UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE INGENIERIA

DISEÑO E IMPLEMENTACION DE UN PROBADOR DE CIRCUITOS INTEGRADOS POR MEDIO DE COMPUTADORA.

ROBERTO KHOURI SOLIS

1989

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE INGENIERIA

DIRECTOR: M. en I. JUAN CARLOS ROA BEIZA

TITULO: DISEÑO E IMPLEMENTACION DE UN PROBADOR DE CIRCUITOS INTEGRADOS POR MEDIO DE COMPUTADORA.

OBJETIVO:

Desarrollar instrumentación digital de alta calidad, que facilite el diagnóstico y mantenimiento de fallas en equipo electrónico para Ingenieros en Computación e Ingenieros en Electrónica.

Permitirá por un método de comparación, detectar circuitos integrados dañados dentro de los sistemas electrónicos, sin necesidad de extraerlos para verificarlos.

El sistema mostrará en pantalla el "CI" con su configuración de patas indicando en forma visual el adecuado o mal funcionamiento de éste.

I N D I C E

CAPITULO 1

I N T R O D U C C I O N

Una de las principales causas por las que se incrementa el costo de mantenimiento del equipo electrónico, es ocasionado por la dificultad que se tiene de verificar un circuito integrado (CI) eficientemente dentro de su gabinete, independientemente de los elementos que puedan alterar su funcionamiento.

Algunos de los caminos a seguir para facilitar el mantenimiento del equipo son:

1.- Substituir los circuitos que se creen dañados hasta encontrar el (los) elemento(s) que realmente está(n) afectando el funcionamiento del equipo.

Esta técnica aun sigue siendo utilizada a pesar de que incrementa en gran escala el costo de reparación, ya que el técnico puede incurrir en los siguientes errores y por ende incrementar los costos:

- Dañar el circuito al momento de desmontarlo y no ser éste, el que provoca la falla. (Costo del CI).
- Dañar el circuito al montarlo, sin obtener

solución satisfactoria. (Costo del CI)

- Cuando dos o mas CI's son los dañados, se incrementa en gran medida el tiempo de su localización y verificación, por consiguiente esta técnica es ineficiente debido a la pérdida de tiempo que ocasiona montar y desmontar los CI's. (Costo Horas/Hombre)

2.- Colocar bases en todos los CI's.

El colocar bases a todos los CI's no es costeable, por lo que, solo en los CI's más costosos, tales como: microporcesadores y memorias, son insertados en bases. Pero puede también incurrirse en los errores mencionados en el punto anterior cuando se trata de circuitos MOS, no obstante, se eliminan los riesgos que existen al desoldar un circuito y el tiempo ocasionado cuando se cambia, disminuye considerablemente.

Notando las desventajas de las técnicas anteriores, el objetivo de este proyecto es diseñar y construir un probador de CI's computarizado, para facilitar el diagnóstico y mantenimiento de fallas en equipo electrónico.

Este proyecto se realizará en la computadora COCO 3 (Radio Shack TRS-80 color computer), debido a las facilidades que se tienen para su adquisición, además, del práctico manejo de los lenguajes con los que cuenta.

Con este probador de CI's no será necesario extraer el CI para verificar su buen o mal funcionamiento, puesto que, se contará con un interfaz que permitirá la conexión entre la computadora y el CI, pudiendo ésta leer los datos proporcionados por el circuito y analizarlos por medio de la comparación de una tabla de verdad almacenada en una memoria EPROM con la obtenida por la computadora, lo que evita:

- Riesgo de dañar algún circuito al extraerlo y probarlo.
- Certeza en el diagnóstico del equipo a probar.
- Reducción drástica del tiempo de revisión y costos de mano de obra.

En el momento de proporcionar a la computadora el número de circuito a probar, se desplegará en pantalla la configuración de patas, así como, cada uno de los estados bajo prueba y el número de circuito interno actual, que se esté probando en caso de que el CI tenga más de uno.

En caso de alguna discrepancia entre la tabla de verdad teórica y la generada se desplegará un letrero de ERROR en la esquina superior derecha, acompañado de un sonido. En este momento observaremos con mas cuidado el estado y el número de circuito interno en el que se encuentra la falla.

Se almacenó en memoria las tablas de verdad de 10 CI's, esto no limita al sistema, pues, si uno desea probar algún circuito que no se encuentre dentro de los seleccionados, tiene como opción el poder almacenar la información necesaria del circuito para después verificar su funcionamiento.

CAPITULO 2

DESCRIPCION DE LA COMPUTADORA COCO 3

En el presente capitulo se hará una breve descripción de la computadora COCO 3, así como su editor assembler.

DESCRIPCIÓN GENERAL

El sistema COCO 3 se compone de:

- Un teclado con 53 teclas para entrada de programas y datos.
- Una interfaz para televisión lo que permite tener un amplio rango de sonidos y gama de colores.
- Un microprocesador 6809E.
- Una ROM que contiene el lenguaje BASIC.
- Una RAM para almacenar programas y datos mientras la computadora está encendida (ésta es expandible de 64 a 128 Kbyte opcional/extra).
- Una entrada de cartucho para cargar paquetes de programación.
- Dos interfases para joystick.
- Una interfase de alta velocidad para cassette,

lo que permite el almacenamiento de programas y datos. Estos son salvados a una velocidad de 1500 baunds aproximadamente (11,000 caracteres por minuto).

- Una interfase para impresora.

Es posible desplegar textos, gráficas (para lo cual se tienen disponibles 9 colores), generar tonos y efectos especiales.

El teclado contiene caracteres de escritura estándar, además de caracteres de control marcados con nombres especiales: break, enter, clear, etc..

DESCRIPCION DEL SISTEMA.

Las funciones principales de la COCO 3 son ejecutadas por 5 circuitos LSI (Larga Escala de Integración) de 40 patas, estos circuitos son etiquetados como CPU, SAM, VDG y dos PIA's; además, una Memoria de Acceso Aleatorio (RAM) y una Memoria de Solo Lectura (ROM). Con los circuitos anteriores y una fuente de poder, la COCO 3 operará y proveerá una salida de video compuesto y permitirá la comunicación con el exterior al conectarle interfases de I/O (fig. 1).

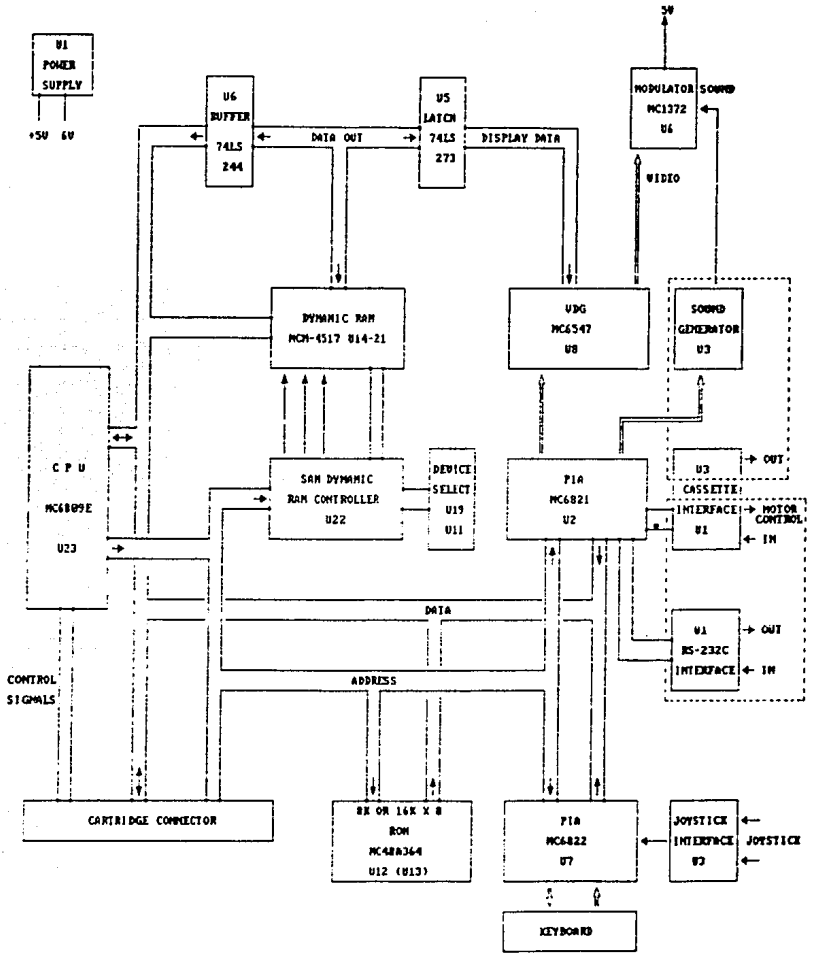

U1
POWER
SUPPLY
+5V 6V
U6
BUFFER
74LS
244
DATA OUT
U5
LATCH
74LS
273
DISPLAY DATA
5V
MODULATOR
MC1372
U6
SOUND
VIDEO
DYNAMIC RAM
MCM-4517 814-21
VDG
MC6547
U8
SOUND
GENERATOR
U3
C P U
MC6809E
U23
SAM DYNAMIC
RAM CONTROLLER
U22
DEVICE
SELECT
U19
U11
PIA
MC6821
U2
U3
OUT
CASSETTE
INTERFACE
U1
MOTOR
CONTROL
IN
DATA
U1
RS-232C
INTERFACE
OUT
IN
CONTROL
SIGNALS
ADDRESS
CARTRIDGE CONNECTOR
8K OR 16K X 8
ROM
MC68A364
U12 (U13)
PIA
MC6822
U7
JOYSTICK
INTERFACE
U2
JOYSTICK
KEYBOARD

FIGURA 1

- CPU .- (Central Processor Unit).

La CPU es la unidad central de procesamiento, su función es ejecutar programas almacenados en la memoria central tomando sus instrucciones, examinándolas y ejecutándolas una tras otra.

La CPU también tiene una pequeña memoria de alta velocidad utilizada para almacenar resultados intermedios y cierta información de control.

Tiene la capacidad de ejecutar un limitado número de operaciones matemáticas y lógicas con los datos.

La CPU ejecuta cada instrucción en una serie de pasos que se muestra a continuación:

a) Extrae de la memoria la siguente instrucción y la lleva al registro de instrucción.

b) Incrementa el contador del programa de modo que apunte a la dirección de la siguiente instrucción.

c) Analiza el tipo de instrucción que acaba de extraer.

d) Verifica si la instrucción requiere datos de la memoria y, si es así,

determina donde están situados.

e) Extrae los datos si los hay y los carga en los registros internos de la CPU.

f) Realiza la instrucción.

g) Almacena los resultados en el lugar apropiado.

h) Se regresa al inciso a).

- ROM .- (Read Only Memory).

Tiene la función de proveer al CPU con un conjunto predefinido de instrucciones en Basic.

En operación normal el CPU da un salto a la dirección de inicio en ROM después de que el switch del RESET ha sido presionado y entonces ejecuta el programa del RESET colocando al mismo nivel todos los dispositivos programables, posteriormente el intérprete de Basic, residente en ROM, se encontrará bajo control del CPU.

- RAM .- (Random Access Memory).

Provee el almacenamiento para los programas

y/o datos. La capacidad de una RAM MCM4517 es de 16K X 1. El número de circuitos a usar depende del tamaño de memoria que se quiera tener en la computadora. Se tiene como opción utilizar memorias MC6665 de 64K X 1, considerando que la memoria de la computadora tiene capacidad para expanderse hasta 128K. La RAM es también usada para desplegar el video, el que ocupa una porción diferente a la utilizada para el almacenamiento de programas. Durante la operación normal, el intérprete de Basic localizado en ROM, controlará la ejecución de los programas, localizados en RAM.

- SAM .- (Syncronous Address Multiplexer).

  Este circuito es un componente central que provee el refresco y multiplexaje de direcciones para la RAM; además, proporciona los tiempos del sistema y de los dispositivos de selección.

- VDG .- (Video Display Generator).

  Provee virtualmente toda la interfase de video

sobre un circuito y permite tener diversos modos gráficos y alfanuméricos. Los modos de operación del VDG son controlados por uno de los dos adaptadores de interfase periféricos (PIA's) usados en la COCO 3. Con esta información y datos de la RAM, el VDG genera información para la circuitería moduladora del video compuesto y de color.

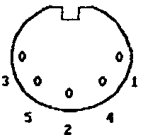
La circuitería restante de la COCO 3 está dedicada a la comunicación de entrada y salida. La parte más importante de la circuitería es el teclado, el que permite la entrada de información introducida por el usuario. Otros circuitos permiten la entrada de joysticks, entrada y salida de cassette y RS-232C.

ESPECIFICACIONES.

Voltaje	105-130 V, 60 Hz.
Corriente	0.18 Amperes.
Temperatura	12.8-29.4 grados centígrados.

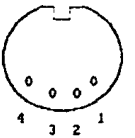
Altitud		-30 a 1830 metros sobre nivel del mar.
Microprocesador		MC6809E de Motorola. Velocidad del reloj: 0.896 MHz.
Memoria	RAM.	64K expandible a 128K.
	ROM.	8K de Basic standar (26-3026). 8K de Basic extendido (26-3027).
Teclado		53 teclas disponibles dentro de una matriz.
Video		Salida modulada RF, canal 3 o 4. 32 caracteres por 16 renglones. 8 colores disponibles. Resolución gráfica de 256 X 192 puntos.

Sonido	6 bit DAC. Bit particular. Entrada de cinta para cassette. Entrada de conector para cartucho.
Cassette	Velocidad de 1500 baund (fig 2).
RS-232C	Interfase de 3 hilos controlada por software (fig. 3).
Joystick	2 conectores. 0.25 a 4.75 V DC de entrada en 64 pasos (fig 4).
Expansión	Conector para cartucho de 40 patas. Todas las señales generales del CPU son necesarias para la interfase (fig 5).



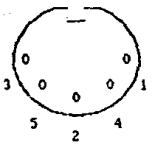
1.- REMOTE CONTROL
2.- SIGNAL GROUND
3.- REMOTE CONTROL
4.- INPUT FROM RECORDER'S EARPHONE JACK
5.- OUTPUT TO RECORDER'S AUX OR MIC JACK

FIGURA 2



SIGNAL RS-232C		PIN #
CD	CARRIER DETECT (STATUS INPUT LINE)	1
RD	RECEIVE DATA	2
GROUND	ZERO VOLTAGE REFERENCE	3
TD	TRANSMIT DATA OUT	4

FIGURA 3



1.- COMPARATOR INPUT (RIGHT-LEFT)

2.- COMPARATOR INPUT (DOWN/UP)

3.- GROUND

4.- "FIRE" BUTTOM, HIGH WHEN OPEN, LOW WHEN CLOSED

5.- Vcc, CURRENT-LIMITED +5VDC

FIGURA 4

PIN #	SIGNAL NAME	
1	N/C	
2	N/C	
3	HALT	HALT INPUT TO THE CPU
4	NMI	NON-MASKABLE INTERRUPT TO THE CPU
5	RESET	MAIN RESET AND POWER-UP CLEAR SIGNAL TO THE SYSTEM
6	E	MAIN CPU CLOCK (0.89 MHz)
7	Q	QUADRATIVE CLOCK SIGNAL WHICH LOADS E
8	CART	INTERRUPT INPUT FOR CARTRIDGE DETECTION
9	+5V	+5 VOLTS (300mA)
10	D0	CPU DATA BIT 0
11	D1	CPU DATA BIT 1
12	D2	CPU DATA BIT 2
13	D3	CPU DATA BIT 3
14	D4	CPU DATA BIT 4
15	D5	CPU DATA BIT 5
16	D6	CPU DATA BIT 6
17	D7	CPU DATA BIT 7
18	R/W	CPU READ-WRITE SIGNAL
19	A0	CPU ADDRESS BIT 0
20	A1	CPU ADDRESS BIT 1
21	A2	CPU ADDRESS BIT 2
22	A3	CPU ADDRESS BIT 3
23	A4	CPU ADDRESS BIT 4
24	A5	CPU ADDRESS BIT 5
25	A6	CPU ADDRESS BIT 6
26	A7	CPU ADDRESS BIT 7
27	A8	CPU ADDRESS BIT 8
28	A9	CPU ADDRESS BIT 9
29	A10	CPU ADDRESS BIT 10
30	A11	CPU ADDRESS BIT 11
31	A12	CPU ADDRESS BIT 12
32	CTS	CARTRIDGE SELECT SIGNAL
33	GND	SIGNAL GROUND
34	GND	SIGNAL GROUND
35	SND	SOUND INPUT
36	SCS	SPARE SELECT SIGNAL
37	A13	CPU ADDRESS BIT 13
38	A14	CPU ADDRESS BIT 14
39	A15	CPU ADDRESS BIT 15
40	SLENB	INPUT TO DISABLE DEVICE SELECTION

FIGURA 5

CARACTERISTICAS DE IMPRESION

- Velocidad de 600 baunds.
- 1 bit de start (cero lógico).
- 8 bits de datos (primer bit menos significativo).
- 2 bits de stop (uno lógico).
- No hay bit de paridad.
- 132 caracteres de impresión.
- Retorno automático del carro al fin de línea.

INTERFASE DE CASSETTE.

- Nivel de entrada sugerido para grabar:

  1-5 Vpp

  Zmin 220 Ohms

- Salida típica de la computadora para grabar:

  800 mVpp

  Z = 1 Kohm

- Switch automático de ON/OFF:

  0.5 Amax

  6 V DC

CONECTOR DE CARTUCHO

Este conector tiene 40 patas y proporciona la posibilidad de utilizar paquetes de utileria de la COCO 3. Todas las señales importantes del bus del CPU están conectadas a éste.

El cartucho usado es ROM. Por medio del reloj Q conectado a la pata de interrupción del microprocesador, se detecta que la ROM está insertada; lo que genera una interrupción y ocasiona que la CPU ordene al Program Counter alterar la dirección en donde inicia la ROM.

TEORIA DE OPERACION.

C P U

El 6809 es un microprocesador en una sola pastilla que contiene una ampliación, y mayores ventajas que el del 6800. Tiene todos los registros programables del 6800 así como, la mayoria de sus lineas externas de datos, dirección y control, haciendolo compatible con el hardware de este microprocesador.

Posee mayor poder de cálculo ya que consta de 1,464

códigos de operación.

El incremento de la potencia del 6809 se debe, en su mayoría, a sus nuevas posibilidades de direccionamiento.

En este microprocesador se tiene una instrucción de multiplicación pero no la de división.El 6809 cuenta con el mismo bus de datos externo de 8 bit que utiliza el 6800; su bus de datos interno y los circuitos asociados, tienen un ancho de 16 bits, por lo que, el 6809 puede considerarse como un microprocesador de 16 bits (ver fig. 6).

Las transferencias de datos a través del bus del sistema 6809 se realizan de la forma semisincrona, representada en la fig 7.

Existen 3 versiones del 6809

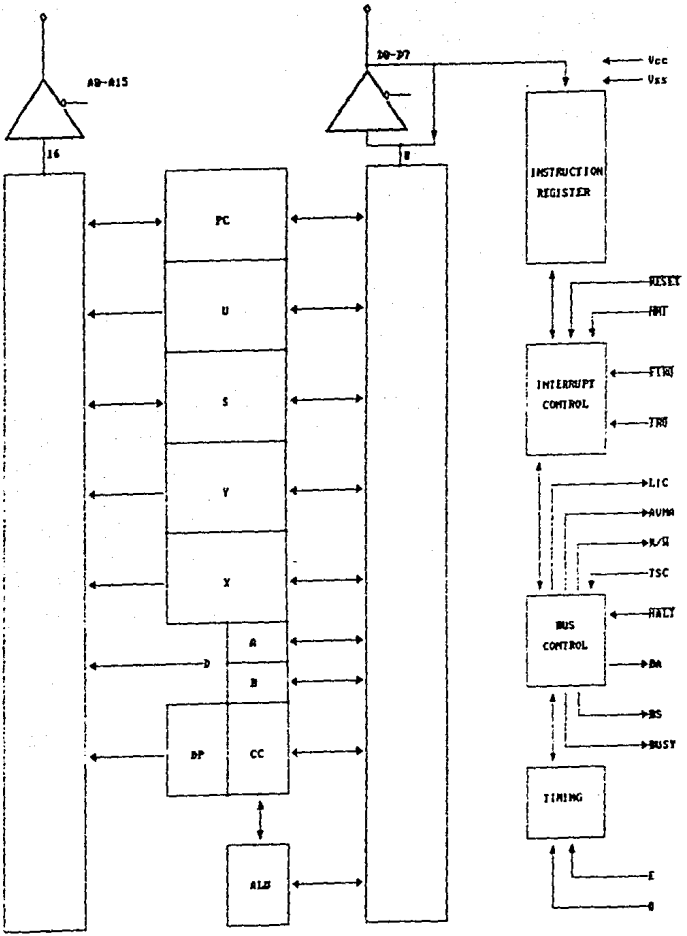
MC6809

MC6809E

MC68HC09E.

El equipo con el que se cuenta posee el microprocesador 6809E, que incluye todas las facilidades del 6809, además, relojes externos para proveer la flexibilidad requerida en un sistema multiprocesador.

El 6809E tiene un set más completo de modos de direccionamiento, comparado con otros microprocesadores de 8 bits.



* INTERNAL THREE STATE CONTROL

FIGURE 6

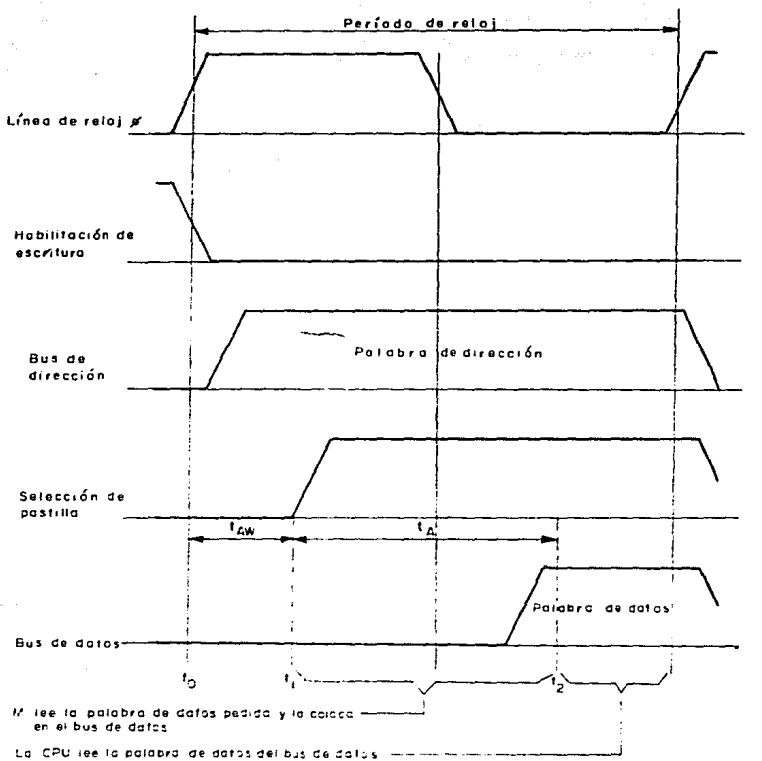

Período de reloj
Línea de reloj
Habilitación de escritura
Bus de dirección
Palabra de dirección
Selección de pastilla
t_{AW}
t_A
Palabra de datos
Bus de datos
t_0
t_1
t_2
M lee la palabra de datos pedida y la coloca en el bus de datos
La CPU lee la palabra de datos del bus de datos

FIGURA 7

Tal microprocesador tiene características de hardware y software que lo hacen ideal para lenguajes de alto nivel o aplicaciones estándar de control. Relojes externos de entrada permiten su sincronización con sistemas periféricos u otros microprocesadores.

COMPATIBILIDAD CON EL 6800.

- HARDWARE.- Interfases con todos los periféricos.
- SOFTWARE.- Código fuente compatible con el set de instrucciones y modos de direccionamiento.

CARACTERISTICAS ARQUITECTONICAS.

- 2 Registros Indice de 16 bits.
- 2 Stack Pointer indexables de 16 bits.
- 2 Acumuladores de 8 bits que pueden ser concatenados para formar uno de 16 bits.
- Registro de paginación directa que permite direccionamamiento para toda la memoria.

CARACTERISTICAS DE HARDWARE.

- Relojes externos (E y Q). Permiten la sincronización.
- Entrada TSC de control interno del bus de buffers.
- LIC. Indicador que trae el código de operación.
- AVMA. Permite el uso eficiente de recursos comunes en un sistema de microporcesador.
- BUSY. Es una línea de estado para el microprocesador.
- Entrada al stack por la interrupción de requerimiento rápido (Interrupción de hardware). Es realizado solamente con el registro de código de condición y el Program Counter.
- Interrupción Acknowledge Output. Permite la vectorización por dispositivos.
- Sincronización de salidas para eventos externos.
- Ciclo sencillo de bus para RESET.
- Operación a 5 Volts.
- NMI. Inhibido después del RESET hasta que carga la primera instrucción que direcciona el Stack Pointer.

  Las primeras direcciones válidas permiten usar

las primeras localidades bajas de memoria (comenzando de 0 a X).

- Los primeros datos escritos son para la memoria dinámica.

CARACTERISTICAS DE SOFTWARE.

- 10 modos de direccionamiento.
  - + Compatibilidad con modos de direccionamiento de la familia MC6800.
  - + Direccionamiento directo a través de todo el mapa de memoria.
  - + Bifurcaciones relativamente largas.
  - + Program Counter (PC) relativo.
  - + Direccionamiento indirecto.
  - + Direccionamiento indexado expandido. Teniendo un offset constante de 0, 5, 8 y 16 bits, offset para el acumulador de 8 o 16 bits.
  - + Autoincremento o decremento por 1 ó 2.
- 1,464 instrucciones con un único modo de direccionamiento.
- Multiplicación sin signo de 8 X 8 bits.
- Aritmética de 16 bit.

- Transferencia/cambio de todos los registros.
- Push y Pop de algunos registros o conjunto de registros del stack.
- Carga efectiva de direcciones.

Los modos de direccionamiento disponibles son:

1.- Indirecto (Incluye acumulador).

2.- Inmediato.

3.- Extendido.

3.1.- Indirecto extendido.

4.- Directo.

5.- De registro.

6.- Indexado.

6.1.- Offset cero.

6.2.- Offset constante.

6.3.- Offset acumulador.

6.4.- Autoincremento/decremento.

6.5.- Indexado indirecto.

7.- Relativo.

7.1.- Derivación relativa larga/corta.

7.2.- Relativo del Program Counter (PC).

DESCRIPCION DE REGISTROS

Acumuladores A, B y D.

Los registros A y B son acumuladores de propósito general los que son usados para cálculos aritméticos y manejo de datos.

Ciertas instrucciones concatenan los registros A y B para tener un acumulador de 16 bits. Este es referido como el registro D y está formado por el registro A como byte más significativo.

Registro de Dirección de Página.

El registro de dirección de página del MC6809E sirve para tener un modo de direccionamiento directo. El contenido de este registro aparece en las direcciones de salidas altas (A8-A15), durante la ejecución de instrucciones de direccionamiento directo, esto permite al modo directo ser usado en algún lugar de memoria, bajo el control del programa. Todos los bits de este registro son limpiados durante el proceso de reset.

Registros Indices (X,Y).

Los registros índices son usados en modos de direccionamiento indexado. La dirección de 16 bits en el

registro toma parte en el cálculo de direcciones efectivas. Estas direcciones pueden ser usadas para apuntar directamente a datos o pueden modificarse por una constante opcional o un registro de offset.

Stack Pointer (U,S).

El stack pointer del hardware (S) es usado automáticamente por el procesador durante el llamado a subrutinas e interrupciones. El stack pointer del MC6809E apunta al tope del stack, en contraste al stack pointer del MC6800 el que apunta a la siguiente localidad libre en el stack.

El stack pointer del usuario (U) es controlado exclusivamente por el programador.

Ambos stack pointer tienen el mismo modo de direccionamiento indexado similar al de X y Y, pero también soportan las instrucciones PUSH y PULL. Esto le permite al MC6809E ser usado eficientemente como un procesador de stack, incrementando su capacidad para soportar lenguajes de alto nivel y programación modular.

Program Counter (PC).

Es usado por el procesador para apuntar a la dirección de la siguiente instrucción a ser ejecutada.

Registro de Código de Condición (CCR).

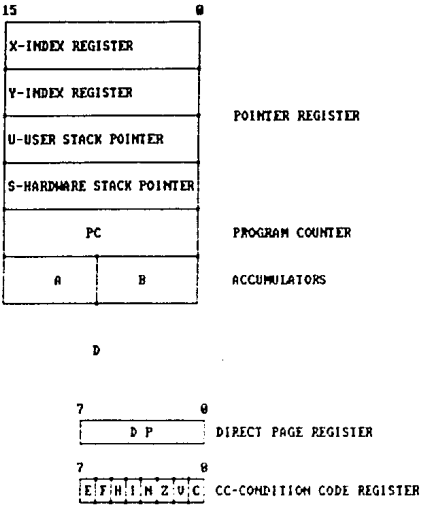
Define el estado del procesador, dependiendo del resultado de algunas operaciones.

NOTA: Para mayor ilustración ver la figura 8.

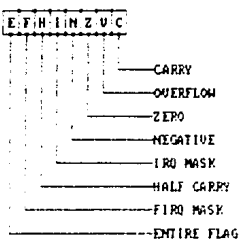
DESCRIPCION DE PINES.

La figura 9 muestra cada uno de los pines del C.I. para mayor ilustración.

Vss	Tierra
Vcc	5 V +/- 5%
A0-A15	Bus de direcciones. Usadas para salida de información desde el MPU sobre el bus.
D0-D7	Bus de datos. Estos 8 pines proveen comunicación con el sistema bidireccional del bus de datos.



PROGRAMMING MODEL OF THE MICROPROCESSING UNIT



CONDITION CODE REGISTER FORMAT

FIGURA 8

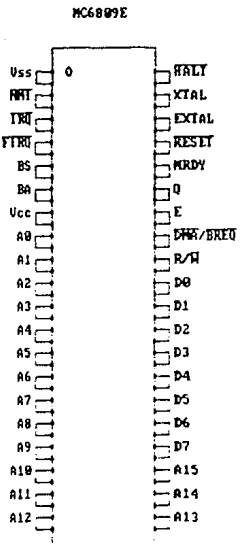

MC6809E
Vss
NMI
IRQ
FIRQ
BS
BA
Vcc
A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
HALT
XTAL
EXTAL
RESET
MRDY
Q
E
DMA/BREQ
R/W
D0
D1
D2
D3
D4
D5
D6
D7
A15
A14
A13

FIGURA 9

R/W	Lectura-Escritura. Esta señal indica la dirección de transferencia de datos en el bus.
RESET	Con un voltaje bajo empieza a funcionar el microprocesador.
HALT	Un nivel bajo de entrada causa que el MPU se detenga al finalizar la instrucción presente y permanezca detenida indefinidamente sin pérdida de datos.
BA, BS	El bus disponible de salida está indicando una señal de control interna que hace que el bus MOS del MPU esté en alta impedancia.

NMI	Interrupción no mascarable. La CPU siempre causa la interrupción del flujo del programa normal y ejecuta una rutina especial de interrupción manual. (FIGURA 10)
FIRQ	Requerimiento de interrupción rápida. Un nivel bajo en esta entrada inicia una secuencia rápida de interrupción previendo un bit de máscara (F) lo que modifica el CC (FIGURA 11).
IRQ	Requerimiento de interrupción. Un nivel bajo de entrada en esta pata permitiría inicializar una secuencia en la interrupción de

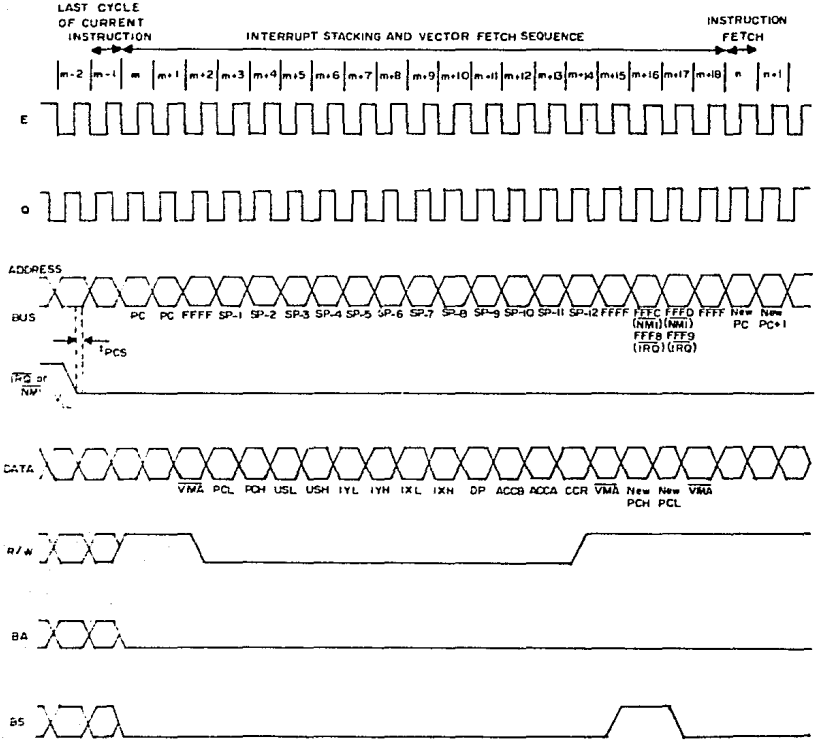

LAST CYCLE OF CURRENT INSTRUCTION
INTERRUPT STACKING AND VECTOR FETCH SEQUENCE
INSTRUCTION FETCH
m-2 m-1 m m+1 m+2 m+3 m+4 m+5 m+6 m+7 m+8 m+9 m+10 m+11 m+12 m+13 m+14 m+15 m+16 m+17 m+18 n n+1
E
Q
ADDRESS
BUS
PC PC FFFF SP-1 SP-2 SP-3 SP-4 SP-5 SP-6 SP-7 SP-8 SP-9 SP-10 SP-11 SP-12 FFFF FFFC (NMI) FFFB (IRQ) FFFD (NMI) FFF9 (IRQ) FFFF New PC New PC+1
tPCS
IRQ or NMI
DATA
VMA PCL PCH USL USH IYL IYH IXL IXH DP ACCB ACCA CCR VMA New PCH New PCL VMA
R/W
BA
BS

FIGURA 10

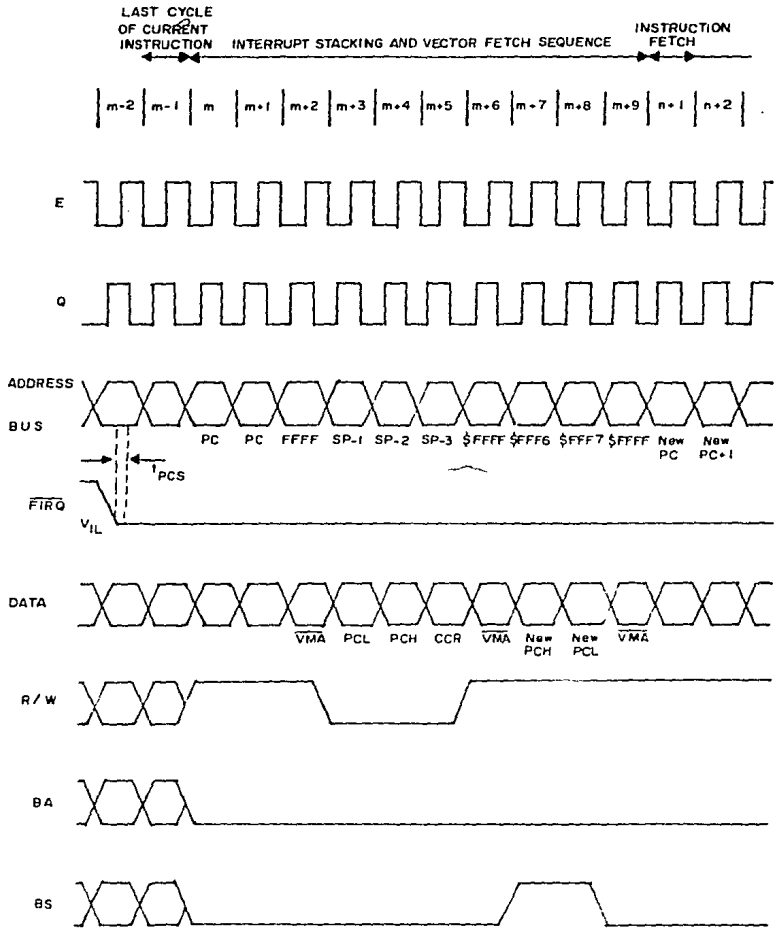

LAST CYCLE OF CURRENT INSTRUCTION
INTERRUPT STACKING AND VECTOR FETCH SEQUENCE
INSTRUCTION FETCH
m-2
m-1
m
m+1
m+2
m+3
m+4
m+5
m+6
m+7
m+8
m+9
n+1
n+2
E
Q
ADDRESS BUS
PC
PC
FFFF
SP-1
SP-2
SP-3
$FFF8
$FFF6
$FFF7
$FFFF
New PC
New PC+1
tPCS
FIRQ
VIL
DATA
VMA
PCL
PCH
CCR
VMA
New PCH
New PCL
VMA
R/W
BA
BS

FIGURA 11

requerimiento. Provee el bit de máscara (I) modificando el CC (fig. 10).

E, Q

Entradas de reloj. Son las señales requeridas para el MC6809E. Q es primero que E, esto es, una transmisión en Q puede ser seguida por una transmisión similar en E después de unos minutos de retardo.

BUSY

Es alto para leer y modificar ciclos de una instrucción de Lectura/Escritura y durante el acceso del primer byte de una operación de doble byte. Es también alto durante el primer byte de alguna

búsqueda indirecta o de otro vector.

AVMA — Es un avance de la señal VMA e indica que el MPU puede usar el bus en el siguiente ciclo BS, Permite compartir eficientemente el sistema del microprocesador.

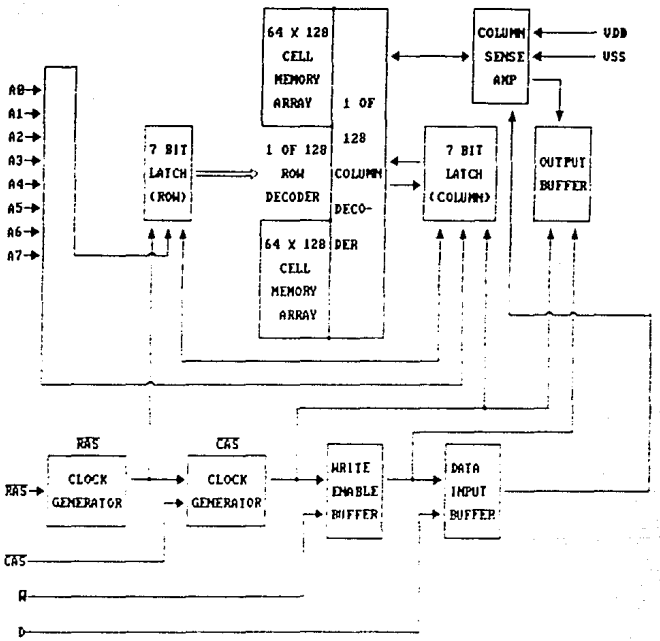
LIC — Ciclo de la última instrucción. En las instrucciones y su transición de alto a bajo, indicará que el primer byte de un código de operación puede ser sujeta al fin de un ciclo presente. Este puede ser alto cuando el MPU está detenido en el fin de una instrucción.

TSC	Control de 3 estados. Causará una dirección MOS, datos, buffers de R/W al asumir un estado de alta impedancia. Este fue creado de modo que permita compartir con otros buses maestros.

DRAM's

La COCO 3 utiliza ocho Memorias Dinámicas de Acceso Aleatorio, cada uno de estos circuitos es capaz de almacenar 16,384 bits (16 K). El CPU necesita accesar 8 bits de datos en un mismo tiempo, lo que origina tener bancos de memoria de 16K X 8 (fig. 12).

Las direcciones son multiplexadas dentro de 2 grupos de 7 renglones y 7 columnas de dirección. Los renglones de dirección se presentan primero y la DRAM reconoce que está la dirección del renglón con la presencia del RAS* (Row Address Strobe) y ausencia del CAS* (Column Address Strobe). Después de que la DRAM obtuvo los bits de los renglones se presentan las direcciones de las columnas con la señal CAS*. Si el ciclo presente es de lectura,



DRAM BLOCK DIAGRAM

FIGURA 12

WE* (Write Enable) se mantiene en alto y los dátos son extraídos de la celda correspondiente y presentados en las patas de salida en un tiempo real. El tiempo real depende del tiempo de acceso de la DRAM. Durante el ciclo de escritura, los datos y la señal WE son activados antes y durante el tiempo de CAS.

ROM

Memoria no volátil. Cuando se aplica voltaje al CPU la ROM inmediatamente trae un vector e inicia la ejecución de instrucciones. La COCO 3 contiene una o dos ROM's dependiendo si tiene Basic Standar o Extendido. Cada ROM contiene 8K bytes que son programados para proveer el uso acertado de comandos y funciones de Basic. La ROM que contiene el Basic Standar está siempre instalada, por lo que, es indispensable para las máquinas que contienen Basic Extendido. Todas las direcciones y líneas de datos son paralelas en las dos ROM's, la única línea que es independiente para cada ROM es CE (Chip Enable).

PIA's

La COCO 3 utiliza 2 adaptadores de interfase para periféricos, los que proveen la interfase universal para el MC6809E CPU, soportando todas las funciones de entrada y salida de la COCO 3.

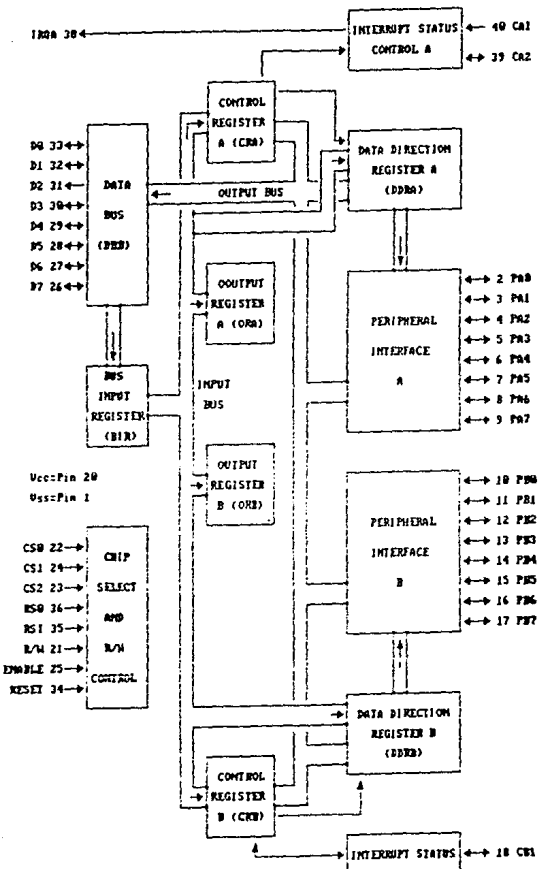
La configuración funcional de las PIA's es programada por el CPU durante la rutina del RESET. Cada línea de datos del periférico puede ser programada para actuar como entrada o salida y cada una de las cuatro líneas de control/interrupción pueden ser programadas para uno de los diversos modos de control (fig. 13).

Una PIA consiste de 2 registros de 8 bit de datos y 4 líneas control/interrupción. Las direcciones de los registros de control son colocados arriba de la rutina del RESET y normalmente no van a ser modificadas.

Las 4 líneas de control/interrupción son controladas por 2 registros, éstos registros de control son dispositivos de selección manejados dentro de la PIA. La función de 2 de las 4 líneas es unicamente como interrupción de entrada y las otras dos líneas pueden ser usadas como interrupción de entradas o salidas.

Una de las PIA's es usada principalmente para el teclado y funciona como sigue:

- El registro de datos B (patas 10-17) es programado



PIA BLOCK DIAGRAM

FIGURE 13

como salida y se utiliza para habilitar las columnas del teclado.

- Las primeras 7 líneas de datos del registro A (patas 2-8) son programadas como entradas y son usadas para leer los renglones del teclado.
- Las patas 2 y 3 son usadas para encender el botón de entrada para los joysticks.
- Las patas restantes tienen diversas funciones. El bit más significativo del registro A (pata 9) es programada como una entrada para la interfase del joystick. CA2 y CB2 (pata 19 y 39) son usadas como salidas. Estas dos líneas seleccionan una de las cuatro entradas del joystick o sonidos. Las patas CA1 y CB1 (patas 40 y 18) son usadas como entradas de interrupciones.

La otra PIA es usada para diferentes funciones:

- Las patas 4-9 del registro A son usadas para la conversión de los 6 bit de digital a analógico.

- La pata 3 del registro A es la señal de salida de la RS-232C, la que es usada para el manejo de impresora y otros tipos de dispositivos RS-232C.

- La pata 2 del registro A es la entrada para datos desde cassette.

- Las patas 13-17 del registro de datos B son usadas para el control de selección de diversos modos alfanuméricos y gráficos del VDG.

- La pata 12 del registro B es una entrada para el tamaño del salto en memoria.

- La pata 11 del registro B es un bit particular para la salida de video.

- La pata 10 es una pata de señal de entrada del RS-232C.

- La pata de control/interrupción del PIA, tiene además otras funciones. CA1 (pata 40) es la entrada para la señal CD (entrada de interrupción de status para la interfase RS-232C). CA2 es una salida para controlar el motor del cassette. CB1 es la entrada de interrupción del cartucho. Finalmente CB2 es usado como una salida para habilitar sonidos desde el circuito DAC.

EDITOR ASSEMBLER

El editor assembler incluye 3 sistemas:

1.- El editor, para introducir programas en lenguaje ensamblador 6809E.

2.- El ensamblador, para ensamblar los programas dentro del código de máquina 6809E.

3.- ZBUG, permite examinar y depurar los programas en código de máquina.

El 6809E puede direccionar 65,536 bytes de localidades de memoria (0000-FFFF).

Los programas editados en ensamblador comienzan en la localidad C000, en la figura 14 se muestra el mapa de memoria.

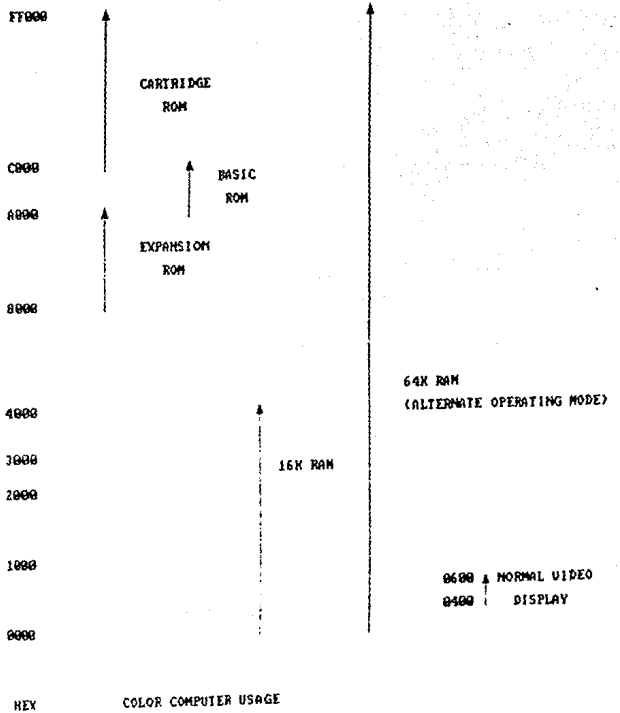
La memoria se puede examinar en 4 diferentes modos:

1.- BYTE. Este modo despliega byte por byte el contenido de memoria a partir de cierta dirección.

2.- WORD. Este modo despliega palabra por palabra el contenido en memoria a partir de cierta direccción. Una palabra está compuesta por 2 bytes.

3.- ASCII. Despliega el contenido de cada localidad de memoria en código ASCII.

4.- MNEMONIC. Este modo es el de default. Despliega



MAPA DE MEMORIA

FIGURA 14 (a)

FF00

BIT0=KEYBOARD ROW 1 AND RIGHT JOYSTICK SWITCH
BIT1=KEYBOARD ROW 2 AND LEFT JOYSTICK SWITCH
BIT2=KEYBOARD ROW 3
BIT3=KEYBOARD ROW 4
BIT4=KEYBOARD ROW 5
BIT5=KEYBOARD ROW 6
BIT6=KEYBOARD ROW 7
BIT7=JOYSTICK COMPARISON INPUT

FF01

BIT0		0=IRQ TO CPU DISABLED 1=IRQ TO CPU ENABLED
	CONTROL OF THE HORIZONTAL SYNC CLOCK (63.5 MICROSECONDS) INTERRUPT INPUT	
BIT1		0=FLAG SET ON THE FAILING EDGE OF HS 1=FLAG SET ON THE RISING EDGE OF HS
BIT2=NORMALLY 1:	0=CHANGES FF00 TO THE DATA DIRECTION REGISTER	
BIT3=SEL 1:	LSB OF THE TWO ANALOG MUX SELECT LINES	
BIT4=1 ALWAYS		
BIT5=1 ALWAYS		
BIT6=NOT USED		
BIT7=HORIZONTAL SYNC INTERRUPT FLAG		

FF02

BIT0=KEYBOARD COLUMN 1
BIT1=KEYBOARD COLUMN 2
BIT2=KEYBOARD COLUMN 3
BIT3=KEYBOARD COLUMN 4
BIT4=KEYBOARD COLUMN 5
BIT5=KEYBOARD COLUMN 6
BIT6=KEYBOARD COLUMN 7/RAM SIZE OUTPUT
BIT7=KEYBOARD COLUMN 8

FF03

BIT0		0=IRQ TO CPU DISABLED 1=IRQ TO CPU ENABLE
	CONTROL OF THE FIEL SYNC CLOCK 16.667 MS INTERRUPT INPUT	
BIT1		0=SETS FLAG ON FALLING EDGE FS 1=SETS FLAG ON RISING EDGE FS
BIT2=NORMALLY 1:	0=CHANGES FF02 TO THE DATA DIRECTION REGISTER	
BIT3=SEL 2:	MSB OF THE TWO ANALOG MUX SELECT LINES	
BIT4=1 ALWAYS		
BIT5=1 ALWAYS		
BIT6=NOT USED		
BIT7=FIELD SYNC INTERRUPT FLAG		

MAPA DE MEMORIA

FIGURA 14 (b)

FF20

BIT0=CASSETTE DATA INPUT
BIT1=RS-232 DATA OUTPUT
BIT2=6 BIT D/A LSB
BIT3=6 BIT D/A
BIT4=6 BIT D/A
BIT5=6 BIT D/A
BIT6=6 BIT D/A
BIT7=6 BIT D/A MSB

FF21

BIT0	CONTROL OF THE CD RS-232 STATUS INPUT	0=FIRQ TO CPU DISABLED 1=FIRQ TO CPU ENABLED
BIT1		0=SET FLAG ON FALLING EDGE CD 1=SET FLAG ON RISING EDGE CD
BIT2=NORMALLY 1:	0=CHANGES FF20 TO THE DATA DIRECTION REGISTER	
BIT3=CASSETTE MOTOR CONTROL 0=OFF 1=ON		
BIT4=1 ALWAYS		
BIT5=1 ALWAYS		
BIT6=NOT USED		
BIT7=CD INTERRUPT FLAG		

FF22

BIT0=RS-232 DATA INPUT	
BIT1=SINGLE BIT SOUND OUTPUT	
BIT2=RAM SIZE INPUT	HIGH=16K CHANGEABLE=64K
BIT3=VDG CONTROL OUTPUT	CSS
BIT4=VDG CONTROL OUTPUT	GM0&INT/EXT
BIT5=VDG CONTROL OUTPUT	GM1
BIT6=VDG CONTROL OUTPUT	GM2
BIT7=VDG CONTROL OUTPUT	A/G

FF23

BIT0	CONTROL OF THE CARTRIDGE INTERRUPT INPUT	0=FIRQ TO CPU DISABLED 1=FIRQ TO CPU ENABLE
BIT1		0=SETS FLAG ON FALLING EDGE CART 1=SETS FLAG ON RISING EDGE CART
BIT2=NORMALLY 1:	0=CHANGES FF22 TO THE DATA DIRECTION REGISTER	
BIT3=SIX BIT SOUND ENABLE		
BIT4=1 ALWAYS		
BIT5=1 ALWAYS		
BIT6=NOT USED		
BIT7=CARTRIDGE INTERRUPT FLAG		

FF40-FFBF NOT USED

MAPA DE MEMORIA

FIGURA 14 (c)

DECIMAL	HEX 16B	CONTENTS	DESCRIPTION
0-105	0-69	DIRECT PAGE RAM	CAN BE USED FOR MACHINE-CODE PROGRAMS
112-255	70-FF		CANNOT BE USED FOR MACHINE-CODE PROGRAMS
256-273	100-111	INTERNAL USE	INTERRUPT VECTORS
274-276	112-114	USRJMP	JUMP TO BASIC'S USR ROUTINE
277-281	115-119		CAN BE USED FOR MACHINE-CODE PROGRAMS
282	11A	KEYBOARD ALPHA LOCK	0=NOT LOCKED; FF=LOCKED
283-284	11B-11C	KEYBOARD DELAY CONSTANT	
285-337	11D-151		CAN BE USED BY MACHINE-CODE PROGRAMS
338-345	152-159	KEYBOARD ROLLOVER TABLES	
346-349	15A-15D	JOYSTICK POT VALUES	
350-1023	15E-3FF	INTERNAL USE	
1024-1535	0400-05FF	VIDEO TEXT MEMORY	
1536-TOP OF THE RAM TOP OF THE RAM IS: 16383 FOR 16K SYSTEM 32767 FOR 32K SYSTEM	0600-TOP OF THE RAM TOP OF THE RAM IS: 3FFF FOR 16K SYSTEM 7FFF FOR 32K SYSTEM	IF THE EDITOR ASSEMBLER IS IN CONTROL, IT ALLOCATES THESE RANDOM ACCESS MEMORY ADDRESS IN THIS MANNER:	
		1.-TEMPORARIES	SPACE RESERVED FOR TEMPORARY STORAGE OF EDTASM'S VARIABLES BUFFERS, AND STACKS (THIS CONSUMES HEXADECIMAL 200 BYTES).
		2.-EDIT BUFFER	STORAGE SPACE FOR THE PROGRAM LINES YOU INSERT WITH THE EDITOR.
		3.-SYMBOL TABLE	STORAGE SPACE FOR ALL SYMBOLS IN YOUR PROGRAM AND THEIR CORRESPONDING VALUES.
		4.-OBJECT CODE	STORAGE SPACE FOR YOUR ASSEMBLED PROGRAM.
		IF BASIC IS IN CONTROL, IT ALLOCATES THESE RANDOM ACCESS MEMORY LOCATIONS IN THIS MANNER:	
		1.-GRAPHICS VIDEO MEMORY	SPACE RESERVED FOR GRAPHICS VIDEO PAGES. 6144 BYTES OR 4 PAGES ARE RESERVED FOR THIS ON START-UP. THIS VALUE CAN BE RESET BY THE PCLEAR STATEMENT; NUMBER OF PAGES RESERVED BY PCLEAR X 1,536 BYTES PER PAGE (NOTE: ALL PAGES MUST START AT A 256-BYTE PAGE BOUNDARY--i.e., A MEMORY LOCATION DIVISIBLE BY 256.)
		2.-BASIC PROGRAM STORAGE	SPACE RESERVED FOR BASIC PROGRAMS AND VARIABLES. 6455 BYTES (16K SYSTEM) OR 22,839 BYTES (32K SYSTEM) ARE RESERVED FOR THIS ON START-UP. THIS VALUE CAN BE RESET BY DIFFERENT SETTINGS OF RANDOM FILE BUFFERS, FCBs, GRAPHICS VIDEO MEMORY, STRING SPACE OR USER MEMORY.
		3.-BASIC VARIABLE STORAGE	
		4.-STACK	
		5.-STRING SPACE	TOTAL SPACE FOR STRING DATA. ON START-UP, 200 BYTES ARE RESERVED, BUT THIS CAN BE BY THE CLEAR STATEMENT.
		6.-USER MEMORY	TOTAL SPACE FOR USER MACHINE-LENGUAGE ROUTINES. NO SPACE IS RESERVED FOR THIS ON START-UP, BUT THIS CAN BE RESET BY THE CLEAR STATEMENT.
32768-40959	8000-9FFF	EXTENDED COLOR BASIC ROM	READ ONLY MEMORY
40960-49151	A000-BFFF	COLOR BASIC ROM	READ ONLY MEMORY
49152-57343	C000-DFFF	EDTASM+ ROM	READ ONLY MEMORY
57344-65279	E000-FEFF	UNUSED	
65280-65535	FF00-FFFF	INPUT/OUTPUT	

FIGURA 14 (d)

cierto número de instrucciones en mnemónico. La longitud de una instrucción varía entre 1 y 5 bytes.

Los comandos que existen para el uso del editor assembler son:

RESET o E <ENTER>	Entrada al editor. Aparece el PROMPT "*", indicando que el editor está disponible.
W NOM-ARC <ENTER>	Salva archivo en cassete.
L NOM-ARC <ENTER>	Carga el programa de cassette a memoria.
	LISTA EN PANTALLA:
PN <ENTER>	La línea N.
PN1:N2 <ENTER>	El rango desde la línea N1 hasta la línea N2.

P# <ENTER>	La primera linea.
P* <ENTER>	La última linea.
P. <ENTER>	La siguiente linea.
P#:* <ENTER>	Todo el programa.
P#!N <ENTER>	Del inicio a la linea N. La "!" equivale a ":".
<SHIFT> *	Detiene el listado.
	IMPRIME:
H#:* <ENTER>	Todo el listado.
TN1:N2 <ENTER>	Imprime N2 lineas empezando por la linea N1.
E LINE <ENTER>	Edita la linea.

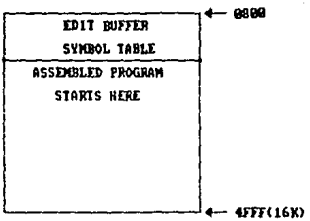
DN1:N2 <ENTER>	Borra desde la línea N1 hasta la línea N2.
IN1,N2 <ENTER>	Inserta a partir de la línea N1 a la línea N2.
<BREAK> I <ENTER>	Para salir del comando de inserción.
NX1,X2 <ENTER>	Renumera desde la línea X1 con incrementos X2.
RN1,N2	Insertar nuevas líneas a partir de N1 con incrementos de N2.
CN1,N2:N3,N4 <ENTER>	Copia de la línea N2 hasta la N3, empezando de la línea N1 con incrementos de N4.
Z <ENTER>	Entra al Zbug. Aparecerá en pantalla prompt "#".

Q <ENTER>	Entras a BASIC.
EXEC 49152 <ENTER>	Regresas de BASIC.
EXEC &HC000 <ENTER>	Regresas de BASIC.

Para ensamblar se tienen los siguientes comandos:

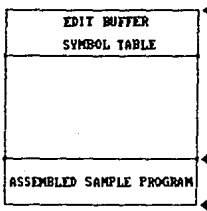
A NOM-ARC <ENTER>	Para ensamblar.
/WE	El ensamblador se detiene cada vez que se encuentra un error.
/SS	Lista una parte pequeña del programa.
/NO	Despliega el programa sin código objeto.
/NS	Despliega el programa sin tabla de símbolos.
/NL	No lista todo el programa.

/LP	Imprime el programa.
/IM	Ensambla el programa en memoria. El ensamblador almacena el programa después de la tabla de símbolos, la que se encuentra debajo del buffer editor (fig. 15).
/AO	Permite determinar la localidad absoluta en memoria del programa (fig. 16).
/MO	Determina manualmente el inicio del programa (fig. 17).
/NO	No almacena el código objeto ni en memoria ni en cinta.



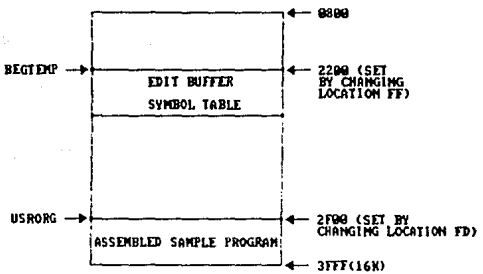
/IM IN MEMORY ASSEMBLY

FIGURA 15



/AO IN MEMORY ASSEMBLY

FIGURA 16



/MO IN MEMORY ASSEMBLY

FIGURA 17

CAPITULO 3

SELECCION DE CIRCUITOS INTEGRADOS

En el desarrollo de este proyecto se realizó una investigación sobre los circuitos integrados utilizados con mayor frecuencia, con la finalidad de iniciar el sistema con los 10 CI más solicitados.

En realidad el número de circuitos es reducido, por lo que, con el fin de dar al usuario mayor posibilidad de usar el sistema, se tiene la opción de dar de alta la información correspondiente del circuito que se desee probar. Esta quedará residente en memoria RAM originando ampliar el sistema de modo que cada vez esté más completo.

Para introducir nuevos circuitos ver manual de usuario.

Los circuitos que se han seleccionados son:

7401	7404	7408	7427
7432	7451	7474	7486
74109	74125	74138	74139
74150	74151	74153	74154
74157	74164	74166	74175

74191	74195	74221	74240
74244	74245	74257	74273
74299	74367	74373	74381
74393	74490	74629	74670

Los circuitos integrados listados anteriormente son los de uso más frecuente, pero debido a la capacidad de almacenamiento del equipo utilizado solamente se introdujeron en EPROM 10 de ellos que son los que aparecen subrayados. La elección de los circuitos se hizo de forma tal que existiera variedad en sus funciones.

7404

HEX INVERTERS

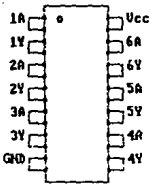


TABLA DE VERDAD

ENTRADAS	SALIDA
A	Y
L	H
H	L

7408

QUADRUPLE 2 INPUT
POSITIVE AND GATES

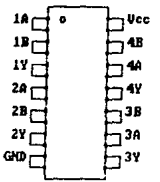


TABLA DE VERDAD

ENTRADAS		SALIDA
A	B	Y
H	H	H
L	X	L
X	L	L

7 4 2 7

TRIPLE 3 INPUT
POSITIVE NOR GATES

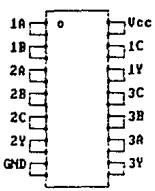


TABLA DE VERDAD

ENTRADAS			SALIDA
C	B	A	Y
L	L	L	H
L	L	H	L
L	H	L	L
L	H	H	L
H	L	L	L
H	L	H	L
H	H	L	L
H	H	H	L

7 4 3 2

QUADRUPLE 2 INPUT
POSITIVE OR GATES

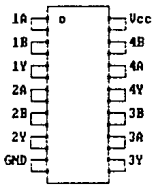


TABLA DE VERDAD

ENTRADAS		SALIDA
A	B	Y
H	X	H
X	H	H
L	L	L

7474

DUAL D-TYPE POSITIVE EDGE TRIGGERED FLIP FLOP WITH PRESET AND CLEAR

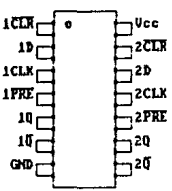


TABLA DE VERDAD

ENTRADAS				SALIDAS	
PRE	CLR	CLK	D	Q	Q̄
L	H	X	X	H	L
H	L	X	X	L	H
L	L	X	X	H*	H*
H	H	↓	H	H	L
H	H	↑	L	L	H
H	H	L	X	Qo	Q̄o

7 4 8 6

QUADRUPLE 2 INPUT
EXCLUSIVE OR GATES

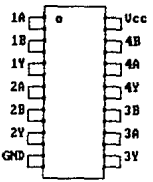


TABLA DE VERDAD

ENTRADAS		SALIDA
A	B	Y
L	L	L
L	H	H
H	L	H
H	H	L

74139

2 TO 4 LINE DECODERS/DEMULTIPLEXERS

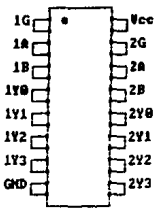


TABLA DE VERDAD

ENTRADAS			SALIDAS			
ENABLE	SELECT					
G	B	A	Y0	Y1	Y2	Y3
H	X	X	H	H	H	H
L	L	L	L	H	H	H
L	L	H	H	L	H	H
L	H	L	H	H	L	H
L	H	H	H	H	H	L

74151

1 TO 8 DATA SELECTORS/MULTIPLEXERS

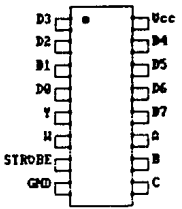


TABLA DE VERDAD

ENTRADAS				SALIDAS	
SELECT			STROBE		
C	B	A	S	Y	W
X	X	X	H	L	H
L	L	L	L	D0	D0
L	L	H	L	D1	D1
L	H	L	L	D2	D2
L	H	H	L	D3	D3
H	L	L	L	D4	D4
H	L	H	L	D5	D5
H	H	L	L	D6	D6
H	H	H	L	D7	D7

74244

OCTAL BUFFERS AND DRIVERS
WITH 3 STATE OUTPUTS

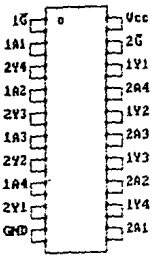


TABLA DE VERDAD

ENTRADAS		SALIDA
G	A	Y
H	X	Z*
L	L	L
L	H	H

* Z=ALTA IMPEDANCIA

7 4 3 9 3

DUAL 4 BIT DECADE AND
BINARY COUNTERS

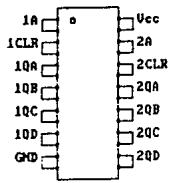


TABLA DE VERDAD

COUNT	SALIDAS			
	QA	QB	QC	QD
0	L	L	L	L
1	L	L	L	H
2	L	L	H	L
3	L	L	H	H
4	L	H	L	L
5	L	H	L	H
6	L	H	H	L
7	L	H	H	H
8	H	L	L	L
9	H	L	L	H
10	H	L	H	L
11	H	L	H	H
12	H	H	L	L
13	H	H	L	H
14	H	H	H	L
15	H	H	H	H

CAPITULO 4

IMPLEMENTACION DEL SOFTWARE

El sistema completo se implementó en la "COCO 3", utilizando sus propios lenguajes (BASIC y ENSAMBLADOR 6809), organizándolo de manera modular para su mejor entendimiento, esto permite que sea transportado facilmente a cualquier otro sistema de cómputo que cuente con puerto paralelo .

El lenguaje en que se implementó no pone restricción a que sea modular, porque la programación se hizo siguiendo los algoritmos que presenta el diseño estructurado.

La ventaja de hacer un sistema en forma modular radica en tener bloques lo más independientemente posible, logrando una mejor visualización cuando se analiza el programa.

Para el desarrollo del sistema se utilizaron algunas técnicas de Ingeniería de Software. Se desarrollaron los siguientes puntos:

1.- Estructura de Software

2.- Diagrama de Flujo de Datos

3.- Descripción de Sistemas y Subsistemas

4.- Diccionario de datos

5.- Carta Estructurada

1.- ESTRUCTURA DE SOFTWARE

Muestra las fases de desarrollo del proyecto. (Ver fig. 18).

2.- DIAGRAMA DE FLUJO DE DATOS

Tiene como finalidad mostrar graficamente la transformación sucesiva de datos a lo largo del proceso (Ver fig. 19).

3. DESCRIPCION DE SISTEMAS Y SUBSISTEMAS.

VERIFICA NUMERO DE CIRCUITO

Verifica si el número de circuito está correcto, considerando que está compuesto por 4 partes.

Ejem.

SN 74LS08 J 00

a) SN = Prefijo.

b) 74LS08 = Descripción única del circuito.

ESTRUCTURA DE SOFTWARE

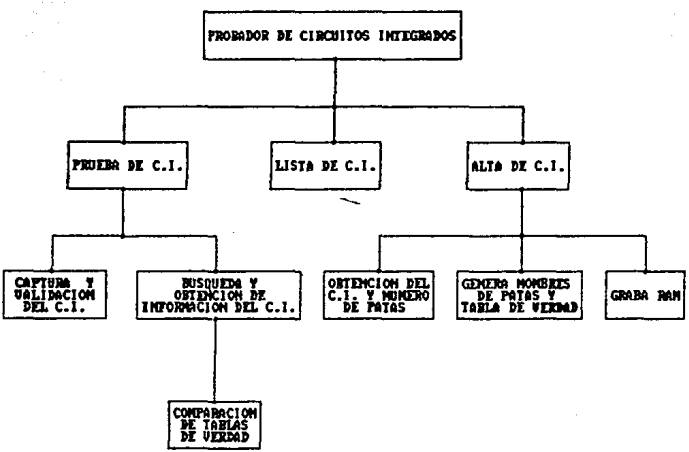


FIGURA 18

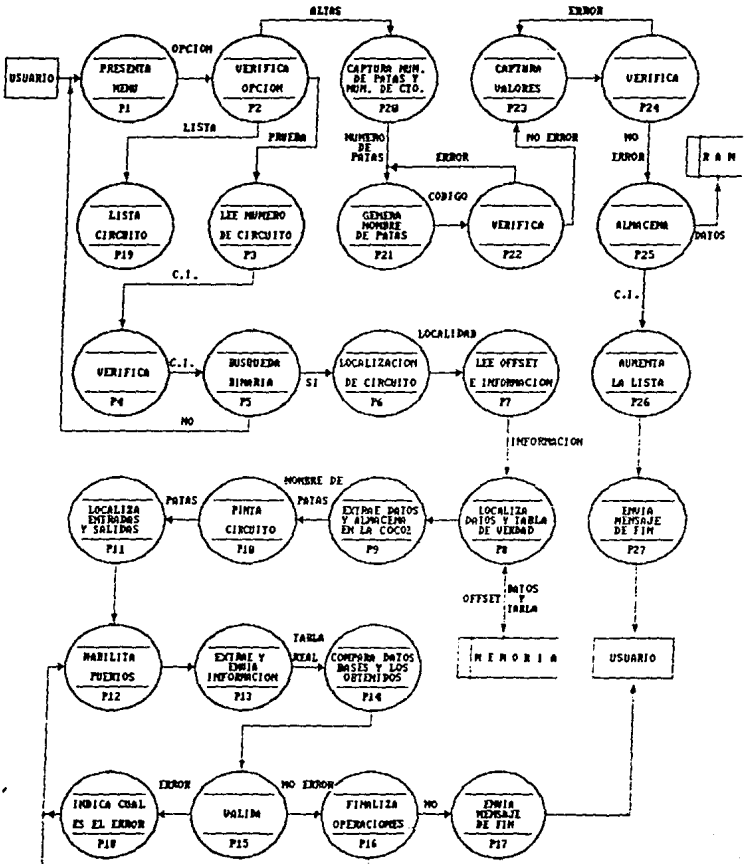

USUARIO
PRESENTA MENU
P1
OPCION
VERIFICA OPCION
P2
ALTAS
CAPTURA NUM. DE PATAS Y NUM. DE CTO.
P20
ERROR
CAPTURA VALORES
P23
VERIFICA
P24
LISTA
PINTA
LISTA CIRCUITO
P19
LEE NUMERO DE CIRCUITO
P3
NUMERO DE PATAS
ERROR
GENERA NOMBRE DE PATAS
P21
CODIGO
VERIFICA
P22
NO ERROR
NO ERROR
RAM
ALMACENA
P25
DATOS
C.I.
C.I.
VERIFICA
P4
C.I.
BUSQUEDA BINARIA
P5
SI
LOCALIZACION DE CIRCUITO
P6
LOCALIDAD
LEE OFFSET E INFORMACION
P7
AUMENTA LA LISTA
P26
NO
INFORMACION
NOMBRE DE PATAS
LOCALIZA ENTRADAS Y SALIDAS
P11
PATAS
PINTA CIRCUITO
P10
PATAS
EXTRAE DATOS Y ALMACENA EN LA COCO2
P9
LOCALIZA DATOS Y TABLA DE VERDAD
P8
ENVIA MENSAJE DE FIN
P27
OFFSET
DATOS Y TABLA
HABILITA PUERTOS
P12
TABLA
EXTRAE Y ENVIA INFORMACION
P13
REAL
COMPARA DATOS BASES Y LOS OBTENIDOS
P14
MEMORIA
USUARIO
ERROR
INDICA CUAL ES EL ERROR
P18
VALIDA
P15
NO ERROR
FINALIZA OPERACIONES
P16
NO
ENVIA MENSAJE DE FIN
P17

FIGURA 19

c) J = Empaquetado.

d) 00 = instrucción.

a) Prefijo.

Compuesto por 2 o 3 letras.

RSN Radiation Hardened Circuit.

SN Standard Prefix.

SNM Mach IV, Level I.

SNC Mach IV, Level III.

SNH Mach IV, Level IV.

SNJ Jan Processor.

b) Descripción única del Circuito.

Contiene de 4 a 8 caracteres, de los cuales, los 2 primeros números indican la familia, estos pueden ser seguidos por 1 o 2 letras: H, S, HC, L o LS o por el número de circuito, este puede constar de 2 a 3 dígitos; a continuación sigue la clase de circuito, el que puede ser A, B, C o ninguna.

c) Empaquetado

Contiene 1 o 2 letras. Los tipos de empaquetado existentes son:

J Ceramic dual-in-line packages.

N Plastic dual-in-line packages.

T Flat package.

W Ceramic flat package.

d) Instruccion.

Contiene 2 números.

Posteriormente se analiza la descripción única del circuito, sin verificar el tipo de prefijo, empaquetado e instrucción .

PRESENTA MENU PRINCIPAL

Este subsistema permite visualizar las diferentes opciones que pueden ser realizadas por el sistema, verificando a su vez que la opcion elegida sea correcta.

Opciones:

Prueba de Circuito

Lista de Circuitos

Altas de Circuitos

BUSQUEDA BINARIA DEL CIRCUITO

Una vez que el nombre del circuito fue verificado,

es verificado en la lista de circuitos integrados que incluye el sistema, por medio de una búsqueda binaria, la que está basada en la bipartición repetida del intervalo de búsqueda.

LOCALIZA OFFSET DEL CIRCUITO

Localiza el offset de la memoria EPROM donde se encuentra almacenada la información (nombre de patas y tabla de verdad), del circuito a probar.

PINTA CIRCUITO INTEGRADO Y ASIGNA NOMBRES A LAS PATAS

Conociendo el número de patas del circuito a probar este es dibujado en pantalla y se le asigna el nombre correspondiente a cada una de sus patas una vez que el código de cada una de éstas ha sido reconocido.

LOCALIZA TABLA EN EPROM

Se localiza la tabla de verdad del circuito a analizar ubicándose en la dirección correspondiente indicada por el offset del circuito.

Se utilizó el código FF como clave para indicar el final de la información de ese circuito.

OBTENCION Y ALMACENAMIENTO DE LA TABLA DE VERDAD

Una vez que es localizada la tabla de verdad en la memoria EPROM, ésta es copiada a una localidad auxiliar (stack) introduciéndola por medio del puerto paralelo a través de la PIA correspondiente, con el fin de disminuir el tiempo de prueba del sistema al evitar consultar continuamente la tabla del circuito desde la EPROM.

LOCALIZA ENTRADAS Y SALIDAS

Identifica cuales son las entradas y salidas del circuito para habilitar las líneas de los puertos de las PIA's según corresponda.

HABILITA PUERTO DE LAS PIA's

Habilita las PIA's U3 y U4 cuando se necesita extraer información de la EPROM. Estas mismas PIA's son habilitadas para leer o escribir en la RAM. LA elección de una u otra memoria, se hace utilizando la línea 6 del puerto B de la PIA U4.

Habilita las PIA's U1 y U2 para mandar y sensar información al circuito a prueba.

SENSADO DE SALIDAS

Una vez aplicados los voltajes de entrada al circuito a prueba, son sensados los voltajes de salida en las patas correspondientes de éste. Los valores son almacenados en el stack para posteriormente ser comparados con los valores teóricos.

PINTA TABLAS DE VERDAD

Despliega los nombres de las entradas y salidas de circuito y posiciona los valores de cada uno de los estados de la tabla de verdad teórica.

COMPARA TABLAS DE VERDAD

Compara cada uno de los estados de la tabla de verdad teórica con los estados de la tabla de verdad obtenida. En caso de que los resultados sean diferentes se despliega un mensaje de error indicando el estado erróneo. Se continúa la prueba unicamente cuando exista otro circuito interno por probar.

LISTA CIRCUITOS DISPONIBLES

En este módulo se listan todos los circuitos disponibles, ya sea que se encuentren almacenados en la EPROM (circuitos iniciales en el sistema) o en la RAM (circuitos que son dados de alta).

CAPTURA NUMERO DE PATAS Y NUMERO DE C.I.

Para llevar a cabo el proceso de alta de un circuito es necesario que el usuario indique el número de patas y número de C.I. que desea almacenar. Este módulo se encarga de obtener y validar los datos antes mencionados.

GENERA NOMBRES DE PATAS

El usuario va generando cada uno de los nombres de las patas del C.I. de acuerdo a los menús que se le presentan.

En el sistema se consideró que cada nombre está compuesto por 4 partes que internamente son representadas por un dígito. Lo que permite reconocer cualquier nombre que sea accesado.

DIGITO 1

Valor	Característica
0	positivo
1	negativo
3	nombre largo (+/-)
4	primo

En caso de que el dígito 1 sea igual a 0 ó 1, el

dígito 2 indicará el caracter anterior al nombre (número de circuito interno). El dígito 3 indica el nombre de la letra o el 3 y 4 indican el código de un nombre corto.

DIGITO 3

(LETRA)

Valor	Letra
3	Q
4	W
5	S
6	K
7	J
8	H
9	G
A-F	A-F

DIGITO 3 Y 4

(NOMBRE CORTO)

Valor	Nombre Corto
A5	CN
A8	RO
A9	Rext/Cext
AA	CLEAR
AB	CLOCK

AC	Cext
AD	CX1
AE	CX2
AF	PRESET

Cuando el nombre está integrado por una letra, éste frecuentemente contiene un caracter posterior, el que es considerado como el dígito 4.

En caso de que el dígito 1 sea igual a 3 ó 4, se trata de un nombre constituído por una palabra o por un nombre compuesto por más de una. En ambos casos el nombre puede ser negativo o positivo o alterno en el caso de Vcc o tierra el que es tomado para el dígito 2.

DIGITO 2

Valor	Característica.
1 ó 2	Número de circuito interno.
3	Alterno.
4	Negado.

Si se trata de un nombre simple, el dígito 3 y 4 contendrá la clave de éste.

DIGITO 3 Y 4

Clave	Nombre
A0	DIR
A1	N.C.
A2	N.U.
A3	Vcc
A4	GND
B0	READ
B1	WRITE
B2	STROBE
B3	CLOCK INHIBIT
B4	QH OUTPUT
B5	STR Y1
B6	STR Y8

Si el nombre es compuesto, es decir, que esté constituído por 2 o más palabras se tiene el siguiente código para cada una de ellas:

DIGITO 3 y/o 4

Valor	Nombre
1	ENABLE
2	PARALLEL INPUT
3	LATCH

4	OUTPUT
5	CONTROL
6	FREQUENCY
7	SELECT
8	RANGE
9	-G
A	SHIFT
B	LOAD
C	RIGHT
D	LEFT
E	SERIAL
F	INPUT

Los nombres compuestos pueden estar precedidos por un caracter que indique el número de circuito interno. Este caracter estará contenido en el dígito 2.

En el caso especial de que el nombre de la pata sea PARALLEL INPUT éste podrá contener un caracter posterior que será almacenado en el dígito 4.

Después de la entrada de cada nombre de pata se muestra el código obtenido para que el usuario verifique si el nombre es o no correcto.

CAPTURA VALORES DE LA TABLA DE VERDAD TEORICA

Captura los valores de las entradas y salidas de cada estado de la tabla de verdad teórica.

GRABA INFORMACION DEL NUEVO C.I.

Una vez que se tiene los códigos de los nombres de las patas y los valores de la tabla de verdad, la información es grabada en la RAM habilitando las PIA's correspondientes.

INCLUYE NUEVO C.I. EN LISTA

Coloca el número de C.I. en la posición correspondiente de la lista, así como las características necesarias para su identificación.

4.- DICCIONARIO DE DATOS

Permite definir de manera completa y formal los datos y flujo de estos.

Los arreglos que fueron utilizados son:

L$ Dimensión (10,5). Contiene la lista en forma creciente de los C.I., disponibles para ser probados, así, como el offset de la memoria EPROM en donde se halla localizada su información (código de patas y tabla de verdad), número de patas, cuantas son entradas y cuantas salidas.

COLUMNA DEL ARREGLO	LONGITUD	CARACTERISTICAS
1	4 - 13	Número de C.I..
2	4	Offset.
3	2	Número de patas.
4	2	Número de entradas.
5	2	Número de salidas.

N$ Dimensión (35). Contiene los nombres de las patas del C.I. que se desplegará en pantalla.

IE Dimensión (6,20). Contiene las patas de entrada del circuito que se está probando. Las patas de entrada de cada uno de los circuitos internos serán almacenados en un renglón diferente del arreglo.

IS Dimensión (6,20). Contiene las patas de salida del circuito que se está probando. Las patas de salida de cada uno de los circuitos internos serán almacenadas en un renglón diferente del arreglo.

Las variables de mayor importancia utilizadas en cada uno de los módulos se listan a continuación:

NOMBRE DE VARIABLE	TIPO	DESCRIPCION
OP	CARACTER	Contiene la opción elegida de los menús principales.
CI	CADENA	Contiene el nombre del C.I. a probar.
LO	ENTERO	Contiene la longitud del nombre del C.I..
OF	CARACTER	Contiene el offset de la EPROM donde se localiza la información del C.I. a prueba.
NP	ENTERO	Número de patas del C.I. a prueba.
NE	ENTERO	Número de entradas de cada circuito interno.
NS	ENTERO	Número de salidas de cada circuito interno.
D1	CARACTER	Primer digito del código del nombre de la pata que se está analizando.

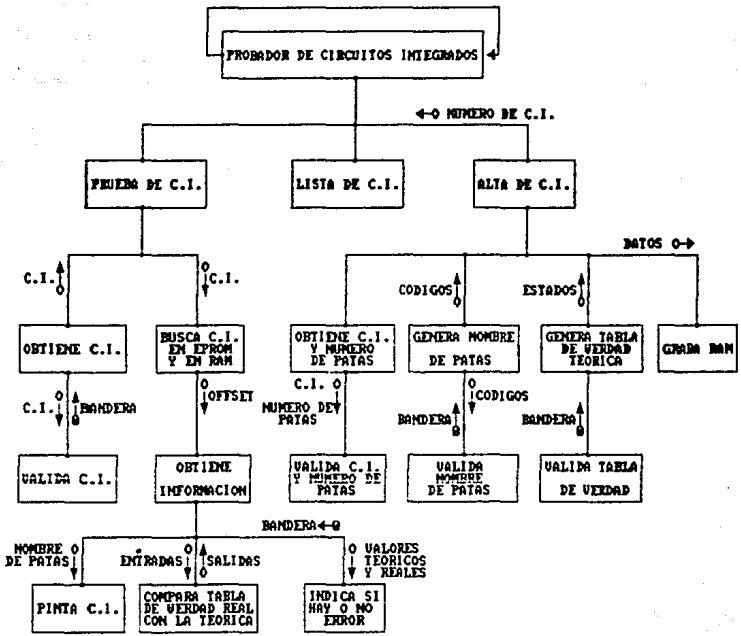
D2	CARACTER	Segundo dígito del código del nombre de la pata que se está analizando.
D3	CARACTER	Tercer dígito del código del nombre de la pata que se está analizando.
D4	CARACTER	Cuarto dígito del código del nombre de la pata que se está analizando.
ET	ENTERO	Número total de entradas del circuito a prueba.
NC	ENTERO	Número de circuitos internos.
ED	ENTERO	Número de estados de la tabla de verdad del C.I. a prueba.

Todas las variables restantes son de uso temporal.

5.- CARTA ESTRUCTURADA

Muestra la partición del sistema en módulos y la relación jerárquica entre estos. Además los flujos de datos y control entre los módulos (Ver fig. 20).

CARTA ESTRUCTURADA



PARAMETROS

DATOS O ESTRUCTURA DE DATOS

VARIABLES DE CONTROL

FIGURA 20

BASIC

```
10 'PROGRAMA PRINCIPAL
   LA FUNCION DEL CLEAR ES PROTEGER CIERTAS  LOCALIDADES
   DE MEMORIA PARA EL MOMENTO EN QUE SE  COMPLEMENTA  EL
   PROGRAMA EN BASIC CON EL PROGRAMA EN ENSAMBLADOR.
20 CLEAR 200,30968:CLOADM"ENSAM"
30 DIM L$(11,5),IE(6,20),IS(6,20),N$(35)
40 CLS:GOSUB 5400                    ;SBR DE INICIALIZACION
                                      DE CI'S DISPONIBLES.
 50 GOSUB120                         ; SBR DE MENU PRINC.


                ANALIZA LA OPCION ELEGIDA

60 'SBR CASE MENU PRINCIPAL
70 IF OP$="P" THEN GOTO 490          ; VA A PROBAR UN CI
80 IF OP$="L" THEN GOSUB 260         ; LISTA LOS CI DISP.
90 IF OP$="A" THEN GOSUB 2410        ; VA A DAR DE ALTA
100 IF OP$="S" THEN CLS:END          ; TERMINA SU ACTIV.
110 GOTO 50

   PRESENTA EL MENU  DE  LAS  OPCIONES  DISPONIBLES.
 120 'MENU1
130 CLS:PRINT @74," M E N U "
140 PRINT @132,"<P> PRUEBA DE CI":PRINT @196,"<L>  LISTA
    DE CI DIPONIBLES":PRINT @260,"<A>  ALTA DE  CI":PRINT
    @324,"<S> FIN DE SESION":PRINT  @396,"OPCION";:INPUT
    OP$
150 IF OP$ <>"P" THEN IF OP$<>"L" THEN IF  OP$<>"A"  THEN
    IF OP$ <>"S" THEN GOSUB 220
160 RETURN


170 'SBR DE ESPERA PARA CONTINUAR UN PROCESO
180 PRINT @453," <ENTER> PARA CONTINUAR"
190 Z$=INKEY$:IF Z$<>"" GOTO 210
200 GOTO 190
210 RETURN


220 'SBR PARA ENVIAR MENSAJE DE DATOS INVALIDOS
230 PRINT @425,"OPCION INVALIDA"
240 GOSUB 170
250 RETURN
```

```
SBR QUE LISTA LOS CI DISPONIBLES EN MEMORIA EPROM  Y  EN
                        RAM.
260 'LISTA DE CI DISPONIBLES
270 GOSUB 300:GOSUB 170
280 GOSUB 350:GOSUB 170
290 RETURN

300 'SBR LISTA DE EPROM
310 CLS:PRINT @70,"CIRCUITOS DISPONIBLES"
320 PRINT @138,"CI EN EPROM"
330 PRINT:PRINT:FOR I=1 TO 5
340 PRINT " ";L$(I,1);" ";L$(I+5,1):NEXT I:RETURN

350 'SBR LISTA DE RAM
360 CLS:PRINT @70,"CIRCUITOS DISPONIBLES"
370 PRINT @138,"CI    EN    RAM":EXEC &H7AFF
380 SS=234:R=242:J=0: NC=PEEK(&H7924):IF  NC=0  OR  NC>8
    THEN GOTO 480
390 FOR I=1 TO NC
400 EXEC(&H7B38):DR=&H7924
410 Q$=STR$(PEEK(DR+1)):L=LEN(Q$):
    Q$=RIGHT$(Q$,L-1):W$=STR$(PEEK(DR+2)): L=LEN(W$):
    W$=RIGHT$(W$,L-1)
420 A$="74"+Q$+W$
430 IF J=0 THEN GOTO 460
440 PRINT @R,A$:R=R+32:J=0
450 GOTO 470
460 PRINT @SS,A$:SS=SS+32:J=1
470 NEXT I
480 RETURN



        SBR DONDE SE RELIZA LA PRUEBA DEL CI.
490 'SBR PRUEBA DE CIRCUITO
500 CLS:PRINT @230," ";:INPUT "NUMERO DE CI";CI$
510 GOSUB 560:YY=0:GOSUB 610:IF YY=0 THEN GOTO 930
520 GOTO 500
530 GOSUB 1410:GOSUB 1510:GOSUB 170
540 GOSUB 1620:GOSUB 1840
550 GOTO 50

560 'SBR INICIALIZA MATRIZ DE E/S.
570 FOR J=1 TO 6:FOR I=1 TO 20
580 IS(J,I)=255:IE(J,I)=255
590 NEXT I:NEXT J
600 RETURN
```

```
610 'SBR CHECA NUMERO DE CI
620 CO=1:LO=LEN(CI$):IF 3<LO AND LO<13 THEN 640
630 GOTO 900
640 L$=MID$(CI$,CO,1):A=ASC(L$)
650 IF (A>47 AND A<58) THEN 680
660 CO=CO+1:IF CO<>LO THEN 640
670 GOTO900
680 CO=CO+1:K$=MID$(CI$,CO,1):NO$=L$+K$
690 IF NOT(NO$="74" OR NO$="54") THEN GOTO 900
700 CO=2:L$=MID$(CI$,CO+1,1):IF  (L$="S"  OR  L$="L"  OR
    L$="H") THEN 730
710 A=ASC(L$):IF A>47 AND A<58 THEN 780
720 GOTO 900
730 CO=CO+1:IF CO>LO THEN 900
740 CO=CO+1:K$=MID$(CI$,CO,1)
750 A=ASC(K$):IF A>47 AND A<58 THEN CO=CO-1:GOTO780
760 IF NOT(A>64 AND A<91) THEN 900
770 N1$=L$+K$:IF NOT(N1$="HC" OR N1$="LS") THEN 900
780 CO=CO+1:IF CO>LO THEN 900
790 L$=MID$(CI$,CO,1):A=ASC(L$)
800 IF NOT(A>47 AND A<58) THEN 900
810 CO=CO+1:IF CO>LO THEN 900
820 K$=MID$(CI$,CO,1):A=ASC(K$)
830 IF NOT(A>47 AND A<58) THEN 900
840 IF CO=LO THEN 870
850 CO=CO+1:IF CO>LO THEN 900
860 J$=MID$(CI$,CO,1):A=ASC(J$)
870 N2$="74"+L$+K$:IF A>47 AND A<58 THEN N2$=N2$+J$
880 L$="":K$="":J$=""
890 RETURN

900 'MENSAJE DE ERROR CUANDO EL NUMERO DE CIRCUITO
    PROPORCIONADO POR EL USUARIO ES INVALIDO
910  CLS:  PRINT  @231,"CIRCUITO  EQUIVOCADO":YY=1:GOSUB
    170:CLS
920 GOTO 890

930 'METODO DE BUSQUEDA.UNA VEZ QUE CHECO QUE EL CI
    EXISTA LOCALIZA SU POSICION EN EPROM O EN RAM.
940 IN=1:SU=10
950 IF IN<=SU THEN GOTO 980 ELSE GOTO 1070
960 EXEC &H7A00:ET=PEEK(&H7924):ST=PEEK(&H7925):
    DS=31014:NC=ET/NE
970 GOSUB 1240:GOTO 530

980 'SBR CHECA EN EPROM
990 ME=INT((IN+SU)/2):N1=VAL(N2$):N3=VAL(L$(ME,1))
1000 IF N1<N3 THEN SU=ME-1:GOTO 950
1010 IF N1>N3 THEN IN=ME+1:GOTO 950
1020 CLS:PRINT @231,"CIRCUITO LOCALIZADO"
```

```
1030 OF=VAL(L$(ME,2)):NP=VAL(L$(ME,3)):NE=VAL(L$(ME,4)):
     NS=VAL(L$(ME,5))
1040 POKE(&H791E),NP:POKE(&H791C),NE:POKE(&H791D),NS:
     POKE(&H7916),0
1050 FO=OF:GOSUB 1210:POKE(&H7914),A2:POKE(&H7915),A3
1060 FO=VAL(L$(ME+1,2)):GOSUB 1210:POKE(&H7919),A2:
     POKE(&H791A),A3:GOTO 960

1070 'SBR CHECA EN RAM
1080 EXEC&H7AFF:      F1=PEEK(&H7925):    F2=PEEK(&H7926):
     DR=PEEK(&H7924)
1090 I=1:NC=PEEK(&H7924):IF NC<1 OR NC>8 GOTO 1140
1100 EXEC(&H7938)
1110 Q$=STR$(PEEK(DR+1)): L=LEN(Q$):  Q$=RIGHT$(Q$,L-1):
     W$=STR$(PEEK(DR+2)): L=LEN(W$): W$=RIGHT$(W$,L-1)
1120 CA$="74"+Q$+W$: IF CA$=CI$ THEN  CLS:  PRINT  @231,
     "CIRCUITO LOCALIZADO": GOSUB 170: GOTO 1150
1130 I=I+1: IF I<=NC THEN GOTO 1100
1140 CLS: PRINT @233, "CIRCUITO  NO  LOCALIZADO":  GOSUB
     170: GOTO 50
1150 UF=PEEK(&H7927):POKE(&H7914),UF:  DF=PEEK(&H79286):
     POKE(&H7915),DF
1160 IF I<NC THEN POKE(&H7919),PEEK(&H792F):
     POKE(&H791A),PEEK(&H7930) ELSE POKE(&H7919),F1:
     POKE(&H791A),F2
1170 SA$=HEX$(F2):  GOSUB  1730:  S7=S5:  SA$=HEX$(F1):
     GOSUB 1780: FO=S7+S6+1
1180 SA$=HEX$(DF): GOSUB 1730: S7=S5: SA=$HEX$(UF):GOSUB
     1780: OF=S7+S6
1190 NP=PEEK(&H7929): NE=PEEK(&H792A):  NS=PEEK(&H792B):
     POKE(&H791E),NP:POKE(&H791C),NE: POKE(&H791D),NS
1200 POKE(&H7916),64:GOTO 960

1210 'SBR DIVIDE OFFSET
1220 FO$=HEX$(FO):   A1=LEN(FO$):      IF  A1<2  THEN
     SA$=LEFT$(FO$,A1-2):  GOSUB   1730:     A2=S5:
     SA$=RIGHT$(FO$,2):GOSUB 1730:A3=S5 ELSE A2=0:A3=FO
1230 RETURN

1240 'SBR CALCULO DE NP+NE+NS
1250 C1=0:C2=0:C3=0:C4=0:C5=0:C6=0:C7=0:C8=0:C=NP+NE+NS:
     IA=0:IB=NP+1
1260 FOR I=1 TO C
1270 AU$=HEX(PEEK(DS))
1280 IF LEN(AU$)>1 THEN D1$=LEFT$(AU$,1) ELSE D1$="0"
1290 D2$=RIGHT$(AU$,1):DS=DS+1
1300 AU$=HEX$(PEEK(DS))
1310 IF LEN(AU$)>1 THEN D3$=LEFT$(AU$,1) ELSE D3$="0"
1320 D4$=RIGHT$(AU$,1):DS=DS+1
1330 IF I<=NP THEN GOSUB 5190
1340 GOSUB 4360
```

```
1350 K=K+1
1360 IF K=9 THEN J=J+1:K=1
1370 N$(I)=D1$+D2$+D3$+D4$
1380 NEXT I
1390 DS=DS-2:POKE(DS),1:DS=DS+1:POKE(DS),0:DS=DS-1
1400 RETURN

1410 'SBR ALMACENA DATAS
1420 POKE(&H7924),FV:POKE(&H7925),PG:POKE(&H7926),255:
     DA=&H7927
1430 FOR I=1 TO NC
1440 FOR J=1 TO NE+1
1450 POKE(DA),IE(I,J):DA=DA+1
1460 NEXT J
1470 FOR J=1 TO NS+1
1480 POKE(DA),IS(I,J):DA=DA+1
1490 NEXT J:NEXT I
1500 RETURN

1510 'SBR IMPRIME CIRCUITO
1520 CLS1:R=44:SS=53:X=1:J=2
1530 FOR JJ=1 TO NP/2
1540 L=LEN(N$(X)):T=R-L
1550 PRINT @T,N$(X);:X=X+1
1560 FOR I=26 TO 38:RESET(I,J):NEXT I:J=J+1:FOR I=25 TO
     39:RESET(I,J):NEXT I:J=J+1
1570 PRINT @SS,N$(X):X=X+1
1580 R=R+32:SS=SS+32
1590 NEXT JJ
1600 SET(27,3,1):FOR I=26 TO 38:RESET(I,J):NEXT I: PRINT
     @440," "
1610 RETURN

1620 'AJUSTE DE NOMBRES.PARA CUANDO PRESENTA LA PRUEBA
     QUE VA REALIZANDO LOS LETREROS QUEDEN CENTRADOS.
1630 FOR I=1 TO NP
1640 Q$=N$(I):K=LEN(Q$)
1650 IF K=1 THEN W$="   "+Q$+"   "
1660 IF K=2 THEN W$="   "+Q$+" "
1670 IF K=3 TEHN W$=" "+Q$+" "
1680 IF K=4 THEN W$=" "+Q$
1690 IF K=5 THEN W$=Q$
1700 N$(I)=W$
1710 NEXT I
1720 RETURN

1730 'CONVERSION DE NUMEROS DECIMALES EN HEXADECIMALES.
1740 S3=ASC(LEFT$(SA$,1)) :   S4=ASC(RIGHT$(SA$)):
     U=LEN(SA$):IF U>1 THEN IF S3>64 THEN S3=S3-55
     ELSE S3=S3-48 ELSE S3=0
1750 IF S4>64 THEN S4=S4-55 ELSE S4=S4-48
1760 S5=S4+S3*16
1770 RETURN
```

```
1780 'SBR CONVERSION 2
1790 GOSUB 1730:S6=S3*4096+S4*256:RETURN

1800 'ALMACENA DS.
1810 SD$=HEX(DS):S1$=LEFT$(SD$,2):S2$=RIGHT$(SD$,2):
     SA$=S1$:GOSUB 1730:POKE(&H7903),S5:SA$=S2$:
     GOSUB 1730:POKE(&H7904),S5:RETURN

1820 'ALMACENA DT. DT ES EL APUNTADOR DE LA TABLA DE
      VERDAD.
1830 SD$=HEX(DT):S1$=LEFT$(SD$,2):S2$=RIGHT$(SD$,2):
     SA$=S1$:GOSUB 1730:POKE(&H7901),S5:SA$=S2$:
     GOSUB 1730:POKE(&H7902),S5:RETURN

1840 'SBR DONDE REALMENTE SE INICIA LA PRUEBA
1850 DT=31012:GOSUB1800:GOSUB1820
1860 NC=ET/NE:ED=VAL(L$(ME+1,2))-OF-C*2)/(NE+NS)-1
1870 EXEC&H7B9A
1880 DT=DT+3:DS=DS+2:TT=DS:GOSUB 1800
1890  CLS:PRINT  @38,"CIRCUITO  INTERNO  #";:PRINT
     @74,"ESTADO #";
1900 GOSUB 2070:GOSUB 2020
1910 FOR ZZ=1 TO NC
1920 Y=1:PRINT @56,ZZ
1930 PRINT @82,Y:Y=Y+1
1940 GOSUB 1820:GOSUB 2160
1950 EXEC&H7D5D
1960 EXEC&H7BF1
1970 EXEC&H7E62
1980 EXEC&H7D67
1990 GOSUB 170
2000 ER=PEEK(&H790B):IF ER=1 THEN PRINT  @85,"ERROR":
     SOUND 6,8:GOSUB170:POKE(&H790B),0:GOTO 2020
2010 IF Y=ED THEN GOTO 1930
2020 DS=TT:GOSUB 1800:DT=DT+NE+2:GOSUB 1820
2030 NEXT ZZ
2040 RETURN

2070 'IMPRESION DE ENTRADAS
2080 T=129:J=1:CN=NE
2090 IF NE>5 THEN CN=5
2100 PRINT @T,"ENTRADAS":T=T+33
2110 S1=T:FOR I=NP+1 TO NP+NE
2120 IF J>CN THEN J=1:S1=S1+64:T=S1
2130 PRINT @T,N$(I);:J=J+1:T=T+6
2140 NEXT I
2150 RETURN

2160 'IMPRIME UNOS Y CEROS
2170 S1=196:J=1:CN=NE
2180 IF NE>5 THEN CN=5
```

```
2190 FOR I=1 TO NE
2200 IF J>CN THEN J=1:S1=S1+64:T=S1
2210 PRINT @T,PEEK(DS)
2220 DS=DS+1:T=T+6:J=J+1
2230 NEXT I
2240 S1=S1+96:T=S1:J=1:CN=NS
2250 IF NS>5 THEN CN=5
2260 FOR I=1 TO NS
2270 IF J>CN THEN J=1
2280 PRINT@S1,PEEK(DS)
2290 DS=DS+1:T=T+6:J=J+1
2300 NEXT I
2310 RETURN

2320 'IMPRESION DE SALIDAS
2330 S1=S1+95:T=S1:J=1:CN=NS
2340 IF NS>5 THEN CN=5
2350 PRINT @T,"SALIDAS":T=T+33
2360 S1=T:FOR I=NP+NE+1 TO C
2370 IF J>CN THEN S1=S1+32:J=1:T=S1
2380 PRINT @T,N$(I):J=J+1:T=T+6
2390 NEXT I
2400 RETURN


        SBR PRINCIPAL PARA ALTAS DE CIRCUITOS.
2410 'SBR A L T A S
2420 CLS:PRINT @229,"ES EL PRIMER CIRCUITO":PRINT  @268,
     "<S/N> ";
2430 PRINT @274,"";:INPUT OP$:IF OP$="S" THEN  CLS:PRINT
     @197,"ESTAS SEGURO DE BORRAR":PRINT @229,"CUALQUIER
     CIRCUITO QUE": PRINT @266,"ESTE EN RAM":PRINT @299,
     "<S/N> ";:INPUT OP$ ELSE IF OP$<>"N" THEN 2430
2440 IF OP$="S" THEN EXEC&H7F93ELSE  IF  OP$<>"N"  THEN
     PRINT @272,"";:INPUT OP$:GOTO 2440
2450 CLS:GOSUB 2530:GOSUB 2570:GOSUB2690
2460 EXEC&H7A81
2470 POKE(&H7927),PEEK(&H791F):POKE(&H7928),PEEK(&H7920)
2490 EXEC&H7A36
2500 GOSUB 2780:GOSUB 2710
2510 EXEC&H7AF2:EXEC&H7A14:EXEC&H7A48
2520 RETURN

2530 'SBR MEMORIA RAM LIBRE
2540 EXEC&H7AFF:IF PEEK(&H7924)=255 THEN FI=0 ELSE
     SA$=HEX$(PEEK(&h7926)):GOSUB 1730:S7=S5:
     SA$=HEX$(PEEK(&h7925)):GOSUB1780:FI=S6+S7
2550 ME=4016-FI:CLS:PRINT @97,"MEMORIA LIBRE EN RAM
     : ":PRINT @148,ME;" BYTES"
2560 GOSUB170:RETURN
```

```
2570 'SBR INFORM. DE CI NUEVO
2580 CLS:PRINT @163,"SI INTRODUCES UN CIRCUITO":
     PRINT @195,"PRESENTADO EN LISTA, ESTE ":
     PRINT @232,"SERA IGNORADO.":GOSUB170:CLS
2590 PRINT @230,"NUMERO DE CI";:INPUT CI$
2600 YY=0:GOSUB 610:IF YY=1 THEN GOTO 2590
2610 A2=LEN(CI$):C1=VAL(LEFT$(CI$,2)):
     C2=VAL(MID$(CI$,3,1)):C3$=VAL(RIGHT$(CI$,A2-2))
2620 CLS:PRINT @229,"NUMERO DE PATAS DEL CI":PRINT @263,
     "(14,16,20,24)";:INPUT NP$
2630 NP=VAL(NP$):IF (NP<>14) AND (NP<>16) AND (NP<>20)
     AND (NP<>24) THEN GOSUB 220:GOTO 2620
2640 CLS:PRINT @226,"NUMERO DE ENTRADAS TOTALES";:INPUT
     NE$:NE=VAL(NE$):IF NE=0 THEN 2640
2650 CLS:PRINT @227,"NUM. DE SALIDAS TOTALES";:INPUT NS$
     :NS=VAL(NS$):IF NS=0 THEN 2650
2660 IF NS+NE>NP-2 THEN CLS:PRINT @197,"EXCESO DE NUM.
     DE PATAS":PRINT @296,"NO INCLUIR PATAS":PRINT @328,
     "DE POLARIZACION.":GOSUB 170:GOTO 2640 ELSE IF
     NS+NE<NP-2 THEN CLS:PRINT @228,"NUM. DE PATAS
     INCOMPLETO":GOSUB 170:GOTO 2640
2670  CLS:PRINT  @229,"NUM. DE CI INTERNOS";:INPUT
     NC$:NC=VAL(NC$):IF NC=0 OR NC>6 THEN GOTO 2660
2680 E=NE/NC:S=NS/NC:RETURN

2690 'SBR POKES
2700 POKE(&H7924),C1:POKE(&H7925),C2:POKE(&H7926),C3:
     POKE(&H7929),NP:POKE(&H792A),NE:POKE(&H792B),NS:
     POKE(&H792C),255:POKE(&H791C),E:POKE(&H791F),S:
     K=&h7924:RETURN

2710 'SBR TABLA DE VERDAD
2720 CLS:PRINT @135,"INTRODUCE TABLA DE":PRINT @166,
     "VERDAD POR RENGLONES"
2730 FOR J=1 TO E+S
2740 INPUT VL$:IF VL$<>"0" AND VL$<>"1" THEN PRINT
     "VALOR INCORRECTO":GOTO 2740
2750 VL=VLA(VL$):POKE(K),VL:K=K+1:NEXT J
2760 PRINT "EXISTEN MAS ESTADOS <S/N> ";:INPUT R$="S"
     THEN GOTO 2730 ELSE IF R$<>"N" THEN 2760
2770 POKE(K),255:RETURN

2780 'SBR ALTA DE PATAS
2790 POKE(K);ET:POKE(K+1),ST:K=K+2
2800 CLS:PRINT @199,"INDICA LOS NOMBRES":PRINT @234,"DE
     LAS PATAS":GOSUB 170:IA=0:IB=NP+1
2810 FOR I=1 TO NP
2820 IC=INT(I/2):IC=I/2-IC
2830 IF IC<>0 THEN GOSUB 2990 ELSE GOSUB 3010
2840 GOSUB 3030:N$(I)=D1$+D2$+D3$+D4$:GOSUB 2940
```

```
2850 POKE(K),N1:POKE(K+1),N2:K=K+2
2860 NEXT I
2870 GOSUB 1510:GOSUB 170
2880 CLS:PRINT @199,"INDICA ENTRADAS":GOSUB 170:CLS
2890  FOR  J=1  TO  NE+NS:IF  J<=NE  THEN  CLS:PRINT
     @201,"ENTRADA  ";J  ELSE  JJ=J-NE:CLS:PRINT  @201,"
     SALIDA ";JJ
2900 GOSUB 170:CLS
2910 GOSUB 3060:I=J:I=J:GOSUB 2940
2920 POKE(K),N1:POKE(K+1),N2:K=K+2:NEXT J
2930 RETURN

2940 'SBR CHECO NOM. PATAS
2950 SA$=D1$+D2$:GOSUB 1730:N1=S5:SA$=D3$+D4$:GOSUB 1730
     :N2=S5
2960 N$(I)=D1$+D2$+D3$+D4$:GOSUB 4360:N$(I)=D1$+D2$+D3$+
     D4$
2970 CLS:PRINT @195,"NOMBRE OBTENIDO: ":PRINT @242,N$(I)
     :PRINT @296,"CORRECTO  <S/N>";:INPUT  R$:IF  R$="N"
     THEN GOSUB 3060:GOTO 2950 ELSE IF R$<>"S" THEN GOTO
     2970
2980 RETURN

2990 'SBR PATAS INICIALES
3000 IA=IA+1:PT=IA:RETURN

3010 'SBR PATAS FINALES
3020 IB=IB-1:PT=IB:RETURN

3030 'SBR ALTA DE NOM. DE PATAS
3040 CLS:PRINT @237,"PATA # ";PT:GOSUB 180
3050 GOSUB 3060:RETURN

3060 'SBR ALTA DE NOM. PATAS 2
3070 GOSUB 3120:D1=VAL(D1$)
3080 IF D1=1 THEN GOSUB  3190:D2=VAL(D2$):IF  D2=1  THEN
     D1$="0" ELSE D1$="1"
3090 IF D1<>1 THEN D1$="3"
3100 IF D1$<>"3" THEN GOSUB 3250 ELSE GOSUB 3290
3110 RETURN

3120 'SBR MENU DESCRIPCION
3130 CLS:PRINT @75,"DESCRIPCION"
3140 PRINT  @130,"<1>  PRECEDIDOS  Y/O  SEGUIDOS":PRINT
     @166,"DE LETRA O NUMERO."
3150 PRINT @226,"<2> NOMBRES CORTOS."
3160 PRINT @290,"<3> NOMBRES COMPUESTOS POR":PRINT @326,
     "DOS PALABRAS."
3170 PRINT @427,"OPCION ";:INPUT  D1$:IF  VAL(D1$)<1  OR
     VAL(D1$)>3 THEN GOSUB 230:GOTO 3030
3180 RETURN
```

```
3190 'SBR MENU POLARIDAD
3200 CLS:PRINT @139,"POLARIDAD"
3210 PRINT @201,"<1> POSITIVO"
3220 PRINT @265,"<2> NEGATIVO"
3230 PRINT @427,"OPCION ";:INPUT  O2$:IF  VAL(O2$)<1  OR
     VAL(O2$)>2 THEN GOSUB 220:GOTO 3200
3240 RETURN

3250 'SBR D1=1 OR D1=0
3260 GOSUB 3240:GOSUB 3390
3270 GOSUB 3470:GOSUB 3520
3280 RETURN

3290 'SBR NOM. CORTOS
3300 IF D1=2  THEN  GOSUB  3920:GOSUB  4030  ELSE  GOSUB
     4110:GOSUB 4260
3310 RETURN

3320 'SBR CARACTER PRECEDENTE
3330 CLS:PRINT @229,"CARACTER ANTECESOR";
3340 INPUT D2$
3350 IF D2$="" THEN D2$="1"
3360 IF  (VAL(D2$)<1)  OR  (VAL(D2$)>8)  THEN  IF  (ASC
     (D2$)<65) OR (ASC(D2$)>72) THEN GOSUB 220:GOTO 3330
3370 CLS:IF D2$="" THEN D2$="1"  ELSE  IF  D2$="G"  THEN
     D2$="9" ELSE IF D2$="H" THEN D2$="8"
3380 RETURN

3390 'SBR CARACTER SUCESOR
3400 CLS:PRINT @229,"CARACTER SUCESOR";
3410 INPUT D4$:CLS
3420 IF D4$="" THEN D4$="1"
3430 IF (VAL(D4$)>9) AND (VAL(D4$)<16) THEN D4$=HEX$(VAL
     (D4$))
3440 IF VAL(D4$)<1 OR VAL(D4$)>9 THEN  IF  (ASC(D4$)>73)
     THEN IF D4$<>"0" THEN GOSUB 220:GOTO 3400
3450 IF  D4$="H"  THEN  D4$="8"  ELSE  IF  D4$="G"  THEN
     D4$="9"
3460 RETURN

3470 'SBR NOMBRES CORTOS
3480 CLS:PRINT @105,"TIPO DE NOMBRE"
3490 PRINT  @200,"<1>  NOMBRE  CORTO  ":PRINT  @264,"<2>
     NOMBRE DE LETRA"
3500 PRINT @426,"OPCION ";:INPUT  OP$:IF  VAL(OP$)<1  OR
     VAL(OP$)>2 THEN GOSUB 220:GOTO 3480
3510 RETURN

3520 'SBR DIGITO 3
3530 OP=VAL(OP$)
3540 ON OP GOSUB 3560,3570
```

```
3550 GOTO 3580
3560 GOSUB 3590:RETURN
3570 GOSUB 3730:RETURN
3580 CLS:RETURN

3590 'SBR MENU DIGITO 3 Y 4
3600 CLS:PRINT @73"NOMBRES CORTOS"
3610 PRINT @137,"<1> CN"
3620 PRINT @169,"<2> RO"
3630 PRINT @201,"<3> REXT/CEXT"
3640 PRINT @233,"<4> CLEAR"
3650 PRINT @265,"<5> CLOCK"
3660 PRINT @297,"<6> CEXT"
3670 PRINT @329,"<7> CX1"
3680 PRINT @361,"<8> CX2"
3690 PRINT @393,"<9> PRESET"
3700 PRINT @458,"OPCION ";:INPUT  OP$:IF  VAL(OP$)<1  OR
     VAL(OP$)>9 THEN GOSUB 220:GOTO 3600
3710 D3$="A":D4$=HEX$(VAL(OP$)+6)
3720 RETURN

3730 'SBR LETRAS
3740 CLS:PRINT @77,"LETRAS"
3750 PRINT @137,"<1> Y     <8> G"
3760 PRINT @169,"<2> Q     <9> A"
3770 PRINT @201,"<3> W     <10> B"
3780 PRINT @233,"<4> S     <11> C"
3790 PRINT @265,"<5> P     <12> D"
3800 PRINT @297,"<6> J     <13> E"
3810 PRINT @329,"<7> H     <14> F"
3820 PRINT @429,"OPCION ";:INPUT  OP$:IF  VAL(OP$)<1  OR
     VAL(OP$)>14 THEN GOSUB 220:GOTO 3740
3830 OP=VAL(OP$)-1:IF D4$="I" THEN D3$="0":IF OP>7  THEN
     D4$=HEX$(OP+2) ELSE IF OP<>0  THEN  D4$=RIGHT$(STR$
     (OP+2),1):GOTO 3860 ELSE D4$="1":GOTO 3860
3840 IF OP>7 THEN D3$=HEX$(OP+2) ELSE IF OP<>0 THEN D3$=
     RIGHT$(STR$(OP+2),1) ELSE D3$="1"
3850 IF OP>7 AND D3$=D4$ THEN D3$="0"
3860 RETURN

3870 'SBR CASE ALTERNO O NEGADO
3880 PRINT @426,"OPCION";:INPUT OP2$
3890 IF VAL(OP2$)<1 OR VAL(OP2$)>2 THEN IF OP2$<>"" THEN
     GOSUB 220:GOTO 3880
3900 IF OP2$="" THEN  D2$="0"  ELSE  D2$=RIGHT$(STR$(VAL
     (OP2$)+2),1)
3910 RETURN

3920 'SBR MENU N.L. 1 PALABRA
3930 CLS:PRINT @73,"NOMBRES CORTOS"
3940 PRINT @131," 1  DIR          <7> WRITE"
```

```
3950 PRINT @163,"<2> NC          <8> STROBE"
3960 PRINT @195,"<3> NU          <9> CK INH"
3970 PRINT @227,"<4> Vcc         <10> QH OUT"
3980 PRINT @259,"<5> GND         <11> STR Y1"
3990 PRINT @291,"<6> READ        <12> STR Y8"
4000 PRINT @426,"OPCION ";:INPUT OP$
4010 IF VAL(OP$)<1 OR VAL(OP$)>12 THEN IF  OP$<>""  THEN
     GOSUB 220:GOTO 3930
4020 RETURN

4030 'SBR CASE DE NOM. CORTOS
4040 OP=VAL(OP$):IF OP<6 THEN OA=1 ELSE OA=2
4050 D2$="0":IF OP=4 OR  OP=5  THEN  CLS:PRINT  @229,"ES
     ALTERNO <S/N>";:INPUT AL$:IF AL$="S"  THEN  D2$="3"
     ELSE IF AL$<>"N" THEN GOSUB 220:GOTO 4050
4060 ON OA GOSUB 4080,4090
4070 GOTO 4100
4080 D3$="A":D4$=RIGHT$(STR$(VAL(OP$)-1),1):RETURN
4090 D3$="B":D4$=RIGHT$(STR$(VAL(OP$)-6),1):RETURN
4100 RETURN

4110 'SBR MENU NOMBRES DOBLES
4120 CLS:PRINT @73,"NOMBRES DOBLES"
4130 PRINT @131,"<1> ENABLE          <9> -G"
4140 PRINT @163,"<2> PLL. INPUT      <10> SHIFT"
4150 PRINT @195,"<3> LATCH           <11> LOAD"
4160 PRINT @227,"<4> OUTPUT          <12> RIGHT"
4170 PRINT @259,"<5> CONTROL         <13> LEFT"
4180 PRINT @291,"<6> FRECUENCY       <14> SERIAL"
4190 PRINT @323,"<7> SELECT          <15> INPUT"
4200 PRINT @355,"<8> RANGE"
4210 PRINT @426,"OPCION 1 ";:INPUT D3$
4220 PRINT @458,"OPCION 2 ";:INPUT D4$
4230 IF VAL(D3$)<1 OR VAL(D3$)>15 THEN GOSUB  220:  GOTO
     4120
4240 IF VAL(D4$)<1 OR VAL(D4$)>15 THEN IF  D4$<>""  THEN
     GOSUB 220:GOTO 4120
4250 RETURN

4260 'SBR CASE NOMBRES DOBLES
4270 IF D4$="" THEN D4$=D3$:D3$="0"
4280 D2$="0":D3$=HEX$(VAL(D3$)):D4$=HEX$(VAL(D4$))
4290 IF D4$="9" OR D3$="9" THEN D2$="4"
4300 IF D3$="2" THEN GOSUB 4320
4310 RETURN

4320 'SBR LETRA DE PLL. INPUT
4330 CLS:PRINT @226,"LETRA POSTERIOR (A-F.";:INPUT D4$
4340 IF  ASC(D4$)<65  OR  ASC(D4$)>70  THEN  GOSUB  220:
     GOTO 4330
4350 RETURN
```

```
4360 'SBR PARA TRADUCIR EL CODIGO DEL NOMBRE DE LOS
     PINES.
4370 IF D1$="0" THEN D1$="":GOTO4410
4380 IF D1$="1" THEN D1$="-":GOTO4410
4390 IF D1$="3" THEN D1$="":GOTO4800
4400 RETURN

4410 IF D2$="8" AND D1$="3" THEN D2$="H"
4420 IF D2$="9" THEN D2$="G"
4430 IF D3$="0" THEN D3$="":GOTO4640
4440 IF D3$="1" THEN D3$="Y":GOTO4730
4450 IF D3$="3" THEN D3$="Q":GOTO4640
4460 IF D3$="4" THEN D3$="W":GOTO4400
4470 IF D3$="5" THEN D3$="S":GOTO4400
4480 IF D3$="6" THEN D3$="K":GOTO4400
4490 IF D3$="7" THEN D3$="J":GOTO4400
4500 IF D3$="8" THEN D3$="H":GOTO4400
4510 IF D3$="9" THEN D3$="G":GOTO4400
4520 IF  D3$="A"  AND  VAL(D4$)<7  AND  VAL(D4$)>0  THEN
     GOTO4730
4530 IF D3$="A" THEN D3$="" ELSE GOTO4400
4540 IF D4$="A" THEN D4$="CLR"
4550 IF D4$="B" THEN D4$="CK"
4560 IF D4$="C" THEN D4$="CEXT"
4570 IF D4$="D" THEN D4$="CX1"
4580 IF D4$="E" THEN D4$="CX2"
4590 IF D4$="F" THEN D4$="PRE"
4600 IF D4$="9" THEN D4$="REXT/CEXT"
4610 IF D4$="8" THEN D4$="R0"
4620 IF D4$="7" THEN D4$="CN"
4630 GOTO 4400
4640 IF D4$="1" THEN D4$="Y"
4650 IF D4$="3" THEN D4$="Q"
4660 IF D4$="4" THEN D4$="W"
4670 IF D4$="5" THEN D4$="S"
4680 IF D4$="6" THEN D4$="K"
4690 IF D4$="7" THEN D4$="J"
4700 IF D4$="8" THEN D4$="H"
4710 IF D4$="9" THEN D4$="G"
4720 GOTO4400
4730 IF D4$="A" THEN D4$="10"
4740 IF D4$="B" THEN D4$="11"
4750 IF D4$="C" THEN D4$="12"
4760 IF D4$="D" THEN D4$="13"
4770 IF D4$="E" THEN D4$="14"
4780 IF D4$="F" THEN D4$="15"
4790 GOTO4400
4800 IF D2$="0" THEN D2$=""
4810 IF D2$="3" THEN D2$="ALT "
4820 IF D2$="4" THEN D2$="-"
4830 IF D3$="A" THEN D3$="":GOTO4960
```

```
4840 IF D3$="B" THEN D3$="":GOTO4880
4850  GOSUB5050:DA$=D3$:D3$=D4$: IF    DA$="P    IN"    THEN
     GOTO4870
4860 GOSUB5050
4870 D4$=D3$:D3$=DA$:GOTO5040
4880 IF D4$="0" THEN D4$="R"
4890 IF D4$="1" THEN D4$="W"
4900 IF D4$="2" THEN D4$="STR"
4910 IF D4$="3" THEN D4$="CK IN"
4920 IF D4$="4" THEN D4$="QH OU"
4930 IF D4$="5" THEN D4$="ST YI"
4940 IF D4$="6" THEN D4$="ST Y8"
4950 GOTO4400
4960 IF D4$="0" THEN D4$="DIR"
4970 IF D4$="1" THEN D4$="NC"
4980 IF D4$="2" THEN D4$="NU"
4990 IF D4$="3" THEN D4$="VCC"
5000 IF D4$="4" THEN D4$="GND"
5010 IF D4$="B" THEN D3$="SH ":D4$="L"
5020 IF D4$="C" THEN D3$="SH ":D4$="RI"
5030 IF D4$="D" THEN D3$="SH ":D4$="LE"
5040 GOTO4400

5050 'PALABRAS COMPUESTAS
5060 IF D3$="0" THEN D3$=""
5070 IF D3$="1" THEN D3$="E "
5080 IF D3$="2" THEN D3$="P IN"
5090 IF D3$="3" THEN D3$="LCH "
5100 IF D3$="4" THEN D3$="OUT "
5110 IF D3$="5" THEN D3$="CTR "
5120 IF D3$="6" THEN D3$="FR "
5130 IF D3$="7" THEN D3$="SEL "
5140 IF D3$="8" THEN D3$="RGE "
5150 IF D3$="9" THEN D3$="G "
5160 IF D3$="E" THEN D3$="SER "
5170 IF D3$="F" THEN D3$="IN "
5180 RETURN

5190 'SBR IDENTIFICA E/S
5200 IC=INT(I/2):IC=I/2-IC:IF IC<>0 THEN GOSUB 2990 ELSE
     GOSUB 3010
5210 IF D3$+D4$="01"  OR  D3$+D4$="03"  OR  D3$+D4$="B7"
     OR D3$+D4$=04 OR  D3$="4"  OR  D3$="3"  OR  D3$="1"
     THEN  ES$="S"  ELSE ES$="E".
5220 PO$=D1$+D3$+D4$
5230 IF PO$="3A3" THEN PV=PT:GOTO 5270 ELSE  IF
     PO$="3A4"  THEN  PG=PT:GOTO5270
5240 IF PO$="0AB" THEN PC=PT:ES$="E"
5250 IF ES$="S" THEN GOSUB 5280
5260 IF ES$="E" THEN GOSUB 5340
5270 RETURN
```

```
5280  SBR IDENT. SALIDAS
5290 IF D2$="2" THEN C1=C1+1:CC=C1 ELSE IF  D2$"3"  THEN
     C2=C2+1:CC=C2 ELSE IF  D2$="4"  THEN  C3=C3+1:CC=C3
     ELSE C4=C4+1:CC=C4
5300 R=VAL(D2$)
5310 IF R=0 THEN R=1
5320 IS(R,CC)=PT
5330 RETURN

5340 'SBR IDENT. ENTRADAS
5350 IF D2$="2" THEN C5=C5+1:CC=C5 ELSE IF D2$="3"  THEN
     C6=C6+1:CC=C6 ELSE IF  D2$="4"  THEN  C7=C7+1:CC=C7
     ELSE C8=C8+1:CC=C8
5360 R=VAL(D2$)
5370 IF R=0 THNE R=1
5380 IE(R,CC)=PT
5390 RETURN

5400 'SBR ASIGNACION DE MEM.
5410 L$(1,1)="7405":L$(1,2)="00":L$(1,3)="14":
     L$(1,4)="1":L$(1,5)="1"
5420 L$(2,1)="7408":L$(2,2)="39":L$(2,3)="14":
     L$(2,4)="2":L$(2,5)="1"
5430 L$(3,1)="7427":L$(3,2)="88":L$(3,3)="14":
     L$(3,4)="2":L$(3,5)="1"
5440 L$(4,1)="7432":L$(4,2)="137":L$(4,3)="14":
     L$(4,4)="4":L$(4,5)="2"
5450 L$(5,1)="7474":L$(4,2)="204":L$(5,3)="14":
     L$(5,4)="4":L$(5,5)="2"
5460 L$(6,1)="7486":L$(6,2)="283":L$(6,3)="14":
     L$(6,4)="2":L$(6,5)="1"
5470 L$(7,1)="74139":L$(7,2)="332":L$(7,3)="16":
     L$(7,4)="3":L$(7,5)="4"
5480 L$(8,1)="74151":L$(8,2)="416":L$(8,3)="16":
     L$(8,4)="12":L$(8,5)="2"
5490 L$(9,1)="74244":L$(9,2)="605":L$(9,3)="20":
     L$(9,4)="5":L$(9,5)="4"
5500 L$(10,1)="74393":L$(10,2)="693":L$(10,3)="14":
     L$(10,4)="2":L$(10,5)="4"
5510 RETURN
```

ENSAMBLADOR.

```
                ORG     $7A00
PIAS
PS              EQU     $78F7
VAR4            EQU     $78F8
DT              EQU     $7901
DS              EQU     $7903
VR              EQU     $7905
C1              EQU     $7906
C2              EQU     $7907
ROT             EQU     $7908
BAND            EQU     $790A
MAL             EQU     $790B
ES1             EQU     $790C
ES2             EQU     $790D
ES3             EQU     $790E
ES4             EQU     $790F
DATO1           EQU     $7910
DATO2           EQU     $7911
DATO3           EQU     $7912
DATO4           EQU     $7913
DIR1            EQU     $7914
DIR2            EQU     $7915
MEM             EQU     $7916
DATO            EQU     $7917
RW              EQU     $7918
DIR3            EQU     $7919
DIR4            EQU     $791A
CCM             EQU     $791B
NPS             EQU     $791D
NP              EQU     $791E
AU1             EQU     $791F
AU2             EQU     $7920
AUX2            EQU     $7921
AUX3            EQU     $7922
AUX1            EQU     $7923
STACK           EQU     $7924
DDR1A           EQU     $FF40
CR1A            EQU     $FF41
DDR1B           EQU     $FF42
CR1B            EQU     $FF43
DDR2A           EQU     $FF44
CR2A            EQU     $FF45
DDR2B           EQU     $FF46
CR2B            EQU     $FF47
DDR3A           EQU     $FF48
CR3A            EQU     $FF49
DDR3B           EQU     $FF4A
```

```
CR3B    EQU     $FF4B
DDR4A   EQU     $FF4C
CR4A    EQU     $FF4D
DDR4B   EQU     $FF4E
CR4B    EQU     $FF4F

LEPROM
        LDY     #STACK
REGRE   LBSR    DIRMEM
        LDA     DIR1
        LDB     DIR2
        CMPD    DIR3
        BLE     REGRE
        RTS

ESRAM
        LDA     #$1
        STA     RW
        LDY     #STACK
        LBSR    P485
REGRE1  LDA     ,Y+
        STA     DATO
        CMPA    #$FF
        BEQ     FINRAM
        LBSR    PCB4A
        LBSR    PCAS
        LBSR    INCREM
        LBSR    REGRE1
FINRAM  RTS

LISTA
        LDA     #$09
        LDB     CCM
        MUL
        ADDB    #$3
        STB     DIR2
        CLR     DIR1
        LBSR    ESRAM
        RTS

ACTUA
        LDA     DIR1
        STA     DIR3
        LDA     DIR2
        STA     DIR4
        CLR     DIR1
        CLR     DIR2
        LDA     CCM
        INCA
        STA     DATO
```

```
                LBSR    GRABA
                LDA     DIR3
                STA     DATO
                LBSR    GRABA
                LDA     DIR4
                STA     DATO
                LBSR    GRABA
                RTS

GRABA
                LBSR    PSB4A
                LBSR    PSAS
                LBSR    INCREM
                RTS

NUEVO
                LDA     #$40
                STA     MEM
                LDA     #$2
                STA     RW
                CLR     DIR1
                CLR     DIR2
                LBSR    P4BS
REGRE2          LBSR    PSB4A
                LBSR    PSAE
                LDB     DATO
                LDA     DIR2
                CMPA    #$00
                BEQ     CCM1
                CMPA    #$01
                BEQ     OFF1
                STB     AU2
                LBRA    CONTI
OFF1            STB     AU1
                LBRA    SIGUE
CCM1            STB     CCM
                LDA     #$FF
                CMPA    CCM
                BNE     SIGUE
                CLR     CCM
                CLR     AU1
                LDA     #$50
                STA     AU2
                LBRA    CONTI
SIGUE           LBSR    INCREM
                LBRA    REGRE2
CONTI           RTS
```

```
DIRMEM
            LBSR        P3B4A
            LBSR        P3AE
            LBSR        INCREM
            LDA         DATO
            STA         ,Y+
            RTS

INCREM
            LDA         DIR2
            LDB         DIR2
            ADDD        #$01
            STA         DIR1
            STB         DIR2
FIN         RTS

ASIGNA      LDA         AU1
            LDB         AU2
            STA         DIR1
            STB         DIR2
            RTS

LIS         LDA         #$2
            STA         RW
            LDA         #$40
            STA         MEM
            CLR         DIR1
            CLR         DIR2
            LBSR        F4BS
            LDY         #STACK
            CLR         AU1
LLIS        LBSR        P3B4A
            LBSR        P3AE
            LDA         DATO
            STA         ,Y+
            LBSR        INCREM
            LDA         AU1
            INCA
            STA         AU1
            CMPA        #$3
            BLT         LLIS
            LDA         #$2
            STA         DIR2
            RTS


LIS1        LDY         #STACK
            LDA         #$1
            STA         AU1
```

```
INLIS   LBSR    INCREM
        LBSR    P3B4A
        LBSR    P3AE
        LDA     DATO
        STA     ,Y+
        LDA     AU1
        INCA
        STA     AU1
        CMPA    #$0D
        BLE     INLIS
        LDA     DIR1
        LDB     DIR2
        SUBD    #$5
        STA     DIR1
        STB     DIR2
        RTS

SACOF   LDY     #STACK
        LDA     #$0
        STA     AUXI
        LDA     #$2
        STA     RW
        LDA     #$40
        STA     MEM
        LBSR    P4BS
SAC1    LBSR    P3B4A
        LBSR    P3AE
        LBSR    INCREM
        LDA     DATO
        STA     ,Y+
        LDA     AUXI
        INCA
        STA     AUXI
        CMPA    #$4
        BLE     SAC1
        RTS

POLAR
        LDY     DS
        LDX     DT
        LDB     #$00
        STB     ES1
        STB     ES2
        STB     ES3
        STB     ES4
        STB     DATO1
        STB     DATO2
        STB     DATO3
        STB     DATO4
        LDA     NP
        CMPA    #$10
```

```
                BLE     LP1416
                LBSR    P2024
                LBRA    SIA
LP1416          LBSR    P1416
SIA             LBSR    COMPA
                STX     DT
                STY     DS
                LDA     #$00
                STA     AUX2
                LDX     #ES1
                LDY     #VAR4
ALMA            LDA     ,X+
                STA     ,Y+
                LDA     AUX2
                INCA
                STA     AUX2
                CMPA    #$7
                BLE     ALMA
                RTS

CORRE
                LDA     #$0
                STA     AUX2
                LDX     #ES1
                LDY     #VAR4
OBTEN           LDA     ,Y+
                STA     ,X+
                LDA     AUX2
                INCA
                STA     AUX2
                CMPA    #$7
                BLE     OBTEN
                LDX     DT
                LDY     DS
                LDA     NP
                CMPA    #$10
                BLE     CP1416
                LBSR    P2024
                LBRA    SIB
CP1416          LBSR    P1416
SIB             STX     DT
                STY     DS
                RTS

P1416
LOOP1           LDA     ,X
                STA     AUX2
                CMPA    #$7
                BLE     SIETE
                CMPA    #$8
                BNE     OCHO
```

```
            LDA     NP
            CMPA    #$0E
            BEQ     OCHO
            LBRA    SIETE
OCHO
            LDA     NP
            INCA
            LBSR    RDAT1
            LBRA    S2
SIETE
            LBSR    RDAT0
S2          LBSR    COMPA
            BNE     LOOP1
            LBRA    FIN1

P2024
LOOP2       LDA     ,X
            STA     AUX2
            CMPA    #$8
            BLE     PRIME
            CMPA    #$0A
            BLE     PRIME1
            CMPA    #$0C
            BLE     PRIME2
            LDA     NP
            CMPA    #$18
            BEQ     SIGUE4
            INCA
            LBSR    RDAT1
            LBRA    S3
SIGUE4      LDA     ,X
            CMPA    #$10
            BLE     SIGUE6
            LDA     NP
            INCA
            LBSR    RDAT1
            LBRA    S3
SIGUE7      LDA     NP
            LBSR    RDAT1
            LBRA    S3
SIGUE6      LDA     NP
            LBRA    SIGUE3
PRIME2      LDA     NP
            CMPA    #$18
            BEQ     PRIME1
SIGUE3      SUBA    #$7
            SUBA    ,X
            STA     AUX2
            LBSR    RDAT3
            LBRA    S3
```

```
PRIME1  LDA     ,X
        SUBA    #$8
        STA     AUX2
        LBSR    RDAT2
        LBRA    S3
PRIME   LBSR    RDAT0
S3      LBSR    COMPA
        BNE     LOOP2
        LBRA    FIN1

RDAT3
        LBSR    CORRE1
        LDA     AU1
        ORA     ES4
        STA     ES4
        LDA     AU2
        ORA     DAT04
        STA     DAT04
        RTS

RDAT2
        LBSR    CORRE1
        LDA     AU1
        ORA     ES3
        STA     ES3
        LDA     AU2
        ORA     DAT03
        STA     DAT03
        RTS

RDAT1
        SUBA    ,X
        STA     AUX2
        LBSR    CORRE1
        LDA     AU1
        ORA     ES2
        STA     ES2
        LDA     AU2
        ORA     DAT02
        STA     DAT02
        RTS

RDAT0
        LBSR    CORRE1
        LDA     AU1
        ORA     ES1
        STA     ES1
        LDA     AU2
        ORA     DAT01
        STA     DAT01
        RTS
```

```
CORRE1
            CLR     AU1
            CLR     AU2
            LDA     AU1
            ADDA    #$01
            STA     AU1
            LDB     AU2
            ORB     ,Y+
LOOP5       LDA     AUX2
            DECA
            STA     AUX2
            BEQ     SIGUE1
            LDA     AU1
            LSLA
            STA     AU1
            LSLB
            LBRA    LOOP5
SIGUE1      STB     AU2
            RTS

COMPA
            STX     AUX1
            LDD     AUX1
            ADDD    #$01
            STD     AUX1
            LDX     AUX1
            LDA     ,X
            CMPA    #$FF
FIN1        RTS

CLOCK
            CLR     DATO1
            CLR     DATO2
            LBSR    P1A2A
            RTS

ANAL
            CLR     C2
            CLR     MAL
            LDX     DT
INICIO      LEAX    +1,X
            STX     DT
            LDA     [DT]
            STA     PS
            LDB     NP
            CMPB    #$0E
            BGT     N16
            CMPA    #$7
            BGT     ASTE
```

```
GAMA    LBSR    R1
        LDA     DATO1
        STA     AUX3
        LBRA    FINO
ASTE    LBSR    CONV
        LBSR    R1
        LDA     DATO2
        STA     AUX3
        LBRA    FINO
N16     CMPB    #$10
        BGT     N20
        CMPA    #$8
        BGT     ASTE
        LBRA    GAMA
N20     CMPB    #$14
        BGT     N24
        CMPA    #$8
        BGT     L31
        LBRA    GAMA
        CMPA    #$0A
        BGT     L32
BETA    LBSR    R2
        LDA     DATO3
        STA     AUX3
        LBRA    FINO
L32     CMPA    #$0C
        BGT     ASTE
ALFA    LBSR    CONV
        LBSR    R2
        LDA     DATO4
        STA     AUX3
        LBRA    FINO
N24     CMPA    #$8
        BGT     L41
        LBRA    GAMA
L41     CMPA    #$0C
        BGT     L42
        LBRA    BETA
L42     CMPA    #$10
        BGT     ASTE
        LBRA    ALFA
FINO    LBSR    ROTA
        LBSR    COMP
        LDB     C2
        INCB
        STB     C2
        CMPB    NPS
        BEQ     FIN2
        LBRA    INICIO
FIN2    LEAX    +2,X
        STX     DT
        RTS
```

```
R1      LDA     PS
        SUBA    #$1
        STA     ROT
        RTS

R2      LDA     PS
        SUBA    #$9
        STA     ROT
        RTS

CONV    LDA     NP
        ADDA    #$1
        SUBA    PS
        STA     PS
        RTS

ROTA    CLRA
PRIN    LDB     AUX3
        CMPA    ROT
        BEQ     TERM
        RORB
        STB     AUX3
        INCA
        LBRA    PRIN
TERM    ANDB    #$1
        STB     AUX3
        RTS

COMP
        LDB     [DS]
        STB     VR
        LDY     DS
        LEAY    +1,Y
        STY     DS
        LDA     AUX3
        CMPA    VR
        BNE     ERROR
        LBRA    FINAL
ERROR   LDA     #$01
        STA     MAL
FINAL   RTS

ELLIGE
        LBSR    P1A2A
        LDA     NP
        CMPA    #$10
        BLE     P12
        LBSR    P1B2B
P12     RTS
```

```
P3B4A
            CLR     CR3B
            CLR     CR4A
            LDA     #$FF
            STA     DDR3B
            STA     DDR4A
            LDA     #$04
            STA     CR3B
            STA     CR4A
            LDA     DIR1
            ORA     MEM
            STA     DDR4A
            LDA     DIR2
            STA     DDR3B
            RTS

P3AS
            CLR     CR3A
            LDA     #$FF
            STA     DDR3A
            LDA     #$04
            STA     CR3A
            LDA     DATO
            STA     DDR3A
            RTS

P3AE
            CLR     CR3A
            CLR     DDR3A
            LDA     #$04
            STA     CR3A
            LDA     DDR3A
            STA     DATO
            RTS

P4BS
            CLR     CR4B
            LDA     #$FF
            STA     DDR4B
            LDA     #$04
            STA     CR4B
            LDA     RW
            STA     DDR4B
            RTS

P1A2A
            CLR     DDR1A
            CLR     DDR2A
            CLR     CR1A
            CLR     CR2A
            LDA     ES1
```

```
            LDB     ES2
            STA     DDR1A
            STB     DDR2A
            LDA     #$04
            STA     CR1A
            STA     CR2A
            LDA     DATO1
            LDB     DATO2
            STA     DDR1A
            STB     DDR2A
            LDA     DDR1A
            LDB     DDR2A
            STA     DATO1
            STB     DATO2
            RTS

P1B2B
            CLR     DDR1B
            CLR     DDR2B
            CLR     CR1B
            CLR     CR2B
            LDA     ES3
            LDB     ES4
            STA     DDR1B
            STB     DDR2B
            LDA     #$04
            STA     CR1B
            STA     CR2B
            LDA     DATO3
            LDB     DATO4
            STA     DDR1B
            STB     DDR2B
            LDA     DDR1B
            LDB     DDR2B
            STA     DATO3
            STB     DATO4
            RTS

FFF         LDA     #$FF
            STA     DATO
            LDA     #$01
            STA     RW
            LDA     #$40
            STA     MEM
            CLR     DIR1
            CLR     DIR2
            LBSR    P4BS
            LBSR    P3B4A
            LBSR    P3AS
            RTS
            END
```

CAPITULO 5

DISEÑO DEL HARDWARE

La elaboración del hardware implementado en este proyecto, tiene como finalidad comunicar al puerto de salida de la "COCO 3" con la circuitería requerida para la realización de las pruebas de los circuitos integrados, los componentes usados tienen compatibilidad con los de la computadora.

DESCRIPCION DEL FUNCIONAMIENTO DE LOS CI'S

PIA

PERIPHERICAL INTERFACE ADAPTER.

Este dispositivo se encarga de establecer la interfase de señales entre el MPU y un periférico. La PIA se comunica por medio de un bus de datos de 8 bits bidireccionales. Como se mencionó en el capitulo dos, la PIA consta de dos puertos (A y B) de 8 bits cada uno, los cuales, son programables como entradas o como salidas (ver figura 21).

La PIA MC6821 tiene dos partes: A y B. Cada una tiene un registro de datos periférico, un registro de

MC68A21P

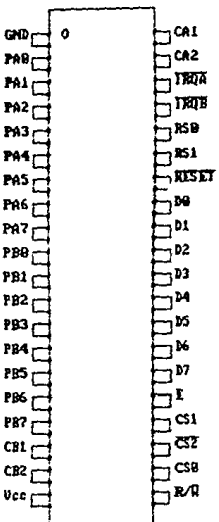


FIGURA 21

dirección de datos y un registro de control. El registro periférico de datos permite la interfase entre la PIA y el mundo exterior a través de 8 bits. El registro de dirección de datos definida por el programador indica si cada línea periférica es utilizada como entrada o como salida.

ENTRADA=0

SALIDA=1

El registro de control es el que permite al MPU el control de operación de las cuatro líneas periféricas de control (CA1, CA2, CB1 y CB2). El bit dos de este registro indica o determina el registro de datos periférico o registro de dirección de datos que ha sido seleccionado (ver figura 22 y 23).

Para tener acceso a la PIA es necesario que las patas de selección estén en los valores de voltaje adecuado, CS0 y CS1 deben tener un voltaje alto, CS2 debe tener un voltaje bajo. RS0 y RS1 están conectadas directamente a las direcciones A0 y A1 del microprocesador, seleccionan uno de los seis registros internos de la PIA.

Como se observa en la figura 21, los pines que

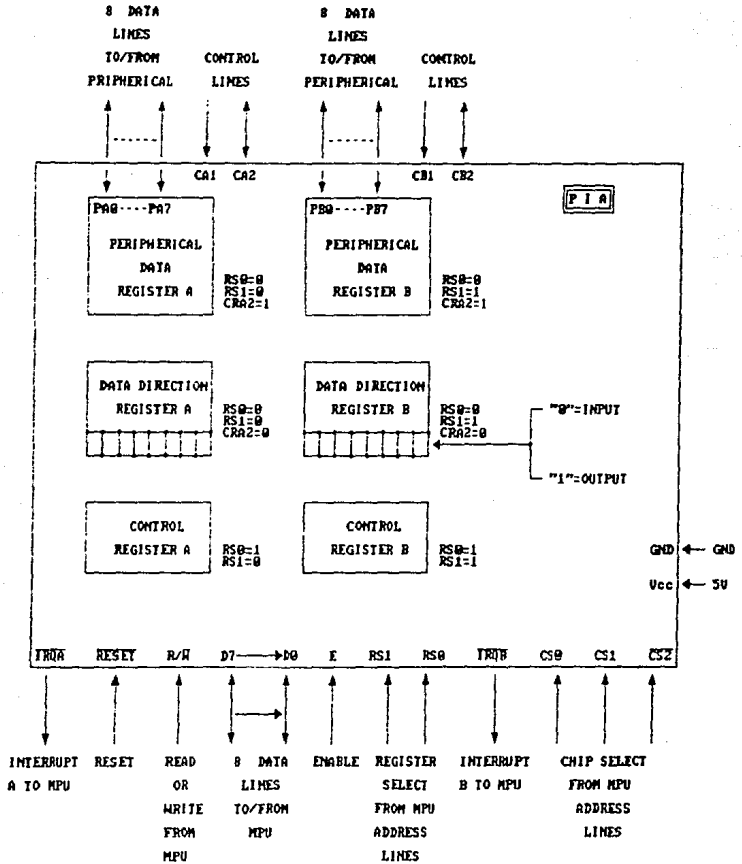

8 DATA LINES TO/FROM PRIPHERICAL
CONTROL LINES
8 DATA LINES TO/FROM PERIPHERICAL
CONTROL LINES
CA1 CA2
CB1 CB2
P I A
PA0----PA7
PERIPHERICAL DATA REGISTER A
RS0=0 RS1=0 CRA2=1
PB0----PB7
PERIPHERICAL DATA REGISTER B
RS0=0 RS1=1 CRA2=1
DATA DIRECTION REGISTER A
RS0=0 RS1=0 CRA2=0
DATA DIRECTION REGISTER B
RS0=0 RS1=1 CRA2=0
"0"=INPUT
"1"=OUTPUT
CONTROL REGISTER A
RS0=1 RS1=0
CONTROL REGISTER B
RS0=1 RS1=1
GND ← GND
Vcc ← 5V
IRQA RESET R/W D7——→D0 E RS1 RS0 IRQB CS0 CS1 CS2
INTERRUPT A TO MPU
RESET
READ OR WRITE FROM MPU
8 DATA LINES TO/FROM MPU
ENABLE
REGISTER SELECT FROM MPU ADDRESS LINES
INTERRUPT B TO MPU
CHIP SELECT FROM MPU ADDRESS LINES

FIGURA 22

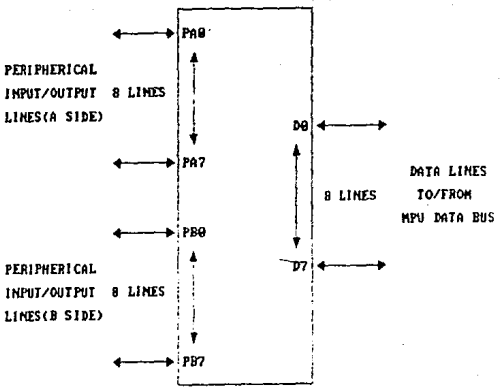

PA0
PERIPHERICAL
INPUT/OUTPUT 8 LINES
LINES(A SIDE)
PA7
D0
8 LINES
DATA LINES
TO/FROM
MPU DATA BUS
PB0
D7
PERIPHERICAL
INPUT/OUTPUT 8 LINES
LINES(B SIDE)
PB7

FIGURA 23

constituyen a la PIA son 40, de los cuales se da una breve explicación a continuación:

PA0-PA7 y PB0-PB7 (Puerto A y Puerto B respectivamente).

Estas líneas pueden ser programadas como entradas o como salidas por medio del DDR (Data Direction Register), correspondiendo un "1" lógico si es salida y un "0" lógico si es entrada. Cuando va ser leído por el MPU es necesario programar a la PIA como entrada y utilizar una instrucción de carga (LD), para obtener los datos de la PIA y almacenarlos en memoria . Al programarla como salida los datos del MPU son transferidos a la PIA por medio de una instrucción de almacenamiento (ST).

D0-D7 (líneas de datos)

Se tienen disponibles 8 líneas de datos bidireccionales que pemiten hacer la transferencia de datos entre la PIA y el MPU.

El bus de datos de salida de la PIA son dispositivos de tres estados que permanecen en alta impedancia, excepto cuando el MPU permite a la PIA hacer una operación de lectura.

Líneas de Selección.

Estas líneas permiten la selección de una PIA en particular lo cual para que funcione la PIA es necesario que CS0 y CS1 estén en un estado alto ("1" lógico) y CS2 debe estar en un estado bajo ("0" lógico).

ENABLE (E)

El pulso del enable es un periodo de tiempo que es controlado por el MPU para activar la PIA y ésta pueda hacer la operación requerida por el MPU.

RESET

Esta línea permite limpiar todos los registros de la PIA, asignándoles un cero lógico. Normalmente estos registros están en un estado alto.

R/W (Read y Write)

Esta señal es generada por el MPU y controla la dirección de transferencia de datos. Un estado bajo transfiere los datos del MPU a la PIA y un estado alto permite a la PIA hacer la transferencia de datos de la PIA al MPU.

IRQA e IRQB

Estas líneas interrumpen al MPU directa o indirectamente por medio de una rutina de software que secuencialmente lee y prueba. La condición para que se lleve a cabo una interrupción es necesario que el bit 6 y 7 del registro de banderas estén en un estado alto.

CA1 y CB1

Estas líneas son usadas unicamente como entradas a la PIA para interrupción.

CA2 y CB2

Estas líneas pueden ser usadas para programar una interrupción de entrada o salida de un periférico.

DECODIFICADOR

Circuito combinacional que convierte la información binaria de N líneas de entrada a 2 a la N líneas de salida. El circuito sólo elige una de las líneas de salida poniéndola en valor inverso a las demás líneas de salida.

EPROM Y RAM

Erasable Programable Read Only Memory y Random Access Memory.

La organización de éstas memorias son de 16K x 8 y 8K x 8 respectivamente. Las cuales son utilizadas para el almacenamiento del software, los nombres de los pines y la tabla de verdad teórica de los posibles circuitos integrados a probar.

LISTA DE COMPONENTES

No. DE C.I.	DESCRIPCION	UBICACION
MC68A21P	PIA	U1, U2, U3 y U4
HM6264	RAM	U5
MC27128	EPROM	U6
MC74LS139	DECODIFICADOR DE 2 A 4 CON SALIDA NEGADA	U7

MC74LS04	INVERSOR	U8
SR0852	PUENTE DE DIODOS A 1 AMPERE	CR1
LM7805	REGULADOR DE VOLTAJE	VR1
	LEDS	CR2 y CR3
	TRANSFORMADOR 127V A 6V 300mA	T1
	MICROSWITCH	SW1, SW2 y SW3.
	RESISTENCIAS 330Ω	R1-R24
	RESISTENCIAS 470Ω	R25 y R26
	CAPACITORES CERAMICOS 1μf A 250V	C1-C8
	CAPACITOR ELECTROLITICO 2200μf A 25V	C9

CAPACITOR ELECTROLITICO C10

4.7μf A 50V

CAPACITOR CERAMICO C11-C12

0.33μf A 250V

FUNCIONAMIENTO

El diagrama de bloques de este diseño se muestra en la figura 24. De éste diagrama se pueden distinguir cuatro partes importantes:

1.- FUENTE DE PODER

2.- BLOQUE DE SELECCION (DECODIFICADOR Y PIAS)

3.- BLOQUE DE ALMACENAMIENTO

4.- BLOQUE DE ZONA DE PRUEBA

FUENTE DE PODER

Se implementó una fuente de poder con la finalidad de no sobrecargar a la computadora; para lo cual, se utilizó un transformador que proporciona un voltaje de 6 volts, ésta señal se manda a un puente de diodos, el que se encarga de rectificar la señal. A la salida del

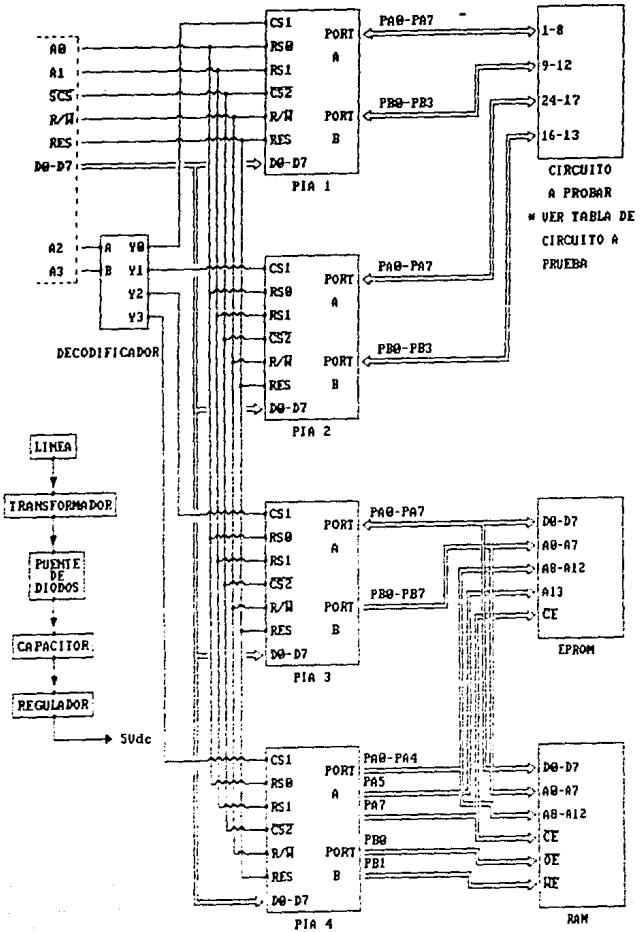

A0
A1
SCS
R/W
RES
D0-D7
A2
A3
A
B
Y0
Y1
Y2
Y3
DECODIFICADOR
CS1
RS0
RS1
CS2
R/W
RES
D0-D7
PORT
A
PORT
B
PIA 1
PIA 2
PIA 3
PIA 4
PA0-PA7
PB0-PB3
1-8
9-12
24-17
16-13
CIRCUITO
A PROBAR
* VER TABLA DE
CIRCUITO A
PRUEBA
LINEA
TRANSFORMADOR
PUENTE DE DIODOS
CAPACITOR
REGULADOR
5Vdc
A0-A7
A8-A12
A13
CE
EPROM
PA5
PA7
PB0
PB1
OE
WE
RAM

FIGURA 24

puente de diodos se coloca un capacitor con la finalidad de eliminar el voltaje de rizo que se forma obteniendo así 6 volt de DC, como este voltaje no es el adecuado para la circuitería, se puso un regulador de voltaje de manera que se obtengan 5 volts de DC a la salida de este y así conectar directamente todos los circuitos al regulador. Para evitar el ruido y que la circuitería no falle se colocaron capacitores a la entrada y salida del regulador.

BLOQUE DE SELECCION

Está constituido por un decodificador que se encarga de seleccionar una de las PIAS que se tienen disponibles.

La selección se hace por medio de las direcciones A2 y A3 de MPU, ya que van conectadas al decodificador y este hace la selección de una de las salidas.

A3	A2	PIA
0	0	1
0	1	2
1	0	3
1	1	4

La tabla de direcciones se muestra a continuación:

DIRECCION	A3	A2	A1	A0	
FF40	0	0	0	0	
					PUERTO A PIA 1
FF41	0	0	0	1	
FF42	0	0	1	0	
					PUERTO B PIA 1
FF43	0	0	1	1	
FF44	0	1	0	0	
					PUERTO A PIA 2
FF45	0	1	0	1	
FF46	0	1	1	0	
					PUERTO B PIA 2
FF47	0	1	1	1	
FF48	1	0	0	0	
					PUERTO A PIA 3
FF49	1	0	0	1	

FF4A	1	0	1	0	
					PUERTO B PIA 3
FF4B	1	0	1	1	
FF4C	1	1	0	0	
					PUERTO A PIA 4
FF4D	1	1	0	1	
FF4E	1	1	1	0	
					PUERTO B PIA 4
FF4F	1	1	1	1	

NOTA: Las líneas de dirección A0 y A1 están conectadas a las líneas RS0 y RS1 de la PIA, en donde RS1 indica la parte de la PIA que ha sido elegida,

A1=0 PUERTO A

A1=1 PUERTO B

y RS0 indica el registro seleccionado,

A0=RS0=0 CRA=0 o CRB=0 selecciona DDR

A0=RS0=0 CRA=1 o CRB=1 selecciona PDR

si A0=RS0=1 selecciono CR.

Cabe mencionar que 2 PIAS (U1 y U2) son utilizadas para entradas y salidas de información del circuito a

prueba. En tanto que las otras dos PIAS (U3 y U4) se utilizan para la obtención de toda la información teórica del circuito que se esté probando.

La información que se tiene puede estar almacenada en la EPROM o en la RAM, por lo que, es necesario habilitar el bit 6 del Puerto A de la PIA 4 (U4), el que activará alguna de las dos memorias.

BIT 6 = 0	EPROM
BIT 6 = 1	RAM

BLOQUE DE ALMACENAMIENTO

Una vez que ha sido seleccionada la memoria de la que se obtendrá o almacenará información, es necesario habilitar la PIA 3 (U3), la que extraerá o transferirá por medio del puerto A al MPU.

El Puerto B de ésta servirá para saber los primeros 8 bits de la dirección a elegir.

Si se seleccionó la EPROM, se requiere de los 6 primeros bits de la PIA 4 (U4) para saber los bits complementarios de la direccion elegida. Si la seleccionada fue la RAM, se requiere de los 5 primeros bits de la misma para saber el complemento de la dirección.

Es necesario habilitar los bit cero y uno del Puerto B de la misma PIA para activar en la RAM la línea de escritura o de lectura.

BLOQUE DE PRUEBA

Los circuitos que se pueden probar son de 14, 16, 20 y 24 pines por lo que, la distribución de las líneas de los puertos para la obtención y transferencia de información se distribuyó como se indica en la siguiente tabla:

TABLA DE CIRCUITO A PROBAR

		PIA 1		
No. DE PATAS	PUERTO A	No. DE PINES	PUERTO B	No. DE PINES
14	PA0-PA6	1 - 7	--------	----------
16	PA0-PA7	1 - 8	--------	----------
20	PA0-PA7	1 - 8	PB0-PB1	9 - 10
24	PA0-PA7	1 - 8	PB0-PB3	9 - 12
		PIA 2		
No. DE PATAS	PUERTO A	No. DE PINES	PUERTO B	No. DE PINES
14	PA0-PA6	14 - 8	--------	----------
16	PA0-PA7	16 - 9	--------	----------
20	PA0-PA7	18 - 11	PB0-PB1	20 - 19
24	PA0-PA7	20 - 13	PB0-PB3	24 - 21

Es necesario habilitar los bit cero y uno del Puerto B de la misma PIA para activar en la RAM la línea de escritura o de lectura.

BLOQUE DE PRUEBA

Los circuitos que se pueden probar son de 14, 16, 20 y 24 pines por lo que, la distribución de las líneas de los puertos para la obtención y transferencia de información se distribuyó como se indica en la siguiente tabla:

TABLA DE CIRCUITO A PROBAR

PIA 1

No. DE PATAS	PUERTO A	No. DE PINES	PUERTO B	No. DE PINES
14	PA0-PA6	1 - 7	--------	----------
16	PA0-PA7	1 - 8	--------	----------
20	PA0-PA7	1 - 8	PB0-PB1	9 - 10
24	PA0-PA7	1 - 8	PB0-PB3	9 - 12

PIA 2

No. DE PATAS	PUERTO A	No. DE PINES	PUERTO B	No. DE PINES
14	PA0-PA6	14 - 8	--------	----------
16	PA0-PA7	16 - 9	--------	----------
20	PA0-PA7	18 - 11	PB0-PB1	20 - 19
24	PA0-PA7	20 - 13	PB0-PB3	24 - 21

La PIA U1 se utiliza para los 7, 8, 10 o 12 primeros pines y la PIA 2 para los restantes de acuerdo al número de pines del circuito integrado a probar.

Cuando se tienen circuitos de 14 o 16 pines sólo se activa el Puerto A de las PIAS 1 y 2; y cuando se tienen circuitos de 20 o 24 pines se activan tanto el Puerto A como el Puerto B de ambas PIAS.

En caso de probar un circuito de OPEN COLLECTOR se tiene disponible una serie de resistencias de 330Ω conectadas a Vcc y a cada una de las patas de la base del C.I. que se va a probar, claro que solo se activan por medio de un microswitch, por lo que, sólo se deben activar las resistencias necesarias para probar el C.I..

NOTA: Para evitar algun daño tanto a la circuitería del diseño como al mismo C.I. a prueba, es de suma importancia colocar correctamente el C.I. a prueba, por lo que, por medio de un led rojo se indica la pata uno del socket.

Una precaución adicional para evitar que el ruido haga que funcionen mal los circuitos integrados es colocar un capacitor de 1μf a cada circuito entre Vcc y tierra.

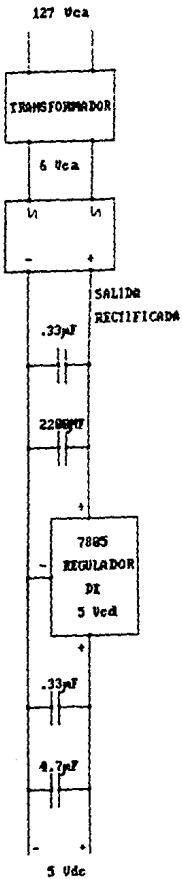

127 Vca
TRANSFORMADOR
6 Vca
SALIDA
RECTIFICADA
.33μF
2200μF
7805
REGULADOR
DE
5 Vcd
.33μF
4.7μF
5 Vdc

FIGURA 25

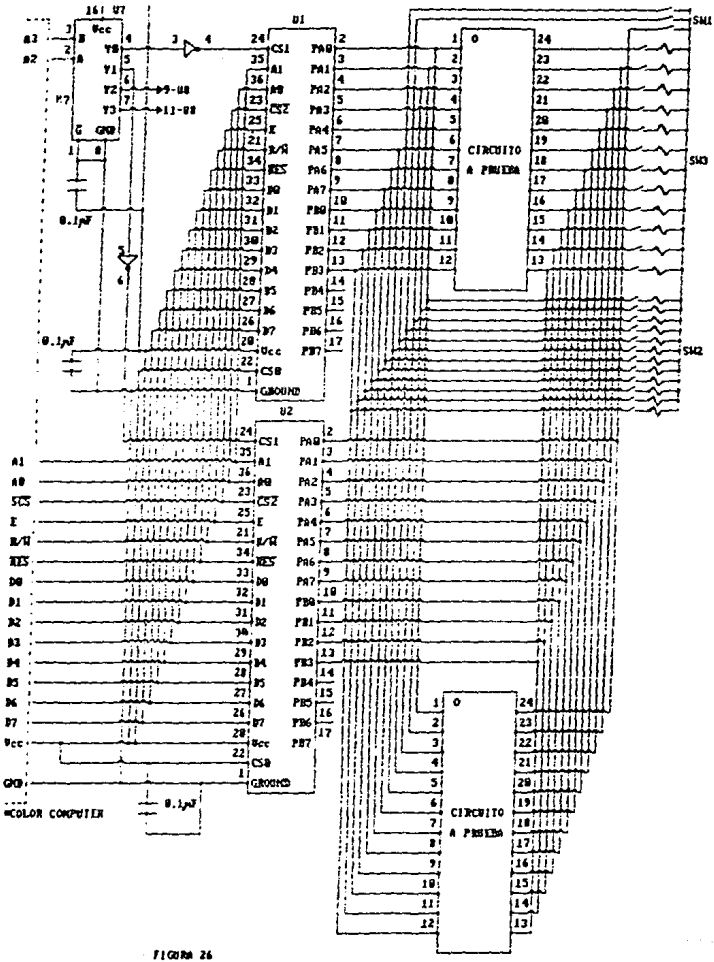

U7
U1
U2
Vcc
GND
A13
A12
A11
A10
Y0
Y1
Y2
Y3
0-7
8-F
8-B
C-F
0.1µF
CS1
CS2
A1
A0
R/W
RES
D0
D1
D2
D3
D4
D5
D6
D7
E
GROUND
PA0
PA1
PA2
PA3
PA4
PA5
PA6
PA7
PB0
PB1
PB2
PB3
PB4
PB5
PB6
PB7
CIRCUITO
A PRUEBA
SM1
SM2
SM3
COLOR COMPUTER

FIGURA 26

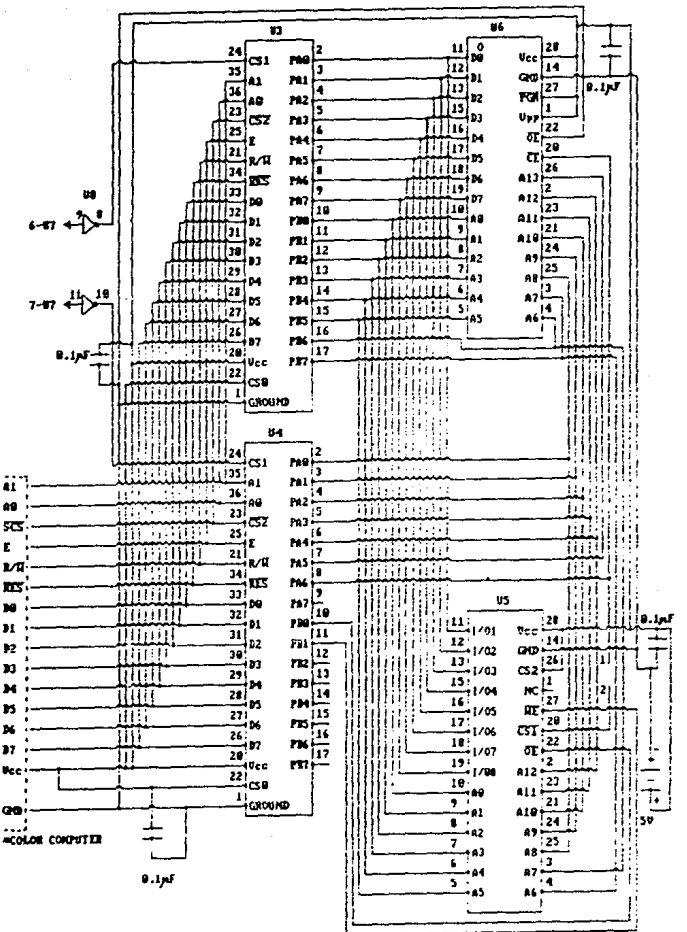

U3
U6
U4
U5
CS1
A1
A0
CSZ
E
R/W
RES
D0
D1
D2
D3
D4
D5
D6
D7
Vcc
CS0
GROUND
PA0
PA1
PA2
PA3
PA4
PA5
PA6
PA7
PB0
PB1
PB2
PB3
PB4
PB5
PB6
PB7
Vcc
GND
PGM
Vpp
OE
CE
A13
A12
A11
A10
A9
A8
A7
A6
A5
A4
A3
A2
A1
A0
I/O1
I/O2
I/O3
I/O4
I/O5
I/O6
I/O7
I/O8
CS2
NC
WE
CS1
U8
6-R7
7-R7
0.1μF
COLOR COMPUTER
S9

FIGURA 27

CONCLUSIONES

Los objetivos de esta tesis se llevaron a cabo con gran satisfacción a pesar de la poca información existente sobre el manejo de la memoria, así como, el de algunas instrucciones tanto de lenguaje Ensamblador como de algunas especificaciones de la computadora COCO 3.

Los problemas encontrados en cuanto al manejo de la memoria fueron ocasionados por el uso de programas realizados en BASIC y en Ensamblador, lo que originó la necesidad de proteger ciertas localidades de memoria, de forma que a la hora de correr el programa de Basic, éste no invada la zona elegida para el Ensamblador y el Stack.

El interés de conocer el microprocesador 6809 fue el principal motivo de utilizar la COCO 3, aun cuando esta sea conocida como una computadora de uso casero, unida a la facilidad de adquirir una computadora de este tipo, consiguiendo así poder usar el paquete sin necesidad de hacer una fuerte inversión.

La estructura del programa se desarrolló lo más modular posible, además, para las tres actividades importantes se tiene una estructura de árbol necesitando

almacenar varios apuntadores en el stack. Al irse introduciendo a lo largo del programa, se van almacenando varios apuntadores ocasionados por llamadas a subrutinas, de tal manera que el stack interno va perdiendo los primeros apuntadores conforme se van llamando a las subrutinas. Ocasionando que al ir retornando a las primeras llamadas no encuentra los apuntadores correctos, provocando que el Program Counter se vaya a localidades no correspondientes. Lo anterior provoca a la computadora el bloqueo del sistema.

La solución a éste problema fue el uso de otras instrucciones que no tuvieran que ocupar localidades del stack.

Las PIAS tienen 3 entradas para permitir habilitarlas, una de ellas se conectó directamente a Vcc, otra a la salida de un decodificador, lo que hizo pensar por un momento que se necesitaba para la tercer señal de un arreglo de compuertas, evitando así, que la PIA se habilitára con cualquier dirección que coincidiera con las entradas del decodificador.

Al analizar más a fondo las salidas del slot se observó que había una señal que se habilitaba unicamente desde la dirección FF00 hasta la FFFF, correspondiente a

la zona de Ent/Sal de la computadora (Ver mapa de memoria (Fig.14d), lo que nos llevo a tomar la decisión de ocupar esta señal, conectándola al $\overline{CS2}$ de la PIA y así ahorrase el arreglo de compuertas evitando un incremento en costo y en tamaño.

En el hardware se encontró un problema: la PIA no alcanza a polarizar el circuito integrado a probar, esto es debido a que la corriente que maneja la PIA no es suficiente para poder polarizar un CI. El problema se localizó al hacer varias mediciones y descartar posibilidades llegando hasta ésta falla, la que se solucionó de la siguienta manera: primeramente se observó en que patas era la polarización de los CI's de 14, 16, 20 y 24, dichas patas fueron: en los de 24 la pata 24, en los de 20 la pata 20, en los de 16 las patas 16 y 5 y en los de 14 las patas 4, 5 y 14; tomando esto en cuenta se colocaron tres microswitch conectando las patas 4, 5 y la última directamente a Vcc y a la base del CI a probar, para así elegir solamente una de las tres patas de polarización por medio de uno de ellos, dependiendo del CI que se le va a hacer la prueba.

MANUAL DE USUARIO

El sistema esta diseñado a base de MENUS, permitiendo al usuario interactuar con este, logrando introducirse al nivel que desee, dependiendo de la opción elegida.

Recomendaciones al usuario

Es recomendable para no dañar el sistema, que el usuario tome las siguentes precauciones:

1.- Asegurarse que la computadora esté apagada antes de colocar el módulo de prueba.
2.- Insertar el módulo de prueba.
3.- Encender la computadora.
4.- Cargar el programa de BASIC.
5.- Correr el programa.
6.- Colocar el circuito a prueba hasta el momento en que se ponga el número del circuito (la computadora indica ese momento).

El sistema cuenta con tres actividades principales que son:

1.- La prueba de un circuito integrado.

Al elegir la opción de prueba <P> la computadora pregunta el número de circuito integrado a probar, una vez proporcionado el número de éste, es preciso que el usuario introduzca el C.I. en la base correspondiente.

NOTA: En el módulo de prueba se pone un led rojo indicando que corresponde al pin # 1 del C.I..
Se sugiere al usuario verificar la posición del C.I., antes de continuar la prueba, para evitar que el sistema y/o el C.I. se dañen.

El ejemplo 1 muestra los pasos a seguir para probar un circuito integrado.

2.- Consultar la lista de circuitos integrados.

En ésta opción <L> el usuario consulta los circuitos integrados almacenados hasta ese momento. El objetivo de ésta lista es saber si el circuito integrado que se quiera probar está ya almacenado, si esto es afirmativo, se continua con la prueba del circuito; de lo contrario primero se tiene que dar de alta el circuito, el que queda almacenado en la RAM, posteriormente se sigue con la prueba de éste (ver ejemplo 2).

3.- Dar de alta un circuito integrado.

Cuando se va a dar de alta un circuito integrado, es necesario tener a la mano los siguientes datos del mismo:

I.- Número de circuito integrado.

II.- Número de pines.

III.- Número de circuitos internos.

IV.- Número de entradas.

V.- Número de salidas.

VI.- Nombre de los pines.

VII.- Tabla de verdad.

La computadora por medio de menús ayuda al usuario a dar de alta el C.I. (ver ejemplo 3).

Después que se dió de alta el C.I., el siguente paso a seguir es hacer la prueba de éste.

NOTA: El C.I. queda almacenado y pasa a formar parte de la lista de C.I. disponibles.

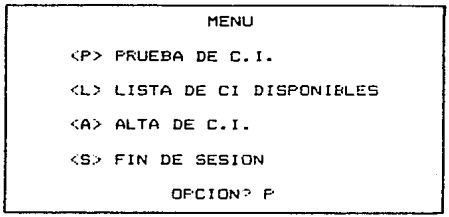
4.- Fin de sesión.

Con esta opción (S) nos regresamos al sistema presentando el OK, el que nos dice que la computadora está habilitada para trabajar en el lenguaje de BASIC.

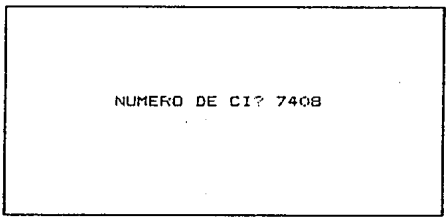
EJEMPLO 1:

El circuito a probar es un 7408 (compuertas AND).

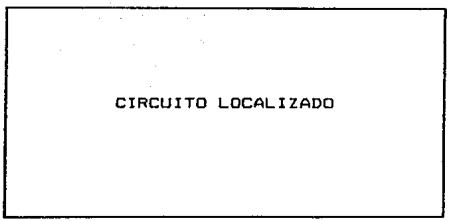
a)



b)



c)



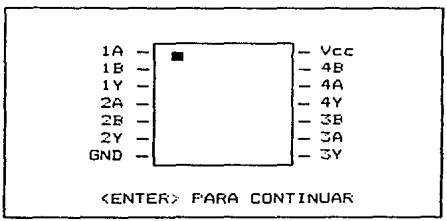
NOTA: El circuito es buscado en EPROM y RAM en este caso es localizado, mostrandose la pantalla anterior, de lo contrario se desplegará la siguiente:

```
CIRCUITO NO LOCALIZADO

<ENTER> PARA CONTINUAR
```

regresandose asi al MENU principal.

d)




```
        CIRCUITO INTERNO # X
            ESTADO # Y

ENTRADAS
 1A     1B
  1      1

SALIDAS
 1Y
  1
      <ENTER> PARA CONTINUAR
```

donde: X indica el número de circuito interno

Y indica el número de estado

Una vez que es oprimido <ENTER> será desplegado el número del siguiente estado con sus respectivos valores, en caso de no haber error en el análisis de los valores reales obtenidos.

Si se detectó algún error se mostrará la siguiente pantalla acompañado de un sonido.

```
        CIRCUITO INTERNO # X
           ESTADO # Y   ERROR
ENTRADAS
 1A     1B
  1      0
SALIDAS
 1Y
  0
      <ENTER> PARA CONTINUAR
```

Esta pantalla muestra al usuario la ubicación de la falla (circuito interno y estado).

Cuando ocurre un error la prueba de ese circuito interno es cancelada, continuando con el siguiente circuito interno si es que existe.

Al terminar la prueba del circuito con éxito o con errores el sistema regresa al menú principal.

NOTA: Los valores de la tabla de verdad que son mostradas en la pantalla son los teóricos.

EJEMPLO 2:

a)

```
                MENU

<P> PRUEBA DE C.I.

<L> LISTA DE CI DISPONIBLES

<A> ALTA DE C.I.

<S> FIN DE SESION

          OPCION? L
```

b)

```
        CIRCUITOS DISPONIBLES
             CI EN EPROM
          7405        7486
          7408        74139
          7427        74151
          7432        74244
          7474        74393
       <ENTER> PARA CONTINUAR
```

Al oprimir <ENTER> se muestra la siguiente pantalla.

```
        CIRCUITOS DISPONIBLES
             CI EN RAM
          7400        7410




       <ENTER> PARA CONTINUAR
```

En este ejemplo fueron dados de alta los circuitos 7400 y 7410. La lista será nula si no se han introducido circuitos y se ira incrementando de acuerdo a la actividad del usuario. Posteriormente se regresará al MENU principal.

EJEMPLO 3

a)

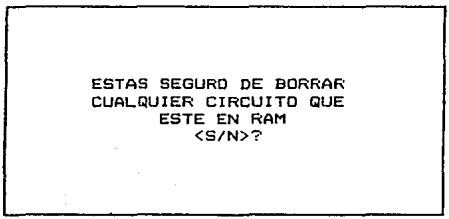
```
                MENU
<P> PRUEBA DE C.I.
<L> LISTA DE CI DISPONIBLES
<A> ALTA DE C.I.
<S> FIN DE SESION
          OPCION? A
```

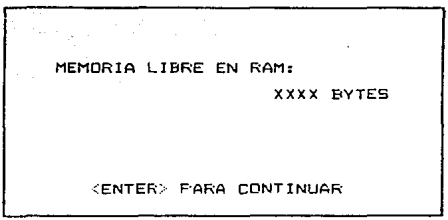
b)

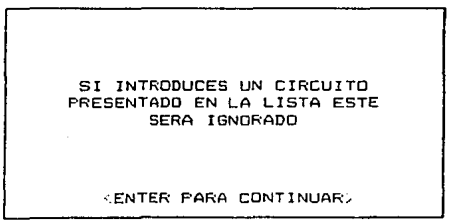
```
ES EL PRIMER CIRCUITO
<S/N>?
```

Esta pregunta da la opción de borrar toda la información de la memoria RAM, si la respuesta es S, debido a la acción a tomar, ocasionada con esta

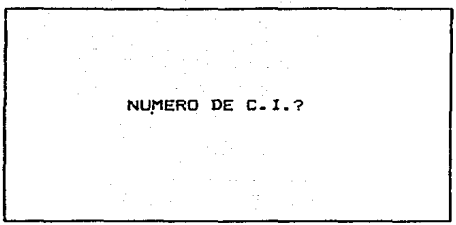
respuesta, la computadora despliega la siguiente pregunta para confirmar y/o rectificar dicha respuesta.

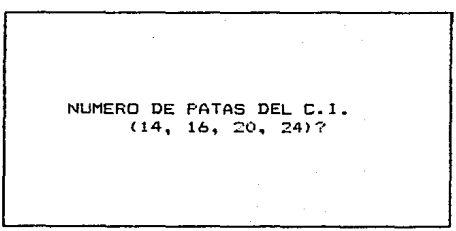




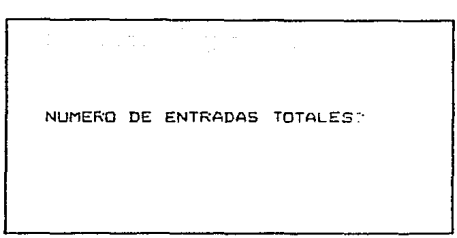


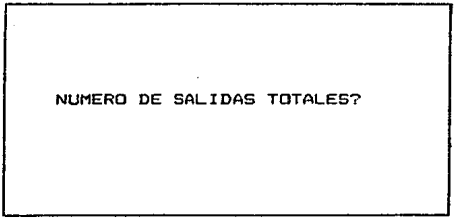
c)


NUMERO DE C.I.?

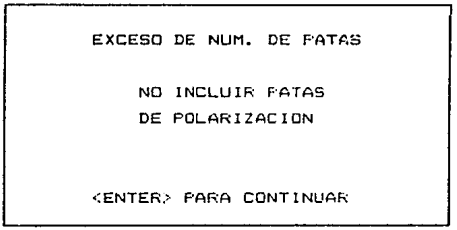

NUMERO DE PATAS DEL C.I.
(14, 16, 20, 24)?

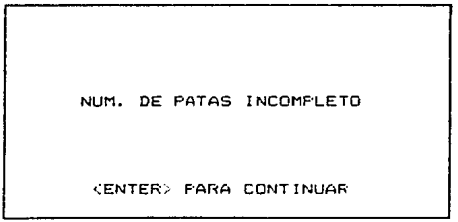
d)


NUMERO DE ENTRADAS TOTALES?

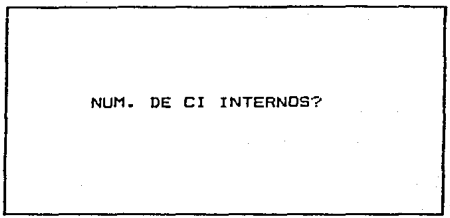


Las respuestas a las preguntas anteriores son validadas, en caso de que exista alguna incongruencia se mostrará alguna de las siguientes pantallas.

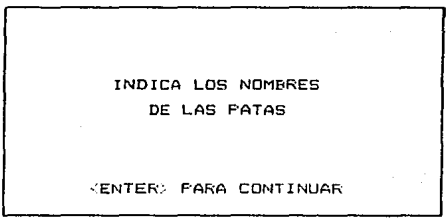




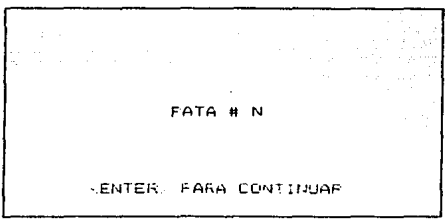
Regresando ambas a d).



e)

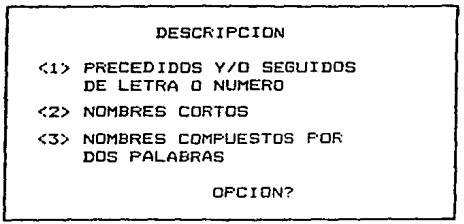


f)



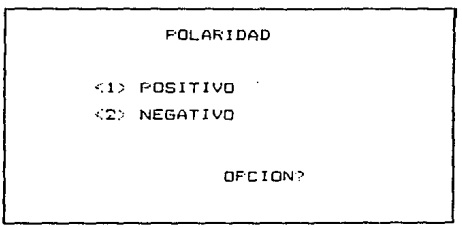
Es recomendable fijarse en el número de pata N, que es mostrado ya que la secuencia no es ascendente.

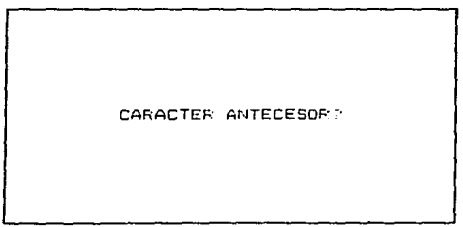
g)

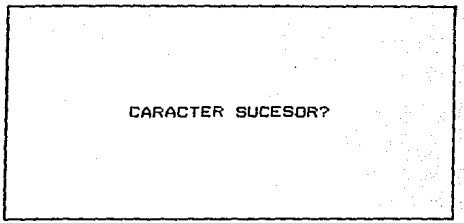


La opción es validada. En caso de existir error se muestra un letrero de OPCION INVALIDA.

Si la opción elegida fue <1> se tendrán los siguientes menus.







Caracter antecesor se refiere a la letra <A-H> o número <1-8> que va antes del nombre.

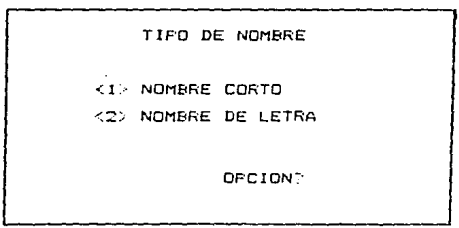
Ejemplo:

1Y	el antecesor es 1
2RXT/CEXT	el antecesor es 2

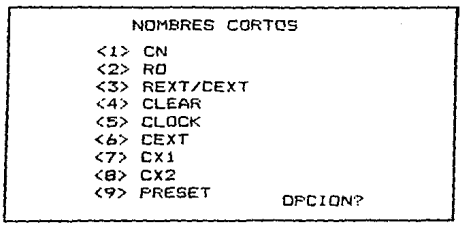
Caracter sucesor se refiere a la letra <A-H> o número <1-8> que va a continuación del nombre.

Ejemplo:

1Y	no tiene caracter sucesor
2WA	caracter sucesor A

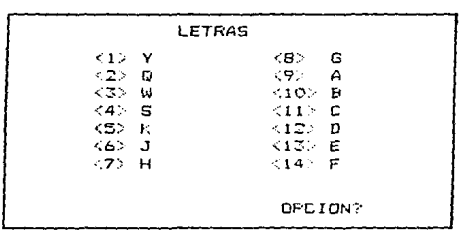


Eligiendo opción 1

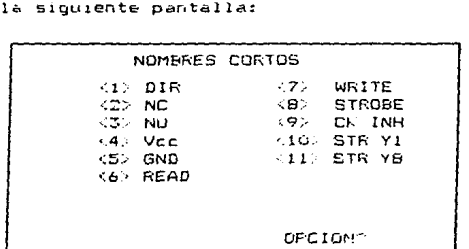


Al haber elegido la opción de nombres cortos es cancelado automáticamente el carácter sucesor en caso de existir.

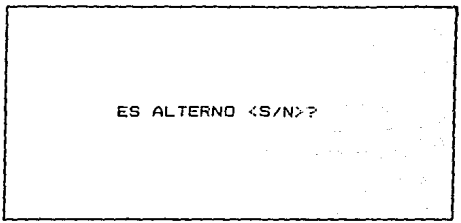
Eligiendo opción 2



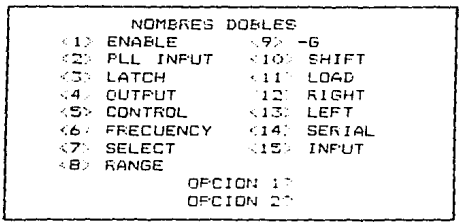
Si se elige la opción 2 de descripción de nombres se muestra la siguiente pantalla:



En caso de ser elegida la opción 4 o 5 aparece la siguiente pantalla.



Si se elige la opción 3 de descripción de nombres se tiene la siguiente pantalla:



El nombre de una pata puede estar compuesto por la concatenación de dos opciones de este menú. En caso de no ser así en la OPCION 2, se teclea simplemente ENTER. No es válido teclear en la OPCION 1 ENTER.

```
NOMBRE OBTENIDO:

                XXXXX

        CORRECTO <S/N>?
```

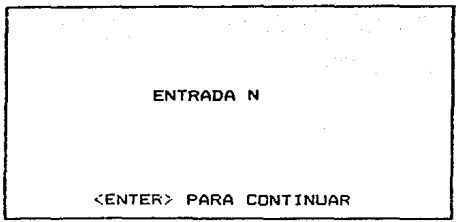
Esta pregunta permite verificar si el nombre de la pata fué correcto, en caso contrario, se regresa al inciso g.

Cuando se termina de dar de alta las patas del C.I. se despliega en pantalla el dibujo del circuito integrado.

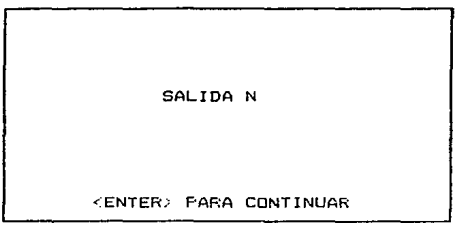
El siguiente paso a seguir es dar de alta las patas que son entradas y las que son salidas.

```
    INDICA ENTRADAS Y SALIDAS


    <ENTER> PARA CONTINUAR
```

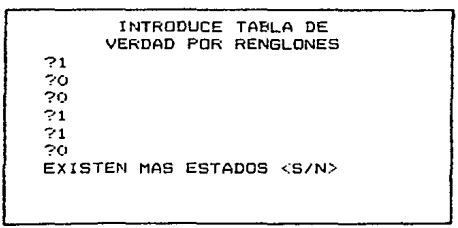


Se realiza el procedimiento indicado en el inciso g).



Se realiza el procedimiento indicado en el inciso g).

A continuación se proseguirá a dar la tabla de verdad del circuito integrado.



Los estados válidos en la tabla de verdad son unos y ceros.

↑ = 1

↓ = 0

⌐¯ = 1

¯_ = 0

Con esta información se concluye el proceso de altas regresando al MENU principal.

BIBLIOGRAFIA

+ BARDEN, WILLIAM

TRS-80 COLOR COMPUTER ASSEMBLY LANGUAGE PROGRAMMING

TANDY CORPORATION, 1983

+ BISHOP, RON

BASIC MICROPROCESSORS AND THE 6800

MOTOROLA INC., 1984

+ HAYES, JOHN P.

DISEÑO DE SISTEMAS DIGITALES Y MICROPROCESADORES

Mc GRAW HILL, 1984

+ LEVENTHAL, LANCE A.

6809 ASSEMBLY LENGUAGE PROGRAMMING

Mc GRAW HILL, 1981

+ MORRIS MANO, M.

LOGICA DIGITAL Y DISEÑO DE COMPUTADORES

PRENTICE HALL, 1985

+ ZAKS, RODNAY AND LABIAK, WILLIAM

PROGRAMMING THE 6809

MOTOROLA INC., 1982

+MOTOROLA INC.

- MEK-6802D3 MICROCOMPUTER UNIT, 1977
- MOTOROLA MICROPROCESSORS, 1983
- PROGRAMMING THE 6800 MICROPROCESORS, 1977

+TANDY CORPORATION

- GETTING STARTED WITH EXTENDED COLOR BASIC, 1984
- INTRODUCING YOUR COLOR COMPUTER 3, 1984
- SERVICE MANUAL FOR THE COLOR COMPUTER 3, 1983

+TEXAS INSTRUMENTS INCORPORATED

- MOS MEMORY DATA BOOK, 1985
- THE TTL DATA BOOK, 1981