29

Universidad Nacional Autónoma de México

Facultad de Ingeniería

DISEÑO Y CONSTRUCCION DE UN SISTEMA PARA CONTROL DE ACCESO

Tesis Profesional

Que para obtener el título de:

INGENIERO EN COMPUTACION

P r e s e n t a :

JORGE RUBIO MONROY

Director: Ing. Juan B. Martínez

México, D. F. 1988

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

T E M A R I O

F I G U R A S

FOTOGRAFIAS

I. INTRODUCCION

La ciencia se preocupa por la exploración de la naturaleza, por la comprensión de los fenómenos naturales y por la necesidad que existe de conocer todo lo que ocurre a nuestro alrededor. Sin embago es la aplicación de los conocimentos adquiridos, lo que permite que nuestra vida diaria pueda llegar a beneficiarse de una manera práctica.

La ciencia pura sólo puede utilizarse de manera provechosa cuando es aplicada de modo inteligente y productivo. La Tecnología es la aplicación de diversas ramas del conocimento a usos prácticos. Algunos tipos de tecnología se generan a partir de un conjunto de hallazgos de diferentes áreas, encausados a un objetivo común e incluso a partir de la aplicación de conocimientos no científicos. La capacidad de estructurar y organizar esos conocimientos es un reto al ingenio del hombre.

Puede ocurrir que la realización práctica de algún objetivo fracase por la limitación de los conocimientos que se tengan sobre una rama de la ciencia, pero lejos de desanimar , estimula al científico a que con investigaciones y experimentos, amplíe las fronteras del conocimento en ese campo y así haga posible su realización práctica y económica.

Los descubrimientos científicos hacen posible un avance constante de las ciencias aplicadas, es la causa de cambios continuos y de una perfección cada vez mayor de la tecnología. También existe una retroalimentación de la ciencia a través de la tecnología, pues al existir limitaciones esta se ve forzada a nuevas investigaciones.

Las fronteras de la ciencia están en constante expansión, y el Ingeniero se enfrenta continuamente al reto de dar un uso práctico a esos nuevos conocimientos.

La evolución de las ciencias por las necesidades de la tecnología, es así como la Electrónica se derivo de la electricidad cuando hubo la necesidad de manejar dispositivos de "Estado Solido". La tecnología fue encontrando aplicaciones a esos descubrimientos y por la necesidad de controlar, manejar y administrar la información en forma recurrente se derivó en conjunto con otras ciencias la Computación.

Dentro del área de control el avance de la tecnología ha sido sorprendente, utilizando diferentes dispositivos se puede dirigir, regular, inspeccionar, comprobar y dominar algún evento o bien un proceso. Todas las variables que intervienen para llevar a cabo el control de un sistema pueden ser procesados por algún dispositivo electrónico que se encargue de ejecutar la acción correspondiente y si además se cuenta con retroalimentación se tendrá un control automático del sistema.

En el campo de la Electrónica Digital la tecnología de semiconductores ha evolucionado en forma sorprendente, en 1970 surgió el microprocesador, este dispositivo permite monitorear variables y tomar acciones de control en fracción de segundos, contiene la Unidad de Control (UC) y la Unidad Aritmética y Lógica (ALU), asociando una unidad de memoria y una de entrada/salida se conforma el dispositivo que conocemos como microcomputadora. Posteriormente con el avance de la tecnología y la necesidad de integrar todo conformando un solo dispositivo se creó la microcomputadora integrada en un Chip que contiene las cuatro unidades básicas.

En el Laboratorio de Automatización del Instituto de Ingeniería se requería de un dispositivo que permitiera controlar el acceso al Laboratorio, ya que existe mucha gente que requiere entrar pero no cualquiera tiene autorización, así que se pensó en un dispositivo tal que a través de un teclado se le alimentara una secuencia de dígitos (clave), la reconociera y permitiera o no el acceso al Laboratorio. Este dispositivo contempla utilizar la microcomputadora integrada en un chip (microcontrolador) por sus características aplicables al control y procesaminto de información, se cuenta con equipo y programas de apoyo que permiten su programación, su costo no es elevado y conviene experimentar aplicaciones con este chip.

2. DESCRIPCION GENERAL

El Dispositivo Para Control de Acceso "DJPAK" se diseño y construyó sobre una tarjeta "Punto Flotante", posteriormente se hizo una tarjeta con el circuito impreso para futuras instalaciones. Para describir mejor su funcionamiento lo dividiremos en 8 bloques que son :

- Fuente de Poder
- Microcontrolador
- Teclado
- Despliegue de Datos
- Número de Clave
- Control Remoto
- Señales de Función
- Accionador de la Contra-Eléctrica

El Micro controlador se encarga del monitoreo y el control de los dispositivos periféricos, a su vez este se encuentra gobernado por un Programa de Control (Fig. 1.1), cada uno de estos bloques tienen la siguiente función :

a) Fuente de Poder :

Su función principal es la de proporcionar un voltaje regulado de 5 Volts a 500 miliampers al Sistema, todos los elementos trabajan a ese voltaje.

b) Microcontrolador :

De acuerdo a la información que monitoree, tome acciones de control sobre los dispositivos periféricos, este microcontrolador es específicamente el 8748.

c) Teclado :

Transmite la información pulsada por el usuario hacia el microcontrolador, este se encargará de interpretarla y saber que tecla fue oprimida. Consta de un juego de 11 teclas conectadas en forma matricial, el teclado cuenta también con un interruptor al microcontrolador que sirve para Resetear (recolocar) al Sistema.

d) Despliegue de Datos :

Por medio de un despliegue de 7 segmentos se presenta el número de la clave seleccionada mientras esté sin utilizarse, al momento de oprimir alguna tecla despliega el número asociado a dicha tecla, también indica cuando la tecla oprimida es equivocada. Este bloque del sistema no lo ve el usuario, pero auxilia para la selección de la clave e indica a la persona que la instala su buen funcionamiento.

e) Número de Clave :

Permite seleccionar con un juego de 4 microswitchs alguna de las 16 claves programadas y será monitoreada por el microcontrolador al iniciar el Sistema.

f) Control Remoto :

Esta opción permite accionar en forma manual y remota la contraeléctrica, esta señal es sensada por el microcontrolador.

g) Señales de Función :

Estas señales indican al usuario que el "DIPAK" está funcionando adecuadamente; consiste básicamente en un Led que se encuentra intermitente, al momento de oprimir alguna tecla el Led se detiene y al mismo tiempo se genera una señal audible en una bocina.

h) Accionador de la Contra-Eléctrica :

Acciona a través de un reelevador la contra-eléctrica. Cuando el microcontrolador detecta una secuencia de 4 dígitos correcta o bien el control remoto ha sido oprimido, se selecciona al reelevador y una vez energizado permite el paso de un voltaje eléctirco no rectificado, tomado directamente del transformador de la Fuente de Poder y así se genera una corriente en el solenoide que libera la contra-chapa de la puerta, permitiendo el acceso.

Los Bloques de la Fuente de Poder,el Microcontrolador, el Despliegue de Datos, el Número de Clave y parte de Señales de Función se encuentran integradas en el Circuito Impreso, el Teclado, el Control Remoto y una parte del accionador de la Contra-eléctrica y las Señales de Función son externos al circuito impreso pero son gobernados por este.

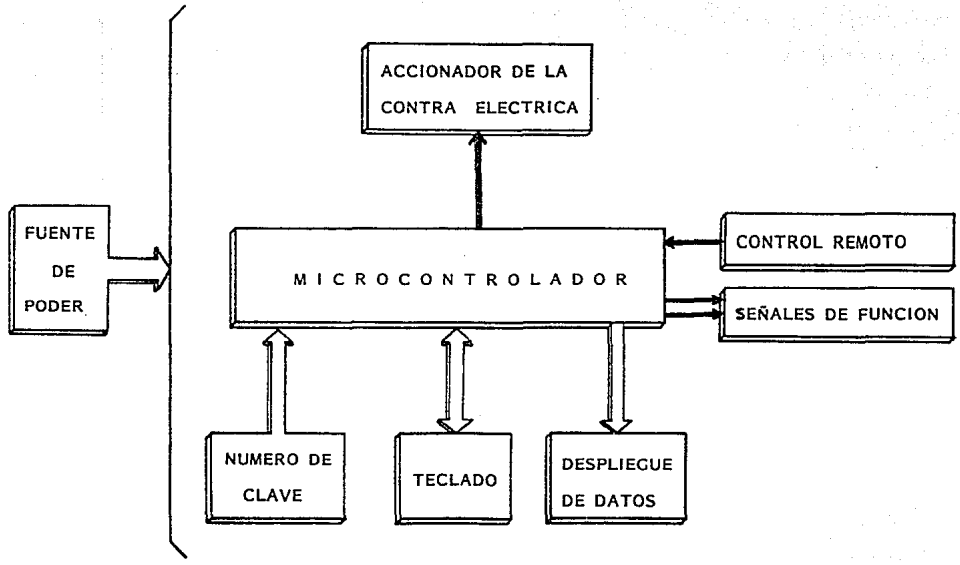

ACCIONADOR DE LA
CONTRA ELECTRICA
FUENTE
DE
PODER
MICROCONTROLADOR
CONTROL REMOTO
SEÑALES DE FUNCION
NUMERO DE
CLAVE
TECLADO
DESPLIEGUE
DE DATOS

Figura 1.1

3. DESCRIPCION HARDWARE

Describiré en detalle cada uno de los bloques que conforman el Dispositivo Para Control de Acceso "DIPAK" en orden de importancia, mencionando su modo de operación, asi como la implementación de cada bloque :

3.1 EL MICROCONTROLADOR

Es el dispositivo central del "DIPAK", constituído por el 8748 que por sus caracteristicas y por los elementos de apoyo que mas adelante describiremos, se acoplo perfectamente para nuestro objetivo.

El avance de la tecnología permitió a INTEL crear una familia de microcomputadoras integradas en un solo chip llamada MCS-48; la microcomputadora que se empleó fue el 8748 que posee memoria programable (EPROM), sus principales funciones son :

* CPU de 8 Bits
* Memoria Programable EPROM de 1K x 8
* Memoria de Datos de 64 bytes
* 27 Lineas de Entrada/Salida
* Contador de Tiempo/Eventos de 8 Bits

La alimentación es de 5 Volts por lo que es compatible con circuitos TTL; tiene un repertorio de mas de 90 instrucciones ejecutables en uno o dos ciclos de máquina.

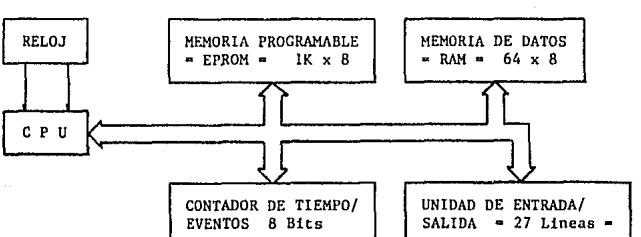


Figura 3.1 Diagrama de Bloques

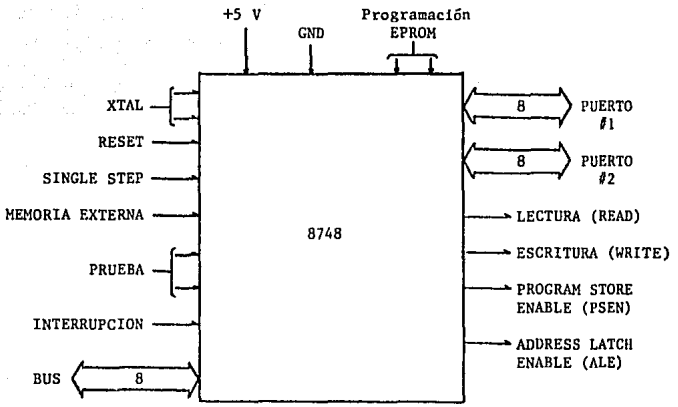


Figura 3.2 Diagrama Lógico del 8748

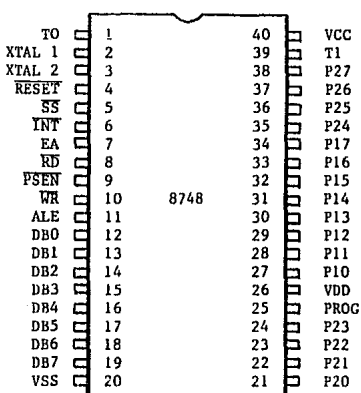


Figura 3.3 Diagrama Físico

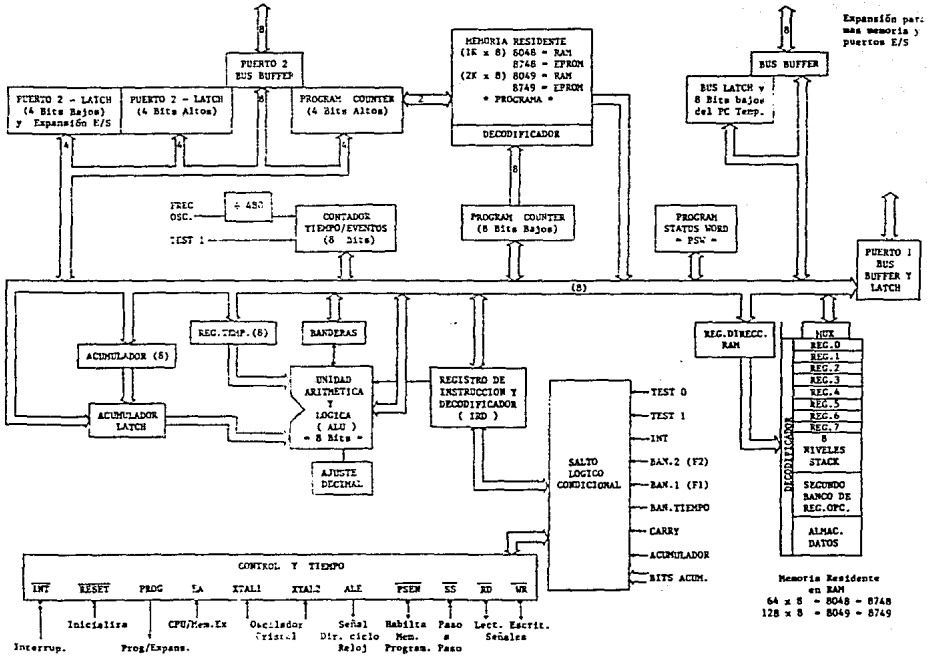


Figura 3.4 Arquitectura MCS-48

3.1.1 Función los Alambres del 8748

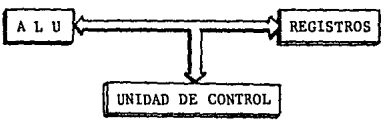
A continuación se describe la función que tiene cada alambre :

ASIGNACION	No.ALAMBRE	FUNCION
VSS	20	Tierra GND
VDD	26	Programación +25 V Operación +5 V (ROM y PROM)
VCC	40	Fuente Principal de Poder +5 V Programación para el 8748
PROG	25	Programa de Pulsos (+25 V) durante la programación del 8748
P10-P17 (Puerto 1)	27-34	Puerto quasi-bidireccional de 8 Bits (Resistencia interna de Pullup ≐ 50 KΩ)
P20-P27 (Puerto 2)	21-24 35-38	Puerto Quasi-bidireccional de 8 Bits (Resistencia interna de Pullup ≐ 50 KΩ) Los Bits P20- P23 del Puerto 2 pueden contener los 4 Bits altos del Program Counter para direcc. la Memoria Externa Programada y sirven como 4 Bits de E/S expandiendo el BUS con el 8243
D0-D7 (BUS)	12-19	Puerto Bidireccional que puede ser escrito o leído en sincronía con las señales RD, WR. El dato es guardado(Latch). Puede contener los 8 Bits bajos del PC para direcc.la mem. ext. programada bajo el control de la señal PSEN. También puede direccionar memoria RAM bajo control del ALE, RD y WR.
T0	1	Entrada de prueba, usada para saltos condicionales. Puede habilitarse como salida de reloj. Se usa durante la programación del chip.
T1	39	Entrada de Prueba, usada para saltos condicionales. Puede habilitarse como la entrada de un contador de eventos.

ASIGNACION	No.ALAMBRE	FUNCION
$\overline{INT}$	6	Entrada de Interrupción. Genera una interrupcion si está habilitada. Si está deshabilitada se utiliza en saltos condicionales. (Activo Bajo)
$\overline{RD}$	8	Señal de Salida, activada durante una lectura al BUS. Puede habilitar un dato de un dispositivo externo conecectado al BUS. Para la Lectura de Datos de Memoria Externa. (Activo Baja)
$\overline{RESET}$	4	Inicia al Procesador. Usado durante la programación y verificación de la Mem. Programada PROM. (Activo Bajo) (Pullup interno = 200 KΩ)
$\overline{WR}$	10	Señal de Salida, activada durante una escritura en el BUS. Utilizada para escritura en la Memoria Externa. (Activo Bajo)
ALE	11	Address Latch Enable. Generada cada ciclo, utilizada como una señal de reloj. Durante el estado negativo del ALE se puede direccionar internamente el programa de memoria y datos externos.
$\overline{PSEN}$	9	Program Store Enable. Esta señal de salida ocurre durante el Fetch del Programa de Memoria Externa. (Activo Baja)
$\overline{SS}$	5	Single Step. Se usa en conjunto con el ALE manejando "paso a paso" cada instrucción del procesador. (Activo Baja) (Resistencia interna de Pullup = 300 KΩ)
EA	7	External Access. Entrada de Acceso Externo el cual provoca que el Fetch de la Memoria Programada se refiera a la Memoria Externa Utilizada para la Emulación y depuración, es escencial para la verificación y programación. (Activo Alta)
XTAL1	2	Una pata del Cristal para generar la oscilación , puede ser una fuente externa.
XTAL2	3	Otro lado de la entrada del Cristal.

3.1.2 Arquitectura del CPU.

El CPU lo podemos subdividir en 3 Bloques :



3.1.2.1 Registros

Podemos subdividir a los registros en :

- Registros De Proposito General :

  Son registros de almacenamiento temporal que interactuan con la Unidad de Control y el ALU (R0 - R7).

- Registros de Proposito Particular :

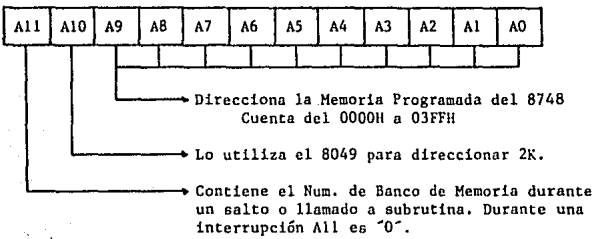
  . Acumulador (A)
  . Program Counter (PC)
  . Stack Pointer (SP)
  . Banderas
  . Instruction Decoder
  . Program Status Word (PSW)

3.1.2.1.1 Acumulador

Es un registro muy importante, es una fuente de entrada al ALU y a menudo es el destino del resultado de la operación ejecutada dentro del ALU. Para intercambiar un Dato de Memoria y en los Puertos de Entrada/Salida.

3.1.2.1.2 Program Counter (Contador del Programa)

Contiene la dirección de la Localidad de Memoria de donde se leerá la siguiente instrucción. Utiliza 10 bits para direccionar las 1024 palabras de la memoria programada, el PC se inicia con cero activando la linea de RESET.



3.1.2.1.3 Stack Pointer (Contador de la Pila)

Contador relativo, contiene la dirección del último nivel del Stack en RAM, el Stack tiene 8 niveles por lo que el SP cuenta del 000 al 111. El Stack es un área dentro de la memoria de datos (RAM), está compuesto de 8 niveles de 2 bytes cada uno, empieza en el (R8,R9) y el último nivel está en el (R22,R23).

Cuando se genera un llamado a subrutina o bien una interrupción se sigue la siguiente secuencia :

```
PC          --→  PC + 1
n           --→  8 + SP * 2
(Rn 0-7)    --→ PC 0-7
(Rn+1 0-3)  --→ PC 8-11
(Rn+1 4-7)  --→ PSW 4-7
SP          --→ SP + 1
PC          --→ Dir. de Salto
```

Cuando se detecta el retorno de subrutina (RTN) se tiene la siguiente secuencia :

```
SP       --→ SP - 1
n        --→ 8 + SP * 2
PC 0-7   --→ (Rn 0-7)
PC 8-11  --→ (Rn+1 0-3)
PSW 4-7  --→ (Rn+1 4-7)
```

... donde :

PC = Program Counter
SP = Stack Pointer
PSW = Program Status Word
Rn = Reg. Proposito General
n = No. de Reg. a utilizar

3.1.2.1.4 Banderas

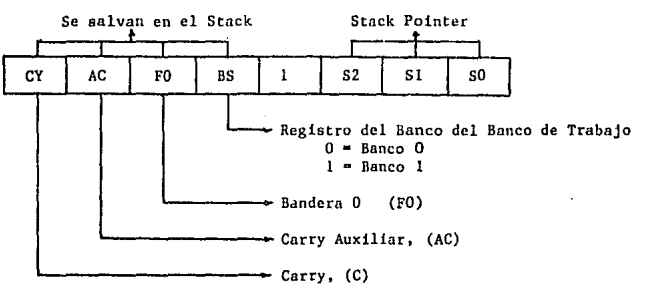
La microcomputadora maneja 4 banderas, posee instrucciones que permite al usuario limpiarlas, complementarlas y probarlas, son :

- Carry (CY) ..... Indica que la operación ejecutada en el Acumulador tuvo un sobreflujo.
- Carry Auxiliar (AC) ... Generada por la instrucción ADD y utilizada por el Ajuste Decimal.
- F0 y F1 ..... Banderas utilizadas por el usuario.

Las Banderas CY, AC y F0 están contenidas en el PSW.

3.1.2.1.5 Program Status Word - Palabra del Estado del Programa (PSW)

Contiene información del Stack Pointer y Banderas que el procesador afecta, dependiendo de la instrucción ejecutada. El PSW realmacena los estados de la máquina después de alguna secuencia.



Los 4 Bits mas significativos se guardan en el Stack de memoria cuando se genera un llamado a subrutina o bien una interrupción.

3.1.2.1.6 Registro Decodificador de Instrucciones (IRD)

Una porción del código de operación de cada instrucción del programa es almacenada en el registro decodificador de instrucciones y convierta a la salida una función de control para cada uno de los bloques de la sección aritmética.

3.1.2.2 Unidad Aritmética y Lógica (ALU)

El ALU permite ejecutar operaciones binarias aritméticas y lógicas entre dos operandos y está bajo el control del Decodificador de Instrucciones de la Unidad de Control. La operación se realiza con un registro de propósito general y el acumulador que es donde se almacena el resultado de la operación. El ALU puede ejecutar las siguientes instrucciones :

- Suma con o sin acarreo.
- Funciones Lógicas : AND, OR, XOR.
- Incremento / Decremento.
- Complemento de Bit.
- Rotar a la Izquierda o a la Derecha.
- Intercambio de Nibbles (4x4 Bits)
- Ajuste Decimal.

3.1.2.3 Unidad de Control (UC)

Se encarga de coordinar las funciones de las otras unidades, llevando control sobre la unidad de memoria, el ALU y la unidad de E/S. La Unidad de Control recibe la instrucción del programa a través del Registro Decodificador de Instrucciones (IRD) y la ejecuta generando señales de control que activan o desactivan según sea el caso las unidades controladas.

Acciones de la Unidad de Control :

- Señales de Prueba
- Salto Lógico Condicional
- Señales de Interrupción

La UC hace uso tanto de los Registros de Propósito Particular, controlando de esta manera la secuencia y el ambiente del sistema. Además toma acciones de control sobre otros bloques de la microcomputadora como el Contador de Estados y de Tiempo, Direccionamiento Externo, etc. que analizaremos en forma separada.

3.1.2.3.1 Señales de Prueba

Los pines TO, T1 e INT son entradas utilizadas como señales de prueba que al ser sensadas por la UC pueden ejecutarse Saltos Condicionales. Al iniciarse el Sistema el valor de TO y T1 es cero y el de INT es uno.

3.1.2.3.2 Salto Lógico Condicional

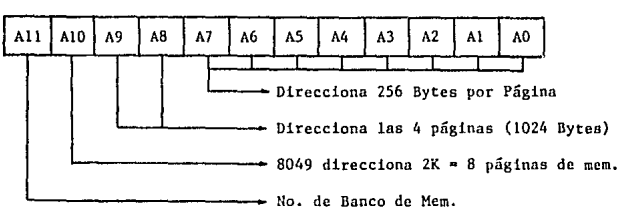
Existen varias condiciones habilitadas en forma interna o externa por las que la UC puede preguntar por medio de instrucciones ya definidas, que permiten cambiar la secuencia del programa en ejecución.

La siguiente tabla muestra que variables (condicionales) puede sensar la UC y el estado que deben tener para tomar una acción de Salto :

PRUEBA	SALTO CONDICIONAL	
	Todos los Bits = 0	No Todos = 0
Acumulador	---	1
Bit-Acumulador	0	1
Bandera de Carry	---	1
Banderas (F0,F1)	---	1
Bandera de Sobreflujo de Tiempo	---	1
Entradas de Prueba (T0,T1)	0	1
Entrada de Interrupción (INT)	0	---

Como vimos anteriormente el Program Counter (Contador de Programa) consta de 12 Bits, en nuestro caso para direccionar 1K de Memoria Programable requerimos de 10 Bits, sin embargo el direccionamiento para Saltos Condicionales utiliza unicamente 8 Bits, por lo que se manejan 4 páginas de 256 Bytes c/u direccionadas por los 2 siguientes Bits del PC.

Program Counter (PC):



Las instrucciones de Salto pueden direccionarse dentro de cada página pues únicamente cambia la secuencia en el PC del Bit 0 al 7, direccionando 256 Bytes, por lo que hay que tener cuidado de no salirse de la página al ejecutar un salto, pues al no modificarse los Bits 8 y 9 se tendría un direccionamiento equivocado. Existe una instrucción de salto directo incondicional que puede direccionar las 4 páginas de la Memoria Programable.

3.1.2.3.3 Señales de Interrupción

El pin de entrada $\overline{INT}$ puede ser habilidada para funcionar como una interrupción EN I y deshabilitada con DIS I o al generar un RESET a la microcomputadora. Una vez habilitada, la interrupción se genera con un nivel bajo "0" en el pin $\overline{INT}$ y se detecta durante el ciclo ALE, causando un "salto a subrutina" en la localidad 3 de la Memoria Programable, donde por lo general se tiene un salto incondicional para manejar la interrupción en otra parte de la memoria; como en una subrutina el Program Counter y el PSW son salvados en el STACK, durante la ejecución de la subrutina los demás pedidos de interrupción serán ignorados hasta finalizar su ejecución con una instrucción de retorno (RETR), regresando al ambiente anterior a la interrupción.

3.1.2.3.5 RESET (Recolocar)

Es una señal de control externa que permite iniciar la Unidad de Control de la microcomputadora, esta constituido por un Schmitt-Trigger con una resistencia interna de Pull-up (= 200 KΩ) que en combinación con un capacitor externo al aplicar un nivel bajo "0" se genera un pulso que recoloca la circuiteria interna afectando las siguientes funciones :

* Program Counter (PC) ----→ 0
* Stack Pointer (SP) ----→ 0
* Banco de Registros ----→ 0
* Banco de Memoria ----→ 0
* BUS (todos los Bits) ----→ 1 (excepto cuando EA=5V)
* Puertos 1 y 2 ----→ Modo de Entrada (INPUT)
* Interrupciones ----→ Deshabilita
* Contador Tiempo/Eventos ----→ Detiene
* Bandera de Tiempo ----→ 0
* Banderas F0 y F1 ----→ 0
* Pin T0 ----→ Deshabilita como reloj

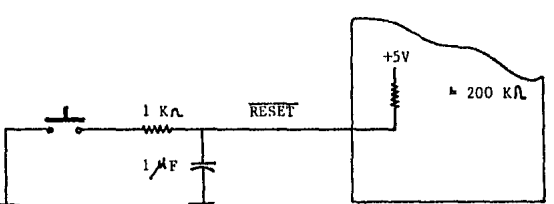


Figura 3.5 RESET

3.1.2.3.4 Ejecución Escalonada

Permite al usuario ejecutar instrucción por instrucción, su modo de operación es el siguiente :

a) Para detener al procesador es necesario aplicar un nivel bajo "0" al pin $\overline{SS}$.
b) El proceso se detiene antes de la ejecución de la siguiente instrucción (Edo.3 de la Sig. instrucción) dejando en alto el ALE. Si se procesaba una instrucción de doble ciclo, ambos ciclos se completarán antes de detenerse.
c) El proceso puede permanecer detenido indefinidamente manteniendo el ALE en alto y la dirección de la siguiente instrucción a ser traída en el BUS y en la mitad mas baja del puerto 2.
d) El procesador sale del modo de detenido aplicando un nivel alto "1" al pin $\overline{SS}$, el ALE muestra un nivel bajo "0".
e) Para detener al procesador en la siguiente instrucción, se da un nivel bajo al $\overline{SS}$ tan pronto como el ALE está en nivel bajo. Si el $\overline{SS}$ se deja en nivel alto el procesador permanece en modo de RUN.

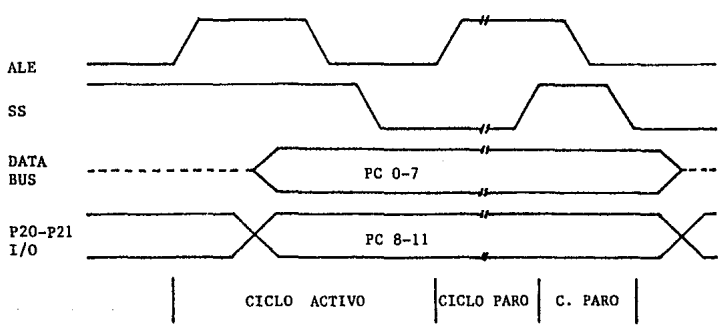


Figura 3.6 Diagrama de Tiempo Single-Step

3.1.3 CIRCUITOS DE RELOJ Y TIEMPO

La generación de pulsos para el MCS48 se maneja a través de la frecuencia de referencia que puede ser un cristal (XTAL), un inductor o un reloj externo.

3.1.3.1 OSCILADOR

Es un circuito resonante de alta ganancia con un rango de frecuencia de 1 a 6 MHz. El pin XTAL1 es la entrada a la etapa amplificadora, mientras XTAL2 es la salida. Un Cristal o un inductor permiten la retroalimentación y la fase de cambio requerida para la oscilación. De ahí se generan las frecuencias para los demás elementos. También podemos utilizar una fuente externa de frecuencia

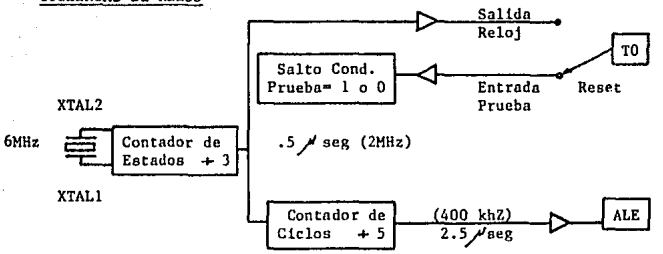
3.1.3.2 CONTADOR DE ESTADOS

La salida del oscilador es dividida entre 3 en el contador de estados, creando un reloj (CLK) que define el tiempo para cada estado y se puede manejar externamente habilitando el pin T0 con la instrucción ENT0 CLK. El reloj es deshabilitado unicamente con un RESET.

3.1.3.3 CONTADOR DE CICLOS

En un Contador de Ciclos CLK es dividido entre 5, dando una señal de reloj cada ciclo de máquina (5 estados de máquina). Este reloj es llamado Address Latch Enable (ALE) y se emplea básicamente para la expansión de memoria.

UTILERIAS DE RELOJ



CICLO DE INSTRUCCION

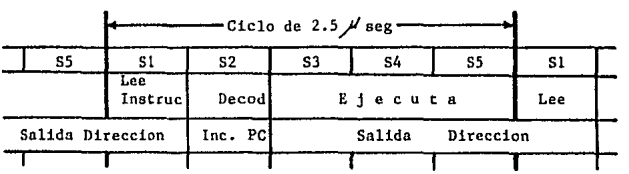


DIAGRAMA DE TIEMPO

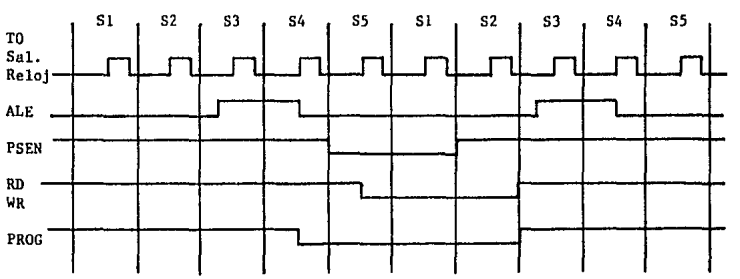


Figura 3.7 Circuitos de Reloj y Tiempo.

3.1.4 CONTADOR DE TIEMPO / EVENTOS

El microcontrolador posee un contador interno que puede ser cargado o leído por software, no se afecta por el RESET y puede funcionar como :

- Un contador de eventos habilitado por la instrucción START CNT
- Un contador de Tiempo habilitado por la instrucción START T
- Detener el Contador por medio de un RESET o por la instrucción STOP TCNT

Cuando el Contador sobrepasa el máximo valor que puede registrar (FFH), reinicia con el valor 00H y se afecta la bandera de sobreflujo que puede ser utilizada para un salto condicional (JTF) o bien si se encuentra habilitada como interrupción (EN TCNTI / DIS TCNTI) se tendrá un salto a la localidad 7 de la Memoria Programada, donde podrá haber un salto incondicional y se hara el tratamiento igual a un llamado a subrutina, por lo que al encontrar la instrucción RETR, regresará al ambiente anterior.

Cuando se utiliza el Contador de Eventos, cualquier transición de Alto a Bajo en el Pin T1 incrementará al Contador.

Si se utiliza el Contador de Tiempo, este se incrementará cada 32 pulsos del ALE (XTAL / 15) osea cada 480 pulsos del Cristal (XTAL), si se considera que el cristal trabaja a una frecuencia de 6 MHz, tenemos que la frecuencia del contador es de 12.5 KHz, osea que se incrementa cada 80 μ seg y cada 20 milisegundos (256 incrementos), se detecta un sobreflujo, que por software puede almacenarse en registros y tomar acciones de control cada intervalo de Tiempo.

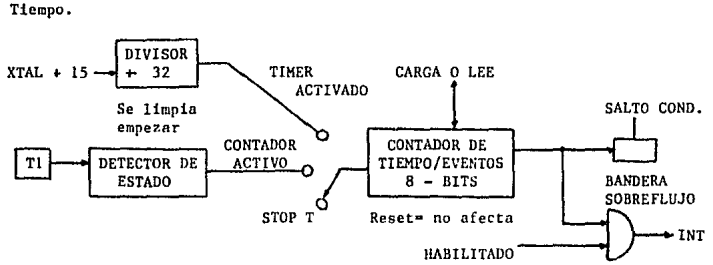


Figura 3.8 Contador de Tiempo y Eventos

3.1.5 UNIDAD DE MEMORIA PROGRAMABLE

La Memoria Programable consiste en el 80/87 48 de 1024 palabras, en el 80/87 49 consiste de 2048 palabras, cada una de 8 Bits de ancho, los cuales son direccionados por el Contador del Programa (PC). En el 8748/49 el usuario puede programar esta memoria pues es EPROM, EN EL 8048/49 la memoria es ROM y es programada de Fábrica.

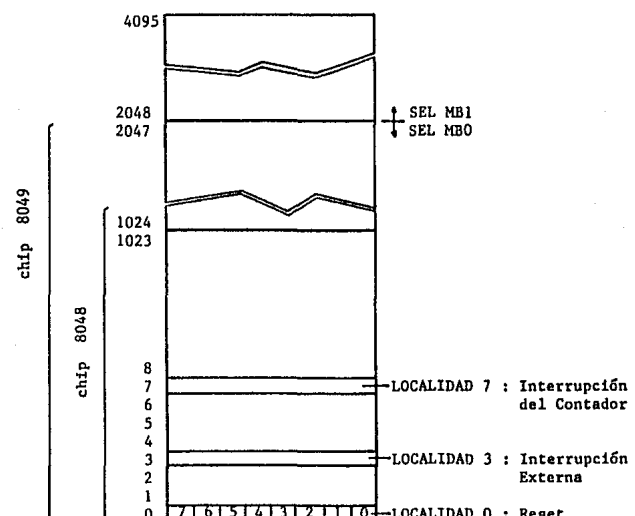


Figura 3.9 Mapa de la Memoria Programable

Existen 3 localidades de la memoria programable de especial importancia:

LOCALIDAD 0 : Activando el RESET del Microcontrolador provocamos que esta sea la primera intrucción a ejecutar.

LOCALIDAD 1 : Activando la linea de entrada de interrupción (INT) del Microcontrolador (la cual deberá estar habilitada) causamos un salto a subrutina a esa localidad.

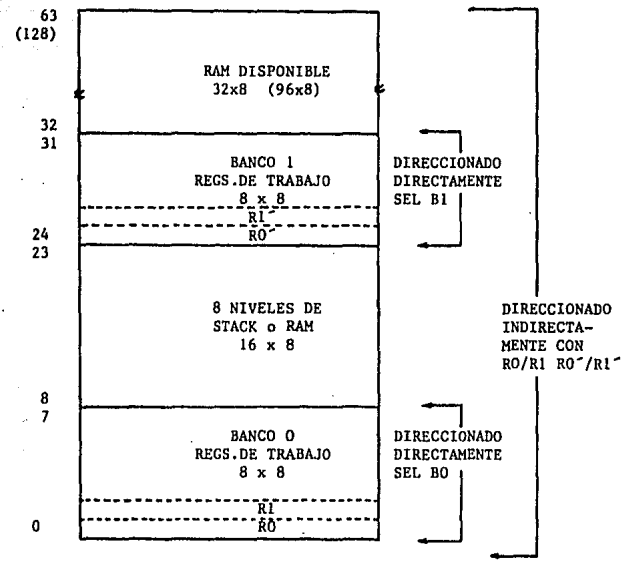
LOCALIDAD 7 : Estando habilitada la interrupción por sobreflujo, al presentarse este en el contador de tiempo / eventos se generará un salto a subrutina a esta localidad.

3.1.6 UNIDAD DE MEMORIA DE DATOS

Esta organizada en 64 (80/8748) o 128 (80/8749) palábras de 8 Bits. Todas las localidades son direccionadas en forma indirecta, a través de 2 registros apuntadores en RAM y residen en la dirección 0 y 1 del vector de registros. Las primeras 8 localidades (0-7) de el vector son designadas para trabajar como registros y son directamente direccionables por diferentes instrucciones. Puesto que estos registros son mas faciles de direccionar, se usarán para almacenar frecuentemente datos de resultados intermedios. La instrucción DJNZ hace muy eficiente el uso de registros de trabajo como cadenas de conteo en un programa, permitiendo a el programador el decremento y prueba de un registro en una sola instrucción.

Podemos seleccionar como Banco de registros a las localidades en RAM (24 - 31) en vez de (0-7) por medio de la instrucción SEL Rn. Este segundo Banco de registros puede ser usado como una extensión de el primer Banco o se reserva para la ejecución de subrutinas.

Puesto que los dos apuntadores en RAM, el registro R0 y R1 son una parte del vector de registros de trabajo, la activación de otro Banco crea dos apuntadores más (R0' y R1') que pueden ser empleados de igual forma. Las localidades en RAM (8-23) son direccionadas por el Stack Pointer durante el llamado a subrutina, donde se tienen máximo 8 niveles, estas localidades pueden ser direccionados también por los registro R0 y R1.



( ) chip 8049 únicamente

Figura 3.10 MAPA DE LA MEMORIA DE DATOS

3.1.7 UNIDAD DE ENTRADA / SALIDA

El microcontrolador cuenta con 27 lineas, las cuales pueden ser usadas para funciones de Entrada o Salida. Estas lineas están agrupadas en 3 puertos de 8 Lineas cada una de las cuales sirven como entrada, salida o bidireccional y 3 entradas de prueba las cuales pueden alterar la secuencia cuando es examinada por una instruccion de salto condicional.

3.1.7.1 PUERTOS 1 Y 2

Ambos puertos se componen de 8 Bits y tienen características identicas. Los datos escritos en estos puertos permanecen estáticos y permanecen así hasta que se vuelva a escribir en ellos. La información que se tenga a la entrada deberá ser leída en el momento que se presenta, pues el puerto no la retiene. Las señales de entrada o salida son de nivel TTL.

Las lineas de los puertos 1 y 2 son llamadas quasibidireccionales porque la estructura del circuito permite que cada linea sirva como una entrada, una salida o ambas, la estructura de este puerto se muestra en la figura 3.11, cada linea tiene un pull-up a través de una resitencia de aprox. 50 K . Este pull-up es suficiente para suministrar la corriente de fuente para un nivel alto de TTL. Al presentarse una conmutación rápida entre "0" y "1" el tiempo de respuesta es de aprox. 500 nseg. debido a la resistencia que debe superar. Un RESET inicializa todas las lineas a una alta impedancia (estado "1"). Esta estructura permite la Entrada/Salida sobre el mismo "pin".

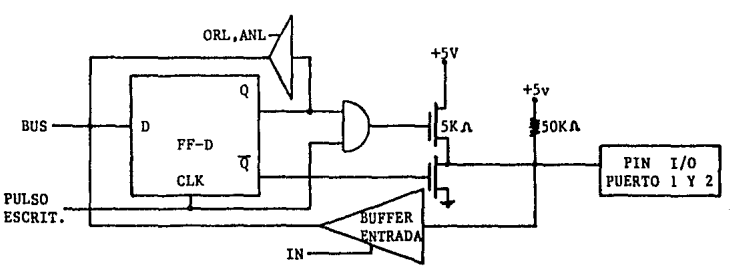


Figura 3.11 Estructura del Puerto Quasi-Bi-Direccional

3.1.7.2 PUERTO BUS (BUS)

Se conforma con 8 Bits, siendo este un puerto bidireccional con entradas y salidas asociadas. Si la caracteristica bidirecional no se necesita, el BUS puede servir para uno u otro, como un puerto que retiene la información que se escriba a la salida o bien un puerto de entrada que no retiene información, las lineas de Entrada/Salida en este puerto no pueden ser mezcaldas.

Los datos de salida se escribe por la instrucción OUTL y para la entrada de datos se emplea la instrucción INS. Las instrucciones INS y OUTL generan pulsos que corresponden a las lineas de Salida RD y WR. Existen instrucciones que permiten emplear al puerto bidireccional (BUS) para expansiones de memoria o puertos. el RESET inicializa las lineas del BUS en alta impedancia "1".

3.1.8. PROGRAMACION, VERIFICACION Y BORRADO DE LA MEM. PROGRAMABLE

La Memoria Programable del 8748 puede ser borrada y reprogramada por el usuario como se indica a continuación :

3.8.1.1 PROGRAMACION Y VERIFICACION

Los pines que el 8748 utiliza para la programación/verificación son los siguientes :

PIN	FUNCION
XTAL1	Entrada de Reloj (1 a 6 MHz)
RESET	Inicializa y guarda la Dirección
TEST0	Selec. el Modo de Programación o Verif.
EA	Activa el Modo de Programación y Verif.
BUS	Entrada de la Direc.y la Instruc.a guardar
P2 0-1	Entrada de la Dirección
Vdd	Suministro de Poder en la Programación
PROG	Entrada del Pulso de Programación

Para llevar a cabo la programación en el 8748 deberá seguirse la siguiente secuencia :

1) Deberán tenerse listas las siguientes señales antes de colocar el 8748

   - Vdd = 5 V
   - El reloj deberá estar funcionando (1 a 6 MHz)
   - RESET = 0 V
   - T0 = 5 V
   - EA = 5 V
   - Los Pines del BUS y PROG = 0V

2) Colocar el 8748 en el Socket
3) T0 = 0V : Selecciona el Modo de Programación
4) EA = 25V : Activa el Modo de Programación
5) Escribe la Dirección en el BUS y en el Puerto 2 (Bits 0-1)
6) RESET = 5 v : Guarada la Dirección
7) Se escribe la instrucción a guardar en el BUS
8) Vdd = 25 V : Fuente de energía para la Programación
9) PROG = 0 V : Seguido por un pulso de 50 mseg a 25 V
10) Vdd = 5 V
11) T0 = 5 V : Modo de Verificación
12) Lee y verifica la instrucción en el BUS
13) T0 = 0 V
14) Para continuar programando saltar al paso 5
15) Para retirar el 8748 deberán cumplirse las condiciones del paso 1

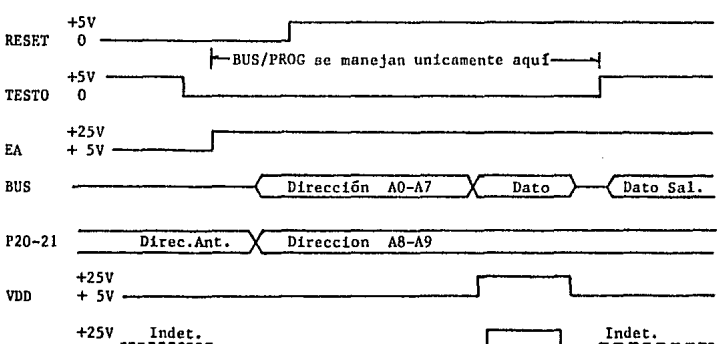


Figura 3.12 Secuencia de Programación / Verificación

Si uno desea unicamente Verificar o leer el programa en memoria la secuencia a seguir es la siguiente :

1) Deberán tenerse listas las siguientes señales antes de colocar el 8748

   - Vdd = 5 V
   - El reloj deberá estar funcionando (1 a 6 MHz)
   - RESET = 0 V
   - T0 = 5 V
   - EA = 5 V
   - Los Pines del BUS y PROG = 0V

2) Insertar el 8048 o el 8748
3) T0 = NC (8048)
   T0 = + 5 V (8748) : Selección del Modo de Verificación
4) EA = +12 V (8048)
   EA = +25 V (8748) : Activa el Modo de Verificación
5) RESET = 0 V
6) Escribe la Dirección en el BUS y en el Puerto 2 (Bits 0-1)
7) RESET = 5 V : Guarda la Dirección
8) La instrucción almacenada se encuetra en el BUS
9) Para continuar verificando salta al paso 5
10) Para retirar el Chip deberán cumplirse las condiciones del paso 1

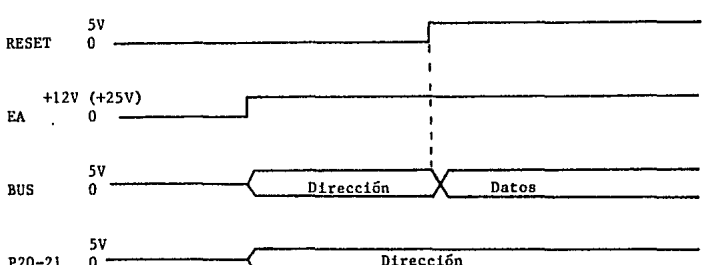


Figura 3.1.3 Secuencia de Lectura del Programa

3.1.8.2 BORRADO DE LA MEMORIA PROGRAMABLE

Para Borrar la Memoria Programable (EPROM) es necesario que se exponga a una luz con baja longitud de onda de aproximadamente 4000 Angstroms. Si el 8748 se encuentra expuesto sin protección a la luz de las lampara fluorecentes puede borrarse la información en aproximadamente 3 años, mientras si está expuesto a la luz del sol se borrará en 1 semana aproximadamente.

El procedimiento para borrar una memoria EPROM recomendable es con luz ultravioleta que tiene una longitud de onda de 2537 Angstroms, el tiempo de borrado es de 15 a 20 minutos.

3.2 FUENTE DE PODER

El Bloque correspondiente a la Fuente de Poder, provee una alimentación constante de +5 Volts a la mayoría de los módulos que conforman al "DIPAK", además de la misma Fuente se derivan 9 Volts No Regulados para activar el reelevador que sirve como interruptor a la Contra-Eléctrica y 9 Volts No Rectificados (AC) que permiten accionarla.

La Fuente de Poder consta de un interruptor de encendido a la entrada de un Transformador de 9 Volts AC a 500 mA, de la salida del Transformador se deriva la alimentación para la Contra-Eléctrica, los 9 V AC se rectifican y sirven para activar al reelevador, este voltaje se regula y de esta manera se obtiene un Voltaje constante de 5 Volts Regulados.

A pesar de que la cantidad de corriente que requiere la Contra-Eléctrica es de aproximadamente 300 mA y del ruido que esta pueda generar, el voltaje se mantiene regulado a 5 Volts constantes y el ruido generado se filtra dentro de la misma Fuente, no afectando a ningún módulo del "DIPAK"

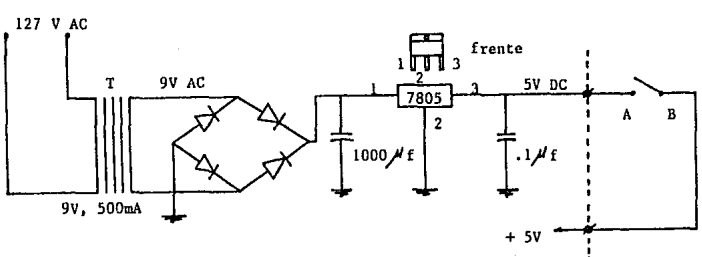
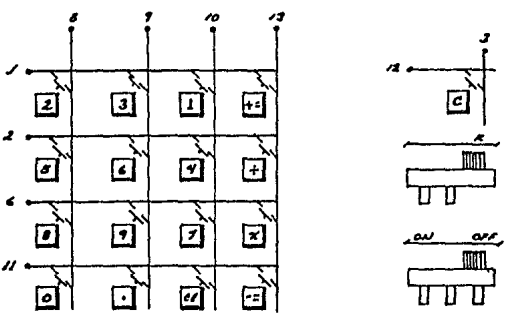


Figura 3.14 Fuente de Poder

3.3 TECLADO

El Teclado está formado por un conjunto de 16 teclas organizadas en forma de matriz de 4 x 4, sin embargo para nuestro propósito empleamos unicamente 12 teclas (4 renglones x 3 columnas) dejando inactiva toda una columna. Además dentro del mismo teclado se cuenta con un interruptor conectado a la alimentación de +5 Volts de la Fuente de Poder, permitiendo de esta manera apagar parcialmente o bien reiniciar (resetear) al "DIPAK".

El microcontrolador se encarga de ir activando en intervalos de tiempo cada uno de los renglones con un cero lógico "0" dejando los demás en uno lógico "1" e irá sensando cada una de las columnas que estan conectadas a +5 Volts, de tal manera que cuando se oprima una tecla el microcontrolador tendrá la posición del renglon activado por un cero lógico "0" y de la columna que contenga un cero lógico "0", detectando de esta manera que tecla fue oprimida.



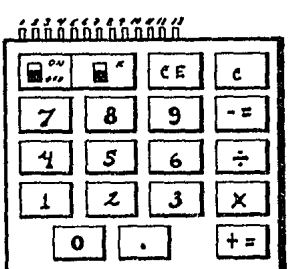


Figura 3.15 Teclado

3.4 SELECCION CLAVE

El DIPAK permite seleccionar de entre un grupo de 16 claves de 4 dígitos cada una, programadas con anterioridad en el microcontrolador en EPROM, para seleccionar una clave se cuenta con 4 microswitchs cuyo valor binario está entre el 0000 y el 1111 osea del 0 al 15 decimal, cada microswitch esta conectado al Puerto 1 de los Bits 2 al 6 en el microcontrolador, únicamente al activar el Sistema el microcontrolador lee el número de clave seleccionada sensando el Puerto 1, para cambiar de Clave, es necesario dar reset (reiniciar) al microcontrolador (Sistema).

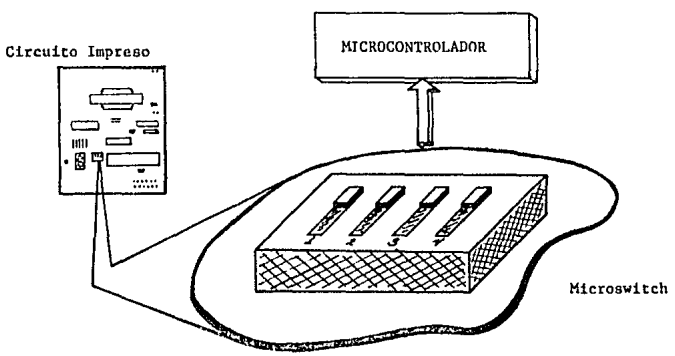


Figura 3.16 Selección Clave

3.5 DESPLIEGUE DE DATOS

El despliegue de Datos está constituido por un módulo de 7 segmentos que se encuentra conectado a un decodificador de 7 segmentos (7448) y éste al BUS del microcontrolador del Bit 0 al 3. Al activar el Sistema el microcontrolador escribe en el BUS el número 0A (Hexadecimal), al ser decodificado, el despliegue de 7 segmentos muestra la letra "c" que indica clave, inmediatamente despues despliega el número de clave seleccionada en los microswitchs y este ciclo lo repite mientras no se oprima ninguna tecla, al momento de oprimir alguna el despliegue muestra el valor asociado a la tecla, a su vez indica si el valor de la tecla oprimida corresponde al la secuencia de la clave seleccionada, si no es igual muestra una "e" de error. El Despliegue de Datos no esta a la vista del usuario externo, su función principal es la de llevar un control interno, indicar el número de clave seleccionada y poder en un momento dado saber que secuencia corresponde a cada clave.

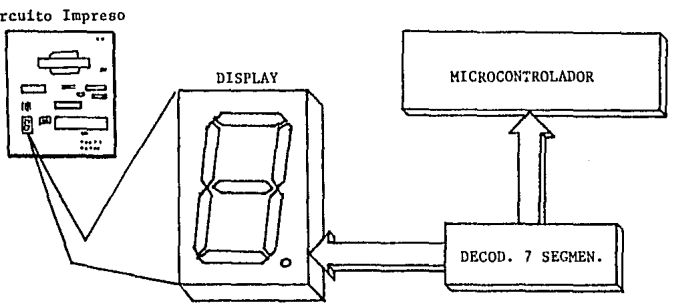


Figura 3.17 Despliegue de Datos

3.6 CONTROL REMOTO

Permite accionar la Contra - Eléctrica desde un punto distante, está constituido por un interruptor (push-botton) conectado a tierra y al pin de prueba T0 del microcontrolador, además en pin T0 está conectado a +5V. La señal de prueba T0 es sensada constantemente por el microcontrolador, mientras no se oprima el interruptor la entrada T0 estará recibiendo un nivel alto "1", al momento de oprimir el interruptor, T0 va a recibir un nivel bajo "0", el microcontrolador al sensar T0 activa la Contra - Eléctrica igual a que se hubiera detectado la secuencia correcta de una clave.

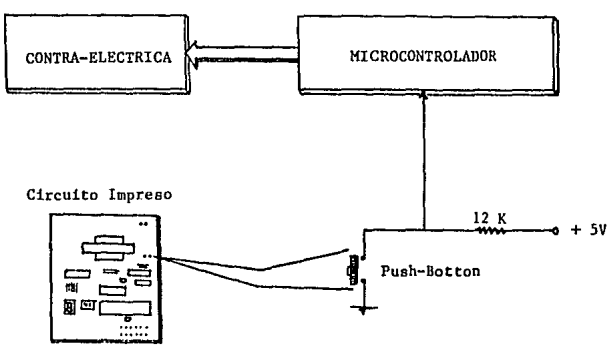


Figura 3.18 Control Remoto

3.7 SEÑALES DE FUNCION

Indican al usuario externo el buen funcionamiento del DIPAK de 2 formas :

Visual : Consiste de 2 Leds conectados en paralelo, uno se encuentra en el circuito impreso para control interno y el otro está conectado en el teclado a la vista del usuario externo. Ambos Leds se encuentran conectados al BUS en el Bit 7 del microcontrolador, el cuál genera una señal visual intermitente en los leds mientras no se oprima ninguna tecla, al momento de oprimir alguna se detiene la señal dejando prendido el Led, al oprimir la siguiente tecla se apaga, continuando así hasta completar la secuencia de 4 y después continúa intermitente

Audible : Esta constituido por una bocina colocada externamente en la caja del teclado. La Bocina esta conectada al Bus en el Bit 6 del microcontrolador. Cuando se oprime alguna tecla, el microcontrolador inmediatamente genera una señal audible en la bocina.

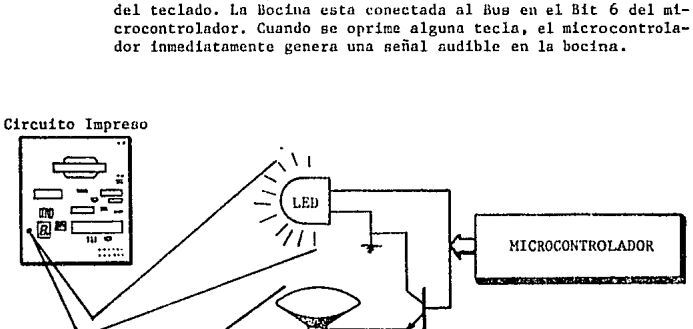


Figura 3.19 Señales de Función

3.8 ACCIONADOR CONTRA-ELECTRICA

El DIPAK permite el acceso accionando una contra-chapa eléctrica que consiste básicamente de un solenoide que al aplicarle un voltaje de 12 V AC libera la contra-chapa. Los 12 V AC se toman del transformador que se emplea para la Fuente de Poder el cual se conecta a un reelevador y este al microcontrolador. Cuando el DIPAK detecta una secuencia de 4 caracteres correcta (CLAVE) o bien accionando el Control Remoto, el microcontrolador genera un nivel alto en el Bit 7 del Puerto 2 conectado a la base de un transistor, el cual a su vez permite el flujo de corriente activando el reelevador y este cierra el circuito aplicando 12 V AC del transformador a la Contra-Chapa. La Contra-Eléctrica tiene conectado al solenoide un vibrador que indica al usuario que puede empujar (o bien jalar) la puerta y poder entrar.

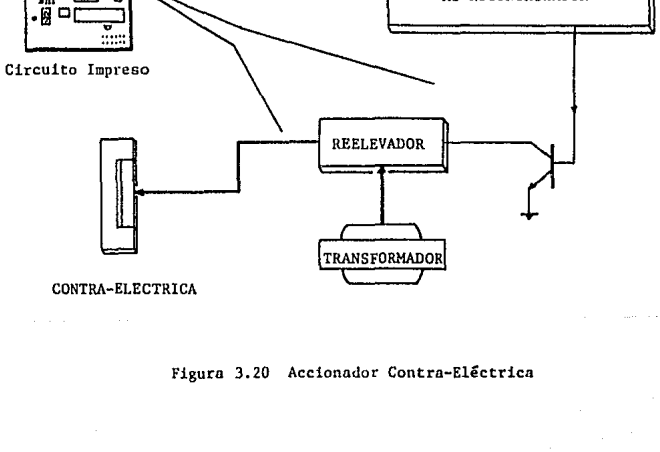


Figura 3.20 Accionador Contra-Eléctrica

3.9 DISPOSICION GENERAL HARDWARE

Los 8 módulos que se describieron anteriormente podemos agruparlos en 3 Tipos de acuerdo a su función :

A) ENTRADA : Son aquellos módulos que proveen de información al Sistema por lo que es necesario sensarlos constantemente.

- Teclado
- Control Remoto

B) PROCESO : Módulos que de alguna manera realizan o participan en el sensado, proceso y control de los demás módulos.

- Microcontrolador
- Fuente de Poder
- Selección Clave
- Despliegue Datos

C) SALIDA : Los módulos de este grupo son aquellos que responden a las acciones de control en el DIPAK de acuerdo a la entrada que se presento en el proceso.

- Señales de Función
- Accionador Contra-Eléctrica

3.9.1 DISPOSICION DE COMPONENTES

A continuación se hará una lista de los componentes que se emplearon para la construcción del DIPAK, así como su disposición dentro del Circuito Impreso.

Componentes Electronicos :

No.	Cantidad	Componenete
1	1	Chip 8748 INTEL
2	1	Cristal 3.5795 MHz (TV Color)
3	1	Capacitor .3 μf Polietileno
4	1	Transformador de 9 V - 500 mA
5	4	Diodos IN4001
6	1	Regulador a 5 V (7805)
7	1	Enfriador Regulador
8	1	Capacitor 1000 μf Electrolítico Axial
9	1	Capacitor .1 μf Polietileno
10	1	Capacitor 10 μF (50v) Electrolítico Radial
11	1	Chip 7448 (Convertidor BCD-7 Seg.)
12	1	Display 7 Seg. (LA-6780 / L / 550 K)
13	10	Resistencias 390 Ω
14	1	Resistencia 12 KΩ
15	1	Microswitch (4)
16	2	Led Rojo (Baja Potencia)
17	2	Transistor BC548
18	1	Reelevador AROMAT AW821198 NC2D-DC12V 4091V 5A 1/10 HP 125,250 V AC 5A 30 V DC

Componentes Mecánicos :

No.	Cantidad	Componenete
1	1	Contra - Eléctrica
2	1	Teclado (Matriz 4x4)
3	1	Bocina
4	1	Portaled
5	1	Interruptor 1 polo 1 tiro
6	1	Push-Botton
7	1	Chasis (12x20) = Circuito Impreso
8	1	Caja Metal.(7x10) = Tecldo/Bocina/Led
9	1	Conector Cable Plano (Hembra y Macho)= Tarjeta
10	1	Conector DB-15 Macho y Hembra
11	1	Concha para DB-15
12	1	Conector Alimentación (Tarjeta-127 V AC)
13	1	Conector Chapa
14	1	Conector Control Remoto
15	4	Aisladores (Tarjeta-Chasis)

Otros :

- 2 Mts. Prom. Cable Telef. (12 alambres)
- 8 Mts. Cable (Alimentación / Contra / Control Remoto)
- Tornillos (Chasis/Teclado/DB-15/Tarjeta)
- Remaches (2)
- Cinta Adhesiva

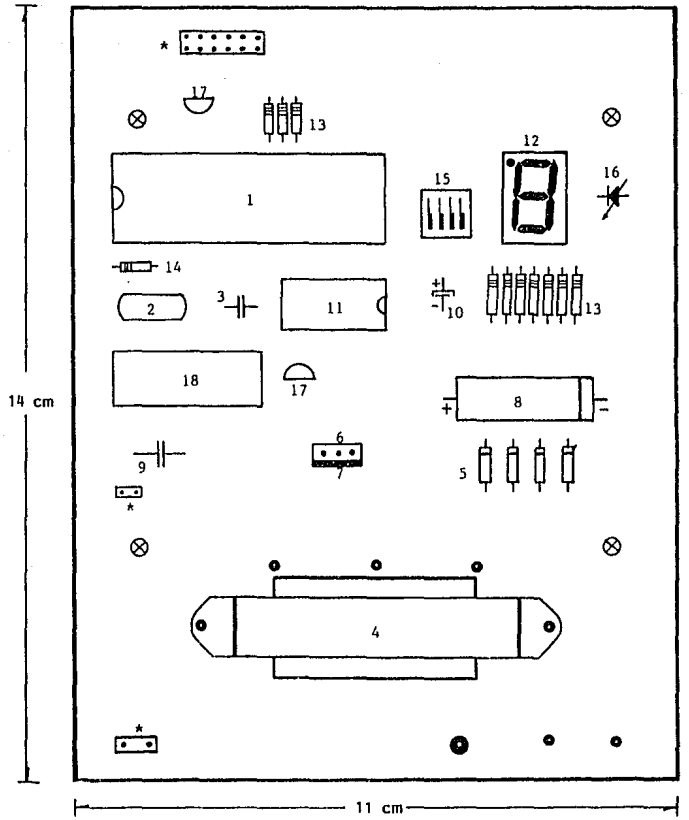


Figura 3.21 LAY OUT DIPAK

* Conectores

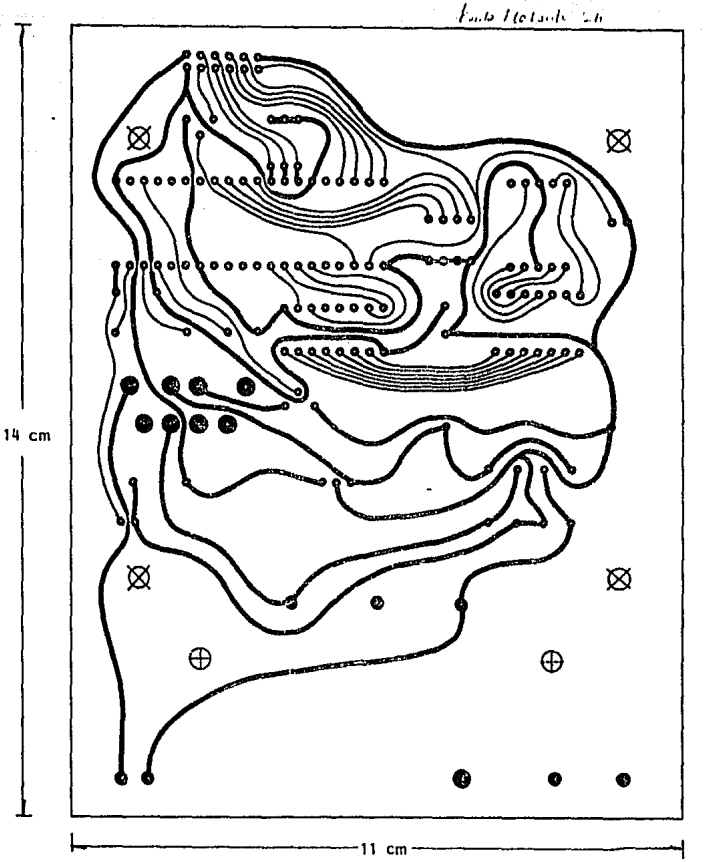


Figura 3.22 Circuito Impreso

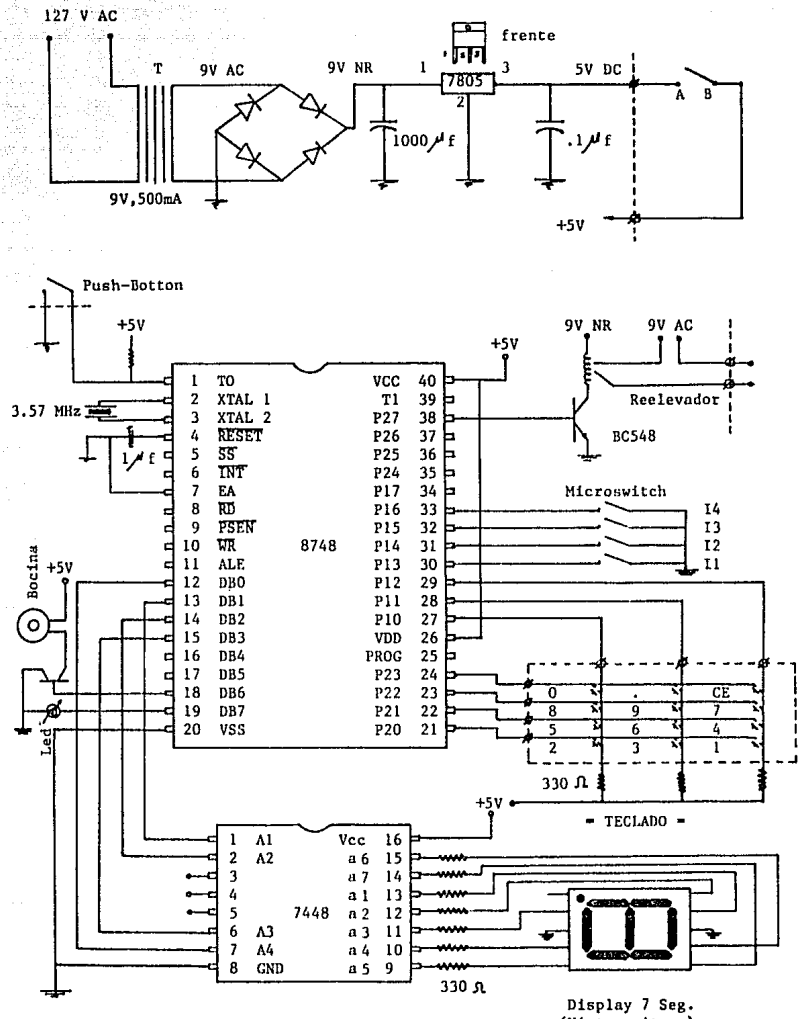


Figura 3.23 Diagrama Eléctrico

4. DESCRIPCION SOFTWARE Y PROGRAMACION

En el Diseño del DIPAK fue necesario hacer una combinación entre Hardware y Software, el primero permite llevar un control físico sobre todas sus componentes, mientras que el Software por medio de instrucciones hace uso del hardware, indicando al microcontrolador sensar variables y ejecutar acciones, por lo que se lleva un control interno.

La microcomputadora 8748 tiene un set de instrucciones mas o menos reducido pero bastante eficiente, este se incluye en el Apéndice A, describiendo cada una de sus instrucciones y su forma de operación.

Para la programación del 8748 (microcontrolador) se utilizó una Computadora PC conectada al Sistema PAT-86 con el programador EXOR-48, además se cuenta con Software de apoyo como el Editor, el Ensamblador, el Simulador y un programa que permite manejar a la Computadora PC como una terminal del PAT-86, todo este soporte se describe con mas detalle en el Apéndice C.

4.1 DESCRIPCION SOFTWARE

Por sus caracteristicas, el programa interno de la microcomputadora (8748) se grabó en EPROM, lo que permitió se hicieran varias pruebas, depurando cada una de ellas antes de llegar a una versión final y será esta la que se describirá a continuación.

4.1.1 PROCEDIMIENTO DE DESARROLLO

Básicamente el obetivo de la microcomputadora es hacer uso del hardware del DIPAK, sensar diferentes señales y tomar acciones de control. En este caso el microcontrolador deberá :

1) Se seleccionará por medio de 4 microswitchs una Clave de entre 16, cada una está compuesta por 4 caracteres los cuales serán asignados por el microcontrolador.
2) Indicar constantemente que todo esta operando correctamente por medio de un led que estará parpadeando mientras todo este bien.
3) Recibir una secuencia de 4 caracteres por medio del teclado.
4) Generar una señal audible en una Bocina si se pulsa alguna tecla.
5) Mostrar en un Display el Num. asignado a la tecla correspondiente cada vez que se oprime alguna e indicar si es correcta o no.
6) Si la secuencia es correcta comparada con la clave asignada o bien cuando se habilita el Control Remoto se deberá activar la Contra - Eléctrica, en caso contrario continuará deshabilitada.

Para poder desarrollar el Software fue necesario tener el diseño completo del hardware del DIPAK, para saber que señales del microcontrolador se implementarían y como se utilizarían.

4.1.1 ALGORITMO

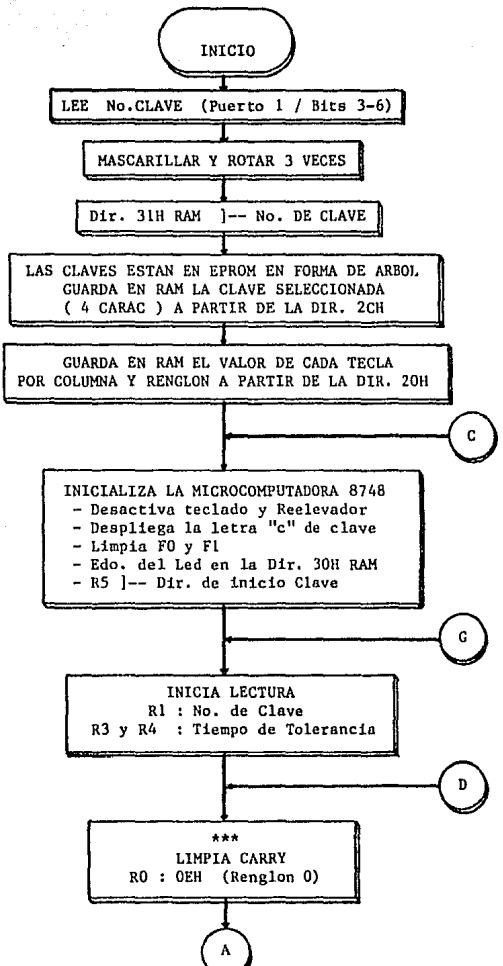
Para poder diseñar y desarrollar el Software se requiere de un Algoritmo que nos muestre de manera sencilla la secuencia que deberá seguir el programa del microcontrolador y de que manera va a afectar a los demás dispositivos. A continuación se muestra esta secuencia :

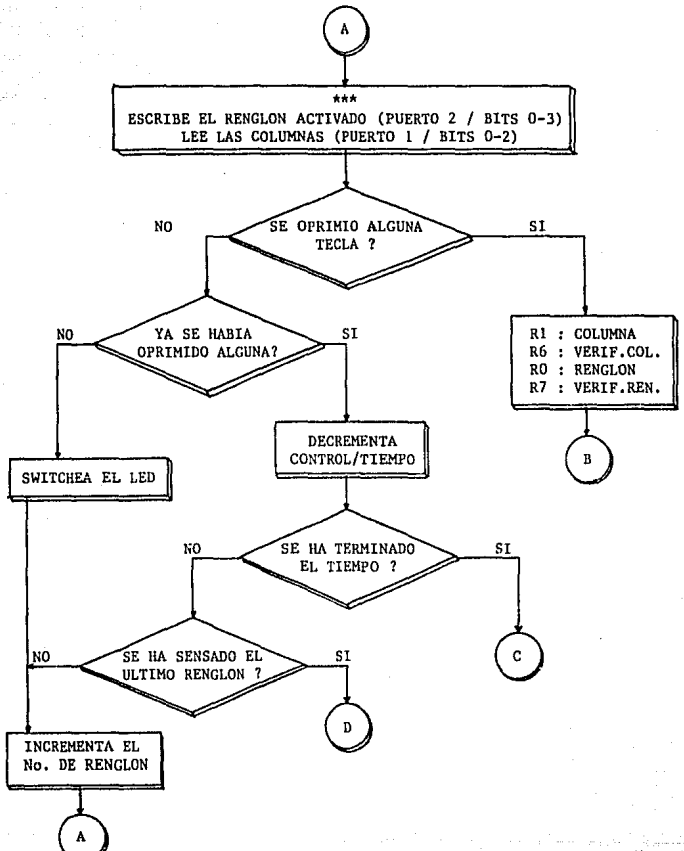
1) Leer el Num. de Clave elegida (0-15) en el microswitch, del Bit 3 al 6 en el Puerto 1.
2) Guardar en RAM la secuencia de caracteres de la Clave que se encuentran en EPROM, de acuerdo al número de la Clave seleccionada.
3) Guarda en RAM el valor de cada Tecla por columna y renglón.
4) Inicializa la Microcomputadora (limpia Banderas, desact. Reelev.,etc.)

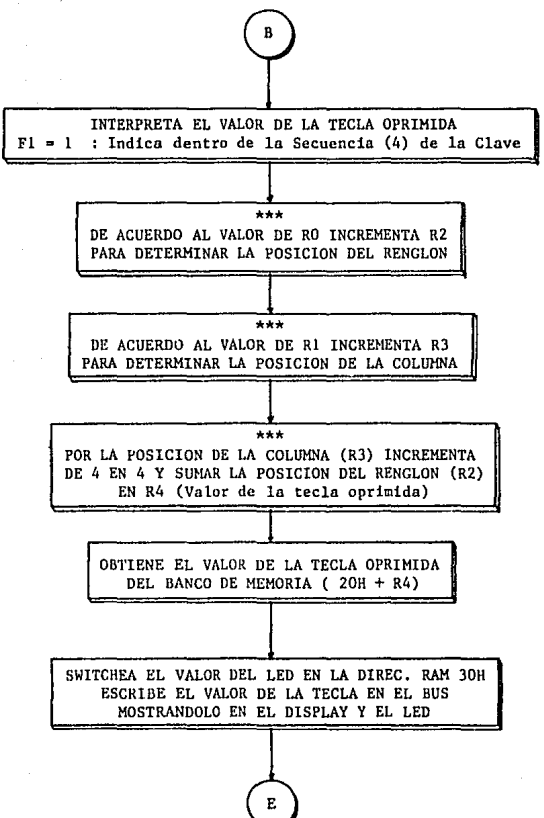
5) Escribe una "c" de clave, muestra el No.de la Clave o bien si ya se ha tecleado algo continuar mostrando el valor de la tecla, el Led debe estar intermitente (ON/OFF) y se debe sensar el Control Remoto, si se activa TO= "0" va al paso 25.
6) Empieza reconocimiento del teclado activando con un nivel bajo "0" renglón por renglón (Puerto 2 Bits 0 al 3).
7) Sensa las columnas en el Puerto 1, Bits 0 al 2.
8) Mientras no se oprima ninguna tecla cambia de renglon y va al paso 5.
9) Si ya se ha oprimido alguna tecla pero no se continua dando la secuencia el micro da un tiempo de tolerancia una vez que transcurre regresa al paso 4.
10) Una vez detectada la tecla procede a hacer un reconocimiento de que tecla fue oprimida asociando columna y renglón.
11) Se asigna el valor de la tecla guardado en RAM.
12) Continua sensando al Control Remoto, si TO=0 salta al paso 25.
13) Switchea el Led, cambia cada vez que se oprime una tecla.
14) Despliega el valor de la Tecla en el Display de 7 segmentos, escribiendo el dato en el BUS del Bit 0 al 3.
15) Da un tiempo de retardo para estabilizar el Teclado.
16) Activa la Bocina generando una señal cuadrada en el BUS Bit 6.
17) Checa si se dejo de oprimir la tecla.
18) Inicia Reconocimiento del valor de la tecla con la secuencia que sigue la Clave.
19) Continua sensando al Control Remoto, si TO=0 salta al paso 25.
20) Compara el valor de la tecla con el valor de la secuencia de la Clave guardada en RAM.
21) Si no es igual activa FO=1 indicando que la secuencia es incorrecta.
22) Si todavía no se completa la secuencia de 4 caracteres, va al paso 5.
23) Si FO esta activado, la clave no fue correcta y regresa al paso 4.
24) Activa el Reelevador durante aprox. 3 seg.
25) Regresa al paso 4.

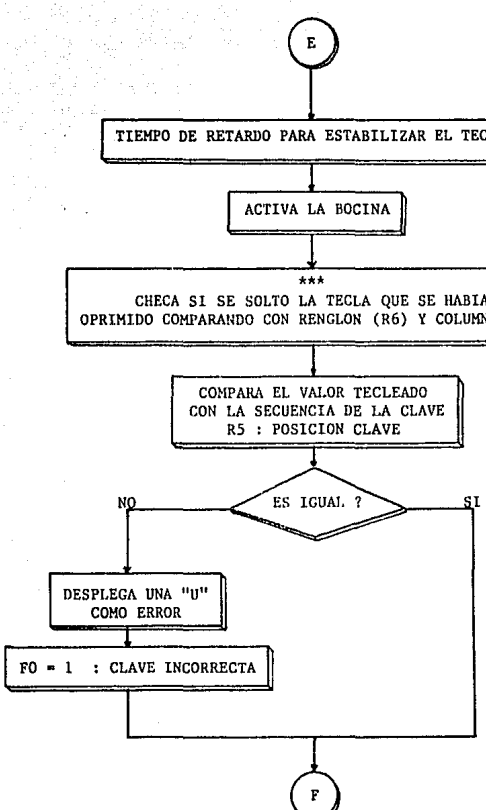
4.1.3 DIAGRAMA DE FLUJO

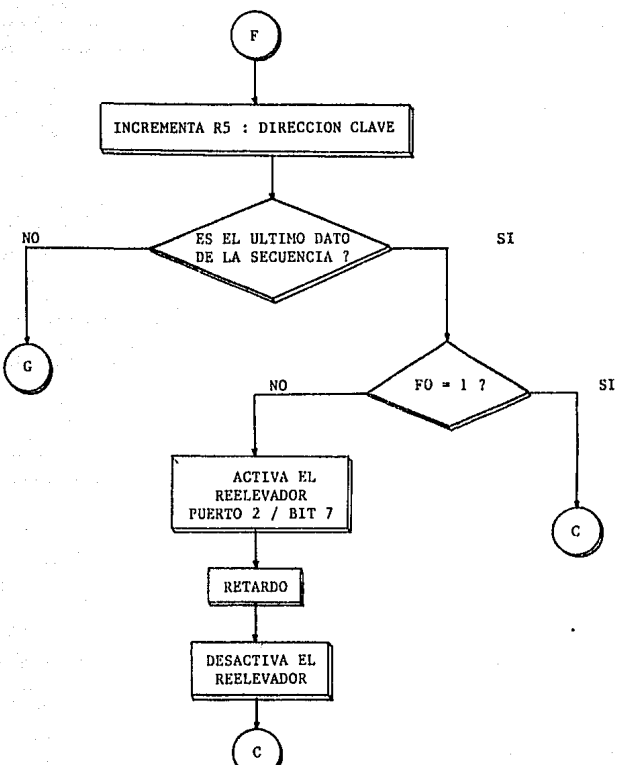
El Diagrama de Flujo permite saber visualmente como funciona el Programa que controla el DIPAK. Todo el programa se estructuró de tal forma que no se utilizaron subrutinas, pues no se encontraron rutinas similares a lo largo del del programa por lo que es un solo bloque. El Listado del Programa se incluye en el Apéndice D.


INICIO
LEE No.CLAVE (Puerto 1 / Bits 3-6)
MASCARILLAR Y ROTAR 3 VECES
Dir. 31H RAM]-- No. DE CLAVE
LAS CLAVES ESTAN EN EPROM EN FORMA DE ARBOL
GUARDA EN RAM LA CLAVE SELECCIONADA
(4 CARAC) A PARTIR DE LA DIR. 2CH
GUARDA EN RAM EL VALOR DE CADA TECLA
POR COLUMNA Y RENGLON A PARTIR DE LA DIR. 20H
C
INICIALIZA LA MICROCOMPUTADORA 8748
- Desactiva teclado y Reelevador
- Despliega la letra "c" de clave
- Limpia F0 y F1
- Edo. del Led en la Dir. 30H RAM
- R5]-- Dir. de inicio Clave
G
INICIA LECTURA
R1 : No. de Clave
R3 y R4 : Tiempo de Tolerancia
D
***
LIMPIA CARRY
R0 : 0EH (Renglon 0)
A


A
***
ESCRIBE EL RENGLON ACTIVADO (PUERTO 2 / BITS 0-3)
LEE LAS COLUMNAS (PUERTO 1 / BITS 0-2)
NO
SE OPRIMIO ALGUNA TECLA ?
SI
NO
YA SE HABIA OPRIMIDO ALGUNA?
SI
R1 : COLUMNA
R6 : VERIF.COL.
R0 : RENGLON
R7 : VERIF.REN.
B
DECREMENTA CONTROL/TIEMPO
SWITCHEA EL LED
NO
SE HA TERMINADO EL TIEMPO ?
SI
C
NO
SE HA SENSADO EL ULTIMO RENGLON ?
SI
D
INCREMENTA EL No. DE RENGLON
A


B
INTERPRETA EL VALOR DE LA TECLA OPRIMIDA
F1 = 1 : Indica dentro de la Secuencia (4) de la Clave
***
DE ACUERDO AL VALOR DE RO INCREMENTA R2
PARA DETERMINAR LA POSICION DEL RENGLON
***
DE ACUERDO AL VALOR DE R1 INCREMENTA R3
PARA DETERMINAR LA POSICION DE LA COLUMNA
***
POR LA POSICION DE LA COLUMNA (R3) INCREMENTA
DE 4 EN 4 Y SUMAR LA POSICION DEL RENGLON (R2)
EN R4 (Valor de la tecla oprimida)
OBTIENE EL VALOR DE LA TECLA OPRIMIDA
DEL BANCO DE MEMORIA (20H + R4)
SWITCHEA EL VALOR DEL LED EN LA DIREC. RAM 30H
ESCRIBE EL VALOR DE LA TECLA EN EL BUS
MOSTRANDOLO EN EL DISPLAY Y EL LED
E


E
TIEMPO DE RETARDO PARA ESTABILIZAR EL TECLADO
ACTIVA LA BOCINA
***
CHECA SI SE SOLTO LA TECLA QUE SE HABIA
OPRIMIDO COMPARANDO CON RENGLON (R6) Y COLUMNA (R7)
COMPARA EL VALOR TECLEADO
CON LA SECUENCIA DE LA CLAVE
R5 : POSICION CLAVE
NO
ES IGUAL ?
SI
DESPLEGA UNA "U"
COMO ERROR
FO = 1 : CLAVE INCORRECTA
F


F
INCREMENTA R5 : DIRECCION CLAVE
ES EL ULTIMO DATO DE LA SECUENCIA ?
NO
SI
G
FO = 1 ?
NO
SI
ACTIVA EL REELEVADOR PUERTO 2 / BIT 7
C
RETARDO
DESACTIVA EL REELEVADOR
C

Donde se sensa el control remoto en el Diagrama de Flujo se indica con tres asteriscos seguidos (***), esa instrucción es " SI TO = 0 ===[ACTIVA EL REELEVADOR". El Arbol al que se hace referencia, es sobre la forma en que se encuentran las 16 Claves almacenadas dentro del programa en EPROM y como al efectuar la lectura del microswitch preguntando por cada uno de los Bits y el árbol podemos asignar la clave correcta, a continuación se muestra como esta estructurado el árbol :

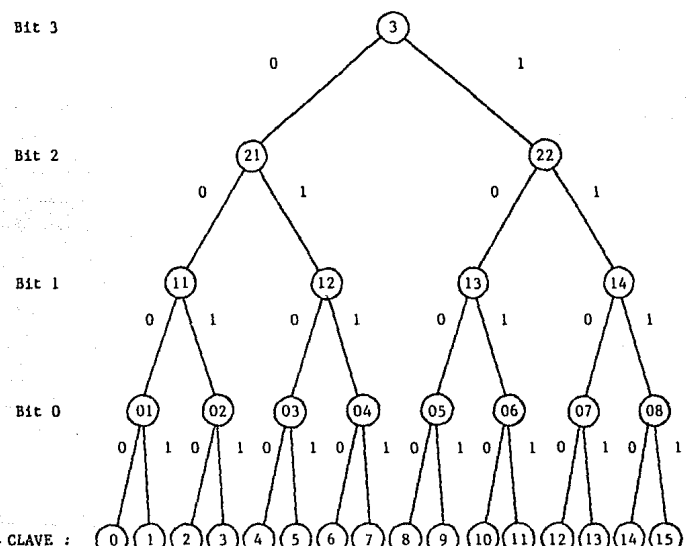


Figura 4.1 Arbol - Selección del No. de Clave

5. INSTALACION Y OPERACION

El conjunto de componentes que conforman el DIPAK y cad uno de sus módulos se encuentran agrupados para su instalación y operación en 4 unidades :

- Proceso
- Teclado
- Acción
- Remoto

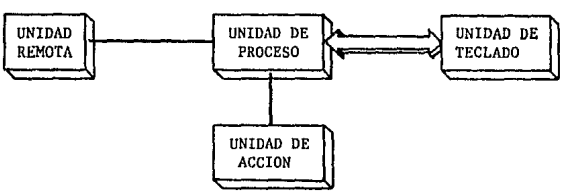


Figura 5.1 Unidades de Instalación y Operación

5.1 UNIDAD DE PROCESO

Está integrada por los módulos que intervienen en el procesamiento, sensado y control de las demás unidades, se conforma por aquellos dispositivos que se encuentran dentro del Circuito Impreso, a su vez este se instala dentro de un gabinete hecho a su medida (figura 5.2), donde se tiene un interruptor para el conector de alimentación (127 V AC) y los conectores externos.

5.2 UNIDAD DE TECLADO

Se compone por el Teclado y las Señales de Función (Bocina y Led), los cuales se instalan dentro de una cja a la medida (figura 5.3). La Comunicación de la Unidad de Teclado con la Unidad de Proceso se hace a través de :

- Un conector de cable plano conectado en el Circuito Impreso.
- Un Cable Plano entre el Circuito Impreso y un Conector DB-15 hembra empotrado en el gabinete de la Unidad de Proceso.
- Un Conector DB-15 macho con Concha, con un cable de 12 alambres y al otro extremo un conector para el teclado instalado dentro de la Caja de la Unidad de Teclado, comunicando la Unidad de Proceso con la Unidad de Teclado.
- La caja de la Unidad de Teclado contendrá el Teclado, la Bocina y el Led

En la Figura 5.4 se muestra como conectar cada uno de los alambres en los conectores para comunicar la Unidad de Proceso con la Unidad de Teclado.

5.3 UNIDAD DE ACCION

Constituida por la Contra-Eléctrica, se aprovechó su existencia en el Mercado para integrarla dentro del DIPAK y es propiamente el dispositivo que responde como salida que la Unidad de Proceso da de acuerdo a la entrada que se haya presentado, en la Figura 5.5. se muestra.

Se puede aprovechar la misma señal que manda la Unidad de Proceso a la Unidad de Acción para conectar otro dispositivo en vez de la Contra-Eléctrica, como puede ser activar un motor que accione una puerta corrediza, plegadiza, etc.

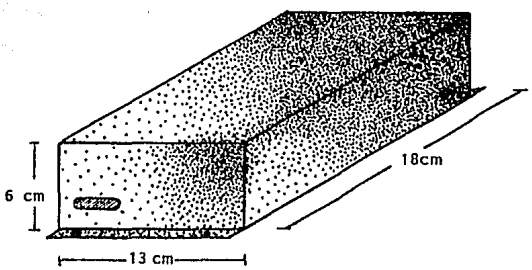


Figura 5.2 Gabinete - Unidad de Proceso

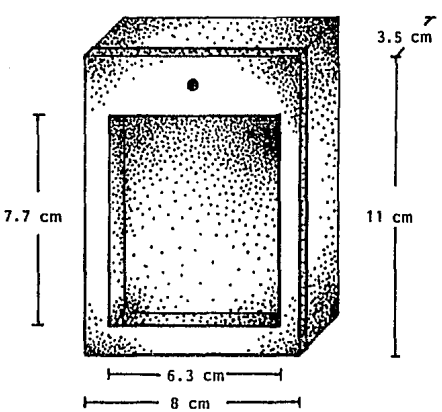


Figura 5.3 Caja de la Unidad de Teclado

5.4 UNIDAD REMOTA

Consiste en uno o varios botones Push-Botton conectados en paralelo a la Unidad de Proceso, permitiendo de esta manera que desde diferentes lugares se pueda activar la Contra-Eléctrica.

5.5. MONTAJE

Se requiere integrar los componentes del Circuito Impreso, el gabinete que lo contendrá requiere perforaciones para los conectores y las conexiones con los dispositivos externos, conformando así la Unidad de Proceso.

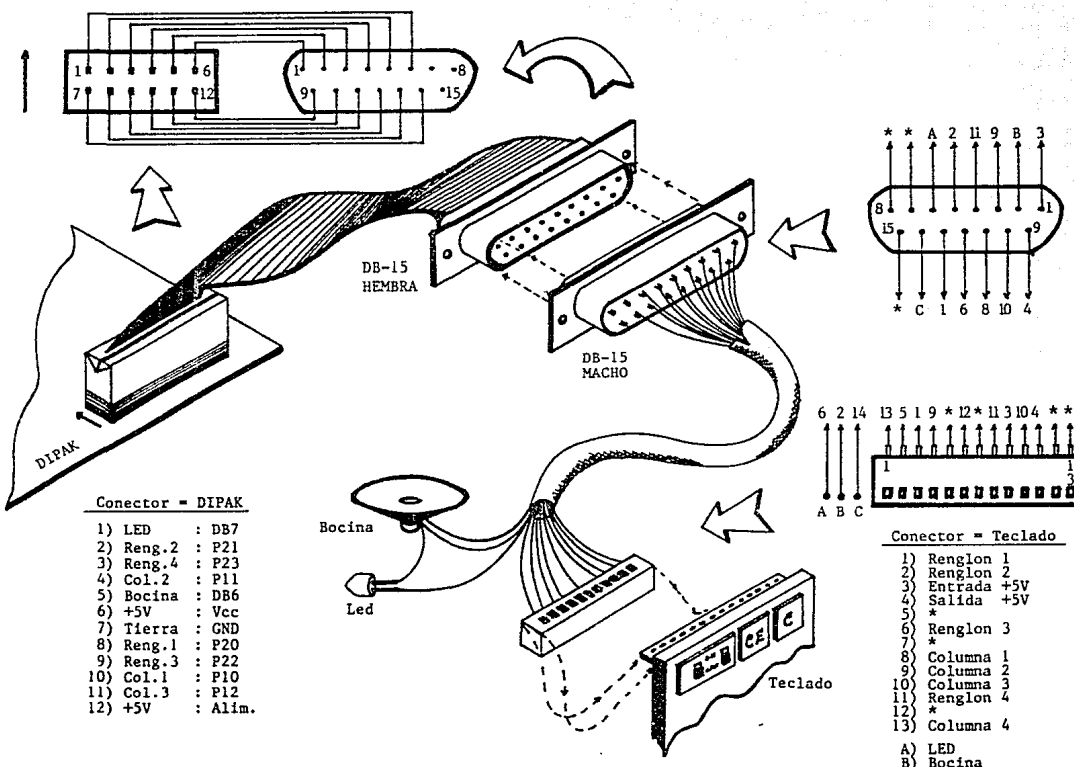
La Unidad de Proceso deberá situerse en el Interior del lugar de acceso, mientras que la Unidad de Teclado deberá estar fuera, se tiene comunicación a través del cable telefónico conectandolo como indica la figura 5.4.

La Unidad de Acción constituida por la Contra-Eléctrica se colocará en la Contra-Chapa de la puerta de acceso, el cable de comunicación deberá estar oculto hasta llegar a la Unidad de Proceso.

Para la Unidad Remota se colocarán en lugares estratégicos dentro del Local, para poder activar la Contra-Eléctrica sin necesidad de ir a la puerta y abrir.

La disposición delos componentes, así como el de las conexiones permiten una rápida instalación, teniendo así una configuración sólida, flexible y confiable.

La operación del DIPAK una vez instalado es muy sencilla, de acuerdo al número de clave seleccionada tenemos una secuencia de caracteres programada dentro del microcontrolador, el usuario la deberá teclear, y si la secuencia es correcta la Contra-Eléctrica se activará, si es incorrecta le impedirá el acceso. Otra forma de activar la Contra-Eléctrica, es por el Control Remoto, permitiendo el acceso a cualquier persona activando la Contra desde el interior del recinto.



Conector = DIPAK

1)	LED	: DB7
2)	Reng.2	: P21
3)	Reng.4	: P23
4)	Col.2	: P11
5)	Bocina	: DB6
6)	+5V	: Vcc
7)	Tierra	: GND
8)	Reng.1	: P20
9)	Reng.3	: P22
10)	Col.1	: P10
11)	Col.3	: P12
12)	+5V	: Alim.

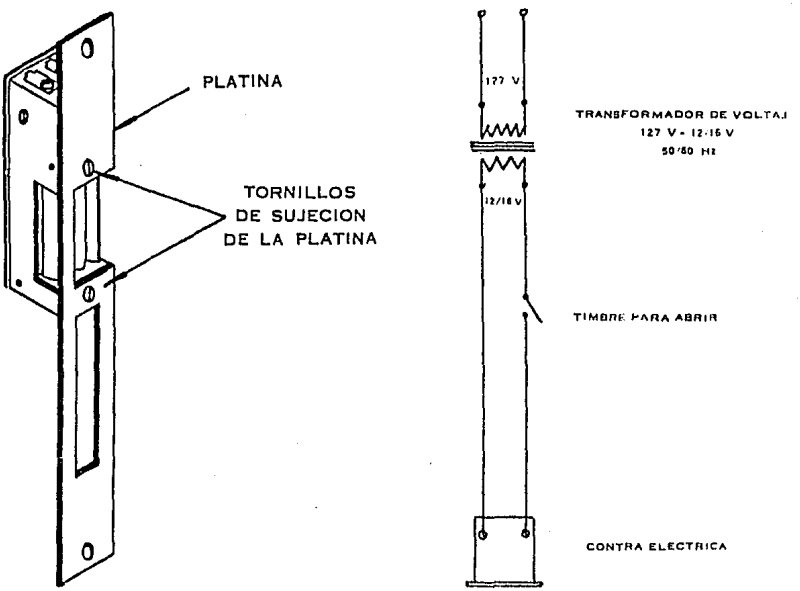
Conector = Teclado

1)	Renglon 1
2)	Renglon 2
3)	Entrada +5V
4)	Salida +5V
5)	*
6)	Renglon 3
7)	*
8)	Columna 1
9)	Columna 2
10)	Columna 3
11)	Renglon 4
12)	*
13)	Columna 4

A) LED
B) Bocina
C) GND

Figura 5.4 Conectores Externos del DIPAK

CONTRA ELECTRICA Lock MOD. 77 SIC-DGE 8482-77



INSTRUCCIONES PARA INSTALACION Y CONEXION

A.—Antes de instalar la contra vea que su posición corresponda al sentido de apertura de la puerta y a la instalación de la cerradura.

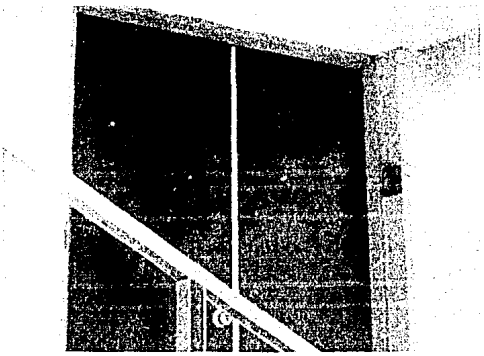
B.—La platina puede voltearse en cualquier posición, soltando los tornillos de sujeción.

C.—La localización de la contra reversible Lock y su ajuste con la cerradura deben quedar de manera que los pestillos no forcen, ni roce la puerta.

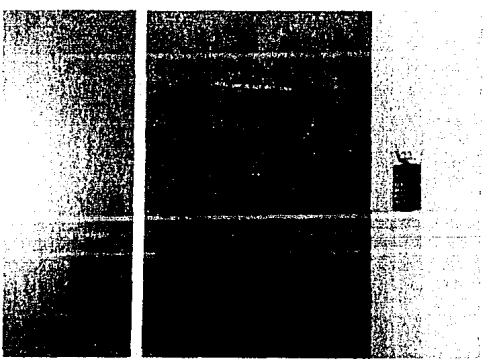
D.—Conéctese la contra como está indicado en el diagrama. Puede utilizar el transformador instalado para el timbre, si tiene el voltaje indicado en el diagrama.

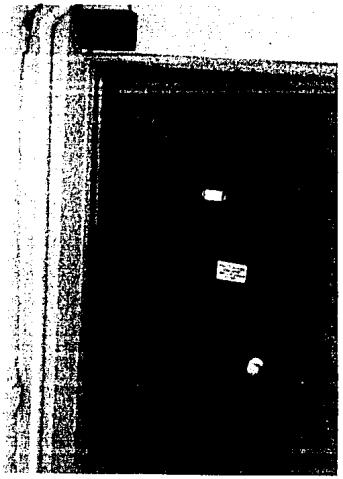
Figura 5.5. Contra-Eléctrica de la Unidad de Acción

A continuación se mostrarán algunas fotografías tomadas en el Laboratorio de Automatización del Instituto de Ingeniería, que darán una idea mas clara de la forma en que se puede implantar un sistema de este tipo.

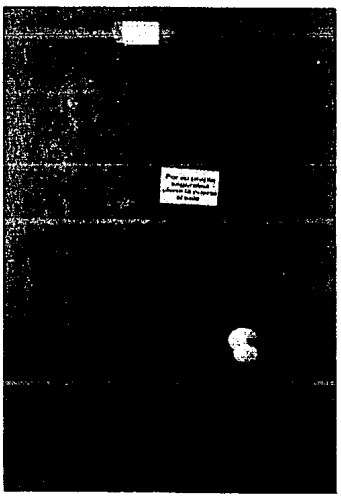


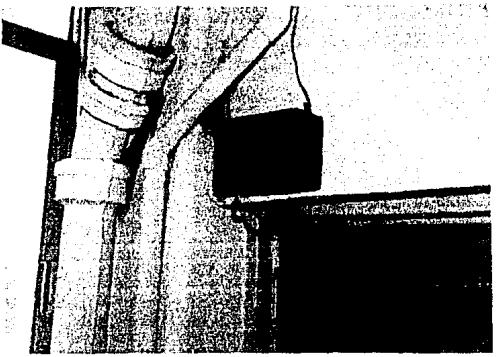
Entrada al Laboratorio de Automatización



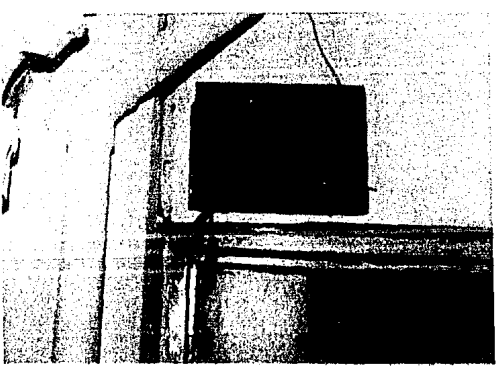


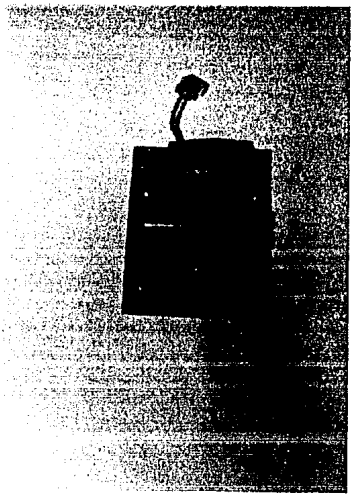
Vista interna a
la puerta de
Acceso



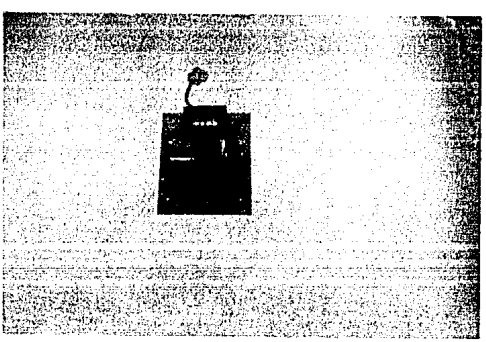


Unidad de Proceso





Circuito Impreso



6. CONCLUSIONES

El Dipak es un dispositivo que soluciona el Control de Acceso a zonas restringidas, donde muchas personas tienen acceso, pero no a todas se les puede proporcionar una llave para entrar. El primer prototipo del Dipak se instaló en Abril de 1987 en el Laboratorio de Automatización del Instituto de Ingeniería y a la fecha a funcionado sin presentar problemas cerca de 1 año en forma continua, habiendose probado los siguientes puntos :

Confiabilidad : El Sistema no ha fallado a pesar de haber trabajado un promedio de 12 horas al día durante aproximadamente 1 año.

Robustez : La distribución de los componentes, el tipo de conexiones que se emplean, así como el montaje en los gabinetes, hacen que el DIPAK sea un dispositivo sólido y resistente.

Facilidad de Operación : Con el fin de poder supervisar y controlar su funcionamiento, el DIPAK cuenta con un Display de 7 Segmentos Interno que muestra el No. de Clave seleccionada, el valor de la Tecla pulsada e indica cuando el dígito no corresponde a la secuencia de la Clave, además indica al usuario externo que el DIPAK se encuentra funcionando por medio de un Led que parpadea y activa una bocina al momento de pulsar alguna tecla.

Facilidad de Instalación : Debido a que el DIPAK se integra por 4 unidades separables, es fácil de instalar en función de las características del lugar.

Dentro de la Universidad, existen muchas Dependencias en donde el Dipak podría ser instalado y tener de esta manera un control efectivo de acceso a sus instalaciones, además previendo el enfoque comercial, se desarrolló el circuito impreso, para posibilitar su producción en serie, creándo de esta manera un dispositivo robusto, práctico y confiable.

Para su comercialización se estima el precio de venta en dólares sin incluir la instalación, de la siguiente manera :

o Componentes Electrónicas :

- Microcontrolador	16
- Fuente de Poder	4
- Despliegue	3
- Reelevador	4
- Otros	3

o Componentes Mecánicos :

- Circuito Impreso	4
- Contra-Eléctrica	12
- Teclado	4
- Bocina	3
- Chasis C.I.	6
- Caja Teclado	4
- Conectores	6
- Otros	3

Costo Componentes Eléctricos	30
Costo Componentes Mecánicos	42
Mano de Obra	20
Total Costo Directo (CD)	92
Costos Indirectos (CI = 50% CD)	46
COSTO DE PRODUCCION (CP) = = = [	138 U.S.
Utilidad (25 % CP)	34.5
P R E C I O D E V E N T A	[[[$ 172.50 U.S.]]]

El Tiempo de Construcción es de aproximadamente 20 horas-hombre. La arquitectura del Dipak permite desarrollos de versiones mejoradas, por ejemplo en lugar de activar una Contra-Eléctrica, activar un motor que abra automáticamente la entrada, colocar un control remoto que opere a través de algún tipo de emisor (ej. frecuencia o luz inflaroja), cambiar el teclado por tarjetas magnéticas, etc. dependiendo también de la aplicación que se le vaya a dar.

Como podemos observar todavía se puede hacer mucho más, empleando la tecnología para mejorar e incrementar la calidad de un primer diseño, proporcionando elementos que resuelvan las necesidades de una comunidad de manera práctica y confiable, es precisamente así como el Ingeniero alcanza un nivel de excelencia en su profesión.

REFERENCIAS

____ M C S 48 USER'S MANUAL
INTEL , 1978

____ MICROCONTROLADOR PAT - 86
ING. JUAN B. MARTINEZ, PROYECTO 3119
INSTITUTO DE INGENIERIA UNAM , 1986

____ MODULOS DE EXPANSION DEL SISTEMA PAT - 86
ING. JUAN B. MARTINEZ , PROYECTO 4126
INSTITUTO DE INGENIERIA UNAM , 1986

____ INTEL MICROSYSTEMAS COMPONENTS HANDBOOK
INTEL CORP. INC., 1985

____ THE TTL HANDBOOK
TEXAS INSTRUMENTS INC. , 1981

____ LOGICA DIGITAL Y DISEÑO DE COMPUTADORAS
MORRIS MANO . PRENTICE - HALL , 1982

APENDICE A

SET DE INSTRUCCIONES DEL 8748

MCS-48™ INSTRUCTION SET

SYMBOLS AND ABBREVIATIONS USED

A	Accumulator
AC	Auxiliary Carry
addr	12-Bit Program Memory Address
Bb	Bit Designator (b=0-7)
BS	Bank Switch
BUS	BUS Port
C	Carry
CLK	Clock
CNT	Event Counter
D	Mnemonic for 4-Bit Digit (Nibble)
data	8-Bit Number or Expression
DBF	Memory Bank Flip-Flop
F0, F1	Flag 0, Flag 1
I	Interrupt
P	Mnemonic for "in-page" Operation
PC	Program Counter
Pp	Port Designator (p=1, 2 or 4-7)
PSW	Program Status Word
Rr	Register Designator (r=0, 1 or 0-7)
SP	Stack Pointer
T	Timer
TF	Timer Flag
T0, T1	Test 0, Test 1
X	Mnemonic for External RAM
#	Immediate Data Prefix
@	Indirect Address Prefix
$	Current Value of Program Counter
(X)	Contents of X
((X))	Contents of Location Addressed by X
—	Is Replaced by

8048/8049
INSTRUCTION SET SUMMARY

	Mnemonic	Description	Bytes	Cycles
Accumulator	ADD A, R	Add register to A	1	1
	ADD A, @R	Add data memory to A	1	1
	ADD A, #data	Add immediate to A	2	2
	ADDC A, R	Add register with carry	1	1
	ADDC A, @R	Add data memory with carry	1	1
	ADDC A, #data	Add immediate with carry	2	2
	ANL A, R	And register to A	1	1
	ANL A, @R	And data memory to A	1	1
	ANL A, #data	And immediate to A	2	2
	ORL A, R	Or register to A	1	1
	ORL A, @R	Or data memory to A	1	1
	ORL A, #data	Or immediate to A	2	2
	XRL A, R	Exclusive Or register to A	1	1
	XRL A, @R	Exclusive or data memory to A	1	1
	XRL A, #data	Exclusive or immediate to A	2	2
	INC A	Increment A	1	1
	DEC A	Decrement A	1	1
	CLR A	Clear A	1	1
	CPL A	Complement A	1	1
	DA A	Decimal Adjust A	1	1
	SWAP A	Swap nibbles of A	1	1
	RL A	Rotate A left	1	1
	RLC A	Rotate A left through carry	1	1
	RR A	Rotate A right	1	1
	RRC A	Rotate A right through carry	1	1
Input/Output	IN A, P	Input port to A	1	2
	OUTL P, A	Output A to port	1	2
	ANL P, #data	And immediate to port	2	2
	ORL P, #data	Or immediate to port	2	2
	INS A, BUS	Input BUS to A	1	2
	OUTL BUS, A	Output A to BUS	1	2
	ANL BUS, #data	And immediate to BUS	2	2
	ORL BUS, #data	Or immediate to BUS	2	2
	MOVD A, P	Input Expander port to A	1	2
	MOVD P, A	Output A to Expander port	1	2
	ANLD P, A	And A to Expander port	1	2
	ORLD P, A	Or A to Expander port	1	2
Registers	INC R	Increment register	1	1
	INC @R	Increment data memory	1	1
	DEC R	Decrement register	1	1
Branch	JMP addr	Jump unconditional	2	2
	JMPP @A	Jump indirect	1	2
	DJNZ R, addr	Decrement register and skip	2	2
	JC addr	Jump on Carry = 1	2	2
	JNC addr	Jump on Carry = 0	2	2
	JZ addr	Jump on A Zero	2	2
	JNZ addr	Jump on A not Zero	2	2
	JT0 addr	Jump on T0 = 1	2	2
	JNT0 addr	Jump on T0 = 0	2	2
	JT1 addr	Jump on T1 = 1	2	2
	JNT1 addr	Jump on T1 = 0	2	2
	JF0 addr	Jump on F0 = 1	2	2
	JF1 addr	Jump on F1 = 1	2	2
	JTF addr	Jump on timer flag	2	2
	JNI addr	Jump on INT = 0	2	2
	JBb addr	Jump on Accumulator Bit	2	2

	Mnemonic	Description	Bytes	Cycles
Subroutine	CALL	Jump to subroutine	2	2
	RET	Return	1	2
	RETR	Return and restore status	1	2
Flags	CLR C	Clear Carry	1	1
	CPL C	Complement Carry	1	1
	CLR F0	Clear Flag 0	1	1
	CPL F0	Complement Flag 0	1	1
	CLR F1	Clear Flag 1	1	1
	CPL F1	Complement Flag 1	1	1
Data Moves	MOV A, R	Move register to A	1	1
	MOV A, @R	Move data memory to A	1	1
	MOV A, #data	Move immediate to A	2	2
	MOV R, A	Move A to register	1	1
	MOV @R, A	Move A to data memory	1	1
	MOV R, #data	Move immediate to register	2	2
	MOV @R, #data	Move immediate to data memory	2	2
	MOV A, PSW	Move PSW to A	1	1
	MOV PSW, A	Move A to PSW	1	1
	XCH A, R	Exchange A and register	1	1
	XCH A, @R	Exchange A and data memory	1	1
	XCHD A, @R	Exchange nibble of A and register	1	1
	MOVX A, @R	Move external data memory to A	1	2
	MOVX @R, A	Move A to external data memory	1	2
	MOVP A, @A	Move to A from current page	1	2
	MOVP3 A, @A	Move to A from Page 3	1	2
Timer/Counter	MOV A, T	Read Timer/Counter	1	1
	MOV T, A	Load Timer/Counter	1	1
	STRT T	Start Timer	1	1
	STRT CNT	Start Counter	1	1
	STOP TCNT	Stop Timer/Counter	1	1
	EN TCNTI	Enable Timer/Counter Interrupt	1	1
	DIS TCNTI	Disable Timer/Counter Interrupt	1	1
Control	EN I	Enable external interrupt	1	1
	DIS I	Disable external interrupt	1	1
	SEL RB0	Select register bank 0	1	1
	SEL RB1	Select register bank 1	1	1
	SEL MB0	Select memory bank 0	1	1
	SEL MB1	Select memory bank 1	1	1
	ENTO CLK	Enable Clock output on T0	1	1
	NOP	No Operation	1	1

APENDICE B

PAT 86 = PROGRAMADOR EXOR 48

EXOR-48, PROGRAMADOR DEL MICROCONTROLADOR 8748.

INTRODUCCION

Se describe un circuito que, conectado al sistema PAT86, permite programar los circuitos 8748 y 8748H.

El módulo EXOR-48 se conecta, a través de una sección de cable plano, con el sistema PAT86, el cual a su vez se comunica con el usuario a través de una terminal de video RS232.

DESCRIPCION GENERAL:

El módulo EXOR-48 realiza su acoplamiento con el sistema PAT86, a través del circuito 8255. En la fig. 1 se muestra el diagrama general de este acoplamiento. Las señales en las patas PC1 y PC5 del 8255 se reemplazaron modificando las conexiones en el sistema PAT-86 por la fuente de 12 y un reloj de 2MHz respectivamente, que se necesitaban para la operación del programador.

En el módulo EXOR-48 (ver fig. 2) se puede distinguir un circuito doblador de voltaje desarrollado a través de un temporizador LM555 y una red de diodos y capacitores. Su funcionamiento es el siguiente:

El 555 está alambrado como un astable que oscila a una frecuencia de aproximadamente 20KHz. El par de transistores BC547-BC328 dan a la salida una gran capacidad de suministro de corriente. La red de diodos y capacitores generan un voltaje de 36 volts en su última etapa. Este voltaje alimenta a 3 reguladores de diodos Zener: 2 de 24 volts y 1 de 27 volts (los dos de 20^V y uno de 22^V, en el caso de programar el 8748H).

Los 3 reguladores descritos se conectan a la base de otros tantos transistores NPN BC547 que completan el circuito de regulación. Los 3 diodos 1N914 conectados al emisor de cada transistor permiten disminuir el voltaje en 0.7^V de tal forma de tener aproximadamente 23^V en las patas 25 y 7 del 8748 y de 26^V en la pata 26 (ó 19^V y 21^V para el 8748H).

El circuito 74LS377 permite almacenar en sus flip flops los datos enviados al módulo. Las salidas en los pines 6, 9 y 12 se conectan a inversores 7406 de colector abierto, las cuales controlan el voltaje de los pi

nes 7, 25 y 26 del 8748: si la salida del inversor está activa (0^V), la base de los transistores se va a 0^V y por tanto el voltaje de salida en el emisor queda a 0^V (pin 25) ó 5^V (pines 7 y 26).

Si, por otro lado, la salida de los inversores es inactiva (desconectado), los reguladores funcionan normalmente y generan sus voltajes altos en lo pines correspondientes.

En la fig. 3 se muestra el LAY-OUT del circuito. Los zeners se encuentran conectados a una base de 24 pines. Si se invierte la posición de esta base se conectan al circuito ya sea los zeners para programar el 8748 ó el 8748H.

En seguida se describe el procedimiento de programación del chip. Al final del reporte se anexa la especificación técnica del circuito, en donde se detallan los diagramas de tiempos, voltajes y corrientes que deben generarse para satisfacer los requerimientos de programación.

CAMBIOS AL MODULO PAT-86, PARA SU ACOPLAMIENTO CON LA TARJETA EXOR-48

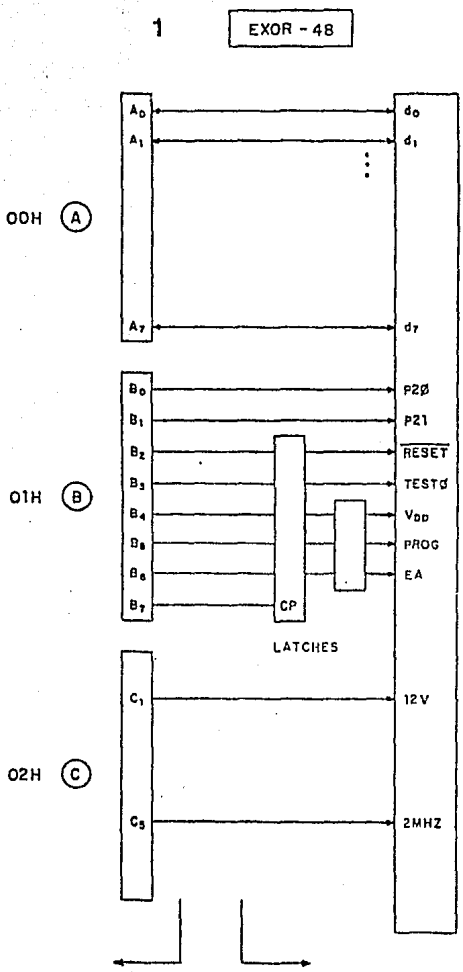
1 Seccionar la pista que conecta la pata 15 del 8255 con la pata 7 del conector J2.

2. Seccionar la pista que conecta la pata 18 del 8255 con la pata 19 del conector J2.

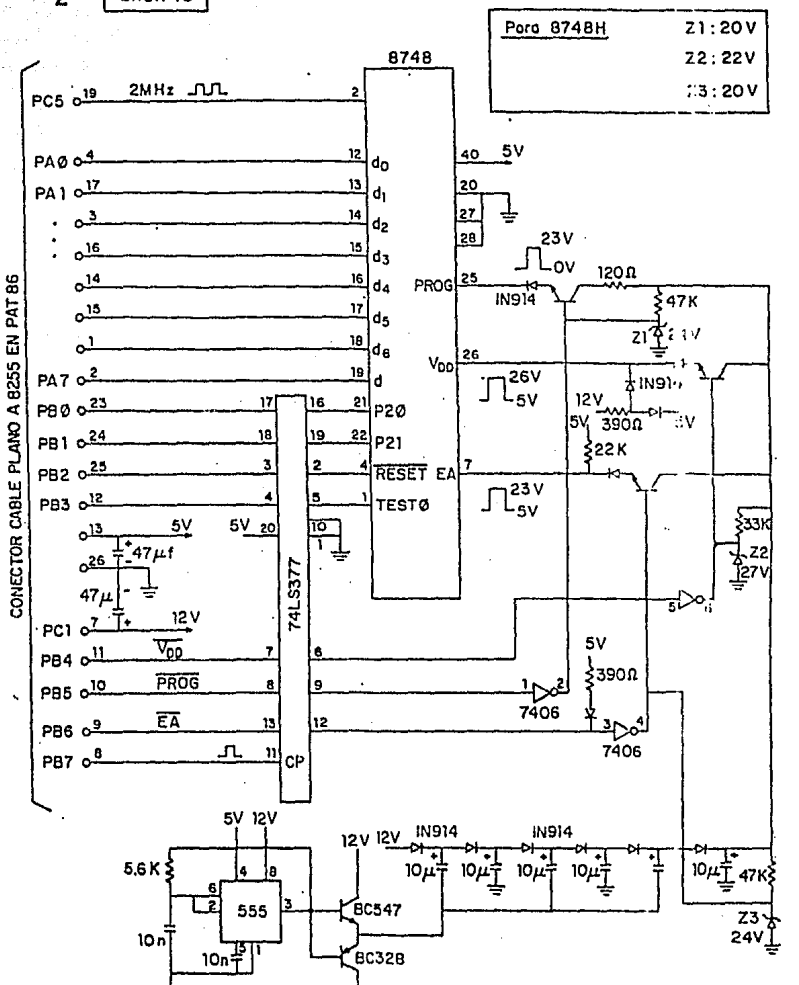
3. Conectar, con alambre wire wrap, la pata 14 del circuito MC1488 ($+12^V$), con la pata 7 del conector J2.

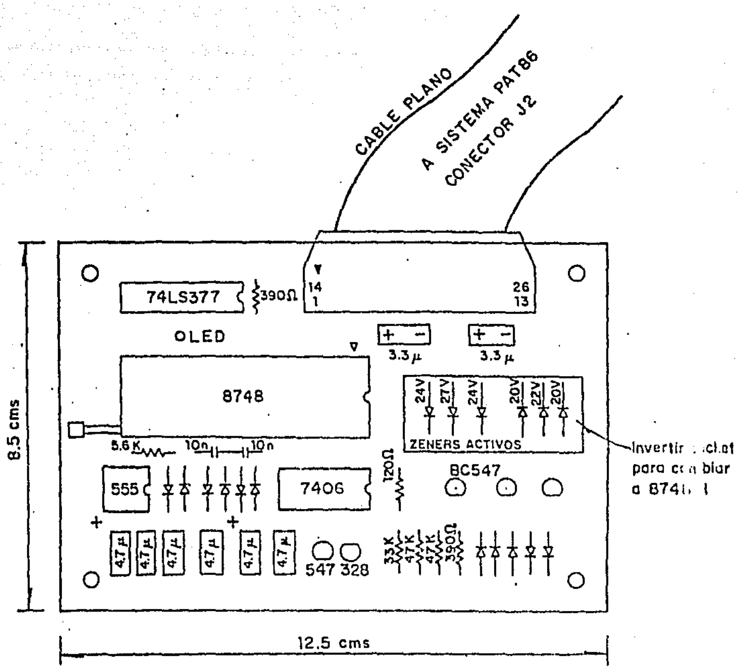
4. Conectar la pata 20 del circuito 8251 (2MHz) con la pata 19 del conector J2.

NOTA: el cable plano que conecta la tarjeta PAT-86 con el módulo EXOR48, no debe exceder una longitud de 40 cms.


1
EXOR - 48
00H
A
A0
A1
A7
d0
d1
d7
01H
B
B0
B1
B2
B3
B4
B5
B6
B7
CP
P2Ø
P21
RESET
TESTØ
VDD
PROG
EA
LATCHES
02H
C
C1
C5
12V
2MHZ
8255
MODULO EXOR48

2 EXOR 48


Para 8748H
Z1: 20V
Z2: 22V
Z3: 20V
8748
CONECTOR CABLE PLANO A 8255 EN PAT 86
PC5
2MHz
PA0
PA1
PA7
PB0
PB1
PB2
PB3
PC1
PB4
PB5
PB6
PB7
74LS377
CP
d0
d1
d2
d3
d4
d5
d6
d7
P20
P21
RESET
TEST0
EA
PROG
VDD
5V
12V
23V
0V
26V
120Ω
47K
IN914
Z1
22K
390Ω
33K
Z2
27V
47µf
7406
555
BC547
BC328
5.6K
10n
10µ
Z3
24V



3 LAY OUT EXOR-48

EXOR 48

PROGRAMADOR DE LA MICROCOMPUTADORA INTEL 8748

El programador para la microcomputadora en un chip 8748 esta diseñado alrededor de la microcomputadora PAT-86 especificamente modificada para extender sus capacidades como controlador a las funciones de programador.

La microcomputadora PAT-86 esta basada en el microprocesador Z80 y cuenta con un máximo de 8 Kbytes en ROM y un máximo de 8 Kbytes en RAM, un puerto serie trabajando a 300 bps para conectarla a una terminal y poder dar comandos, y un puerto paralelo con 24 lineas disponibles. En la configuración del PROGRAMADOR, cuenta con 4 Kbytes de ROM y 2 Kbytes de RAM, de los cuales, las primeras 256 localidades son empleadas por el sistema supervisor, quedando disponibles 1.8 Kbytes para el usuario comenzando desde la localidad 2100H hasta la localidad 27FFH inclusive.

Este sistema supervisor provee de tres funciones necesarias para el programador, despliegue de la memoria de la PAT-86, modificaciones a la memoria de la PAT-86 y ejecucion de las rutinas de acceso al 8748; estos comandos se detallan a continuación:

DM <dir_inicial> <dir_final>

Despliega los contenidos de la memoria de la PAT-86 entre las localidades <dir_inicial> y <dir_final> en hexadecimal.

SM <dir_inicial>

Despliega el contenido de la localidad <dir_inicial> y espera recibir una de tres cosas; un espacio, con lo que incrementa la direccion sin alterar el contenido; la tecla [RETURN], que termina el comando y regresa al modo supervisor; o bien la información que reemplazara a la existente en la localidad, seguida esta de, o bien un espacio para revisar y posiblemente modificar la siguiente localidad, o bien de la tecla [RETURN] para terminar el comando y regresar al modo supervisor.

G <direccion>

Comienza la ejecución de un programa escrito para el Z80 a partir de la localidad <dirección>. En la configuración del programador las rutinas de acceso al 8748 inician en 1000H.

Las rutinas de acceso al 8748 proveen de tres funciones a su vez, permiten verificar si la memoria del 8748 esta borrada, programar cualquier region del 8748 con verificación inmediata de las localidades que no se pudieron programar correctamente, y la lectura de cualquier región de la memoria del 8748, vaciando su contenido a la memoria de la PAT-86. El simbolo '>' indica que se encuentran activas las rutinas de acceso al 8748 y esta esperando un comando. La descripción de los comandos de las rutinas de acceso al 8748 se detalla a continuación:

V

Oprimiendo la tecla [V] se inicia el proceso de verificación, reportando por cada 64 bytes, un '*' si está completamente borrada esa zona, o un '?' si existe al menos una localidad no borrada en la zona, en total verifica 16 zonas, ya que el 8748 tiene 1024 bytes.

P <dir_ini_8748> <dir_fin_8748> <dir_ini_RAM>

El comando P pide tres parámetros, la dirección inicial en la memoria del 8748 que se desea programar, la dirección final en la memoria del 8748 que se desea programar y la direccion inicial en la RAM de la PAT-86 de donde se tomaran los datos a grabar en la memoria del 8748. Si al verificar detecta alguna localidad que no graba correctamente, reporta la dirección del dato en la RAM de la PAT-86, el dato que quería grabar, y el dato que se grabó realmente en el 8748.

P <dir_ini_8748> S<num_bytes> <dir_ini_RAM>

Esta forma alterna del comando P permite comunicar directamente el número de bytes que se desean grabar en la memoria del 8748. La letra S debe ser dada en mayúsculas, al igual que todos los comandos.

L <dir_ini_8748> <dir_fin_8748> <dir_ini_RAM>

El comando L permite la lectura de la memoria del 8748 entre las direcciones <dir_ini_8748> y <dir_fin_8748> inclusive, y las vacia en la memoria de la PAT-86 comenzando en la localidad <dir_ini_RAM>.

L <dir_ini_8748> S<num_bytes> <dir_ini_RAM>

Esta es la forma alterna del comando L que permite indicar el número de bytes que se van a leer en vez de indicar una localidad final de lectura.

E

Al oprimir la tecla [E] terminan las rutinas de acceso al 8748 y se regresa al sistema supervisor cuyo aviso de espera de comandos es ':'

Para programar un 8748, primero se debe encender la PAT-86, conectar la tarjeta de expansión para el PROGRAMADOR teniendo cuidado en la alineación de las flechas de referencia marcada en los conectores, escribir los datos en la memoria de la PAT-86,e iniciar las rutinas de acceso al 8748, INSERTAR el 8748 solo hasta este momento y con mucho cuidado, programar la zona del 8748 que se quería quemar y retirarlo del socket de programación con mucho cuidado. Si el LED piloto esta encendido o si la tarjeta de expansión no esta conectada, no se debe retirar o insertar un 8748 en el socket de programación.

Para leer o verificar un 8748, se siguen los mismos pasos que para programarlo excepto de que no se necesitan cargar datos a la memoria de la PAT-86.

Por último es necesario indicar que el PROGRAMADOR puede no solo programar la versión 8748H sino también la versión 8748 sencilla, cambiando únicamente los zeners de programación

TRANSFER.ASM

```
; ESTA SUBRUTINA SE ENCUENTRA ALMACENADA EN LA DIRECCION 700, EN EL SISTEMA
; PAT86.SU EMPLEO PERMITE LA TRANSFERENCIA DE ARCHIVOS DIRECTAMENTE DE UN
; SISTEMA TIPO PC COMPATIBLE, HACIA EL PAT 86.


; Subrutina nivel usuario para tranferir
; Intel.hex files de disco a memoria del
; sistema prototipo.


        ORG     2000H
        .Z80


; Uso de registros :

;                       A : Scratch
;                       B : Contador de data bytes
;                           por linea
;                       C : Scratch
;                       HL: Apuntador de memoria destino


; Uso de subrutinas monitor :

;                       0460 : Lee 2 car. ascii y
;                              regresa hex en A.


; Parametros :

8251D   EQU     10H
8251C   EQU     11H


; Programa :

        PUSH    B
        PUSH    D
        PUSH    H
        PUSH    PSW
        PUSH    X
        PUSH    Y


LOOP1:  IN      11H     ; Lee status 8251
        ANI     02H     ; RxRDY ?
        JZ      LOOP1   ; No regresa a leer status
        IN      10H     ; Si Lee caracter
        MOV     C,A
        CMP     3       ; Es "
        JZ      LOOP2   ; Si S brinca a LOOP2
        MOV     A,C     ; De otro modo lee el caracter
```

```
LOOP2   CALL    0460H   ; Lectura de caracteres control
                        ; de linea.
        MOV     E,A
        CALL    0460H
        MOV     H,A
        CALL    0460H
        MOV     L,A
        CALL    0460H
        MVI     C,00H
        CMP     C
        JZ      LOOP3   ; Determina si el byte separador
                        ; es tipo "00"
                        ; Si: lee los data bytes; LOOP3


        POP     B
        POP     D
        POP     H
        POP     PSW
        POP     X
        POP     Y
        RST     0       ; No: entonces es tipo "01" y
                        ; termina transfer.
                        ; regresa a monitor


LOOP3   CALL    0460H   ; Loop de lect data bytes linea
        MOV     M,A     ; Carga A en mem apuntada por HL
        INX     H       ; Incr. HL
        DCR     B       ; Decr. B
        MVI     A,00H
        CMP     B       ; B=00H ?
        JZ      LOOP1   ; Si: regresa a buscar sig.
                        ; inicio de linea
        JMP     LOOP3   ; No: sigue leyendo data bytes
                        ; de linea


        END
```

```
                ; ESTA SUBRUTINA SE ENCUENTRA ALMACENADA EN LA DIRECCION 700, EN EL SISTEMA
                ; PAT86.SU EMPLEO PERMITE LA TRANSFERENCIA DE ARCHIVOS DIRECTAMENTE DE UN
                ; SISTEMA TIPO PC COMPATIBLE, HACIA EL PAT 86.


                ; Subrutina nivel usuario para tranferir
                ; Intel.hex files de disco a memoria del
                ; sistema prototipo.


2000                    ORG     2000H
                        .Z80


                ; Uso de registros :

                ;                       A : Scratch
                ;                       B : Contador de data bytes
                ;                           por linea
                ;                       C : Scratch
                ;                       HL: Apuntador de memoria destino


                ; Uso de subrutinas monitor :


                ;                       0460 : Lee 2 car. ascii y
                ;                              regresa hex en A.


                ; Parametros :


0010            8251D   EQU     10H
0011            8251C   EQU     11H


                ; Programa :


2000 C5                 PUSH    B
2001 D5                 PUSH    D
2002 E5                 PUSH    H
2003 F5                 PUSH    PSW
2004 DDE5               PUSH    IX
```

```
2006 FDE5            PUSH   Y


2008 DB11    LOOP1   IN     11H     ; Lee status 8251
200A E602            ANI    02H     ; RxRDY ?
200C CA0820          JZ     LOOP1   ; Not regresa a leer status
200F DB10            IN     10H     ; Si: Lee caracter
2011 063A            MVI    B,3AH
2013 B8              CMP    B       ; Es ":"?
2014 CA1A20          JZ     LOOP2   ; Si: brinca a LOOP2
2017 C30820          JMP    LOOP1   ; Not sigue leyendo caracteres



201A CD6004  LOOP2   CALL   0460H   ; Lectura de caracteres control
201D                                ; de linea.
201D 47              MOV    B,A
201E CD6004          CALL   0460H
2021 67              MOV    H,A
2022 CD6004          CALL   0460H
2025 6F              MOV    L,A
2026 CD6004          CALL   0460H
2029 0E00            MVI    C,00H
202B B9              CMP    C
202C CA3920          JZ     LOOP3   ; Determina si el byte separador
202F                                ; es tipo "00"
202F                                ; Si: lee los data bytes; LOOP3
202F 00              NOP

2030 C1              POP    B
2031 D1              POP    D
2032 E1              POP    H
2033 F1              POP    PSW
2034 DDE1            POP    X
2036 FDE1            POP    Y
2038 C7              RST    0       ; Not entonces es tipo "01" y
2039                                ; termina transfer.
2039                                ; regresa a monitor



2039 CD6004  LOOP3   CALL   0460H   ; Loop de lect data bytes linea
203C 77              MOV    M,A     ; Carga A en mem apuntada por HL
203D 23              INX    H       ; Incr. HL
203E 05              DCR    B       ; Decr. B
203F 3E00            MVI    A,00H
2041 B8              CMP    B       ; B=00H ?
2042 CA0820          JZ     LOOP1   ; Si: regresa a buscar sig.
2045                                ; inicio de linea
2045 C33920          JMP    LOOP3   ; Not sigue leyendo data bytes
2048                                ; de linea
```

```
0000                    END
```

---- SYMBOL TABLE ----

8251C	0011	8251D	0010	LOOP1	2008	LOOP2	201A	LOOP3	2039

***** NO ERRORS DETECTED *****

APENDICE C

SOPORTE PAT 86 - COMPUTADORA PERSONAL

1. RESUMEN.

El sistema PAT86 es un módulo microcontrolador desarrollado en el Instituto de Ingeniería. Entre sus aplicaciones se encuentran el control de procesos, la róbotica y las telecomunicaciones.

Por otro lado con la adquisición de computadoras del tipo PC, se abrieron nuevas posibilidades en cuanto a su uso como sistemas de soporte para el desarrollo de circuitos basados en microprocesadores.

Tal uso se describe en el presente reporte: como conectar el sistema PAT86 -basado en el microprocesador Z80- con una computadora PC y como utilizar los diferentes paquetes de apoyo disponibles que, junto con el sistema PAT86, conforman una poderosa infraestructura para el desarrollo de sistemas basados en los procesadores Z80 y 8748:

-El programa VTERM que convierte a la PC en una terminal de video sumamente flexible.
-El programa XASM85, que ensambla programas escritos para los procesadores 8080, 8085, Z80.
-El programa XASM48, que ensambla programas escritos para procesadores de la familia 48.
-El programa AVSIM48, que es un simulador para los procesadores de la familia 48.

Estos paquetes, usados junto con el sistema PAT86, permiten escribir, almacenar, probar y transferir programas para los procesadores Z80 y 8748. Además, es posible también usar, desde la PC, el programador de memorias 2716 y 2532, y el programador del chip 8748, ambos módulos de expansión del sistema PAT86.

En los siguientes capítulos se describen con todo detalle el empleo de cada una de estas posibilidades.

2. INTERCONEXION DEL SISTEMA CONTROLADOR PAT86 CON UNA COMPUTADORA PC.

Ambos sistemas se conectan a través de sus puertos serie. Sin embargo, no todos los sistemas PC tienen integrado el puerto serie a su hardware. Aquí se describirá la interfaz con 2 modelos de computadora PC fabricados por la PRINTAFORM: la Columbia y la Popular 500.

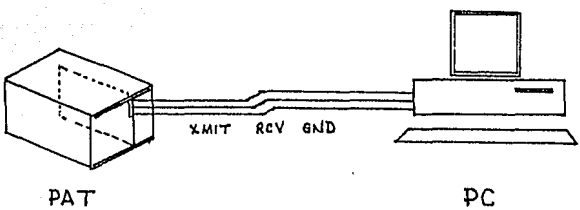
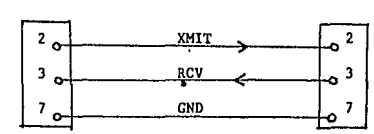


FIG. 1

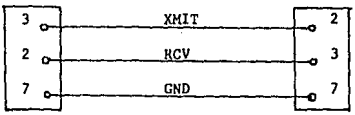
En la fig. 1 se muestra el acoplamiento: del puerto serie del PAT86 se conectan las señales XMIT, RCV, GND, hacia el conector del puerto serie de la computadora PC.

INTERFAZ PAT86-COLUMBIA



CONECTOR DB25 CONECTOR CABLE PLANO 26 PINES. (J1)

INTERFAZ PAT86 - POPULAR 500



POPULAR 500
CONECTOR DB9

PAT86
CONECTOR CABLE PLANO 26 PINES. (J1)

El chassis del sistema PAT86 se encuentra flotando con respecto a la alimentación de 127AC. Su tierra electrónica, se encuentra a su vez flotando con respecto al chasis y a la alimentación.

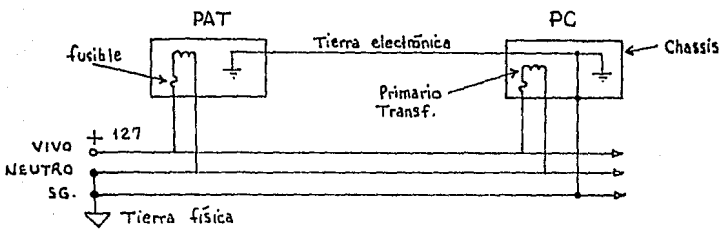


FIG. 2

En cambio, la computadora PC tiene por lo general la tierra electrónica conectada al chassis, y éste conectado a la tierra de seguridad de la alimentación, como se muestra en la fig. 2.

La interconexión así realizada, no presenta mucho riesgo. Sin embargo, se recomienda, como precaución adicional, DESCONECTAR el cable de tierra de seguridad (S.G.) que va del chasis de la PC a la tierra física de la instalación. La acción anterior evitaría daños en caso de un hipotético cortocircuito entre el vivo (+127 AC) y la tierra electrónica del sistema PAT86.

Por supuesto, la otra posibilidad que previene un accidente como el arriba descrito, sería el de conectar la tierra electrónica del PAT86 con su chassis, y de ahí a la tierra física de la instalación. Sin embargo, considerando el hecho que no siempre se tiene tierra física disponible en la alimentación, se aconseja por sencilla, la conexión descrita en el párrafo de arriba.

3. SOPORTE DE SOFTWARE PC.

Una vez realizada la conexión entre ambos sistemas, pueden usarse los programas PC compatibles, que permitirán emplear la computadora como un sistema de desarrollo de sistemas basados en microprocesadores.
Estos programas son:

3.1 VTERM

Este programa convierte a la PC en una terminal de video superversátil. En su aplicación más simple, el VTRM hace de la PC una terminal estándar con capacidad de trasmisión - recepción asíncrona. Sus rutinas permiten programar la velocidad de trasmisión - recepción en una amplia gama de valores, así como fijar los parámetros de la trasmisión asíncrona: número de stop bits, número de bits de información y paridad.

Entre otras posibilidades, el programa permite también recibir y almacenar archivos de texto en disco, ó leer y trasmitir por el puerto serie archivos previamente almacenados.

3.2 Ensamblador Z80.

Si se desea, puede editarse y ensamblarse archivos en la PC, para posteriormente trasmitirlos al PAT86, en donde son ejecutados y debugeados. Para realizar la operación anterior, se dispone de 3 programas:

- -El editor de texto Turbo-87 (utilizado para el lenguaje turbopascal), para teclear y editar los programas escritos en mnemónicos (lenguaje ensamblador Z80),
- -En ensamblador XASM85 que ensambla el programa descrito y produce como salida un archivo con los códigos de máquina del programa y,
- -El programa VTERM, que envía el archivo mencionado hacia el sistema PAT86.

Los programas anteriores pueden ser usados en combinación con el sistema PAT86 y su programador de EPROMS 2716 y 2532 (tarjeta "EXOR"). La información de este programador se da en el reporte "SISTEMA PAT86".

3.3 Ensamblador 8748.

El 8748 es una computadora en un solo chip. En el reporte "Módulos de expansión para el sistema PAT86", se describe el módulo programador EXOR48 que permite, desde el sistema PAT86 programar ó leer los chips 8748 u 8748H.

Usando este módulo y el sistema PAT86, junto con una PC, pueden teclearse en esta última los programas en lenguaje ensamblador para luego obtener el código de máquina a través del programa XASM48 y trasmitir el archivo correspondiente al PAT86, de donde posteriormente se programa el chip 8748 en el módulo EXOR48.

Para realizar la operación anterior, se dispone de 3 programas:

-Un editor de texto, TURBO-87
-El ensamblador XASM48
-El programa "VTERM" (ver punto 3.2)

3.4 Simulador 8748.

Para el procesador 8748, no se cuenta con un sistema de desarrollo desde donde se puedan ejecutar y corregir en RAM los programas antes de programarlos en EPROM, como es el caso del sistema PAT86 y su procesador Z80.

Por eso resulta sumamente útil el uso de un programa simulador en PC, por medio del cual se pueden ejecutar y corregir los programas escritos para el chip 8748, pudiendo observar paso a paso el contenido de sus registros y puertos. Estos programas pueden editarse y probarse previamente antes de trasmitirlos al sistema PAT86 y su módulo EXOR 48.

Instrucciones para el uso de los programas:

Para el uso de los programas anteriores, se requiere del siguiente material:

-Diskette con la etiqueta PAT-PC, que contiene, además del sistema operativo MSDOS, todos los programas descritos en el punto 3.

-Reporte "Sistema PAT86" con la descripción y uso de su programa monitor y del programador EXOR.

-Reporte "Módulos de expansión para el sistema PAT86" que describe el uso de los módulos EXOR, programador de memorias 2716 y 2532, y EXOR48, programador del chip 8748.

Además, si se quiere hacer uso completo de la capacidad de desarrollo del sistema PAT86, se requieren los módulos EXOR Y EXOR 48.

En seguida se describe, paso a paso, el empleo de los programas descritos en los puntos 3.1, 3.2, 3.3 y 3.4.

4. LA PC EMPLEADA COMO TERMINAL.

Introduzca el diskette con la etiqueta PAT-PC en el drive A de la PC y encienda su máquina. Automáticamente, el sistema operativo pregunta fecha y hora (en ambos casos puede teclearse solamente CR) y llama al programa VTERM. Por default, la velocidad es de 300 bauds, 2 stop bits, sin paridad. A partir de este momento la PC se comporta como terminal y debe comunicarse con el monitor del PAT86 "ONLINE". Pruebe algunos de los comandos del monitor y compruebe que funcionan (ver manual del SISTEMA PAT86).

Ahora pulse la tecla F5. Verá en pantalla las opciones del programa. Moviendo el cursor, puede posicionarse sobre la etiqueta que se desea cambiar. Ya en posición, con la tecla "+" se elige la nueva opción. Por ejemplo: ponga el cursor en la etiqueta "COMMUNICATIONS RATE". Pulse varias veces "+" y verá pasar todos los distintos valores posibles de BAUD RATE. Pulse de nuevo F5. El despliegue muestra el formato para trasmisión de archivos. En la etiqueta "Filename" debe teclearse el nombre del archivo a trasmitirse. Posteriormente se darán más detalles para el uso de esta opción.

Oprima F5. Ahora verá las opciones para usar el programa en conjunción con módems y efectuar llamadas y respuestas automáticas. Esta opción está sin embargo, fuera de los objetivos de este reporte.

Si teclea de nuevo F5, el VTERM vuelve al modo "ON LINE" y se comunica con el PAT86.

Si oprime ahora F6, verá información adicional sobre facilidades del programa. De la lista mostrada, aquí solo nos interesan dos:

ALT T: si VTERM está "on line", manda al archivo previamente seleccionado por su puerto serie.

SHIFT-SHIFT: es un switch toggle de VTERM hacia el Sistema Operativo y viceversa. Esta opción es de gran utilidad y permite pasar rápidamente de uno a otro programa.

Con "Escape" regrese a "on line" y oprima SHIFT-SHIFT. Compruebe que funciona. La opción "ALT T" se verá más adelante.

5. ENSAMBLADOR Z80

En esta sección se ilustrarán los pasos a seguir para:

a) Escribir y ensamblar un programa Z80.
b) Transferir el código de máquina al PAT86.

Dentro del MS-DOS del diskette PAT-PC, observe el directorio:

VTERM	emula la PC como terminal
XASM48	ensamblador para procesadores de la familia 48
AVSIM48	simulador para procesadores de la familia 48
TURBO-87	editor de texto para escribir los programas
XASM85	ensamblador para el procesador 8085 y Z80
TRANSFER	programa de demostración Z80
RATON	programa de demostración 8748

Escribir y ensamblar un programa Z80

Haga primeramente las siguientes pruebas a guisa de ejemplo:

Escriba: type transfer.prn
Observe el programa Z80 "transfer" ya ensamblado

Ahora teclee: type transfer .asm
con lo que se despliega el programa fuente,
antes de ensamblar.

Recuerde que el set de instrucciones del 8085 es un subconjunto del set de instrucciones del Z80 ; por esta razón, los mnemónicos empleados en las instrucciones comunes al 8085 y Z80, se escriben con el formato del 8085, aunque el programa sea para el Z80.

Como ejercicio, con ayuda del editor TURBO-87, abra un archivo con nombre BITO.ASM de la siguiente manera:

```
A> TURBO 87
Include error messages (Y/N)  N
Edit
Work file name: BITO.ASM
```

(Aquí se asume que el usuario está familiarizado con el editor de turbo Pascal).

Escriba, ya dentro del editor, el siguiente programa:

```
        ORG 2000H
        .Z80

BITO:   MOV A, B
        MVI A, 3EH
        JMP BITO
```

Almacene este archivo bajo el mismo nombre, y regrese al sitema operativo.

En seguida, con el comando
A> XASM85 BITO.ASM

corra el programa ensamblador del Z80 y espere los resultados.

El ensamblador produce 2 archivos:

BITO.PRN con el programa fuente y los códigos de operación y
BITO.HEX que es el archivo que se enviará hacia el sistema PAT86, formado de caracteres ASCII.

Dé los comandos:
A> TYPE BITO.PRN y
A> TYPE BITO.HEX y observe ambos archivos en pantalla.

El método expuesto se puede hacer extensivo para escribir programas más largos. Si es necesario, utilice un diskette adicional en el drive "b", teniendo cuidado de teclear siempre b: antes del nombre del archivo que se lee ó que se almacena.

Transferir el código objeto al sistema PAT86:

Con lo visto en el punto anterior, ya es posible escribir y ensamblar programas en lenguaje del procesador Z80, creando un archivo ASCII (BITO.HEX), listo pra trasmitirse por el puerto serie.

Para realizar la transferencia, conecte primero, siguiendo los lineamientos del capítulo 1, el sistema PAT86 a la PC. Ahora, corra el programa VTERM y asegúrese que hay comunicación PAT-PC, "ON LINE".

Pulse 2 veces la tecla F5 y, en la etiqueta "Filename", escriba el nombre del archivo que desee trasmitir, por ejemplo, TRANSFER.HEX. En seguida, mediante el "ESCAPE" regrese a "ON LINE" y escriba:

:G700(CR)

Este comando ejecuta el programa residente en ROM del PAT86, cuya función es la recepción del archivo ASCII proveniente de la PC. En este momento, el PAT86 está en "Stand by", esperando el bloque de información.

Como paso final, oprima las teclas ^ALT T^. Este es el comando que indica al programa VTERM que debe trasmitir su archivo por el puerto serie. Espere el final de la transferencia. El PAT86 envía el mensaje
CONII85
:

El bloque enviado debió almacenarse a partir de la dirección 2000H en el PAT86, ya que esta es la dirección inicial que se le dió al programa fuente TRANSFER.ASM.

Verifique lo anterior con el comando
:SM2000 (CR).

Pulsando la barra de espacio, aparecerán cada uno de los bytes transferidos.

Compare esta información con la obtenida en el archivo TRANSFER.PRN. Recuerde que para switchear al MS-DOS solo necesita teclear SHIFT-SHIFT.

No olvide que el sistema PAT86 tiene su stack pointer inicializado en la localidad 20FFH. Para evitar cualquier posibilidad de traslape, escriba sus programas a partir de la 2100H.

Desde luego, una vez teniendo la información en la RAM del PAT86, pueden programarse memorias 2716 ó 2532 haciendo uso del programador EXOR. Refiérase al manual del Sistema PAT86 para hacer uso de este módulo.

Las técnicas y programas descritos conforman un respetable soporte para el desarrollo y prueba de programas basados en el procesador Z80. No se agotan aquí, sin embargo, las posibilidades de la interfaz PAT-PC. En los siguientes capítulos se describen técnicas de desarrollo para los procesadores de la familia 48, el 8048 y el 8748 entre otros.

6. ENSAMBLADOR 8748

En esta sección se describirán los pasos para escribir y ensamblar programas para el procesador 8748, así como la transferencia de los mismos a la RAM del sistema PAT86.

Escribir y ensamblar un programa 8748.

Haga, como práctica, las siguientes pruebas, escribiendo:
A> type raton.asm
Observe el aprograma de prueba ^raton^, antes de ser ensamblado

Transferir el programa objeto 8748 al PAT86.

El módulo EXOR48, programador del chip 8748 puede conectarse al sistema PAT86, haciendo de Este un programador controlado desde la PC.

Para transferir el programa de la PC al PAT86, repita los mismos pasos realizados para el caso de archivos Z80 (ver capítulo 5).

Una vez con el programa en la RAM del PAT86, haga uso del programador EXOR48. En el reporte "Módulos de expansión para el sistema PAT86" se da toda la información al respecto.

En el siguiente capítulo se describe la utilización de un paquete de extrema utilidad: un simulador para los chips de la familia 48, entre ellos el 8748. Con ayuda de este paquete, pueden escribirse y ejecutarse los programas en el simulador, antes de programar el EPROM de los chips.

7. SIMULADOR DEL CHIP 8048, 8748.

En el capítulo anterior, se dieron los pasos para ensamblar y preparar archivos del 8748, para su transferencia al PAT86, sistema desde donde es posible programar el EPROM del chip 8748.

Se cuenta, además, con un paquete que permite la simulación de los programas de procesadores de la familia 48. El uso de este paquete representa un considerable ahorro en tiempo, en el desarrollo de sistemas basados en estos chips.

El paquete se encuentra en el diskette con la etiqueta PAT-PC.

Primeramente, llame al programa tecleando:

A> AVSIM48

El sistema pregunta por el procesador con el cual se va a trabajar. Elija la opción A, para el 8048 y 8748.

Ahora, dentro del despliegue mostrado oprima "L" load, Y LUEGO "A". En seguida, cuando el programa nos pida el nombre del archivo (filename), se debe teclear el nombre del programa que deseamos simular, en este caso RATON.HEX.

En la siguiente hoja se muestra un resumen de los comandos del paquete de simulación AVSIM48. Si requiere más información, refiérase al manual de este programa.

En seguida teclee

A> XASM48 RATON.ASM

Este programa ensamblador produce los archivos
RATON.PRN y RATON.HEX.

Observe el archivo ensamblado tecleando:

A>type raton.prn

y el archivo de trasmisión escribiendo:

A>type ratón.hex

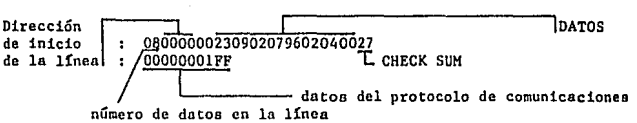
Observe que ahora, a diferencia del caso Z80, la dirección inicial del programa es 0000H, puesto que el 8748 solo tiene 1K de memoria ROM y 64 localidades en RAM.

Esto imposibilita la trasmisión al sistema PAT86, dado que, en la dirección 0000H se encuentra su ROM.

Con el objeto de permitir la transferencia de este archivo, debe editarse, en el programa RATON.HEX, las direcciones iniciales de transferencia. Para tal fin, escriba:

A>TURBO-87
y traiga a edición el archivo RATON.HEX

En pantalla deberá aparecer:



Edite la dirección de inico y escriba el número 2100.

Almacene este archivo, bajo el nombre RATON.PAT. El programa está ahora, listo para su transferencia.

En programas con más líneas, debe cambiarse la dirección de inicio de todas las líneas. El CHECK SUM puede dejarse sin cambio, dado que el programa de transferencia no lo toma en cuenta.

COMANDOS DEL AVSIM.

CTRL C	interrumpe cualquier comando.
MODO COMANDO	(simulación).
ESC	toggle hacia modo display.
F1	start/stop simulación.
F2	mueve cursor de breakpoint hacia arriba.
F3	pone breakpoint dinámico.
F4	mueve cursor de breakpoint hacia arriba.
F5	controla la velocidad de simulación.
F6	display on/off.
F7	control binario /hex/ASCII.
F8	salta la instrucción BSR y JSR.
F9	una instrucción hacia arriba.
F10	paso a paso.

para salir ´Q´ y luego ´E´.

MODO DISPLAY (Edición de áreas resaltadas en pantalla.) Mueva el cursor con las teclas de flechas o sitúese directamente con:

CTRL A	Acumulador
CTRL X	X reg.índice
CTRL P	PC
CTRL H	H Flag
CTRL I	I Flag
CTRL N	N Flag
CTRL Z	Z Flag
CTRL C	C Flag
CTRL S	SP
CTRL R	TDR timer control register
CTRL T	TCR timer counter.

Fije la zona de memoria:

CTRL PGUP	toggle hacia modo scrolling
PG UP	Scrolling hacia arriba
PG DN	Scrolling hacia abajo
ALT F6	toggle si se quiere actualizar datos.

Teclas de edición:

+	incrementa byte /palabra/bandera
-	decrementa byte /palabra/bandera
Ins	inserta caracter y corre hacia la derecha el resto de la línea
Del	borra caracter y corre hacia la izquierda el resto de la línea
CTRL END	clear to end of object
CTRL HOME	clear entire object

AVSIM 8048 Simulator/Debugger

Serial # 00120 Licensed by Avocet Systems, Inc.

Intel 8048 Family Microcomputers

	HMOS		CMOS ROM		OTHERS
A:	8048/8748/8035	D:	80C48/80C35	F:	8020
B:	8049/8749/8039	E:	80C49/80C39	G:	8021
C:	8050/8040			H:	8022

Intel 8041 Family Universal Peripheral Interface

I:	8041	J:	8041A/8641A/8741A	K:	8042/8742

Choose a CPU for simulation:

```
ADDR    OPERATION          8048/8748/8035 AVSIM 8048 Simulator/Debugger    V1.0
0000H   no     memory      CPU REGISTERS          FLAGS    SCL SPD DSP SKP CURSOR
0001H   no     memory      C  ACCUMULATOR      AC F0 F1    OFF HI  ON  OFF  MENU
0002H   no     memory      0  00000000:00:0    0  0  0              Cycles:    OFF
0003H   no     memory          addr      data
0004H   no     memory      PC:0000 / FF FF FF FF        TIMER
0005H   no     memory      SP:  00 / 00 00 00 00     ctr:00   OFF    BUS:    Out
0006H   no     memory                00 00 00 00     tov:0    tf:0            In
0007H   no     memory      RB:0    MB:0                                FF:0:11111111
0008H   no     memory      R0:00:0 / 00:0          INT FFs    PINS     FF:0:11111111
0009H   no     memory      R1:00:0 / 00:0            ie:0     t0:0
000AH   no     memory      R2:00   R4:00   R6:00     tie:0    tf:0     PORT:  Latch
000BH   no     memory      R3:00   R5:00   R7:00     iip:0    int:1           Pins
000CH   no     memory      Data Space                                   P1   11111111
000DH   no     memory      0000  00 00 00 00 00 00 00 00 00000000 FF:0:11111111
000EH   no     memory      0008  00 00 00 00 00 00 00 00 00000000 P2   11111111
000FH   no     memory      0010  00 00 00 00 00 00 00 00 00000000 FF:0:11111111
0010H   no     memory      0018  00 00 00 00 00 00 00 00 00000000
0011H   no     memory      Data Space                                   Expanded I/O
0012H   no     memory      0020  00 00 00 00 00 00 00 00 00000000 P4/5     Pins
0013H   no     memory      0028  00 00 00 00 00 00 00 00 00000000 FF:0:11111111
0014H   no     memory      0030  00 00 00 00 00 00 00 00 00000000 P6/7     Pins
0015H   no     memory      0038  00 00 00 00 00 00 00 00 00000000 FF:0:11111111
>Select Command:
Dump  Expression  commandFile  Help  IO  Load  --space--  ESC to screen
```

```
ADDR    OPERATION          8048/8748/8035 AVSIM 8048 Simulator/Debugger    V1.0
0000H   IN     A,P1        CPU REGISTERS         FLAGS     SCL SPD DSP SMP CURSOR
0001H   ANL    A,#78H      C  ACCUMULATOR      AC F0 F1    OFF HI  ON  OFF  MENU
0003H   RR     A           0  00000000:00:0     0  0  0                Cycles:    OFF
0004H   RR     A              addr      data
0005H   RR     A           PC:0000 / 09 53 78 77        TIMER
0006H   MOV    R0,#31H     SP:  00 / 00 00 00 00     ctr:00  OFF    BUS:    Out
0008H   MOV    @R0,A                00 00 00 00     tov:0   tf:0            In
0009H   MOV    R0,#2CH     R0:0    MB:0                                    FF:0:11111111
000BH   JB3    0083H       R0:00:0 / 00:0          INT FFs    PINS         FF:0:11111111
000DH   JB2    0049H       R1:00:0 / 00:0            ie:0     t0:0
000FH   JB1    002BH       R2:00  R4:00  R6:00     tie:0     t1:0     PORT:  Latch
0011H   JB0    0020H       R3:00  R5:00  R7:00     iip:0    int:1            Pins
0013H   MOV    @R0,#04H    Data Space                                      P1   11111111
0015H   INC    R0          0000  00 00 00 00 00 00 00 00 00000000 FF:0:11111111
0016H   MOV    @R0,#05H    0008  00 00 00 00 00 00 00 00 00000000 P2   11111111
0018H   INC    R0          0010  00 00 00 00 00 00 00 00 00000000 FF:0:11111111
0019H   MOV    @R0,#02H    0018  00 00 00 00 00 00 00 00 00000000
001BH   INC    R0          Data Space                                      Expanded I/O
001CH   MOV    @R0,#00H    0020  00 00 00 00 00 00 00 00 00000000 P4/5     Pins
001EH   JMP    00F7H       0028  00 00 00 00 00 00 00 00 00000000 FF:0:11111111
0020H   MOV    @R0,#08H    0030  00 00 00 00 00 00 00 00 00000000 P6/7     Pins
0022H   INC    R0          0038  00 00 00 00 00 00 00 00 00000000 FF:0:11111111
>Select Command:
Dump  Expression  commandFile  Help  IO  Load   --space--  ESC to screen
```

TURBO PASCAL

Algunos comandos del editor

	mueve cursor
insert	modo inserción
del	borra caracter
PG DN	baja página
PG UP	sube página
CTRL Y	borra línea
CTRL K D	se sale al menú
CR	añade línea

APENDICE D

LISTADO DEL FIRMWARE DEL DIPAK

```
                        ;
                        ; ***********************************************
                        ; **        LABORATORIO DE AUTOMATIZACION      **
                        ; **          :: JORGE RUBIO MONROY ::         **
                        ; **             .. U N A M ..                 **
                        ; **  = Dispositivo Para Control de Acceso =    **
                        ; **              << D I P A K >>              **
                        ; ***********************************************

                        ;    << Guarda Secuencia de la Clave en la Dir. #2CH en RAM >>

0000 09                         IN   A,P1       ; Lee el Puerto 1 (Clave Elegida)
0001 5378                       ANL  A,#78H     ; Deja unicamente del Bit 3 al Bit 6
0003 77                         RR   A          ; Rota a la Derecha el Acumulador -
0004 77                         RR   A          ; 3 veces
0005 77                         RR   A
0006 B831                       MOV  R0,#31H    ; 31 <== Dir. Mem. de Clave
0008 A0                         MOV  @R0,A      ; Guarda el Num.de Clave en (31H)
0009 B82C                       MOV  R0,#2CH    ; Dir. de inicio (grabar clave)
000B 7293                       JB3  SAL22
000D 5249                       JB2  SAL12
000F 3220                       JB1  SAL02
0011 1220                       JB0  SAL1

0013 B004                       MOV  @R0,#04H   ; Digito (1)   *** CLAVE  0 ***
0015 18                         INC  R0
0016 B005                       MOV  @R0,#05H   ; Digito (2)
0018 18                         INC  R0
0019 B002                       MOV  @R0,#02H   ; Digito (3)
001B 18                         INC  R0
001C B000                       MOV  @R0,#00H   ; Digito (4)
001E 04F7                       JMP  GUARDA

0020 B008       SAL1:           MOV  @R0,#08H   ; Digito (1)   *** CLAVE  1 ***
0022 18                         INC  R0
0023 B003                       MOV  @R0,#03H   ; Digito (2)
0025 18                         INC  R0
0026 B001                       MOV  @R0,#01H   ; Digito (3)
0028 18                         INC  R0
0029 B002                       MOV  @R0,#02H   ; Digito (4)
002B 04F7                       JMP  GUARDA

002D 123C       SAL02:          JB0  SAL3

002F B003                       MOV  @R0,#03H   ; Digito (1)   *** CLAVE  2 ***
0031 18                         INC  R0
0032 B002                       MOV  @R0,#02H   ; Digito (2)
0034 18                         INC  R0
0035 B004                       MOV  @R0,#04H   ; Digito (3)
0037 18                         INC  R0
```

```
0033 B005           MOV   @R0,#05H    ; Digito (4)
0035 04F7           JMP   GUARDA

0037 B009     SAL3: MOV   @R0,#09H    ; Digito (1)   *** CLAVE  3 ***
0039 18             INC   R0
003A B000           MOV   @R0,#00H    ; Digito (2)
003C 18             INC   R0
003D B000           MOV   @R0,#00H    ; Digito (3)
003F 18             INC   R0
0040 B005           MOV   @R0,#05H    ; Digito (4)
0042 04F7           JMP   GUARDA

0044 3267     SAL12: JB1   SAL04
0046 1254           JB0   SAL5

0048 B003           MOV   @R0,#03H    ; Digito (1)   *** CLAVE  4 ***
004A 18             INC   R0
004B B001           MOV   @R0,#01H    ; Digito (2)
004D 18             INC   R0
004E B006           MOV   @R0,#06H    ; Digito (3)
0050 18             INC   R0
0051 B002           MOV   @R0,#02H    ; Digito (4)
0053 04F7           JMP   GUARDA

0055 B007     SAL5: MOV   @R0,#07H    ; Digito (1)   *** CLAVE  5 ***
0057 18             INC   R0
0058 B008           MOV   @R0,#08H    ; Digito (2)
005A 18             INC   R0
005B B007           MOV   @R0,#07H    ; Digito (3)
005D 18             INC   R0
005E B003           MOV   @R0,#03H    ; Digito (4)
0060 04F7           JMP   GUARDA

0062 1276     SAL04: JB0   SAL7

0064 B004           MOV   @R0,#04H    ; Digito (1)   *** CLAVE  6 ***
0066 18             INC   R0
0067 B002           MOV   @R0,#02H    ; Digito (2)
0069 18             INC   R0
006A B008           MOV   @R0,#08H    ; Digito (3)
006C 18             INC   R0
006D B005           MOV   @R0,#05H    ; Digito (4)
006F 04F7           JMP   GUARDA

0071 B006     SAL7: MOV   @R0,#06H    ; Digito (1)   *** CLAVE  7 ***
0073 18             INC   R0
0074 B007           MOV   @R0,#07H    ; Digito (2)
0076 18             INC   R0
0077 B002           MOV   @R0,#02H    ; Digito (3)
0079 18             INC   R0
007A B000           MOV   @R0,#00H    ; Digito (4)
```

```
0091 04F7               JMP  GUARDA

0093 528F      SAL22:   JB2  SAL14
0095 22A3               JB1  SAL06
0097 1296               JB0  SAL9

0099 B002               MOV  @R0,#02H   ; Digito (1)   *** CLAVE  8 ***
009B 18                 INC  R0
009C B005               MOV  @R0,#05H   ; Digito (2)
009E 18                 INC  R0
009F B004               MOV  @R0,#04H   ; Digito (3)
00A1 18                 INC  R0
00A2 B001               MOV  @R0,#01H   ; Digito (4)
00A4 04F7               JMP  GUARDA

00A6 B009      SAL9:    MOV  @R0,#09H   ; Digito (1)   *** CLAVE  9 ***
00A8 18                 INC  R0
00A9 B007               MOV  @R0,#07H   ; Digito (2)
00AB 18                 INC  R0
00AC B002               MOV  @R0,#02H   ; Digito (3)
00AE 18                 INC  R0
00AF B005               MOV  @R0,#05H   ; Digito (4)
00B1 04F7               JMP  GUARDA

00B3 12B2      SAL06:   JB0  SAL11

00B5 B000               MOV  @R0,#00H   ; Digito (1)   *** CLAVE 10 ***
00B7 18                 INC  R0
00B8 B003               MOV  @R0,#03H   ; Digito (2)
00BA 18                 INC  R0
00BB B003               MOV  @R0,#03H   ; Digito (3)
00BD 18                 INC  R0
00BE B004               MOV  @R0,#04H   ; Digito (4)
00C0 04F7               JMP  GUARDA

00C2 B003      SAL11:   MOV  @R0,#03H   ; Digito (1)   *** CLAVE 11 ***
00C4 18                 INC  R0
00C5 B007               MOV  @R0,#07H   ; Digito (2)
00C7 18                 INC  R0
00C8 B003               MOV  @R0,#03H   ; Digito (3)
00CA 18                 INC  R0
00CB B000               MOV  @R0,#00H   ; Digito (4)
00CD 04F7               JMP  GUARDA

00CF 32D8      SAL14:   JB1  SAL03
00D1 12D8               JB0  SAL13

00D3 B005               MOV  @R0,#05H   ; Digito (1)   *** CLAVE 12 ***
00D5 18                 INC  R0
00D6 B001               MOV  @R0,#01H   ; Digito (2)
00D8 18                 INC  R0
```

```
00C9 B007           MOV  @R0,#07H   ; Digito (3)
00CB 18             INC  R0
00CC B002           MOV  @R0,#02H   ; Digito (4)
00CE 04F7           JMP  GUARDA

00D0 3097     SAL13:MOV  @R0,#07H   ; Digito (1)   *** CLAVE 13 ***
00D2 18             INC  R0
00D3 B009           MOV  @R0,#09H   ; Digito (2)
00D5 18             INC  R0
00D6 B004           MOV  @R0,#04H   ; Digito (3)
00D8 18             INC  R0
00D9 B003           MOV  @R0,#03H   ; Digito (4)
00DB 04F7           JMP  GUARDA

00DD 12EC     SAL08:JB0  SAL15

00DF B001           MOV  @R0,#01H   ; Digito (1)   *** CLAVE 14 ***
00E1 18             INC  R0
00E2 B005           MOV  @R0,#05H   ; Digito (2)
00E4 18             INC  R0
00E5 B000           MOV  @R0,#00H   ; Digito (3)
00E7 18             INC  R0
00E8 B002           MOV  @R0,#02H   ; Digito (4)
00EA 04F7           JMP  GUARDA

00EC B006     SAL15:MOV  @R0,#06H   ; Digito (1)   *** CLAVE 15 ***
00EE 18             INC  R0
00EF B000           MOV  @R0,#00H   ; Digito (2)
00F1 18             INC  R0
00F2 B007           MOV  @R0,#07H   ; Digito (3)
00F4 18             INC  R0
00F5 B001           MOV  @R0,#01H   ; Digito (4)

                ;   << Guarda el valor de cada tecla Dir. #20H en RAM >>

00F7 B920     GUARDA:MOV R0,#20H    ; R0 <== Dir. de mem. RAM (Col,Ren)

00F9 B002           MOV  @R0,#02H   ; Tec(00) == 00
00FB 18             INC  R0
00FC B005           MOV  @R0,#05H   ; Tec(01) == 01
00FE 18             INC  R0
00FF B008           MOV  @R0,#08H   ; Tec(02) == 02
0101 18             INC  R0
0102 B000           MOV  @R0,#00H   ; Tec(03) == 03
0104 18             INC  R0
0105 B003           MOV  @R0,#03H   ; Tec(10) == 04
0107 18             INC  R0
0108 B006           MOV  @R0,#06H   ; Tec(11) == 05
010A 18             INC  R0
010B B009           MOV  @R0,#09H   ; Tec(12) == 06
010D 18             INC  R0
```

```
010E B008           MOV  @R0,#08H   ; Tec(13) == 07
0110 18             INC  R0
0111 B009           MOV  @R0,#01H   ; Tec(20) == 03
0113 18             INC  R0
0114 B004           MOV  @R0,#04H   ; Tec(21) == 09
0116 18             INC  R0
0117 B007           MOV  @R0,#07H   ; Tec(22) == 0A
0119 18             INC  R0
011A B00C           MOV  @R0,#0CH   ; Tec(23) == 0B

                ;   << Inicializa la Micro y Comienza Reconocimiento >>

011C 230F   INIC:   MOV  A,#0FH     ; Desactiva el Teclado y el -
011E 3A             OUTL P2,A       ; Reelevador
011F 230A           MOV  A,#0AH     ; Despliega una (>) de Clave
0121 02             OUTL BUS,A      ; Escribe el Dato en el BUS
0122 85             CLR  F0         ; Limpia la Ban.F0 (<- Clave Correcta => 0
0123 A5             CLR  F1         ; Limpia la Ban.F1 (<- No se ha tecleado => 0
0124 B930           MOV  R0,#30H    ; Dir. de Mem.RAM (<= ON/OFF LED
0126 B008           MOV  @R0,#08H   ; Mascara (ON/OFF) LED cuando se teclea
0128 BA08           MOV  R2,#08H    ; Mascara (ON/OFF) LED mientras no se teclea
012A BD2C           MOV  R5,#2CH    ; R5 <= Contador del Num.de teclas oprimidas
012C B931   LECT:   MOV  R1,#31H    ; Dir. que guarda el Num.de Clave
012E BB9F           MOV  R3,#09FH   ; Uso en la lectura
0130 BCFF           MOV  R4,#0FFH   ; del Teclado ( 5 segundos )
0132 26D7   LEE1:   JNT0 ACTIVA     ; Si T0=0 --> Activa el Reelevador
0134 97             CLR  C          ; Limpia Carry
0135 B8FE           MOV  R0,#0FEH   ; R0 <= Renglon 0
0137 F8             MOV  A,R0
0138 2607   LEE2:   JNT0 ACTIVA     ; Si T0=0 --> Activa el Reelevador
013A 3A             OUTL P2,A       ; Escribe al Puerto 2
013B 09             IN   A,P1       ; Lee al Puerto 1
013C 5307           ANL  A,#07H     ; Limpia el Dato
013E D307           XRL  A,#07H     ; Checa si se tecleo algo
0140 C643           JZ   CONT1
0142 A9             MOV  R1,A       ; R1 <= Guarda la Columna
0143 AE             MOV  R6,A       ; R6 <= Guarda Col.(Suelta Tecla)
0144 F8             MOV  A,R0       ; R0 <= Guarda el Renglon
0145 AF             MOV  R7,A       ; R7 <= Guarda Ren.(Suelta Tecla)
0146 2464           JMP  INTER      ; Salta a Interpreta (INTER)
0148 EC5C   CONT1:  DJNZ R4,CICLO   ; Control de Tiempo -
014A BCFF           MOV  R4,#0FFH   ; de Teclado ( R4 y R3 )
014C 7653           JF1  CONT11     ; Si se ha tecleado ==> CONT11
014E FB             MOV  A,R3       ; Movemos el Reg3 a el Acum.
014F 3256           JB1  CONT11     ; Si el Bit 1 esta prendido ve a CONT11
0151 FA             MOV  A,R2       ; A <= Guarda Mascara (ON/OFF) LED
0152 47             SWAP A          ; Intercambia 4 X 4 Bits (Switchea el LED)
0153 AA             MOV  R2,A       ; Guarda el valor switcheado en el Reg.2
0154 5338           ANL  A,#38H     ; Mascarea el valor switcheado
0156 41             ORL  A,@R1      ; Combina el val.del acum. y el Num.de Clave
0157 02             OUTL BUS,A      ; Escribe el Dato en el BUS
```

```
0158 EB5C    CONT11:  DJNZ R2,CICLO  ; Si termina el tiempo asignado -
015A 241C             JMP  INIC      ; Ve a INIC (inicializa)
015C F8      CICLO:   MOV  A,R0      ; Si el Bit6 = 1 llego al
015D D232             JB6  LEE1      ; reng. 3 regresa al 0
015F A7               CPL  C         ; Prende el Carry
0160 F7               RLC  A         ; Activa el Sig. renglon
0161 A8               MOV  R0,A      ; y lo guarda en el Reg.0
0162 2438             JMP  LEE2      ; Salta a LEE2

                ;   << Interpreta el Valor de la Tecla oprimida >>

0164 7667    INTER:   JF1  OPRIM     ; Si F1=1 --> Ya estaba activado
0166 95               CPL  F1        ; F1 <== 1
0167 BA00    OPRIM:   MOV  R2,#00H   ; R2 : Posicion del Renglon
0169 BB00             MOV  R3,#00H   ; R3 : Posicion de la Columna
016B F8               MOV  A,R0      ; R0 = Renglon  //  R1 = Columna
016C 26D7    RENG:    JNT0 ACTIVA    ; Si T0=0 --> Activa el Reelevador
016E 67               RRC  A         ; Busca la Posicion del Renglon
016F E674             JNC  FINREN    ; Salta si ya la encontro
0171 1A               INC  R2
0172 246C             JMP  RENG      ; Salta a RENG
0174 F9      FINREN:  MOV  A,R1
0175 26D7    COLUM:   JNT0 ACTIVA    ; Si T0=0 --> Activa el Reelevador
0177 67               RRC  A         ; Busca la Posicion de la Col.
0178 7A7D             JC   JUNTA     ; Salta si la Encuentra
017A 1B               INC  R3
017B 2475             JMP  COLUM     ; Salta a COLUM

017D BC00    JUNTA:   MOV  R4,#00H   ; R4 : Localidad de Memoria-Tecla
017F 26D7    POSCOL:  JNT0 ACTIVA    ; Si T0=0 --> Activa el Reelevador
0181 FB               MOV  A,R3
0182 C68B             JZ   POSREN
0184 FC               MOV  A,R4      ; Suma de 4 en 4 buscando la Loc.
0185 0304             ADD  A,#04H    ; decrementando la posicion de Col.
0187 AC               MOV  R4,A
0188 CB               DEC  R3
0189 247F             JMP  POSCOL    ; Salta a POSCOL
018B FC      POSREN:  MOV  A,R4      ; Suma la Posicion del Renglon
018C 6A               ADD  A,R2

                ;   << Despliega la Tecla oprimida y activa o desactiva el LED >>

018D 0320             ADD  A,#20H    ; Banco de Memoria
018F A8               MOV  R0,A      ; Guarada la Loc. de Memoria-Tecla
0190 F0               MOV  A,@R0     ; Saca el Valor de la Tecla
0191 AA               MOV  R2,A      ; R2 <== Guarda el valor del Acum.
0192 B830             MOV  R0,#30H   ; Dir. RAM de la mascara a switchear LED
0194 F0               MOV  A,@R0     ; A <== Mascara
0195 47               SWAP A         ; Intercambia 4 y 4 Bits (Switchea LED)
0196 A0               MOV  @R0,A     ; Guarda en RAM la mascara Switcheada
0197 5380             ANL  A,#80H    ; Mascara que apaga el bit 3
```

```
0199 4A                 ORL  A,R2        ; Combina el valor del Led con el Dato Tecla
019A 02                 OUTL BUS,A       ; Escribe el Dato en el BUS

                ;   << Tiempo de Retardo para Estabilizar el Teclado y activ. Bocina >>

019B AB                 MOV  R3,A        ; R3 : Guarda el Dato escrito en el Bus
019C 4340               ORL  A,#40H      ; A  : Mezcla el Dato y Activa Bit 6 prend.
019E B80F               MOV  R0,#0FH     ; R0 : Tiempo de Retardo 1
01A0 B968       RETAR1: MOV  R1,#068H    ; R1 : Tiempo de Retardo 2  (250 mseg)
01A2 E9A2       RETAR2: DJNZ R1,RETAR2   ; Decrementa R6 y Salta a RETAR2
01A4 2B                 XCH  A,R3        ; Intercambia el Acum. y el Reg.3 para -
01A5 02                 OUTL BUS,A       ; activar o desactivar el Bit 6 ==> Bocina
01A6 E8A0               DJNZ R0,RETAR1   ; Decrementa R5 y Salta a RETAR1
01A8 B80A               MOV  R0,#0AH     ; Tiempo de retardo para -
01AA B962       RETAR3: MOV  R1,#62H     ; estabilizar el teclado
01AC E9AC       RETAR4: DJNZ R1,RETAR4
01AE E8AA               DJNZ R0,RETAR3

                ;   << Checa si se Solto la Tecla >>

01B0 FF         SUELTA: MOV  A,R7        ; Checa si ya se solto la
01B1 3A                 OUTL P2,A        ; tecla que se habia oprimido
01B2 09                 IN   A,P1
01B3 5307               ANL  A,#07H
01B5 D307               XRL  A,#07H
01B7 DE                 XRL  A,R6
01B8 2607               JNTO ACTIVA      ; Si T0=0 --> Activa el Reelevador
01BA C6B0               JZ   SUELTA

                ;   << Compara el Valor Tecleado con la Clave = Reconocimiento >>

01BC FD                 MOV  A,R5        ; R0 <=== R5 (Dir. de  la Clave)
01BD A8                 MOV  R0,A        ; R2 <== Contiene el Dato Tecleado
01BE FA                 MOV  A,R2
01BF D0                 XRL  A,@R0       ; Checa si son iguales
01C0 C6C8               JZ   RECON1      ; Si es igual salta a RECON1
01C2 B830               MOV  R0,#30H     ; Dir. RAM de la mascara a switchear LED
01C4 F0                 MOV  A,@R0       ; A <== Mascara
01C5 5308               ANL  A,#08H      ; Mascara que apaga el Bit 3
01C7 430C               ORL  A,#0CH      ; Despliega una (u) como error
01C9 02                 OUTL BUS,A       ; Escribe el Dato
01CA B6CD               JF0  RECON1      ; Si F0=1 ==) No saca su Complemento
01CC 95                 CPL  F0          ; Si no prende F0
01CD 2607       RECON1: JNTO ACTIVA      ; Si T0=0 --> Activa el Reelevador
01CF 1D                 INC  R5          ; Incrementa la Direccion
01D0 2330               MOV  A,#30H      ; Checa si es el ultimo valor de la Clave
01D2 00                 XRL  A,R5        ; y lo compara
01D3 962C               JNZ  LECT        ; si no es el ultimo salta a LECT
01D5 B61C               JF0  INIC        ; Si es el ultimo y F0=1 ===) INIC
01D7 23FF       ACTIVA: MOV  A,#0FFH     ; Activa el Relelevador
01D9 3A                 OUTL P2,A        ; y permite la entrada
```

```
010A B903           MOV  R0,#03H    ; Retardo para mantener activa la
010C B9FF     RET1: MOV  R1,#0FFH   ; la chapa
010E BAFF     RET2: MOV  R2,#0FFH
01E0 EAE0     RET3: DJNZ R2,RET3
01E2 E9DE           DJNZ R1,RET2
01E4 E8DC           DJNZ R0,RET1
01E6 241C           JMP  INIC       ; Salta al inicio ( Inicializa )
0000                END
```

---- SYMBOL TABLE ----

ACTIVA	01D7	ARRITA	0179	RET1	01DC	SAL06	00A3	SAL3	003C		
CICLO	015C	LECT	012C	RET2	01DE	SAL08	00D0	SAL5	005A		
COLUM	0175	LEE1	0132	RET3	01E0	SAL1	0020	SAL7	0076		
CONT1	0148	LEE2	0133	RETAR1	01A0	SAL11	0082	SAL9	0096		
CONTT1	0158	OPRIM	0167	RETAR2	01A2	SAL12	0049	SUELTA	01B0		
FINREN	0174	POSCOL	017F	RETAR3	01AA	SAL13	00D0				
GUARDA	00F7	POSREN	018B	RETAR4	01AC	SAL14	00BF				
INIC	011C	RECONT	01C0	SAL02	0020	SAL15	00EC				
INTER	0164	RENG	016C	SAL04	0067	SAL22	0083				

dirak.hex

```
:1000000009537877777B031A0B82C7283524933B8
:10010002D1220B00418B00518B00218B00004F773
:1002000B00818B00318B00118B00204F7123CB0C1
:10030000318B00218B00418B00504F7B00918B0DE
:1004000018B00018B00504F73267125AD0031850
:1005000B00118B00618B00204F7B00718B00818BD
:1006000B00718B00304F71276B00418B00218B045
:1007000818B00504F7B00418B00718B00218B099
:100800000004F752BF32A31296B00218B00518B0A0
:1009000418B00104F7B00918B00718B00218B07E
:100A0000504F712B2B00018B00818B00318B00475
:100B00004F7B00318B00718B00318B00004F73203
:100C0000012D0B00518B00118B00918B00204F75D
:100D000B00718B00918B00418B00304F712ECB058
:100E0000118B00518B00018B00204F7B00618B037
:100F0000018B00718B001B820B00218B00518B049
:1010000018B00018B00318B00618B00918B00BE2
:10110001B00118B00418B00718B00C230F3A2318
:10120000A0285A5B830B008BA08BD20B931B89F0A
:10130000BCFF26D797B8CEF826D73A095307D3073E
:10140000C648A9AEF8AF2464EC5CBCFF7658F8321D
:1015000058FA47AA53304102EB5C241CF5D232A71C
:10160000F7A824387667B5BA00BB00F826D767E64B
:10170007041A246CF926D767F67D1B2475BC0026FB
:10180000D7FBC68BFC0304ACCB247FFC6A0320A3FE
:10190000F0AA8830F047A053804A02AB4340B80FF2
:101A0000B968E9A23B02E8A0B3DAB962E9ACE8AA1A
:101B000FF3A095307D307DE26D7C6B0FDA8FAD009
:101C0000C6CDB330F0538043CC02B6CD9526D71D6E
:101D0002330D0962CB61C23FF3AB803B9FFBAFFD3
:0801E000EAE0E9DEE5DC241C82
:00000001FF
```

dipak.pat

```
:1021000009537877777B831A0B92C728352493288
:10211000D1220B00418B00518B00218B00004F773
:10212000B00818B00318B00118B00204F7123CB0C1
:10213000031BB00218B00418B00504F7B00918B0DE
:10214000018B00018B00504F73267125AB0031850
:10215000B00118B00618B00204F7B00718B00818BD
:10216000B00718B00304F71276B00418B00218B045
:10217000061BB00504F7B00618B00718B00218B099
:10218000004F752BF32A31296B00218B00518B0A0
:10219000413B00104F7B00918B00718B00218B07E
:1021A0000504F712B3B00018B00818B00318B00475
:1021B00004F7B00318B00718B00318B00004F73203
:1021C000DD12D0B00518B00118B00918B00204F75D
:1021D000B00718B00918B00418B00304F712ECB058
:1021E00001188B00518B00018B00204F7B00618B037
:1021F00000138B00718B001B820B00218B00518B049
:10220000818B00018B00318B00618B00918B00BE2
:10221000183B0011BB00418B00718B00C230F3A2318
:10222000A0285A5B830B008BA08BD2CB931BB9F0A
:10223000BCFF26D797B80EF826D73A0953307D3073E
:10224000C648A9AEF8AF2464EC5CBCFF7658F8321D
:1022500058FA47AA53804103EB5C241CF8D232A71C
:10226000F7A824387667B5BA00BB00F826D767E64B
:10227000741A246CF926D767F67D1B2475BC0036FB
:10228000D7FBC68BFC0304ACCB247FFC6A0320A8FE
:10229000F0AAB830F047A053804A02AB4340B80FF2
:1022A000B960E9A22B02E8A0B8DAB962E9ACE8AA1A
:1022B000FF3A025307D307DE26D7C6B0FDA8FAD009
:1022C000C4CDB830F053B0430C02B6CD9526D71D6E
:1022D0002330DD962CB61C23FF3AB303B9FFBAFFD3
:0922E000EAE0F9DEE8DC241C82
:00000001FF
```

AVSIM 8048 Simulator/Debugger

Serial # 00120 Licensed by Avocet Systems, Inc.

Intel 8048 Family Microcomputers

	HMOS		CMOS ROM		OTHERS
A:	8048/8748/8035	D:	80C48/80C35	F:	8020
B:	8049/8749/8039	E:	80C49/80C39	G:	8021
C:	8050/8040			H:	8022

Intel 8041 Family Universal Peripheral Interface

I:	8041	J:	8041A/8641A/8741A	K:	8042/8742

Choose a CPU for simulation:

```
ADDR    OPERATION          8048/8748/8035 AVSIM 8048 Simulator/Debugger   V1.0
0000H   no      memory     CPU REGISTERS        FLAGS    SCL SPD DSP SKP CURSOR
0001H   no      memory     C  ACCUMULATOR     AC F0 F1    OFF H1  ON  OFF  MENU
0002H   no      memory     0  00000000:00:0    0  0  0             Cycles:    OFF
0003H   no      memory        addr      data
0004H   no      memory     PC:0000 / FF FF FF FF        TIMER
0005H   no      memory     SP:  00 / 00 00 00 00     ctr:00  OFF    BUS:   Out
0006H   no      memory                00 00 00 00     tov:0   t1:0           In
0007H   no      memory     RB:0    MB:0                                 FF:0:11111111
0008H   no      memory     R0:00:0 / 00:0          INT FFs    PINS      FF:0:11111111
0009H   no      memory     R1:00:0 / 00:0           ie:0      t0:0
000AH   no      memory     R2:00   R4:00   R6:00    tie:0     t1:0      PORT:  Latch
000BH   no      memory     R3:00   R5:00   R7:00    iip:0     int:1            Pins
000CH   no      memory     Data Space                                    P1   11111111
000DH   no      memory     0000   00 00 00 00 00 00 00 00 00000000 FF:0:11111111
000EH   no      memory     0008   00 00 00 00 00 00 00 00 00000000 P2   11111111
000FH   no      memory     0010   00 00 00 00 00 00 00 00 00000000 FF:0:11111111
0010H   no      memory     0018   00 00 00 00 00 00 00 00 00000000
0011H   no      memory     Data Space                                    Expanded I/O
0012H   no      memory     0020   00 00 00 00 00 00 00 00 00000000 P4/5    Pins
0013H   no      memory     0028   00 00 00 00 00 00 00 00 00000000 FF:0:11111111
0014H   no      memory     0030   00 00 00 00 00 00 00 00 00000000 P6/7    Pins
0015H   no      memory     0038   00 00 00 00 00 00 00 00 00000000 FF:0:11111111
>Select Command:
Dump  Expression  @commandFile  Help  IO  Load  ---space---  ESC to screen
```

```
ADDR    OPERATION          8049/8748/8035 AVSIM 8048 Simulator/Debugger    V1.0
0000H   IN     A,P1        CPU REGISTERS         FLAGS     SCL SPD DSP SKP CURSOR
0001H   ANL    A,#78H      C  ACCUMULATOR      AC F0 F1    OFF HI  ON  OFF  MENU
0003H   RR     A           0  00000000:00:0     0  0  0              Cycles:    OFF
0004H   RR     A              addr      data
0005H   RR     A           PC:0000 / 09 53 78 77        TIMER
0006H   MOV    R0,#31H     SP:  00 / 00 00 00 00     ctr:00  OFF   BUS:   Out
0008H   MOV    @R0,A                 00 00 00 00     tov:0   tf:0          In
0009H   MOV    R0,#2CH     RB:0    MB:0                                 FF:0:11111111
000BH   JB3    0063H       R0:00:0 / 00:0          INT FFs    PINS      FF:0:11111111
000DH   JC2    0049H       R1:00:0 / 00:0            ie:0     t0:0
000FH   JB1    002DH       R2:00  R4:00  R6:00      tie:0    t1:0      PORT:  Latch
0011H   JB0    0020H       R3:00  R5:00  R7:00      iip:0    int:1           Pins
0013H   MOV    @R0,#04H    Data Space                                   P1   11111111
0015H   INC    R0          0000  00 00 00 00 00 00 00 00 00000000 FF:0:11111111
0016H   MOV    @R0,#05H    0008  00 00 00 00 00 00 00 00 00000000 P2   11111111
0018H   INC    R0          0010  00 00 00 00 00 00 00 00 00000000 FF:0:11111111
0019H   MOV    @R0,#02H    0018  00 00 00 00 00 00 00 00 00000000
001BH   INC    R0          Data Space                                   Expanded I/O
001CH   MOV    @R0,#00H    0020  00 00 00 00 00 00 00 00 00000000 P4/5     Pins
001EH   JMP    00F7H       0028  00 00 00 00 00 00 00 00 00000000 FF:0:11111111
0020H   MOV    @R0,#08H    0030  00 00 00 00 00 00 00 00 00000000 P6/7     Pins
0022H   INC    R0          0038  00 00 00 00 00 00 00 00 00000000 FF:0:11111111
>Select Command:
Dump  Expression  commandFile  Help  IO  Load   --space--  ESC to screen
```