UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE INGENIERIA

MICROCOMPUTADORAS PARA EL STD BUS

T E S I S

QUE PARA OBTENER EL TITULO DE:

INGENIERO MECANICO ELECTRICISTA

P R E S E N T A N :

ELIZABETH TREJO HERNANDEZ

JORGE CORDOVA LANDIN

ASESOR: M. EN I. LUIS ALVAREZ - ICAZA L.

MEXICO, D. F. 1992

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

ÍNDICE

RESUMEN

En este trabajo se describen dos microcomputadoras para emplearse en el STD BUS (estándar IEEE 961) basados en los circuitos integrados 8031 de INTEL y MC68HC11 de MOTOROLA, respectivamente.

Para cada tarjeta se desarrolló además un programa monitor para facilitar la elaboración y depuración de los programas de los usuarios, de tal forma que la combinación de la tarjeta, el programa monitor y un ensamblador residente en una computadora PC compatible constituyen en cada caso un módulo de desarrollo.

1. INTRODUCCIÓN

En la Coordinación de Automatización del Instituto de Ingeniería se han desarrollado tarjetas bajo el estándar STD BUS para aplicaciones en control. Actualmente se cuenta con una microcomputadora basada en el microprocesador 8088 de INTEL para un sistema basado en este estándar. Sin embargo se requiere también contar con sistemas para aplicaciones de control sencillas con base en microcontroladores cuya estructura interna les permita resolver este tipo de problemas sencillos sin necesidad de circuitería adicional, y adaptarse al mencionado estándar para mantener compatibilidad con los desarrollos existentes.

Después de un análisis sobre las diferentes posibilidades, se optó por seleccionar los microcontroladores INTEL 8031 y MOTOROLA 68HC11. En ambos casos se trata de dispositivos de ocho bites, que coinciden con el tamaño de la palabra para el STD BUS.

El objetivo de este trabajo es entonces, desarrollar dos procesadores, cada uno de ellos basado en los circuitos mencionados según el estándar STD BUS. Los procesadores deberán contar con un programa monitor que facilite el desarrollo de los programas para las aplicaciones en que se emplearán los

procesadores.

El programa monitor se encuentra residente en una memoria de sólo lectura que permite al usuario interactuar con el módulo a través del puerto serie del microcontrolador y del puerto serie de una microcomputadora PC-compatible mediante un paquete de comunicaciones. El paquete de comunicaciones permite transmitir el código de máquina correspondiente a un programa codificado en ensamblador a una memoria RAM. El programa así cargado, puede entonces manejarse con facilidad.

El programa monitor permite realizar las siguientes funciones:

- Examinar y modificar el contenido de una localidad de memoria.
- Examinar y modificar registros.
- Desplegar el contenido de un bloque de memoria.
- Desplegar la pantalla de ayuda.
- Mover bloques de memoria.
- Borrar el contenido de un bloque de memoria.
- Calcular desplazamientos relativos.
- Leer un programa.
- Verificar que un programa fue correctamente cargado.
- Imprimir un programa.
- Correr un programa residente en memoria RAM.
- Ejecutar un programa paso a paso (trazarlo).
- Cambiar la velocidad de transmisión del puerto serie.

Además de estas funciones, se le permite al usuario ejecutar directamente algunas subrutinas del programa monitor para apoyar su programación.

El escrito se divide en cinco capítulos principales: el capítulo 2 contiene las características y especificaciones técnicas del estándar STD BUS. Los dos siguientes capítulos

describen en forma general la arquitectura y funcionamiento de los microcontroladores 8031 y 68HC11. Enseguida, el capítulo 5 proporciona una descripción de la circuitería que compone ambas tarjetas. Por último, el capítulo 6 contiene una descripción del programa monitor y la información necesaria para su uso.

En la parte final de este escrito, se proporcionan 6 apéndices que contienen: diagramas electrónicos, de disposición, circuitos impresos, lista de partes, listado del programa monitor y el juego de instrucciones para cada uno de los microcontroladores.

2. DESCRIPCIÓN DEL STD BUS

El STD BUS está descrito por el estándar IEEE 961; se trata de un esquema de interconexión poderoso, modular y robusto para estandarizar los aspectos físicos y eléctricos de sistemas modulares que utilizan microprocesadores de ocho bites en aplicaciones de control industrial. El estándar establece los requisitos para una interconexión interna, pero deja libres las conexiones externas para realizarlas por medio de conectores apropiados al acoplamiento de entrada/salida (E/S). El tamaño pequeño de la tarjeta aumenta la resistencia a choques y vibraciones lo cual contribuye a mejorar la precisión del sistema; la modularidad facilita el servicio y mantenimiento, pues los problemas del sistema se pueden aislar a nivel de módulo y resolverse de inmediato con la sustitución de la tarjeta.

2.1 Especificaciones lógicas

El conjunto de dientes del STD BUS está organizado en cuatro grupos funcionales:

- Líneas duales de alimentación (dientes 1-6 y 53-56)
- Canal de datos (dientes 7-14)
- Canal de direcciones (dientes 15-30)
- Canal de control (dientes 31-52)

Las Tablas 2.1 y 2.2 muestran la función nemotécnica y la dirección del flujo de la señal, referida a la tarjeta del procesador maestro, para cada diente del STD BUS en la cara correspondiente.

CARA DE COMPONENTES				
GRUPO	DIENTE	NEMONICO	DIRECCION	DESCRIPCION
LINEAS DE ALI-MENTACION	1	+5VDC	ENTRADA	POTENCIA DIGITAL
	3	GND		TIERRA DIGITAL
	5	VBB #1		-5 V DC
CANAL DE DATOS	7	D3	BIDIRECCIONAL	PARTE BAJA DEL CANAL DE DATOS
	9	D2		
	11	D1		
	13	D0		
CANAL DE DIREC-CIONES	15	A7	SALIDA	PARTE BAJA DEL CANAL DE DIRECCIONES
	17	A6		
	19	A5		
	21	A4		
	23	A3		
	25	A2		
	27	A1		
	29	A0		
CANAL DE CONTROL	31	WR*	SALIDA	ESCRITURA A MEMO-RIA O A PUERTOS
	33	IORQ*	SALIDA	SELECTOR DE E/S
	35	IOEXP	BIDIRECCIONAL	EXPANSION DE E/S
	37	REFRESH*	SALIDA	SEÑAL DE REFRESCO
	39	STATUS1*	SALIDA	ESTADO DEL CPU
	41	BUSAK*	SALIDA	RECONOCIMIENTO DEL CANAL
	43	INTAK*	SALIDA	RECONOCIMIENTO DE INTERRUPCION
	45	WAITRQ*	ENTRADA	PETICION DE ESPE-RA
	47	SYSRESET*	SALIDA	REESTABLECIMIENTO DEL SISTEMA
	49	CLOCK*	SALIDA	SEÑAL DE RELOJ
	51	PCO	SALIDA	SALIDA DE LA CA-DENA DE PRIORIDAD
LINEAS DE ALI-MENTACION	53	AUX GND	ENTRADA	TIERRA AUXILIAR
	55	AUX +V	ENTRADA	+12 V DC

Tab. 2.1 Señales del STD BUS en la cara de componentes.

CARA DE SOLDADURA				
GRUPO	DIENTE	NEMONICO	DIRECCION	DESCRIPCION
LINEAS DE ALI-MENTACION	2 4 6	+5VDC GND VBB #2	ENTRADA	POTENCIA DIGITAL TIERRA DIGITAL -5 V DC
CANAL DE DATOS	8 10 12 14	D7 D6 D5 D4	BIDIRECCIONAL	PARTE ALTA DEL CANAL DE DATOS
CANAL DE DIREC-CIONES	16 18 20 22 24 26 28 30	A15 A14 A13 A12 A11 A10 A9 A8	SALIDA	PARTE ALTA DEL CANAL DE DIRECCIONES
CANAL DE CONTROL	32	RD*	SALIDA	LECTURA A MEMORIA O A PUERTOS
	34	MEMRQ*	SALIDA	SELECTOR DE DI-RECCION A MEMORIA
	36	MEMEX	BIDIRECCIONAL	EXPANSION DE ME-MORIA
	38	MCSYNC*	SALIDA	SINCRONIA CON EL CICLO DE MAQUINA
	40	STATUSO*	SALIDA	ESTADO DEL CPU
	42	BUSRQ*	ENTRADA	PETICION DEL CA-NAL
	44	INTRQ*	ENTRADA	PETICION DE INTE-RRUPCION
	46	NMIRQ*	ENTRADA	INTERRUPCION NO-ENMASCARABLE
	48	PBRESET*	ENTRADA	REESTABLECIMIENTO CON PUSH-BUTTON
	50	CNTRL*	ENTRADA	TEMPORIZACION AU-XILIAR
	52	PCI	ENTRADA	ENTRADA DE LA CA-DENA DE PRIORIDAD
LINEAS DE ALI-MENTACION	54 56	AUX GND AUX -V	ENTRADA ENTRADA	TIERRA AUXILIAR -12 V DC

Tab. 2.2 Señales del STD BUS en la cara de soldadura.

2.2 Descripción de las señales

Líneas duales de alimentación

Las líneas duales de alimentación permiten la distribución de la potencia digital y analógica; se pueden usar hasta cinco fuentes de potencia con dos tierras separadas, estas tierras separadas permiten aislar a los circuitos analógicos de los digitales.

Canal de datos

El canal de datos es de ocho bites, bidireccional, de tres estados y nivel activo alto. La dirección de los datos se controla normalmente por la tarjeta del procesador maestro. Las señales de lectura (RD*), escritura (WR*) y reconocimiento de interrupción (INTAK*) indican la dirección de los datos.

Es requisito que todas las tarjetas dejen al canal en un estado de alta impedancia cuando no esté en uso. La tarjeta del procesador maestro puede ceder el control del canal de datos en respuesta a una entrada de petición del canal (BUSRQ*) desde un sistema controlador alternativo, como en las transferencias de DMA o sistemas multiprocesador.

Canal de direcciones

El canal de direcciones es de 16 bites, de tres estados con nivel activo alto; suministra 16 líneas de dirección para la decodificación de memoria o puertos de E/S, donde las líneas de control de petición de memoria (MEMRQ*) y petición de E/S (IORQ*) distinguen entre estas dos operaciones. El microprocesador que se use en particular determina el número de líneas de dirección y cómo son aplicadas.

La tarjeta maestra puede dejar libre al canal como

respuesta a una entrada de petición del canal (BUSRQ*) desde un sistema controlador alternativo.

Canal de control

Al canal de control se debe la flexibilidad del STD BUS; sus señales se agrupan en cinco áreas separadas: a)memoria y puertos de E/S, b)temporización de periféricos, c)interrupción y control del canal, d)reloj y reestablecimiento y e)encadenamiento de prioridad serie.

a) Señales de control para memoria y puertos de E/S

- WR* .- Escritura a memoria o a puertos de E/S (tres estados, activa baja).

  Esta señal indica que el canal mantiene un dato válido para ser escrito en la memoria direccionada o en el dispositivo de salida. WR* es el pulso de reloj que genera el procesador para habilitar la escritura de datos a la memoria o a los puertos de salida.

- RD* .- Lectura de memoria o a puertos de E/S (tres estados, activa baja).

  Esta señal indica que el procesador u otro dispositivo controlador del canal necesita leer datos de la memoria o desde un dispositivo de E/S. El dispositivo de entrada o la memoria seleccionados utilizan esta señal para accesar los datos dentro del canal.

- IORQ* .- Petición de E/S (tres estados, activa baja).

  Esta señal indica que en las líneas de direcciones se mantiene una dirección válida de E/S para una operación de lectura o escritura a puertos; se utiliza en combinación

con RD* y WR* para designar operaciones de E/S.

- MEMRQ* .- Petición de memoria (tres estados, activa baja).

  Esta señal indica que el canal de direcciones contiene una dirección válida para operaciones de lectura o escritura a memoria.

- IOEXP .- Expansión de puertos de E/S (nivel alto expande, bajo habilita).

  Esta señal expande o habilita el direccionamiento a puertos de E/S. Un nivel activo bajo habilita el sistema primario de puertos de E/S. En sistemas sencillos esta señal se conecta generalmente a tierra.

- MEMEX .- Expansión de memoria (nivel alto expande, bajo habilita).

  Esta señal expande o habilita el direccionamiento a memoria. Un nivel activo bajo habilita el sistema primario de memoria. MEMEX permite un traslape de memoria tal como el que se encuentra en las operaciones de arranque. Los sistemas sencillos generalmente pueden conectar esta señal a tierra.

b) Señales de control para temporización de periféricos

- REFRESH*.- Señal de refresco (tres estados, activa baja).

  Esta señal refresca la memoria dinámica, se puede generar en la tarjeta del procesador o en una tarjeta de control separada. La naturaleza y temporización de la señal es una función del dispositivo de memoria o del microprocesador. En sistemas sin refresco, esta señal puede usarse para control especial de memoria. Los sistemas sencillos con

memoria estática no requieren de esta señal.

- MCSYNC* .- Sincronía con el ciclo de máquina (tres estados, activa baja).

  Esta señal ocurre una vez durante cada ciclo de máquina del procesador y sirve para mantener dispositivos periféricos especializados sincronizados con las operaciones del procesador. (El ciclo de máquina se define como la secuencia que involucra direccionamiento, transferencia de datos, y ejecución). MCSYNC* define el comienzo del ciclo de máquina. La naturaleza y temporización exacta de esta señal depende también del procesador.

- STATUS1* .- Línea de control de estado 1 (tres estados, activa baja).

  Esta señal provee temporización secundaria para dispositivos periféricos. Cuando STATUS1* está disponible se considera como una señal para identificar la búsqueda de una instrucción.

- STATUS0* .- Línea de control de estado 0 (tres estados, activa baja).

  Esta señal suministra temporización adicional para dispositivos periféricos.

La Tab. 2.3 muestra las líneas de control de temporización para algunos microprocesadores de 8 bites.

MICROPRO-CESADOR	REFRESH*	MCSYNC*	STATUS1*	STATUS0*
8051	-	ALE*	PSEN*	-
68HC11	-	E	AS*	R/W*
NCS800	REFRESH*	ALE*	SI*	SO*
8088	-	ALE*	DT/R*	SSO*
Z80	REFRESH*	RD*+WR*+INTAK*	MI*	-
6800	-	ϕ2*	VMA*	R/W*
6809	-	EOUT* (ϕ2*)	-	R/W*
6502	-	ϕ2*	SYNC*	R/W*

Tab. 2.3 Líneas de control y temporización.

c) Líneas de interrupción y control del canal.

- BUSAK* .- Reconocimiento del canal (activa baja).

  Esta señal indica que el canal está disponible para uso del controlador solicitante. El procesador controlador responde a una señal de BUSRQ* liberando el bus y enviando una señal de reconocimiento en la línea de BUSAK*, la cual ocurre al completarse el ciclo de máquina presente.

- BUSRQ* .- Petición del canal (activa baja, colector abierto).

  Esta señal causa que el procesador controlador suspenda las operaciones en el STD BUS dejando todas las líneas en alta impedancia para uso de otro procesador. El STD BUS se libera cuando se completa el ciclo de máquina presente. BUSRQ* se usa en aplicaciones que requieren acceso directo a memoria (DMA). En sistemas complejos puede ser una entrada, salida o bidireccional, dependiendo de la circuitería.

- INTAK* .- Reconocimiento de interrupción (activa baja).

  Esta señal le indica al dispositivo interruptor que la tarjeta del procesador está lista para atender la interrupción. Para el uso de interrupciones vectorizadas, el dispositivo que interrumpe coloca la dirección del vector en el canal de datos durante INTAK*. Esta señal se puede combinar con una señal de prioridad cuando múltiples controladores requieran acceso al canal. INTAK* no se usa en esquemas de interrupciones no vectorizados.

- INTRQ* .- Petición de interrupción (activa baja, colector abierto).

  Esta señal de entrada a la tarjeta del procesador interrumpe condicionalmente al programa. Es enmascarada e ignorada por el procesador a menos que se habilite deliberadamente por una instrucción de programa. Si el procesador acepta la interrupción, usualmente la reconoce por medio de INTAK*. Otras acciones dependen del tipo específico del procesador, de las instrucciones y del mecanismo de interrupción.

- WAITRQ* .- Petición de espera (activa baja, colector abierto).

  Esta señal de entrada al procesador suspende las operaciones mientras se conserve en un estado bajo. Normalmente, el procesador se mantiene en un estado que conserva una dirección válida en el canal de direcciones. WAITRQ* se puede usar también para insertar estados de espera al ciclo del procesador.

- NMIRQ* .- Interrupción no enmascarable (activa baja, colector abierto).

Esta señal es una entrada de interrupción a la tarjeta del procesador de la más alta prioridad.

d) Líneas de control de reloj y reestablecimiento

- SYSRESET* .- Reestablecimiento del sistema (activa baja).

  Esta señal es una salida desde el circuito de reestablecimiento del sistema, el cual se dispara al detectar encendido o accionar el botón de reestablecimiento.

- PBRESET* .- Botón de reestablecimiento (activa baja).

  Esta señal es una línea de entrada al circuito de reestablecimiento del sistema.

- CLOCK* .- Reloj del procesador.

  Esta señal se utiliza para la sincronización del sistema o como una fuente de reloj general.

- CNTRL* .- Control.

  Esta señal es auxiliar para temporización especial, puede ser un múltiplo de la señal de reloj del procesador, una señal del reloj de tiempo real o una entrada externa al procesador.

e) Líneas de encadenamiento de prioridad serie.

- PCO .- Salida de la cadena de prioridad (activa alta).

  Esta señal se envía a la entrada PCI de la siguiente tarjeta con prioridad menor. Una tarjeta que requiera prioridad debe mantener PCO baja.

- PCI .- Entrada de la cadena de prioridad (activa alta).

  Esta señal es provista directamente desde el PCO de la siguiente tarjeta con prioridad mayor. Un nivel alto en PCI proporciona prioridad a la tarjeta que sensa a PCI.

  Si las tarjetas no requieren prioridad PCO y PCI se unen.

2.3 Especificaciones eléctricas

Las tarjetas desarrolladas bajo este estándar requieren normalmente un voltaje de +5 V para llevar a cabo sus operaciones lógicas. Según el tipo de dispositivos que se utilicen en la tarjeta o su funcionamiento, se pueden necesitar otros voltajes de operación.

Las Tablas 2.4 y 2.5 muestran las especificaciones eléctricas de los voltajes de operación del STD BUS.

PARÁMETRO	LIMITE	REFERENCIA
Voltaje positivo aplicado a una entrada lógica o deshabilitada en tres estados	+5.5 V	GND
Voltaje negativo aplicado a una entrada lógica o deshabilitada en tres estados	-0.4 V	GND

Tab. 2.4 Rangos máximos absolutos.

VOLTAJE	TOLERANCIA	REFERENCIA
VCC (+5 V)	± 0.25 V	GND
VBB# 1 (-5 V)	± 0.25 V	GND
VBB# 2 (-5 V)	± 0.25 V	GND
AUX +V (+12 V)	± 0.50 V	AUX GND
AUX -V (-12 V)	± 0.50 V	AUX GND

Tab. 2.5 Tolerancia de los voltajes de alimentación.

2.4 Especificaciones mecánicas

En la Fig. 2.1 se aprecian las dimensiones de la tarjeta y del peine de 56 dientes que requiere el STD BUS y en las Tablas 2.6 y 2.7 se muestran las tolerancias en las dimensiones, respectivamente.

PARÁMETROS	PULGADAS		MILÍMETROS	
	NOMINAL	TOLERANCIA	NOMINAL	TOLERANCIA
LARGO	6.500	±0.025	165.10	±0.64
ANCHO	4.500	+0.005,-0.025	114.30	+0.13,-0.64
ESPESOR	0.062	+0.007,-0.003	1.58	+0.18,-0.08
ESPACIO ENTRE TARJETAS	0.500		12.70	

Tab. 2.6 Dimensiones de la tarjeta.

PARÁMETROS	PULGADAS		MILÍMETROS	
	NOMINAL	TOLERANCIA	NOMINAL	TOLERANCIA
LARGO DEL PEINE	3.610		91.70	
ANCHO DEL DIENTE	0.062	±0.003	1.58	±0.08
LARGO DEL DIENTE	0.300		7.60	
DISTANCIA ENTRE LOS CENTROS DE LOS DIENTES	0.125	±0.002	3.18	±0.05
DISTANCIA ENTRE LA ORILLA DE LA TARJETA Y EL CENTRO DEL DIENTE PROXIMO	0.117	±0.100	2.97	±0.25

Tab. 2.7 Dimensiones del peine y sus dientes.

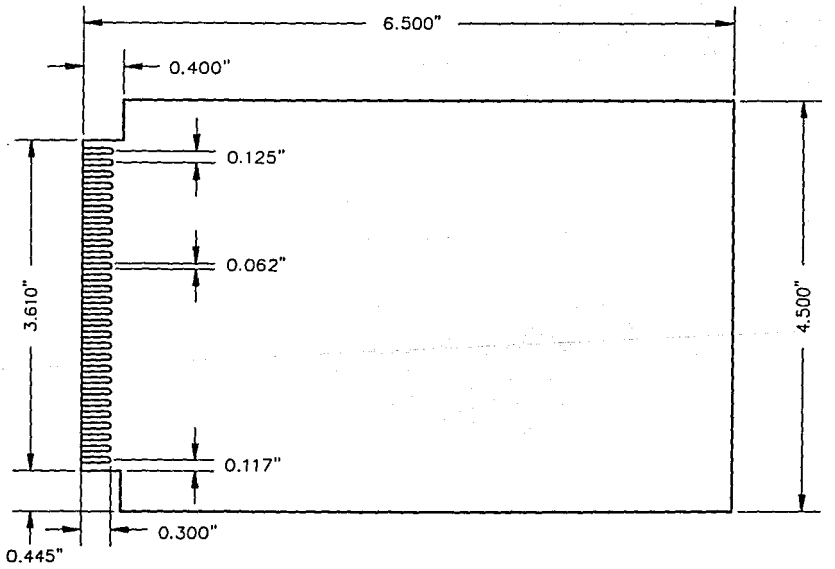


Fig 2.2 Dimensiones de la tarjeta y del peine.

3. DESCRIPCIÓN DEL MICROCONTROLADOR 8031

El 8031 es un microcontrolador de ocho bites de la familia MCS-51 de INTEL el cual tiene las siguientes características principales:

- Unidad Central de Proceso (CPU) de ocho bites.
- Oscilador interno y circuitería de reloj.
- 64 Koctetos de espacio de direccionamiento de memoria externa de datos.
- 64 Koctetos de espacio de direccionamiento de memoria externa de programa.
- 128 octetos de memoria de acceso aleatorio (RAM) interna.
- Cuatro bancos de registros internos.
- 16 líneas de entrada/salida (E/S).
- Dos temporizadores/contadores de 16 bites.
- Estructura de cinco fuentes de interrupción con dos niveles de prioridad.
- Puerto serie de doble vía.
- Procesador booleano.

La Fig. 3.1 muestra la arquitectura de este microcontrolador mediante un diagrama de bloques.

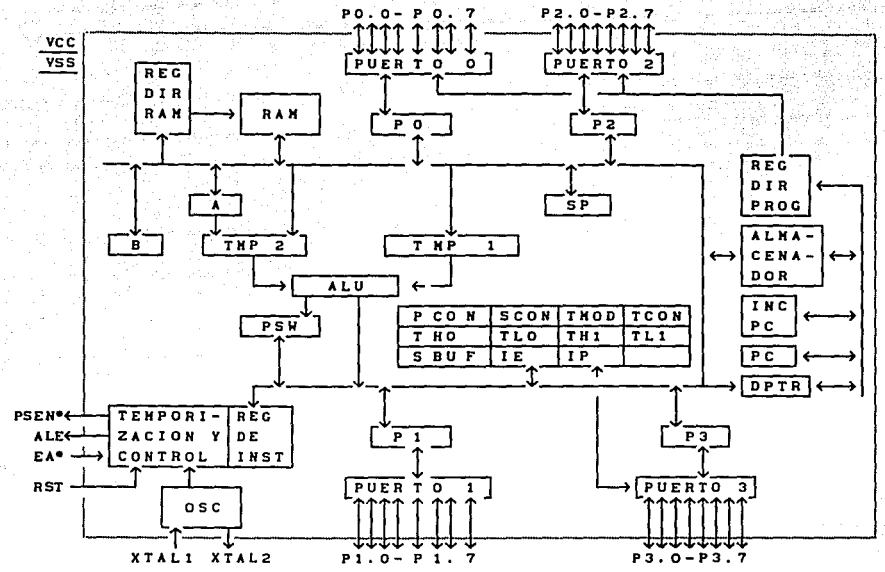


Fig. 3.1 Arquitectura del microcontrolador 8031.

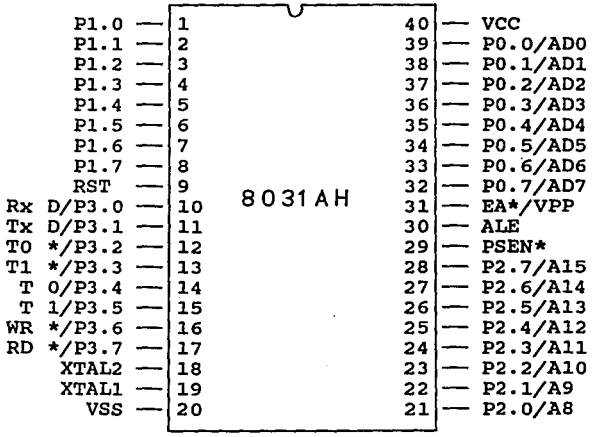


Fig. 3.2 Distribución de alfileres del 8031

3.1 Descripción de alfileres

La Fig. 3.2 presenta la distribución de los alfileres del 8031 en un encapsulado de 40 alfileres en doble fila.

VCC

Voltaje de alimentación.

VSS

Tierra del circuito.

PUERTO 0

Es un puerto bidireccional de ocho bites que opera como un canal multiplexado de direcciones bajas y datos durante accesos a memoria externa.

PUERTO 1

Es un puerto bidireccional de ocho bites.

PUERTO 2

Es un puerto bidireccional de ocho bites; emite la dirección alta durante ciclos de búsqueda de código desde memoria externa de programa y durante accesos a memoria externa de datos que utilizan direcciones de 16 bites (MOVX @DPTR). El puerto 2 emite el contenido del registro de función especial P2, durante accesos a memoria externa de datos que utilizan direcciones de ocho bites (MOVX @Ri).

PUERTO 3

Es un puerto bidireccional de ocho bites. Los alfileres de

este puerto sirven también para las funciones de características especiales listadas a continuación:

P3.0 RxD (entrada del puerto serie).
P3.1 TxD (salida del puerto serie).
P3.2 INT0* (interrupción externa).
P3.3 INT1* (interrupción externa).
P3.4 T0 (entrada externa del temporizador/contador 0).
P3.5 T1 (entrada externa del temporizador/contador 1).
P3.6 WR* (habilitador de escritura de memoria externa de datos).
P3.7 RD* (habilitador de lectura de memoria externa de datos).

RST (Reset)

Entrada de reestablecimiento. Un nivel alto en este alfiler durante dos ciclos de máquina, mientras el oscilador está funcionando, reestablece el dispositivo.

ALE* (Adress Latch Enable)

Es una línea en que se presenta un pulso de salida para la retención del octeto bajo de la dirección durante accesos a la memoria externa. En operación normal ALE* se emite a 1/6 de la frecuencia del oscilador y se puede usar para temporización externa. Cabe señalar, sin embargo, que el pulso ALE* no está presente durante cada acceso a memoria externa de datos.

PSEN* (Program Store Enable)

Señal de salida para la lectura de memoria externa de programa. PSEN* se activa dos veces cada ciclo de máquina cuando el dispositivo está ejecutando un código desde memoria externa de programa, y no está presente durante los accesos a memoria externa de datos.

EA*/VPP (External Access)

Este alfiler se debe conectar externamente a un nivel bajo.

XTAL1

Entrada del oscilador.

XTAL2

Salida del oscilador.

3.2 Organización de memoria

La memoria de programa está separada y es distinta de la memoria de datos. Cada tipo de memoria tiene un mecanismo de direccionamiento distinto, diferentes funciones y señales de control.

El acceso a la memoria externa de programa utiliza la señal PSEN* como habilitador de lectura. El acceso a la memoria externa de datos usa las señales RD* o WR* para habilitar la memoria.

La arquitectura de este microcontrolador es de tipo "Harvard", lo que significa que el procesador de memoria de datos (interno y externo) no se puede usar para códigos de programa, en otras palabras, no pueden ejecutarse instrucciones en la memoria de datos. La arquitectura está optimada para aplicaciones eficientes de control: un gran programa localizado en memoria de sólo lectura (ROM), cientos de variables en RAM y diferentes métodos de direccionamiento para cada tipo de memoria.

Los sistemas con arquitectura "Von Neumann", que no separan las memorias de datos y programa, son útiles para el desarrollo y depuración de programas. Un sistema basado en el 8031 se puede modificar para tener este tipo de arquitectura, es decir, ocupar un solo espacio de memoria fuera de la pastilla. Esto se logra conectando a las entradas de una compuerta AND las señales de control PSEN* y RD*. El CPU puede entonces escribir datos en la memoria común utilizando la señal WR* e instrucciones de

transferencia externa de datos, también puede leer instrucciones o datos con la señal de salida de la compuerta AND e instrucciones de transferencia de datos o de búsqueda de memoria de programa.

Además de los arreglos de memoria, existe otro espacio físico de direcciones con registros de propósito general de ocho bites que se encuentran conectados al canal interno de datos. En conjunto, estos registros son designados como "Registros de Funciones Especiales" (SFR).

REGISTRO	DIRECCIÓN	FUNCIÓN
P0	80 H	Puerto 0
SP	81 H	Apuntador de pila
DPL	82 H	Apuntador de datos (bajo)
DPH	83 H	Apuntador de datos (alto)
PCON	87 H	Control de potencia
TCON	88 H	Control del temp./cont.
TMOD	89 H	Control de modo del temp./cont.
TL0	8A H	Octeto bajo del temp./cont. 0
TL1	8B H	Octeto bajo del temp./cont. 1
TH0	8C H	Octeto alto del temp./cont. 0
TH1	8D H	Octeto alto del temp./cont. 1
P1	90 H	Puerto 1
SCON	98 H	Control del puerto serie
SBUF	99 H	Almacenador de datos seriales
P2	0A0 H	Puerto 2
IE	0A8 H	Control de habilit. de int.
P3	0B0 H	Puerto 3
IP	0B8 H	Control de prioridad de int.
PSW	0D0 H	Palabra de estado del programa
ACC	0E0 H	Acumulador A
B	0F0 H	Registro B

Tab. 3.1 Registros de Funciones Especiales.

ACC

El acumulador A, es un registro de ocho bites, el cual contiene un operando fuente y recibe el resultado de una operación aritmética. El acumulador puede ser la fuente o el destino para operaciones lógicas o de movimiento especial.

B

El registro B, de ocho bites, es usado en conjunto con el acumulador A durante operaciones de multiplicación y división para contener el segundo operando. Para otras instrucciones puede ser tratado como cualquier registro.

PSW (Program Status Word)

El registro PSW contiene información del estado del programa, detallado en la siguiente figura.

(MSB) (LSB)

CY	AC	F0	RS1	RS0	OV	--	P

Símbolo	Posición	Significado
CY	PSW.7	Bandera de acarreo.
AC	PSW.6	Bandera de medio acarreo.
F0	PSW.5	Bandera de propósito general.
RS1	PSW.4	Bit 1 de selección del banco de registros
RS0	PSW.3	Bit 0 de selección del banco de registros
OV	PSW.2	Bandera de saturación.
--	PSW.1	(Reservado).
P	PSW.0	Bandera de paridad.
	NOTA:	Los contenidos de (RS1,RS0) habilitan el banco de registros de trabajo, como se indica:
		(0,0) BANCO 0 (00 H-07 H)
		(0,1) BANCO 1 (08 H-0F H)
		(1,0) BANCO 2 (10 H-17 H)
		(1,1) BANCO 3 (18 H-1F H)

Fig. 3.3 PSW - Palabra de Estado del Programa.

SP (Stack Pointer)

Es el registro para el apuntador de pila de ocho bites que

indica la dirección del último octeto almacenado en la pila. Se incrementa o decrementa automáticamente en todas las instrucciones de "PUSH" o "POP" y todos los llamados a subrutinas y retornos de éstas. La pila puede residir en cualquier parte de RAM interna. Después de un reestablecimiento, este registro es inicializado en la dirección 07 H, por lo que la pila comienza en la localidad 08 H.

DPTR (Data Pointer)

Es el registro apuntador de datos de 16 bites que sirve como base para brincos indirectos y para transferencia de datos externos. Está formado por una parte alta (DPH) y una baja (DPL), que pueden ser manipuladas como registros separados o bien como un solo registro con las instrucciones especiales de carga o incremento, ambas de 16 bites.

PCON (Power Control Register)

El registro de control de potencia sirve para elegir, si así se requiere, entre dos modos de operación para bajo consumo de potencia, además contiene un bit (SMOD) para el control de la velocidad de transmisión y dos banderas de propósito general.

Los dispositivos HMOS y HCMOS proporcionan modos de operación de potencia reducida. Para la versión HCMOS existen los modos: inactivo y de baja potencia como características estándar. En las versiones de HMOS está disponible un modo de potencia reducida pero no como una característica estándar. Este modo permite reducir VCC a cero, mientras se salva el contenido de la RAM interna a través de una fuente de respaldo conectada al alfiler RST.

Para mayor claridad, los registros de funciones especiales restantes se describen en la parte correspondiente a su uso.

(MSB) (LSB)

SMOD	--	--	--	GF1	GF0	PD	IDL

Símbolo	Posición	Significado
SMOD	PCON.7	SMOD=1 la velocidad del puerto serie es duplicada en los modos 1, 2 ó 3.
--	PCON.6	(Reservado)
--	PCON.5	(Reservado)
--	PCON.4	(Reservado)
GF1	PCON.3	Bandera de propósito general.
GF0	PCON.2	Bandera de propósito general.
PD	PCON.1	PD=1 activa el modo de operación de baja potencia.
IDL	PCON.0	IDL=1 activa el modo de operación inactivo.

Fig. 3.4 PCON - Registro de Control de Potencia.

3.3 Puertos de entrada/salida

Los microcontroladores de la familia MCS-51 tienen cuatro puertos bidireccionales. El 8031 siempre utiliza los alfileres del puerto 0 (P0) y del puerto 2 (P2) para direccionamiento externo, el puerto 1 (P1) y el puerto 3 (P3) están disponibles para E/S.

En los diferentes modos de operación o expansión algunos de estos alfileres de E/S se usan para funciones especiales. Las instrucciones que accesan memoria externa utilizan a P0 como un canal multiplexado de direcciones/datos: al inicio de un ciclo de memoria externa, los ocho bites menos significativos de la dirección salen por P0, después el dato se transfiere por los mismos ocho alfileres. Las instrucciones de transferencia externa de datos que requieren una dirección de 16 bites y cualquier instrucción que accesa memoria externa de programa envían los ocho bites más significativos por P2 durante el ciclo de acceso.

Aun dentro de un solo puerto, las funciones de E/S se

pueden combinar de varias formas: puede haber entrada y salida y llevarse a cabo utilizando diferentes alfileres al mismo tiempo, o los mismos alfileres en tiempos distintos; en algunas ocasiones en paralelo y otras en serie; como alfileres de prueba o como funciones especiales adicionales (en el caso de P3).

P0, P1, P2 y P3, son los alimentadores de los puertos 0, 1, 2 y 3, respectivamente. El registro P3 se muestra a continuación:

(MSB) (LSB)

RD	WR	T1	TO	INT1	INT0	TxD	RxD

Símbolo	Posición	Significado
RD	P3.7	Salida de control de lectura de datos.
WR	P3.6	Salida de control de escritura de datos.
T1	P3.5	Alfiler de prueba o entrada externa del temp./cont. 1.
TO	P3.4	Alfiler de prueba o entrada externa del temp./cont. 0.
INT1	P3.3	Alfiler de entrada de interrupción 1.
INT0	P3.2	Alfiler de entrada de interrupción 0.
TxD	P3.1	Alfiler de transmisión de datos para el puerto serie en modo serializado. Salida de reloj en el modo 0.
RxD	P3.0	Alfiler de recepción de datos para el puerto serie en modo serializado. Alfiler de E/S de datos en el modo 0.

Fig. 3.5 P3 - Funciones de entrada/salida del puerto 3.

3.4 Temporizadores/contadores

Existen dos temporizadores/contadores de modo múltiple con registros de 16 bites divididos en un octeto alto y uno bajo; cada uno de estos octetos son llamados TH0, TL0, TH1, TL1. Cada temporizador/contador puede ser programado independientemente

para trabajar en varios modos con un registro designado TMOD y controlado con un registro TCON.

(MSB) (LSB)

GATE	C/T*	M1	M0	GATE	C/T*	M1	M0
CONT1				CONT0			

Símbolo	Significado
GATE	Con GATE=1, el temporizador/contador "x" es habilitado solamente si INTx=1 y TRx=1. Con GATE=0, el temporizador/contador "x" es habilitado siempre que TRx=1.
C/T*	Con C/T*=1, selecciona operación como contador. Con C/T*=0, selecciona operación como temporizador.
M1 M0	Bites de selección de modo.
0 0	MODO 0
0 1	MODO 1
1 0	MODO 2
1 1	MODO 3
NOTA:	Los bites 0-3 y 4-7 se utilizan para el control de los temporizadores/contadores 0 y 1, respectivamente.

Fig. 3.6 TMOD - Registro de Control de Modo del temporizador/contador.

En la función de temporizador el registro se incrementa cada ciclo de máquina; debido a que un ciclo de máquina es de 12 periodos del oscilador, la velocidad de conteo es de 1/12 la frecuencia de éste.

En la función de contador, el registro se incrementa en respuesta a una transición de 1 a 0 en el alfiler de entrada externa correspondiente, T0 o T1. Debido a que se necesitan dos ciclos de máquina para reconocer la transición de 1 a 0, la velocidad máxima de conteo es de 1/24 la frecuencia del oscilador.

(MSB) (LSB)

TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

Símbolo	Posición	Significado
TF1	TCON.7	Bandera de saturación del temporizador 1.
TR1	TCON.6	Bit de control de ejecución del temporizador 1.
TF0	TCON.5	Bandera de saturación del temporizador 0.
TR0	TCON.4	Bit de control de ejecución del temporizador 0.
IE1	TCON.3	Bandera de interrupción 1.
IT1	TCON.2	IT1=1 interrupción externa 1 activada por flanco de bajada. IT1=0 interrupción externa 1 activada por nivel bajo.
IE0	TCON.1	Bandera de interrupción 0.
IT0	TCON.0	IT0=1 interrupción externa 0 activada por flanco de bajada. IT0=0 interrupción externa 0 activada por nivel bajo.
	NOTA:	Las banderas se encienden por circuitería y se borran cuando la interrupción es servida. Los bites de control activan o desactivan por programa al temp./cont. con uno y cero, respectivamente.

Fig. 3.7 TCON - Registro de Control del temporizador/ contador.

Además de la selección como temporizador o contador, se puede seleccionar uno de cuatro modos de operación:

Los modos 0, 1 y 2, son los mismos para ambos temporizadores/contadores. El modo 3 es diferente.

MODO 0. En este modo, se configura al temporizador como un

contador de ocho bites con un preescalador de 32, el registro emplea sólo 13 bites y utiliza la bandera de interrupción TFx. La entrada para contar se conecta al temporizador cuando TRx=1 y GATE=0 o INTx*=1 (con GATE=1, el temporizador es controlado por la entrada externa INTx* para facilitar mediciones de ancho de pulso). TRx es un bit de control del registro TCON y GATE de TMOD. El registro de 13 bites está formado por los ocho bites de THx y cinco bites bajos de TLx.

MODO 1. El modo 1 es similar al modo 0, excepto que el registro temporizador usa los 16 bites.

MODO 2. Este modo configura al registro temporizador como un registro contador de ocho bites (TLx) con recarga automática. La saturación de TLx no sólo enciende TFx, sino que también recarga TLx con el contenido de THx, previamente prefijado por programa. La recarga no altera THx.

MODO 3. En este modo, el temporizador 1 detiene su cuenta. El efecto es el mismo que TR1=0. El temporizador 0 establece a TL0 y TH0 como dos contadores separados. TL0 usa los bites de control del temporizador 0: C/T*, GATE, TR0, INT0* y TF0. TH0 se programa para la función de temporizador y utiliza TR1 y TF1. Por lo tanto, TH0 controla la interrupción del temporizador 1.

3.5 Puerto serie

El puerto serie es de doble vía y de alta velocidad; se puede programar para trabajar en cuatro modos básicos:

MODO 0. Los datos seriales se transmiten y reciben a través de RxD. El alfiler TxD es la salida del reloj de

corrimiento. Se transmiten y reciben: ocho bites de datos (el bit menos significativo LSB, primero). La velocidad de transmisión es 1/12 la frecuencia del oscilador.

MODO 1. Se transmiten (a través de TxD) o reciben (por RxD) 10 bites: un bit de inicio (0), ocho bites de datos (LSB primero) y un bit de paro (1). En la recepción el bit de paro se almacena en el bit RB8 del registro SCON. La velocidad de transmisión es variable y está dada por la fórmula:

$$Vel = \frac{2^{SMOD}}{32} \times \frac{Frecuencia\ del\ oscilador}{12x[256-(TH1)]} \quad ...(1)$$

MODO 2. Se transmiten (TxD) o reciben (RxD) 11 bites: un bit de inicio (0), ocho bites de datos (LSB primero), un noveno bit de datos programable y un bit de paro. En la transmisión, el noveno bit de datos (TB8 en SCON) puede ser asignado con el valor de 0 ó 1, o con el bit de paridad (P en PSW). En la recepción este bit se almacena en RB8 de SCON, mientras el bit de paro es ignorado. La velocidad de transmisión se puede programar a 1/32 ó 1/64 la frecuencia del oscilador.

MODO 3. Este modo es similar al modo 2, excepto que la velocidad de transmisión es variable (fórmula 1).

Los registros de transmisión y recepción se accesan a través del registro de función especial SBUF. Escribiendo en SBUF se carga el registro de transmisión y leyendo SBUF se accesa a un registro de recepción, separado físicamente. En el modo 0 la recepción se inicia por la condición RI=0 y REN=1, y en los otros modos por el bit de inicio que llega si REN=1. Estos modos se controlan con el registro SCON.

(MSB) (LSB)

SM0	SM1	SM2	REN	TB8	RB8	TI	RI

Símbolo		Significado
SM0	SM1	Bites de selección de modo.
0	0	MODO 0
0	1	MODO 1
1	0	MODO 2
1	1	MODO 3
SM2		Habilita la característica de comunicación multiprocesador en los modos 2 y 3. En el modo 2 ó 3, cuando SM2=1, RI no será activada si RB8=0. En el modo 1, cuando SM2=1, RI no será activada si un bit de paro válido no fue recibido. En el modo 0, SM2 debe ser cero.
REN		Habilita/deshabilita recepción serial.
TB8		Noveno bit de datos que será transmitido en los modos 2 y 3.
RB8		Noveno bit de datos que fue recibido en los modos 2 y 3. En el modo 1, cuando SM2=0, RB8 es el bit de paro que fue recibido.
TI		Bandera de interrupción de transmisión. Encendida por circuitería al final del octavo bit en el modo 0, o al principio del bit de paro en los otros modos. Debe ser borrada por programa.
RI		Bandera de interrupción de recepción. Encendida por circuitería al fin del octavo bit en el modo 0, o a la mitad del bit de paro en los otros modos. Debe ser borrada por programa.

Fig. 3.8 SCON - Registro de Control del Puerto Serie.

3.6 Interrupciones

Existen cinco fuentes de interrupción: una del puerto serie, cuando una transmisión o recepción se completa, dos de los temporizadores, cuando ocurre una saturación, y dos de los alfileres de entrada INT0* e INT1*.

Cada fuente de interrupción se puede habilitar o deshabilitar independientemente para permitir la encuesta de distintas fuentes en algún momento determinado. Cada una puede ser clasificada para tener prioridad alta o baja; una fuente de prioridad alta puede interrumpir una rutina de servicio de prioridad baja. Estas opciones son seleccionadas con los registros de habilitación de interrupción y de control de prioridad, IE e IP respectivamente.

(MSB) (LSB)

EA	--	--	ES	ET1	EX1	ET0	EX0

Símbolo	Posición	Significado
EA	IE.7	Con EA=0, deshabilita todas las interrupciones. Con EA=1 cada fuente puede ser habilitada individualmente.
--	IE.6	(Reservado)
--	IE.5	(Reservado)
ES	IE.4	Habilita/deshabilita la interrupción del puerto serie.
ET1	IE.3	Habilita/deshabilita la interrupción de saturación del Temporizador 1.
EX1	IE.2	Habilita/deshabilita la interrupción externa 1.
ET0	IE.1	Habilita/deshabilita la interrupción de saturación del Temporizador 0.
EX0	IE.0	Habilita/deshabilita la interrupción externa 0.
	NOTA:	Un valor de 1 habilita, un 0 deshabilita.

Fig. 3.9 IE - Registro de Habilitación de Interrupción.

(MSB) (LSB)

--	--	--	PS	PT1	PX1	PT0	PX0

Símbolo	Posición	Significado
--	IP.7	(Reservado)
--	IP.6	(Reservado)
--	IP.5	(Reservado)
PS	IP.4	Define el nivel de prioridad de interrupción del puerto serie.
PT1	IP.3	Define el nivel de prioridad de interrupción del Temporizador 1.
PX1	IP.2	Define el nivel de prioridad de la interrupción externa 1.
PT0	IP.1	Define el nivel de prioridad de interrupción del Temporizador 0.
PX0	IP.0	Define el nivel de prioridad de la interrupción externa 0.
	NOTA:	Encendiendo cada uno de estos bites, se asigna prioridad alta a la interrupción, de no ser así, tiene prioridad baja.

Fig. 3.10 IP - Registro de Prioridad de interrupción.

Una fuente de interrupción no puede ser interrumpida por otra del mismo nivel de prioridad o menor, pero sí por una de mayor prioridad. En caso de que dos peticiones de interrupción del mismo nivel de prioridad ocurran simultáneamente se atenderán en el siguiente orden.

1.	IE0	De mayor prioridad
2.	TF0	
3.	IE1	
4.	TF1	
5.	RI o TI	De menor prioridad

Estas fuentes de interrupción tienen una dirección de memoria de programa particular, comienzan con la dirección 0003 H y continúan con intervalos de ocho octetos. (Ver Tab. 3.2). Cuando ocurre una interrupción, el CPU ejecuta automáticamente un llamado interno a subrutina a la dirección correspondiente. Una rutina del usuario inicia en esta localidad (o salta desde esta

localidad) y ejecuta las instrucciones para servir a esa fuente en particular. Después de completar la rutina de servicio de interrupción, la ejecución regresa al programa principal.

FUENTE DE INTERRUPCIÓN	VECTOR
Reset	0000 H
Externa 0	0003 H
Temporizador/Contador 0	000B H
Externa 1	0013 H
Temporizador/Contador 1	001B H
Puerto Serie	0023 H

Tab. 3.2 Vectores de interrupción.

3.7 Modos de direccionamiento

Las instrucciones del lenguaje ensamblador de la familia MCS-51 consisten de un nemónico de operación y de cero a tres operandos separados por comas. En las instrucciones de dos operandos, el destino se especifica primero y después la fuente. Muchas operaciones de datos de ocho bites (tales como ADD o MOV) usan inherentemente el acumulador como un operando fuente o para recibir el resultado.

El operando fuente se encuentra según cuál de los cuatro modos de direccionamiento se emplea, a saber:

- Registro: en uno de los registros de trabajo del banco habilitado.
- Directo: en una localidad de RAM interna, un puerto de E/S, o SFR's.
- Registro indirecto: en una localidad de RAM interna apuntada por un registro de trabajo Ri.
- Dato inmediato: en una constante de 8 bites incorporada dentro de la instrucción.

Direccionamiento de registro

En este modo, el programador tiene acceso a ocho registros de trabajo (R0-R7). Los tres bites menos significativos del código de operación de la instrucción, indican un registro dentro del banco usado. (Ver Fig. 3.11a).

Direccionamiento directo

El direccionamiento directo permite accesar cualquier variable dentro de RAM interna o un registro de función especial. Un octeto adicional al código de operación especifica la localidad a ser usada. (Ver Fig. 3.11b). Dependiendo del bit más significativo del octeto de dirección, se selecciona uno de los dos espacios físicos de memoria. Cuando la dirección está entre 0 y 127 (00 H-7F H), se usa una de las 128 localidades bajas de RAM interna. Todos los registros de puertos de E/S, de función especial, de control o de estado, están asignados a las direcciones entre 128 y 255 (80 H-0FF H).

Existen instrucciones que operan sobre variables Booleanas (un bit) que usan un modo de direccionamiento directo de bit comparable al modo directo. Un octeto adicional se agrega al código de operación que especifica la variable Booleana, según los mapas de dirección de bit. (Ver Figs. 3.12 y 3.13).

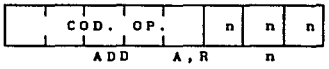
Direccionamiento de registro indirecto

En este modo, R0 y R1 de cada banco de registros pueden operar como registros apuntadores o índices, sus contenidos indican una dirección de RAM. La localidad de RAM interna así direccionada es el operando. El bit menos significativo del código de operación de la instrucción determina qué registro es usado como el apuntador. (Ver Fig. 3.11c).

Direccionamiento inmediato

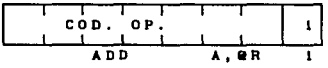
Cuando un operando fuente es una constante mas que una variable, entonces dicha constante puede ser incorporada dentro de la instrucción. Un octeto adicional a la instrucción especifica el valor usado. (Ver Fig. 3.11d).

a) Direccionamiento de registro



b) Direccionamiento directo

c) Direccionamiento de registro indirecto



d) Direccionamiento inmediato

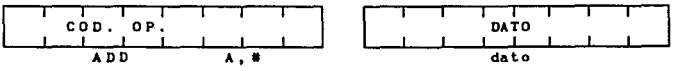


Fig. 3.11 Formatos de código de máquina de los modos de direccionamiento.

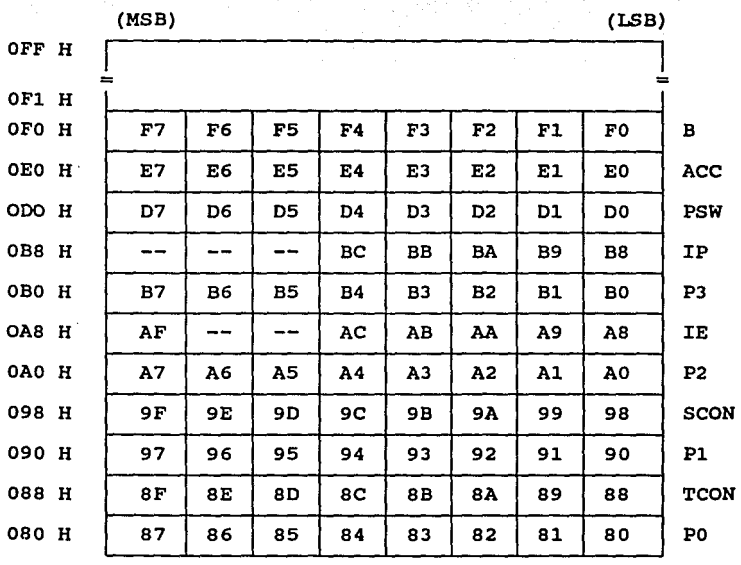


Fig. 3.12 Mapa de dirección de bit de los Registros de Funciones Especiales.

	(MSB)							(LSB)
7F H 30 H								
2F H	7F	7E	7D	7C	7B	7A	79	78
2E H	77	76	75	74	73	72	71	70
2D H	6F	6E	6D	6C	6B	6A	69	68
2C H	67	66	65	64	63	62	61	60
2B H	5F	5E	5D	5C	5B	5A	59	58
2A H	57	56	55	54	53	52	51	50
29 H	4F	4E	4D	4C	4B	4A	49	48
28 H	47	46	45	44	43	42	41	40
27 H	3F	3E	3D	3C	3B	3A	39	38
26 H	37	36	35	34	33	32	31	30
25 H	2F	2E	2D	2C	2B	2A	29	28
24 H	27	26	25	24	23	22	21	20
23 H	1F	1E	1D	1C	1B	1A	19	18
22 H	17	16	15	14	13	12	11	10
21 H	0F	0E	0D	0C	0B	0A	09	08
20 H	07	06	05	04	03	02	01	00
1F H 18 H	BANCO 3 (8 REGISTROS)							
17 H 10 H	BANCO 2 (8 REGISTROS)							
0F H 08 H	BANCO 1 (8 REGISTROS)							
07 H 00 H	BANCO 0 (8 REGISTROS)							

Fig. 3.13 Mapa de dirección de bit de RAM interna.

4. DESCRIPCIÓN DEL MICROCONTROLADOR MC68HC11

El MC68HC11 es un microcontrolador de ocho bites con capacidades periféricas internas que puede alcanzar una velocidad nominal de canal de 2 MHz. Las funciones periféricas principales son: un convertidor analógico-digital (A/D) de ocho canales y ocho bites de resolución, un acomplamiento serial asíncrono (SCI) y otro serial síncrono (SPI). El sistema temporizador gira alrededor de un contador libre de 16 bites y tiene tres líneas de entrada para captura, cinco líneas de salida por comparación y una función de interrupción en tiempo real, además de un subsistema acumulador de pulsos de ocho bites que puede contar eventos o medir periodos externos.

La tecnología HCMOS (High-density Complementary Metal-Oxide Semiconductor) usada en el MC68HC11 proporciona un tamaño reducido, alta velocidad, bajo consumo de potencia y alta inmunidad al ruido. Los sistemas con memoria interna incluyen 512 octetos de memoria programable y borrable eléctricamente (EEPROM) y 256 octetos de memoria de acceso aleatorio (RAM).

El circuito incluye circuitería de automonitoreo para protección contra errores: un sistema de vigilancia de operación apropiada de la computadora (COP) que la protege contra fallas de programa, un sistema monitor del reloj que genera un

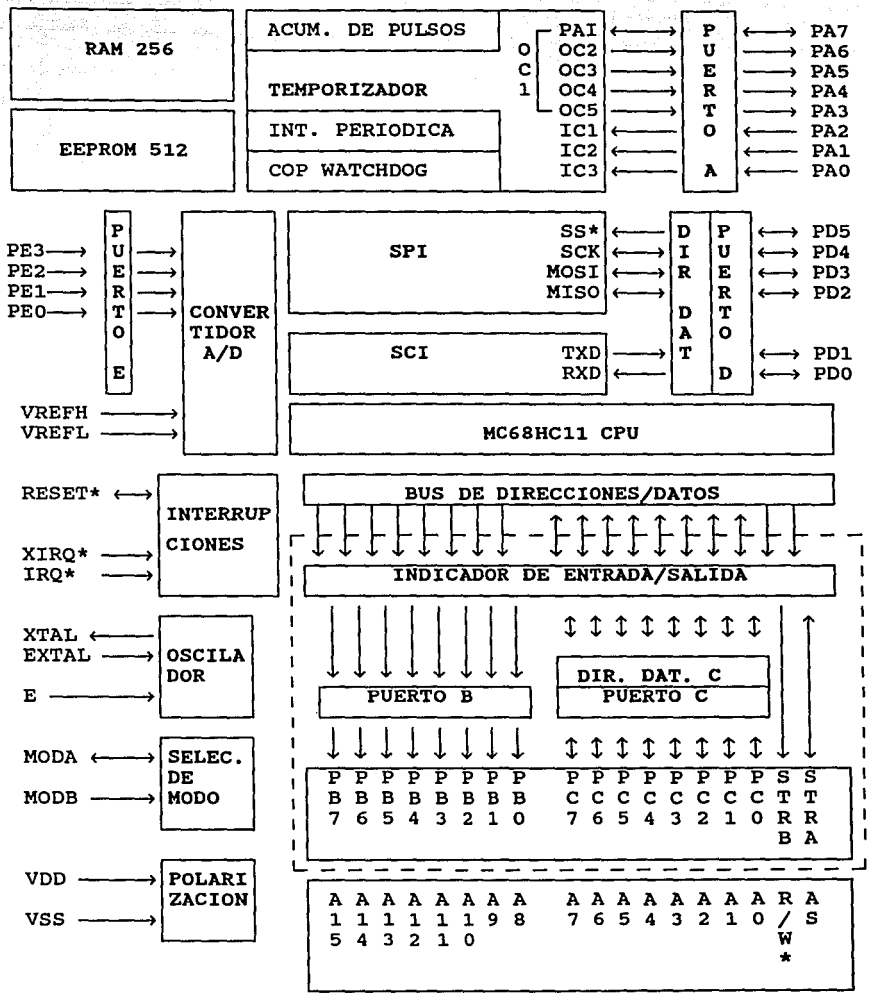


Fig. 4.1 Diagrama de bloques del MC68HC11.

reestablecimiento en caso de que el reloj se pierda o funcione demasiado lento y, finalmente, un circuito de detección de código de operación ilegal que provee una interrupción no enmascarable si se detecta un código de operación ilegal. Existen dos modos, WAIT y STOP, para ahorro de potencia que se controlan por programa y hacen que esta familia sea especialmente atractiva para aplicaciones accionadas por baterías.

La Fig. 4.1 muestra los subsistemas principales y cómo se relacionan con los alfileres del MCU (Microntroller Unit). La parte encerrada en una línea punteada indica el subsistema de entrada/salida (E/S) paralelo para el modo de operación sencillo. Las funciones de este subsistema se pierden cuando el MCU opera en los modos expandidos.

4.1 Modos de operación

Los modos de operación del MC68HC11 son: sencillo y expandido; cada modo tiene una variación normal y una especial. Las cuatro posibilidades se seleccionan con los niveles de los alfileres MODA y MODB. La variación especial del modo sencillo se denomina de arranque y la variación especial del modo expandido, modo de prueba.

Los temporizadores, el convertidor analógico/digital (A/D) y las funciones de E/S serial trabajan de la misma manera en los modos expandidos y sencillos.

El mecanismo de circuitería para la selección del modo depende de los niveles lógicos en los alfileres MODA y MODB mientras el MCU está en estado de reestablecimiento, dichos niveles determinan el estado de los bites de control de modo especial (SMOD y MDA) que a su vez controlan los circuitos lógicos involucrados en la circuitería del modo de selección. Estos bites pertenecen al registro de prioridad más alta de interrupción (HPRIO). (Ver Fig. 4.9).

La línea MODA selecciona entre los modos sencillo y expandido mientras que MODB selecciona entre la variación normal y especial del modo de operación elegido. La Tab. 4.1 muestra el estado de los bites afectados con los modos de selección.

ENTRADAS		MODO	Bites de control en HPRIO			
MODB	MODA	SELECCIONADO	RBOOT	SMOD	MDA	IRV
1	0	Sencillo normal	0	0	0	0
1	1	Expandido normal	0	0	1	0
0	0	Arranque especial	1	1	0	1
0	1	Prueba especial	0	1	1	1

Tab. 4.1 Modos de Operación.

Modo sencillo

Este modo no requiere de las funciones del canal de direcciones y datos, por lo tanto, los puertos B y C y los alfileres STRA y STRB (18 líneas en total) están disponibles como líneas de E/S de propósito general. Los registros necesarios para el control del MCU se encuentran en las memorias internas.

Modo expandido

Este modo de operación permite que tanto la memoria externa como los dispositivos periféricos se accesen por medio de un canal de direcciones/datos multiplexado en el tiempo sobre los ocho alfileres del puerto C. Durante la primera mitad de cada ciclo de canal, la dirección baja (A7-A0) está presente en ocho alfileres del puerto C y la dirección alta (A8-A15) en las líneas del puerto B durante la segunda mitad del ciclo, el puerto C se usa como el canal de datos bidireccional. La señal AS actúa como un habilitador activo alto para un circuito externo retenedor de

direcciones: la dirección será válida mientras que AS es alta y se retiene cuando AS baja. La señal R/W* indica la dirección de los datos: alta para ciclos de lectura y baja para los ciclos de escritura.

Las funciones de E/S paralela de estos 18 alfileres se pierden en los modos expandidos pero se pueden recuperar con un circuito externo llamado unidad de reemplazo de puerto (PRU) MC68HC24.

Modo de arranque

En este modo se utiliza un programa fijo contenido en ROM que sirve para que el usuario cargue programas a través del SCI dentro de la RAM interna y después pueda ejecutarlos. El programa cargado se puede usar para una variedad de tareas, tales como grabar valores de calibración dentro la EEPROM interna o llevar a cabo diagnósticos en un módulo terminado. Mientras el MCU opera en este modo, se puede escribir sobre el bit de control de selección de modo A (MDA) que se encuentra en el registro HPRIO, para conectar el canal multiplexado. Esta posibilidad hace que el modo de arranque sea útil en los sistemas expandido y sencillo.

Modo de prueba

El modo de prueba fue desarrollado principalmente para pruebas de fábrica. Sin embargo, hay algunos casos donde el usuario puede utilizarlo incluyendo la programación del registro CONFIG, programación de datos de calibración dentro de la EEPROM, emulación y depuración.

4.2 Descripción de alfileres

ALFILERES DE SUMINISTRO DE POTENCIA (V_{DD} y V_{SS})

V_{DD} es la entrada de potencia positiva, y V_{SS} es tierra. El

MCU usa una sola fuente de potencia, pero en algunas aplicaciones, puede utilizar otra fuente opcional para la referencia del A/D y/o para la batería de respaldo de la memoria RAM interna.

Un sistema de modo expandido típico debe incluir un capacitor de 1 μF y un capacitor separado de 0.01 μF. Ambos capacitores deben estar lo más cerca posible (física y eléctricamente) al MCU y deben tener buenas características para altas frecuencias (por ejemplo como las que muestran los capacitores de tantalio). Algunos sistemas que operan en modo sencillo y con carga ligera pueden trabajar correctamente con un solo capacitor de paso de 0.1 μF.

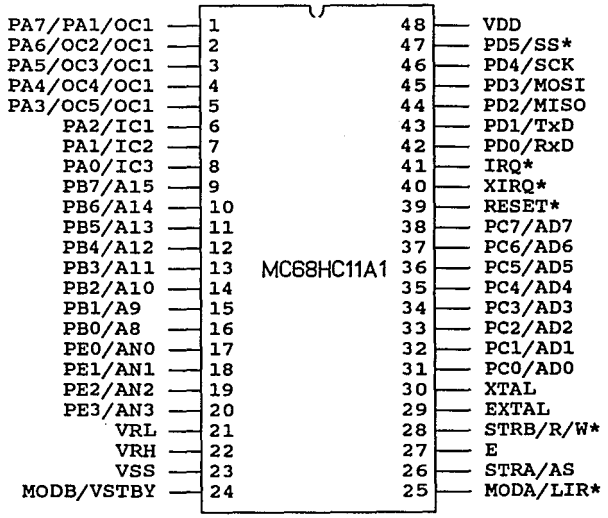


Fig. 4.2 Distribución de alfileres del MC68HC11 en encapsulado de doble fila.

ALFILERES DE SELECCIÓN DE MODO (MODB/V_{STBY} y MODA/LIR*)

El alfiler MODB/V_{STBY} funciona como entrada de selección de modo y como una entrada para fuente de potencia. El alfiler MODA/LIR* se usa para seleccionar el modo de operación del MCU mientras que éste se reestablece y opera como una señal de salida de diagnóstico mientras que el MCU está ejecutando instrucciones. Después del reestablecimiento, los alfileres del modo de selección no tienen más influencia sobre el modo de operación del microcontrolador.

La función alternativa como señal de salida del alfiler MODA indica la carga de instrucción (LIR*) y se puede usar como un monitor en un analizador lógico durante la depuración de un sistema; el programa se puede seguir fácilmente debido a que el indicador de estado muestra cuando comienza cada instrucción. En aplicaciones de modo sencillo, este alfiler simplemente se conecta a V_{SS} porque en este modo de operación no existe visibilidad externa de los canales de datos y direcciones y no hay razón para supervisarlos. En sistemas en modo expandido, el alfiler MODA/LIR* se conecta a un nivel alto a través de una resistencia de sostén de 4.7 KΩ.

El alfiler MODB tiene una función alternativa para mantener el contenido de la RAM cuando V_{DD} no está presente (V_{STBY}).

ALFILERES DE CRISTAL OSCILADOR Y RELOJ (EXTAL, XTAL y E)

Los alfileres del oscilador se pueden conectar a un cristal o reloj externo con salida CMOS. La frecuencia en ambos casos debe ser cuatro veces mayor que la frecuencia del canal deseada que está presente en la línea E, la señal en la misma actúa como referencia de temporización básica; cuando E es baja ocurre un proceso interno en el procesador y cuando es alta, se direccionan los datos. La señal de reloj E oscila mientras el oscilador esté activo. (Ver la instrucción STOP más adelante).

El oscilador interno en el MC68HC11 consiste de una compuerta NAND de dos entradas, una de estas entradas está conectada a la señal que deshabilita al oscilador cuando el MCU está en el modo de paro, la otra entrada de la compuerta es el alfiler EXTAL del MCU y la salida, el alfiler XTAL.

ALFILER DE REESTABLECIMIENTO (RESET*)

Es una señal de control bidireccional activa baja que se usa como una entrada para dar al MC68CH11 un estado inicial conocido y como una salida para indicar que el monitor del reloj o el circuito COP han detectado una falla interna.

ALFILERES DE INTERRUPCIÓN (XIRQ*, IRQ*)

El alfiler XIRQ* provee un medio para manejar interrupciones no enmascarables después de la inicialización. Durante el reestablecimiento, el bit X en el registro de condición de código (CCR) se enciende y cualquier interrupción es enmascarada hasta que se apague dicho bit por programa. La entrada XIRQ* se usa generalmente como una interrupción para detectar la pérdida de potencia.

La entrada IRQ* sirve para tratar interrupciones asíncronas enmascarables. IRQ* se selecciona por programa (registro OPTION) y tiene opción para dispararse por nivel o por flanco. Después del reestablecimiento IRQ* se configura siempre para operar con nivel.

Además de XIRQ* e IRQ*, también se pueden usar otros cinco alfileres en el dispositivo para generar interrupción al MCU. Los alfileres PA0/IC3, PA1/IC2, PA2/IC1, PA7/PAI/OC1 y AS/STRA están asociados con periféricos internos tales como el temporizador y sistemas de manejo de E/S con la ventaja que cada una de estas cinco interrupciones es enmascarable independientemente con un bit de control local así como con el bit global I del CCR; para

cada una de estas cinco señales se tiene un indicador de estado y uno de interrupción pendiente, que se puede borrar sin haber sido servida.

ALFILERES DE REFERENCIA A/D Y PUERTO E (V_{REFL}, V_{REFH}, PE7 - PE0)

Los alfileres V_{REFH} y V_{REFL} sirven para aplicar los voltajes de referencia a la circuitería del convertidor A/D. Estos alfileres están conectados normalmente a V_{DD} y V_{SS} a través de un filtro paso bajas para aislar el ruido de la fuente de potencia lógica de las mediciones analógicas. Debe haber al menos 2.5 V entre V_{REFL} y V_{REFH} y no existe impedimento para mantener V_{REFH} abajo de V_{DD}.

Los alfileres del puerto E se usan como entradas de propósito general y/o analógicas. Las funciones analógica y digital de este puerto normalmente no interfieren entre ellas, por lo tanto, se puede usar cualquier combinación de alfileres como entradas digitales mientras se utilizan los alfileres restantes como entradas analógicas.

ALFILERES DEL PUERTO A TEMPORIZADOR

El puerto A incluye tres alfileres de entrada, cuatro de salida y uno bidireccional. Las líneas de entrada PA0/IC3, PA1/IC2, PA2/IC1 se pueden configurar para detectar flancos de subida, de bajada o ambos y capturar el valor del temporizador de acuerdo con el nivel de la señal presente en el alfiler. Las líneas de salida PA3/OC5/OC1, PA4/OC4/OC1, PA5/OC3/OC1 y PA6/OC2/OC1 sirven para presentar valores preestablecidos en ellas cuando ocurran valores especificados del temporizador; siempre que una función de salida por comparación se habilita, ese alfiler no se puede usar como salida de propósito general. Estas cuatro líneas se pueden controlar por una sola salida por comparación (OC1). La línea PA7/PAI/OC1 se puede usar como un

alfiler de E/S de propósito general, como una entrada del acumulador de pulsos o como un alfiler de salida OC1.

ALFILERES DEL PUERTO D SERIAL

El puerto D incluye seis alfileres de E/S bidireccionales de propósito general que se pueden configurar como entradas o como salidas individualmente. Cuando el receptor del SCI se habilita, el alfiler PD0/RxD es una entrada dedicada a la función RxD y cuando el transmisor se habilita, el alfiler PD1/TxD es una salida dedicada a la función TxD. Si el SPI se habilita los alfileres PD2-PD5 se usan para las funciones MISO, MOSI, SCK y SS*, respectivamente. Si el SPI se habilita en modo maestro el alfiler PD5/SS* se puede usar como una salida de propósito general si se enciende el bit DDRD5 correspondiente.

Los seis alfileres de este puerto, se pueden configurar para operar en modo de OR alambrada. Esta opción permite conectar dos o más salidas sin contención. Si se utiliza este modo, se requieren resistencias de sostén externas en todas las salidas del puerto D. Este modo no afecta el uso de los alfileres como entradas.

PUERTOS B, C y ALFILERES STRA y STRB

Cuando el MCU opera en modo sencillo estos 18 alfileres se utilizan para E/S de propósito general; si el modo es expandido, estos alfileres forman un canal multiplexado de direcciones/datos con líneas de control para habilitación de direcciones (AS) y lectura/escritura (R/W*).

En los modos sencillos, el puerto B de salida y el puerto C bidireccional son de ocho bites. Algunos bites del puerto C se pueden configurar como salidas y el resto como entradas. (Ver Fig. 4.4). Las funciones de manejo de E/S automatizadas están asociadas con estos puertos; para este fin, se utilizan los

alfileres STRA y STRB como líneas de habilitación y control.

El alfiler STRA es una entrada detectora de nivel que ocasiona que los datos del puerto C se retengan dentro de un registro especial interno. El nivel activo para este alfiler se selecciona por programa. (Ver Fig. 4.3). Si las funciones de habilitación y control no se usan, STRA puede ser una entrada de interrupción detectora de nivel pero no puede ser una entrada de propósito general.

PIOC $ 1002

7							0
STAF	STAI	CWOM	HNDS	OIN	PLS	EGA	INVB

STAF =1 se enciende con el nivel activo del alfiler STRA.
=0 inactiva.
STAI =1 petición de interrupción por circuitería cuando STAF=1.
=0 no se genera interrupción por circuitería.
CWOM =1 activa salidas del puerto C como OR alambrada.
=0 salidas del puerto C normales.
HNDS =1 modos de aviso-reconocimiento.
=0 modo de habilitación simple.
OIN =1 selecciona puerto C como salida de aviso-reconocimiento.
=0 selecciona puerto C como entrada de aviso-reconocimiento.
PLS =1 pulsos en STRB.
=0 nivel activo en STRB.
EGA =1 selecciona flanco de subida para STRA.
=0 selecciona flanco de bajada para STRA.
INVB =1 STRB activa alta.
=0 STRB activa baja.

Fig. 4.3 Registro PIOC.

El alfiler STRB es un habilitador de salida asociado con las funciones de manejo de E/S de los puertos B y C. Si las

funciones de control no se utilizan, STRB puede ser una salida de propósito general, aunque es más difícil de controlar que un alfiler normal de salida. La pareja STRA y STRB está presente para funcionar como una dupla aviso-reconocimiento (STRB-ACK).

DDRC $ 1007

7							0
DDC7	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0

=1 salidas.
=0 entradas.

Fig. 4.4 Registro DDRC.

4.3 Modelo de programación

La Fig. 4.5 muestra los registros del CPU disponibles para el programador; estos siete registros son direccionables directamente y no como localidades de memoria.

ACUMULADORES A, B y D

Los acumuladores de propósito general A y B son de ocho bites y se usan para contener operandos y resultados de cálculos aritméticos, lógicos o de manipulación de datos. Ambos acumuladores se pueden concatenar para formar un solo acumulador de 16 bites llamado registro D, que es accesible en algunas instrucciones.

REGISTROS ÍNDICES X y Y

Los registros índices de 16 bites X (IX) y Y (IY) se usan para el modo de direccionamiento indexado. Las instrucciones que

involucran a IY toman un octeto extra de código objeto y un ciclo extra de tiempo de ejecución comparados con las instrucciones equivalentes que usan IX. El segundo registro índice (IY) es especialmente útil para movimientos y cálculos que requieren operandos en dos tablas separadas.

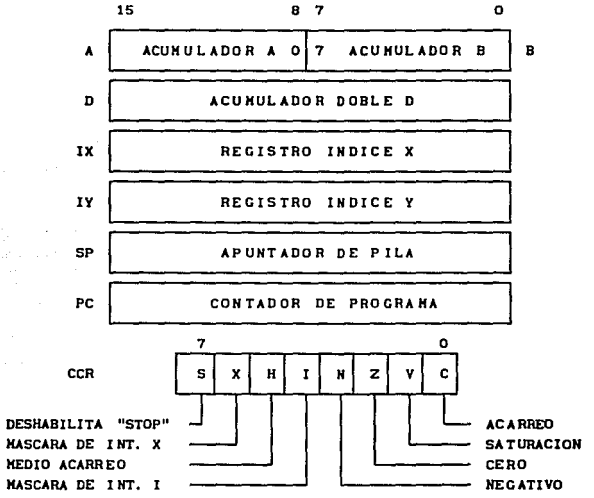


Fig. 4.5 Modelo de Programación.

SP (Stack Pointer)

El apuntador de pila se puede colocar en cualquier lugar dentro de los 64 Koctetos de espacio direccionable. Se incrementa y decrementa automáticamente con las instrucciones de "PULL" y "PUSH", respectivamente. En cualquier momento, este registro mantiene la dirección de 16 bites de la próxima localidad libre en la pila. La pila se usa para llamados a subrutina, interrupciones y almacenamiento temporal de datos.

PC (Program Counter)

El contador de programa es un registro de 16 bites que mantiene la dirección de la siguiente instrucción a ser ejecutada.

CCR (Condition Code Register)

El registro de condición de código contiene cinco indicadores de estado, dos bites de interrupciones enmascarables, y un bit deshabilitador de STOP.

Las cinco banderas de estado reflejan los resultados de operaciones aritméticas y otras del CPU, estas banderas son: medio acarreo (H), negativo (N), cero (Z), saturación (V) y acarreo (C). La bandera de medio acarreo se usa para operaciones aritméticas BCD y se afecta solamente por las instrucciones ABA, ADD, y ADC. Los bites de estado N, Z, V y C, permiten saltos basados en los resultados de una operación previa.

El bit N refleja el estado del bit más significativo (MSB) de un resultado. Para una representación en complemento a dos, un número es negativo cuando el MSB es un 1 y positivo cuando es 0.

El bit Z se enciende cuando todos los bites del resultado son ceros.

El bit V se usa para indicar si ocurrió una saturación del complemento a dos como resultado de una operación.

El bit C se utiliza para indicar si ocurrió un acarreo en operaciones de suma o resta. También sirve como una bandera de error para operaciones de multiplicación y división. Las instrucciones de corrimiento y rotación operan a través de este bit.

El bit S permite deshabilitar la instrucción STOP.

El bit I es un máscara global que deshabilita todas las fuentes de interrupción enmascarables.

El bit X se utiliza para deshabilitar interrupciones a través del pin XIRQ*.

4.4 Modos de direccionamiento

En el CPU se pueden usar seis modos de direccionamiento para referenciar la memoria: inmediato, directo, extendido, indexado, inherente y relativo.

Inmediato

En el modo de direccionamiento inmediato, el argumento real se encuentra inmediatamente después del código de operación de la instrucción; el número de octetos lo marca el tamaño del registro que se emplea por lo que estas instrucciones son de dos o tres octetos.

Extendido

En el modo de direccionamiento extendido, la dirección efectiva de la instrucción aparece explícitamente en los dos octetos siguientes al código de operación. La longitud de la mayoría de las instrucciones que usan este modo es de tres octetos.

Directo

En el modo de direccionamiento directo, el octeto menos significativo de la dirección efectiva de la instrucción aparece en el octeto siguiente al código de operación. El octeto más significativo de la dirección efectiva se asume como $ 00 y no

está incluído como un octeto de la instrucción. Este hecho limita el uso de este modo a operandos en un área de memoria de $ 0000 a $ 00FF (llamada página directa). Este modo es llamado algunas veces modo de direccionamiento de página cero y usa un octeto menos de espacio de memoria de programa que las instrucciones equivalentes que utilizan el direccionamiento extendido. La longitud de la mayoría de las instrucciones que usan este modo es de dos octetos.

Indexado

En el modo de direccionamiento indexado, los registros índices X o Y se usan en el cálculo de la dirección efectiva. En este caso, la dirección efectiva es variable y depende del contenido corriente de los registros índices y una combinación del corrimiento de ocho bites contenido en la instrucción. Este modo se puede usar para referenciar cualquier localidad de memoria en el espacio direccionable de 64 Koctetos.

El corrimiento es un valor no signado de un octeto que, cuando se suma al valor actual en el registro índice, produce la dirección efectiva del operando sin alterar el registro índice. Debido a que el octeto de corrimiento es no signado sólo se pueden especificar valores positivos en el rango [0,255].

Las instrucciones que usan este modo son usualmente de dos o tres octetos.

Las instrucciones de manipulación de bit soportan los modos de direccionamiento directo e indexado, pero no el extendido.

Inherente

En el modo de direccionamiento inherente el CPU conoce todo lo necesario para ejecutar la instrucción. Los operandos (si los hay) son registros del CPU y no necesitan ser direccionados. Las

instrucciones que usan este modo son usualmente de uno o dos octetos.

Relativo

El modo de direccionamiento relativo se usa sólo para instrucciones de salto. Estas instrucciones generan dos octetos de código máquina: uno para el código de operación y otro para el corrimiento relativo. El corrimiento es signado, lo cual permite saltos en ambas direcciones dentro de un rango de [-128,+127] octetos. Si la condición de salto es verdadera, se agrega el corrimiento al contenido del PC para formar la dirección efectiva de salto; de otro modo, procede el control a la instrucción inmediata siguiente de la instrucción de salto.

4.5 Organización de memoria

El MC68HC11 tiene un solo espacio direccionable de memoria de 64 Koctetos. Incluye 256 octetos de memoria RAM estática interna que se usa para el almacenamiento de variables e información temporal, 64 octetos de RAM para registros de control, 512 octetos de EEPROM interna donde se pueden grabar o borrar datos bajo control de programa. La EEPROM se utiliza para información semipermanente, tal como tablas de calibración, datos personalizados, etc. También se puede usar para memoria de programa sin necesidad de otra fuente de voltaje además de la normal de V_{DD} (5 V).

Los 256 octetos de RAM y los 64 octetos de registros, se pueden mapear al principio de cualquier bloque de 4 Koctetos en el espacio direccionable de 64 Koctetos. Esta ubicación permite utilizar el modo de direccionamiento directo ahorrando un octeto de memoria de programa y un ciclo de tiempo de ejecución. Por omisión, se encuentran en las localidades $ 0000-$ 00FF y $ 1000-$ 103F, respectivamente. El registro INIT permite programar la posición de la memoria RAM interna y de los 64

octetos de espacio de registros de control.

Los 512 octetos de EEPROM están ubicados en las localidades $ B600-$ B7FF. Este bloque está arreglado lógicamente en 32 renglones de 16 octetos cada uno. El primer renglón ocupa las localidades $ B600-$ B60F, el segundo las $ B610-$ B61F, etc. Estas localidades se pueden borrar individualmente, por renglones o las 512 de una sola vez. Además del bloque de 512 existe un octeto de EEPROM llamado registro CONFIG que se usa para el control de algunas características básicas del MCU. El registro PPROG controla la programación y el borrado de la EEPROM interna.

CONFIG $ 103F

7							0
0	0	0	0	NOSEC	NOCOP	ROMON	EEON

NOSEC =1 deshabilita modo de seguridad de la EEPROM.
=0 habilita modo de seguridad.
NOCOP =1 deshabilita sistema COP.
=0 habilita sistema COP.
ROMON =1 habilita los 8 Koctetos de memoria de programa interna.
=0 deshabilita ROM interna y no ocupa espacio en el mapa de memoria.
EEON =1 habilita memoria EEPROM interna.
=0 deshabilita la EEPROM y no ocupa espacio en el mapa de memoria.

Fig. 4.6 Registro CONFIG.

INIT $ 103D

7							0
RAM3	RAM2	RAM1	RAM0	REG3	REG2	REG1	REG0

RAM3-RAM0.- Posición de la memoria RAM en el mapa.
REG3-REG0.- Posición del bloque de registros de 64 octetos.

Fig. 4.7 Registro INIT.

PPROG $ 103B

7							0
ODD	EVEN	0	BYTE	ROW	ERASE	EELAT	EEPGM

ODD.- Programa renglones pares en la mitad de la EEPROM.
EVEN.- Programa renglones impares en la mitad de la EEPROM.
BYTE, ROW.- Bites de selección del tipo de operación de borrado.
Estos bites no tienen significado cuando ERASE=0.
ERASE = 1 lectura normal o modo de programa.
= 0 modo de borrado.
EELAT = 1 canal de direcciones y datos de la EEPROM configurado para programa o borrado.
= 0 canales de la EEPROM configurados para lecturas.
EEPGM = 1 habilita el voltaje de programación.
= 0 deshabilita el voltaje de programación.

Fig. 4.8 Registro PPROG.

4.6 Interrupciones

El CPU ejecuta instrucciones secuencialmente, sin embargo soporta la ejecución asíncrona de conjuntos de instrucciones en respuesta a eventos asíncronos o interrupciones de dispositivos periféricos. Una interrupción causa que el flujo normal del programa se suspenda tan pronto finaliza la instrucción

corriente; la lógica de interrupción almacena los contenidos de todos los registros del CPU dentro de la pila para que estos valores se puedan recuperar después de que la interrupción haya sido servida, enseguida, el vector de la fuente de interrupción pendiente de mayor prioridad se carga en el PC y la ejecución continúa con la primera instrucción de la rutina de servicio de interrupción. Una interrupción finaliza con una instrucción de retorno de interrupción (RTI), la cual origina que los registros del CPU y la dirección de regreso se recuperen de la pila para que el programa interrumpido pueda continuar.

FUENTE DE INTERRUPCIÓN	MÁSCARA LOCAL
Saturación del temporizador	TOI
Saturación del acumulador de pulsos	PAOVI
Flanco de entrada del acumulador de pulsos	PAII
Transferencia completa del SPI	SPIE
Sistema SCI	Tab. 4.2.1
IRQ*	Tab. 4.2.2
Interrupción de tiempo real	RTII
Captura por entrada 1 del temporizador	IC1I
Captura por entrada 2 del temporizador	IC2I
Captura por entrada 3 del temporizador	IC3I
Captura por ent. 4/salida por comp. 5	I4O5I
Salida por comparación 1 del temporizador	OC1I
Salida por comparación 2 del temporizador	OC2I
Salida por comparación 3 del temporizador	OC3I
Salida por comparación 4 del temporizador	OC4I

Tab. 4.2 Fuentes de interrupción enmascarables.

Las interrupciones se pueden habilitar o deshabilitar con los bites de máscara (X e I) del registro CCR y con los bites de

máscara de habilitación local de algunos registros de control internos. Existen fuentes de interrupción que están siempre habilitadas y se denominan interrupciones no enmascarables. El alfiler de petición de interrupción no enmascarable (XIRQ*) se deshabilita después del reestablecimiento, la interrupción de detección de código de operación ilegal proporciona integridad al sistema, la interrupción por programa (SWI) es una instrucción más que una interrupción asíncrona.

Los sistemas periféricos internos generan interrupciones enmascarables, las cuales se reconocen solamente cuando el bit I del registro CCR está apagado. Las interrupciones enmascarables tienen por omisión una prioridad predeterminada, sin embargo, cualquier fuente se puede elevar a la prioridad más alta por medio del registro HPRIO.

CAUSA DE INTERRUPCIÓN	MÁSCARA LOCAL
Registro receptor de datos lleno	RIE
Sobreescritura del receptor	RIE
Detección de línea desocupada	ILIE
Registro transmisor de datos vacío	TIE
Transmisión completa	TCIE

Tab. 4.2.1 Interrupciones del sistema SCI.

CAUSA DE INTERRUPCIÓN	MÁSCARA LOCAL
Alfiler externo	----
Manejo de E/S paralelo	STAI

Tab. 4.2.2 Interrupciones de IRQ*.

Estas interrupciones enmascarables se describen a lo largo del texto en cada uno de los subsistemas del MCU.

FUENTE DE INTERRUPCIÓN	PSEL			
	3	2	1	0
Saturación del temporizador	0	0	0	0
Saturación del acumulador de pulsos	0	0	0	1
Nivel de entrada del acumulador de pulsos	0	0	1	0
Transferencia completa del SPI	0	0	1	1
Sistema serial SCI	0	1	0	0
Reservado (por omisión para IRQ*)	0	1	0	1
IRQ*	0	1	1	0
Interrupción de tiempo real	0	1	1	1
Captura por entrada 1 del temporizador	1	0	0	0
Captura por entrada 2 del temporizador	1	0	0	1
Captura por entrada 3 del temporizador	1	0	1	0
Salida por comparación 1 del temporizador	1	0	1	1
Salida por comparación 2 del temporizador	1	1	0	0
Salida por comparación 3 del temporizador	1	1	0	1
Salida por comparación 4 del temporizador	1	1	1	0
Captura por ent. 4/salida por comp. 5	1	1	1	1

Tab 4.3 Alta prioridad de interrupción según PSEL3-PSEL0.

HPRIO $ 103C

7							0
RBOOT	SMOD	MDA	IRV	PSEL3	PSEL2	PSEL1	PSEL0

RBOOT =1 programa de arranque en las localidades $ BF40 a $ BFFF.
=0 programa de arranque deshabilitado.
SMOD =1 selecciona variación de modo especial.
=0 selecciona variación de modo normal.
MDA =1 selecciona modo expandido o de prueba.
=0 selecciona modo sencillo o de arranque.
IRV =1 datos manejados sobre el canal externo durante lecturas internas.
=0 datos de lecturas internas no visibles en el canal.
PSEL3-PSEL0.- Bites de selección de prioridad para fuentes de interrupción enmascarables. (Ver Tab. 4.3).

Fig 4.9 Registro HPRIO.

4.7 Reestablecimientos

El reestablecimiento se utiliza para forzar al MCU a ejecutar un conjunto de condiciones iniciales y comenzar con la ejecución de instrucciones a partir de una dirección predeterminada.

Existen cuatro sistemas internos separados que pueden detectar fallas en el sistema del MCU y generar un nivel en el alfiler RESET* para reinicializar al sistema; se utilizan vectores separados para distinguir entre las causas que originan un reestablecimiento. Las cuatro fuentes posibles de reestablecimiento son: a)un circuito interno que detecta el flanco de subida en VDD e inicializa un reestablecimiento de encendido; b)la aplicación de un nivel bajo en el alfiler RESET* para inicializar un reestablecimiento externo requerido por el

usuario; c)un sistema temporizador de vigilancia (COP) que monitorea la ejecución apropiada de un programa: si el programa no da respuesta al temporizador dentro de su periodo de tiempo de espera se genera un reestablecimiento; y d)un circuito interno que monitorea la frecuencia del reloj y genera un reestablecimiento cuando el reloj se detiene o funciona demasiado lento.

El sistema temporizador COP está diseñado para detectar errores en el procesamiento de un programa. Este sistema se habilita por medio del bit NOCOP del registro CONFIG.

El periodo de tiempo de espera del COP se maneja a través de los bites CR1 y CR0 del registro OPTION, según se muestra en la Tab 4.4. Después del reestablecimiento estos bites son ceros, lo cual selecciona los tiempos de espera más cortos.

OPTION $ 1039

7							0
ADPU	CSEL	IRQE	DLY	CME	0	CR1	CR0

ADPU =1 conecta potencia al sistema A/D.
=0 desconecta potencia al sistema A/D.

CSEL =1 A/D y EEPROM utilizan oscilador interno RC.
=0 A/D y EEPROM utilizan el sistema de reloj E.

IRQE =1 IRQ opera con flancos de bajada.
=0 IRQ opera con nivel bajo.

DLY =1 impone un retardo al salir del modo de paro.
=0 sin retardo.

CME =1 relojes lentos o detenidos ocasionan reestablecimiento.
=0 deshabilitado.

CR1, CR0.- Bites de selección de velocidad del temporizador COP. (Ver Tab. 4.4).

Fig. 4.10 Registro OPTION.

CR1	CR0	E / 2^{15} dividido por	FRECUENCIA DEL CRISTAL		
			2^{23} Hz	8 MHz	4 MHz
			TIEMPO DE ESPERA NOMINAL		
0	0	1	15.625 ms	16.384 ms	32.768 ms
0	1	4	62.500 ms	65.536 ms	131.070 ms
1	0	16	250.000 ms	262.140 ms	524.290 ms
1	1	64	1.000 s	1.049 s	2.100 s
			2.1 MHz	2.0 MHz	1.0 MHz
			FRECUENCIA DEL CANAL (RELOJ E)		

Tab. 4.4 Tiempos de espera del sistema COP.

El temporizador COP se debe reestablecer como sigue: 1) Escribir $ 55 en el registro COPRST para armar el mecanismo de borrado. 2) Escribir $ AA en el mismo registro para borrar el temporizador del COP.

Debido a que el temporizador del COP está basado en el reloj del MCU, el sistema de vigilancia no puede detectar errores que causen que el reloj del MCU se detenga. El sistema monitor de reloj se puede usar como un respaldo para forzar un reestablecimiento si los relojes del MCU fallan. El circuito monitor de reloj está basado en el retardo producido por un circuito RC. Si no se detectan niveles en el reloj del MCU dentro de él, el monitor de reloj puede generar opcionalmente un reestablecimiento. La función de este monitor se habilita por medio del bit de control CME del registro OPTION. Una frecuencia de reloj E menor de 10 KHz se detectará como un error; un posible uso del monitor es proteger la ejecución no intencional de la instrucción STOP.

Mientras el MCU opera en los modos de prueba o arranque, los reestablecimientos desde los sistemas COP y monitor de reloj

se deshabilitan inicialmente con un uno en el bit DISR del registro TEST1.

TEST1 $ 103E

7							0
TILOP	0	OCCR	CBYP	DISR	FCM	FCOP	TCON

TILOP =1 habilita prueba de código de operación ilegal.
=0 deshabilita la función.
OCCR =1 salida del estado del CCR al puerto A.
=0 el puerto A opera en los modos normales.
CBYP =1 el temporizador de corrida libre de 16 bites se divide en dos y se desacopla el preescalador.
=0 el sistema temporizador opera normalmente.
DISR =1 deshabilita los reestablecimientos del COP y del monitor del reloj.
=0 los reestablecimientos del COP y del monitor del reloj operan normalmente.
FCM * =1 fuerza un reestablecimiento del monitor del reloj.
=0 el sistema opera normalmente.
FCOP* =1 fuerza un reestablecimiento del sistema de vigilancia COP.
=0 el sistema opera normalmente.
TCON =1 configuración de prueba sin considerar al registro CONFIG.
=0 las opciones de configuración son controladas por el registro CONFIG.

* El bit de control DISR tiene prioridad sobre este bit e inhibe las funciones del reestablecimiento forzado.

Fig. 4.11 Registro TEST1.

4.8 Sistema temporizador

El sistema temporizador está basado en un contador de 16 bites de oscilación libre con un preescalador programable de cuatro niveles, tres funciones independientes de captura por entrada, cinco de salida por comparación y un circuito de interrupción periódica programable (interrupción de tiempo real, RTI).

Funciones de captura por entrada

Cada función de captura por entrada incluye un alimentador de 16 bites (TICx, x=1, 2 ó 3), una lógica de detección de nivel de entrada y una lógica de generación de interrupción. El alimentador captura el valor corriente del contador cuando se detecta el nivel seleccionado en el alfiler de entrada correspondiente; dicho valor se puede leer por programa como un par de registros de ocho bites y no se afecta por un reestablecimiento.

La lógica de generación de interrupción utiliza una bandera de estado y un bit de habilitación local (ICxF e ICxI). La primera se enciende cuando se detecta el nivel seleccionado, encender la segunda permite generar una interrupción, la captura opera por encuesta cuando este bit está apagado. Si se genera una interrupción debe borrarse el bit ICxF correspondiente antes de salir de la rutina de servicio.

La lógica de detección de nivel utiliza un par de bites de control (EDGxB,EDGxA) del registro TCTL2 para seleccionar los flancos detectados por cada función de entrada por captura de acuerdo con la Tab. 4.5.

TFLG1 $ 1023

7							0
OC1F	OC2F	OC3F	OC4F	I4O5F	IC1F	IC2F	IC3F

OC1F-OC4F.- Banderas de salida por comparación 1-4.
I4O5F.- Bandera de captura por entrada 4 o de salida por comparación 5.
IC1F-IC3F.- Banderas de captura por entrada 1-3.
Escribir unos para borrar la bandera correspondiente.

Fig. 4.12 Registro TFLG1.

TMSK1 $ 1022

7							0
OC1I	OC2I	OC3I	OC4I	I4O5I	IC1I	IC2I	IC3I

OC1I-OC4I.- Habilitadores de interrupción de salida por comparación 1-4.
I4O5I.- Habilitador de interrupción de captura por entrada 4 o salida por comparación 5.
IC1I-IC3I.- Habilitadores de interrupción de captura por entrada 1-3.
=1 habilita petición de interrupción si la bandera está encendida.
=0 inhibe interrupción.

Fig. 4.13 TMSK1.

TCTL2 $ 1021

7							0
EDG4B	EDG4A	EDG1B	EDG1A	EDG2B	EDG2A	EDG3B	EDG3A

Fig. 4.14 Registro TCTL2.

EDGxB	EDGxA	CAPTURA
0	0	Deshabilitada
0	1	Sólo con flanco de subida
1	0	Sólo con flanco de bajada
1	1	Con cualquier flanco

Tab. 4.5 Configuración de la función de captura por entrada.

Funciones de salida por comparación

Cada salida por comparación tiene un registro comparador y un comparador, ambos de 16 bites. El comparador contrasta el valor del temporizador del oscilador libre contra el registro comparador durante cada cuenta del temporizador, cuando éstos coinciden se enciende una bandera de estado (OCxF, x=1, 2, 3, 4 ó 5), se genera una interrupción opcionalmente y los alfileres de salida del temporizador se cambian automáticamente de acuerdo con los bites de control especificados en la Tab. 4.7 Debido a que cada una de las cinco interrupciones es enmascarable individualmente con un bit de habilitación local de interrupción (OCxI) y a que tiene su propio vector de interrupción, no hay necesidad de llevar a cabo una encuesta para determinar la causa de una interrupción.

La salida por comparación OC1 puede controlar cualquier combinación de los cinco alfileres de salida del temporizador con prioridad al uso de esta línea por otro comparador.

Existen para OC5-OC2 un par de bites (OMx, OLx, x=2, 3, 4 ó 5) del registro de control del temporizador (TCTL1) que controlan la acción automática que ocurrirá en el alfiler de salida correspondiente del temporizador cuando suceda una salida por

comparación. Estas acciones se explican en la Tab. 4.6.

TCTL1 $ 1020

7							0
OM2	OL2	OM3	OL3	OM4	OL4	OM5	OL5

Fig. 4.15 Registro TCTL1.

OMx	OLx	CONFIGURACIÓN
0	0	OCx no afecta al alfiler
0	1	Cambia el estado actual del alfiler
1	0	Borra el alfiler
1	1	Enciende el alfiler

Tab. 4.6 Configuración de la función de salida por comparación.

Para OC1 las acciones automáticas se controlan por los registros de máscara (OC1M) y de datos (OC1D). El primero especifica qué alfileres del puerto A se afectan por OC1 y el segundo indica los datos que se enviarán a los alfileres afectados del puerto A cuando existe una igualdad exitosa en OC1. Esta salida por comparación tiene mayor prioridad que las otras, como se mencionó.

OC1D $ 100D

7							0
OC1D7	OC1D6	OC1D5	OC1D4	OC1D3	0	0	0

Si se enciende el bit OC1Mx, el dato en OC1Dx se envía al puerto A bit x cuando ocurre una comparación OC1 exitosa.

Fig. 4.16 Registro OC1D.

OC1M $ 100C

7							0
OC1M7	OC1M6	OC1M5	OC1M4	OC1M3	0	0	0

Si se encienden los bites se habilita el control de OC1 sobre los alfileres correspondientes del puerto A.

Fig. 4.17 Registro OC1M.

Los alfileres para las cinco funciones de comparación se pueden usar como salidas de propósito general, siempre y cuando no se ocupen para el sistema temporizador.

Por otra parte, si el usuario necesita cambiar el estado del alfiler de salida del temporizador sin esperar a que ocurra una salida por comparación, se puede utilizar una función especial para forzar este estado a través del encendido de los bites correspondientes del registro CFORC. Esta función es útil para inicializar el estado de las salidas del temporizador.

CFORC $ 100B

7							0
FOC1	FOC2	FOC3	FOC4	FOC5	0	0	0

=1 fuerza comparación.

Fig. 4.18 Registro CFORC.

Los registros de salida por comparación (TOCx, x=1, 2, 3, 4 ó 5) se pueden leer o escribir por programa como un par de registros de ocho bites y contienen un valor de $ FFFF después de un reestablecimiento.

Mientras una función de comparación está configurada para cambiar el estado de un alfiler o generar una interrupción, dicha acción ocurre cada vez que el valor del temporizador coincide con el del registro comparador, no solamente la primera vez que ocurra una igualdad. Para generar una sola interrupción después de algún retardo, se debe leer el registro TCNT, sumar un valor correspondiente al retardo deseado, escribir el resultado en el registro de comparación y habilitar la interrupción respectiva, por último, hay que deshabilitar la interrupción y borrar el bit OCxF dentro de la rutina de servicio.

Preescalador

El preescalador permite seleccionar una de cuatro velocidades para manejar el temporizador/contador principal de 16 bites. Esta selección combina resolución y rango del temporizador, según se muestra en la Tab. 4.7. El rango del temporizador es importante debido a que la programación requerida para las funciones de temporización es mucho más compleja si se considera la saturación del temporizador.

PR1	PR0	FACTOR DE PREES-CALA-MIENTO	FRECUENCIA DEL CRISTAL		
			2^{23} Hz	8 MHz	4 MHz
			RESOLUCIÓN/RANGO		
0	0	1	477ns/31.25ms	500ns/32.77ms	1us/65.54ms
0	1	4	191us/125ms	2us/131.1ms	4us/262.1ms
1	0	8	3.81us/250ms	4us/262.1ms	8us/524.3ms
1	1	16	7.63us/0.5s	8us/524.3ms	16us/1.049s
			2.1 MHz	2.0 MHz	1.0 MHz
			FRECUENCIA DEL CANAL (RELOJ E)		

Tab. 4.7 Velocidad, rango y resolución del temporizador.

Otro factor a considerar en la selección del preescalador es el consumo de potencia. Debido a que el consumo de potencia CMOS es directamente proporcional a la frecuencia de operación, se ahorra potencia reduciendo la frecuencia. Los bites de selección del factor de preescalamiento son PR1 y PR0 correspondientes al registro TMSK2.

TMSK2 $ 1024

7							0
TOI	RTII	PAOVI	PAII	0	0	PR1	PR0

TOI.- Habilita interrupción por saturación del temporizador.
RTII.- Habilita interrupción de tiempo real.
PAOVI.- Habilita interrupción por saturación del acumulador de pulsos.
PAII.- Habilita interrupción de entrada al acumulador de pulsos.
=1 interrumpe si la bandera correspondiente está encendida.
=0 inhibe la interrupción.
PR1, PR0.- Bites de selección del preescalador del temporizador. (Ver Tab. 4.7).

Fig. 4.19 Registro TMSK2.

Saturación

La función de saturación del temporizador se debe usar en casos donde se desee medir o producir periodos mayores al rango del temporizador/contador. La bandera de saturación del temporizador (TOF) del registro TFLG2 se enciende cada vez que el temporizador/contador se satura (de $ FFFF a $ 0000). Este bit puede generar una interrupción opcionalmente si se enciende el bit de habilitación local de interrupción (TOI) del registro TMSK2. Estos bites de control funcionan de la misma manera que los correspondientes a las funciones de captura por entrada o de salida por comparación.

TFLG2 $ 1025

7							0
TOF	RTIF	PAOVF	PAIF	0	0	0	0

TOF.- Bandera de saturación del temporizador.
RTIF.- Bandera de interrupción (periódica) de tiempo real.
PAOVF.- Bandera de saturación del acumulador de pulsos.
PAIF.- Bandera de nivel de entrada del acumulador de pulsos.
Escribir unos para borrar la bandera correspondiente.

Fig. 4.20 Registro TFLG2.

Interrupción de tiempo real (RTI)

La función del RTI se utiliza para generar interrupciones de circuitería a una velocidad periódica fija. Una práctica común para organizar las rutinas que componen el programa de una aplicación es organizarlo como una secuencia de llamados a subrutina, como el tiempo requerido para ejecutar cada rutina es variable, después de completar un paso a través de todas las rutinas con esta interrupción se puede generar una referencia de tiempo periódica que sincronice el inicio de la secuencia de llamados.

Están disponibles cuatro velocidades diferentes para la señal RTI, que son una función de la frecuencia del oscilador del MCU y del valor de los bites de control RTR1 y RTR0 del registro PACTL, de acuerdo con la Tab. 4.8.

La fuente de reloj para la función del RTI es un reloj de oscilación libre que no se puede detener o interrumpir. Esto origina que el tiempo sea constante entre las salidas sucesivas del RTI.

La bandera de estado RTIF del registro TFLG2 se enciende

automáticamente al final de cada periodo RTI.

PACTL $ 1026

7							0
DDRA7	PAEN	PAMOD	PEDGE	DDRA3	I4/O5	RTR1	RTR0

DDRA7 =1 bit 7 del puerto A como salida.
=0 bit 7 del puerto A como entrada.
PAEN =1 habilita sistema acumulador de pulsos.
=0 deshabilita sistema acumulador de pulsos.
PAMOD =1 modo de acumulación de tiempo con disparo por compuerta.
=0 modo contador de eventos del acumulador de pulsos.
PEDGE =1 el acumulador de pulsos responde a flancos de subida.
=0 el acumulador de pulsos responde a flancos de bajada.
DDRA3 =1 bit 3 del puerto A como salida.
=0 bit 3 del puerto A como entrada.
I4/O5 =1 habilita la función de captura por entrada 4.
=0 habilita la función de salida por comparación 5.
RTR1, RTR0.- Bites de selección de la velocidad de interrupción del RTI.

Fig. 4.21 Registro PACTL.

RTR1	RTR0	E / 2^{13} dividido por	FRECUENCIA DEL CRISTAL		
			2^{23} Hz	8 MH	4 MHz
			VELOCIDAD NOMINAL RTI		
0	0	1	3.91 ms	4.10 ms	8.19 ms
0	1	2	7.81 ms	8.19 ms	16.38 ms
1	0	4	15.62 ms	16.38 ms	32.77 ms
1	1	8	31.25 s	32.77 s	65.54 s
			2.1 MHz	2.0 MHz	1.0 MHz
			FRECUENCIA DEL CANAL (RELOJ E)		

Tab. 4.8 Velocidad nominal del RTI.

4.9 Acoplamiento periférico serial síncrono (SPI)

El acoplamiento periférico serial síncrono se usa principalmente para permitir al MCU comunicarse con dispositivos periféricos. Además, tiene la capacidad de comunicación entre procesadores (interprocesador) en un sistema de maestro múltiple. El sistema SPI se puede configurar como un dispositivo maestro o esclavo; como maestro trabaja a una velocidad de hasta 1 Mbit/seg y como esclavo puede trabajar hasta 2 Mbit/seg.

Durante una transferencia del SPI los datos se transmiten y reciben simultáneamente; una línea de reloj serial sincroniza el corrimiento y muestreo de la información sobre las dos líneas seriales de datos. Una línea de selección de esclavo (SS*) permite la selección individual de un dispositivo SPI esclavo; los dispositivos esclavos que no son seleccionados no interfieren con las actividades del canal del SPI. En un dispositivo SPI maestro, la línea SS* se puede usar opcionalmente para indicar una contención del canal en un sistema de maestro múltiple.

El SPI permite la selección por programa de una de cuatro combinaciones de fase y polaridad del reloj serial (SCK) con dos bites del registro de control del SPI (SPCR). La polaridad se especifica con el bit de control CPOL que selecciona un reloj activo alto o bajo y no influye en el formato de transferencia. El bit de control de fase CPHA selecciona uno de dos formatos de transferencia.

Existen cuatro alfileres de E/S asociados con las transferencias del SPI: SCK, MISO, MOSI y SS*; éstos se configuran como entradas de propósito general cuando el SPI se deshabilita mediante el bit SPE del registro SPCR, la dirección del dato para cada alfiler se controla con el registro de dirección de datos del puerto D (DDRD). Cuando el SPI se habilita, los bites de control de dirección de dato aún tienen influencia sobre estos alfileres.

SPCR $ 1028

7							0
SPIE	SPE	DWOM	MSTR	CPOL	CPHA	SPR1	SPR0

SPIE.- Habilitador de interrupción del SPI.

SPE.- Habilitador del sistema SPI.

DWOM =1 salidas del puerto D en modo de OR alambrada.
=0 salidas del puerto D normales.

MSTR =1 modo maestro.
=0 modo esclavo.

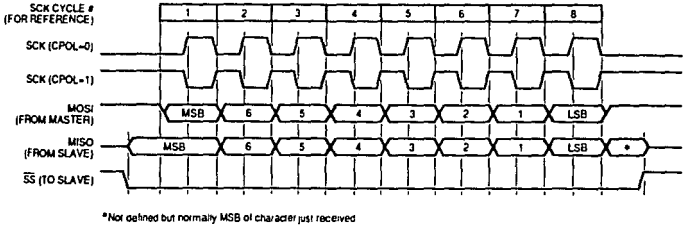
CPOL.- Polaridad del reloj.

CPHA.- Fase del reloj.

SPR1, SPR0.- Bites de selección de la velocidad del reloj del SPI. (Ver Tab. 4.9).

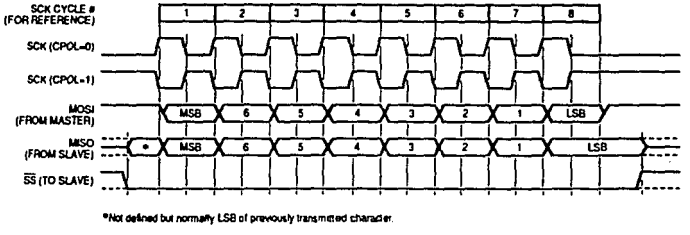
Fig. 4.22 Registro SPCR.

CPHA=0



*Not defined but normally MSB of character just received

CPHA=1



*Not defined but normally LSB of previously transmitted character

Fig. 4.23. Formatos de transferencia.

DDRD $ 1009

7							0
0	0	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0

=1 salidas.
=0 entradas.

Fig. 4.24 Registro DDRD.

Modo maestro

En el modo maestro, el alfiler SCK es una salida de reloj generada internamente; cuando el maestro inicia una transferencia se generan automáticamente ocho ciclos de reloj en este alfiler. Los alfileres MOSI y MISO se usan para transmisión y recepción serial de datos, respectivamente. El alfiler SS* actúa como una entrada de detección de error si DDR5=0, de lo contrario es una salida de propósito general.

En este modo, una transferencia inicia cuando el dato se escribe en el registro de datos (SPDR) y termina cuando se enciende la bandera de fin de transmisión del SPI (SPIF) del registro de estado (SPSR).

SPSR $ 1029

7							0
SPIF	WCOL	0	MODF	0	0	0	0

SPIF.- Petición de interrupción del SPI.
WCOL.- Bandera del estado de colisión de escritura.
MODF.- Bandera de interrupción del modo de error del SPI.

Fig. 4.25 Registro SPSR.

Modo esclavo

En el modo esclavo, el alfiler SCK es una entrada de la señal de reloj de un maestro que sincroniza la transferencia de datos. Los alfileres MOSI y MISO se utilizan para recepción y transmisión serial, respectivamente. El alfiler SS* se utiliza para habilitar al SPI esclavo para una transferencia, los dispositivos esclavos ignoran la señal SCK y mantienen el alfiler de salida MISO en un estado de alta impedancia a menos de que SS* sea bajo.

En este modo, con un formato CPHA=0, una transferencia inicia cuando el alfiler SS* va a un nivel bajo y termina cuando regresa a un nivel alto. Para el otro formato (CPHA=1), la transferencia comienza cuando la línea SCK va a su nivel activo y termina cuando se enciende el bit SPIF.

Tipos de error en el SPI

Se pueden detectar dos tipos de error en un sistema SPI: el primero se denomina "falla de modo" y ocurre cuando más de un SPI trata de ser maestro simultáneamente en un sistema de maestro múltiple; el segundo, llamado "colisión de escritura", ocurre si se escribe en el registro SPDR mientras una transferencia está en proceso.

Cuando se detecta un error de falla de modo se ejecutan las siguientes acciones:

1.- Los bites del registro DDRD correspondientes al SPI se fuerzan a cero para deshabilitar las salidas.

2.- El bit de control MSTR del registro SPCR se fuerza a cero para reconfigurar al SPI como esclavo.

3.- El bit SPE se fuerza a cero para deshabilitar al

sistema.

4.- La bandera de estado MODF del registro SPSR se enciende y se genera una interrupción del SPI.

Cuando ocurre una colisión de escritura, la transferencia continúa sin alteración y el dato que provocó el error no se escribe en el registro de corrimiento. Este tipo de error se presenta normalmente en un sistema esclavo porque no tiene control de cuándo el maestro iniciará una transferencia.

4.10 Acoplamiento serial asíncrono (SCI)

El SCI es un sistema asíncrono, serializado, de doble vía y formato NRZ (un bit de inicio, ocho o nueve bites de datos y un bit de paro). Un generador interno deriva las frecuencias de comunicación a partir del oscilador del MCU. El transmisor y el receptor son funcionalmente independientes, pero usan el mismo formato de datos y velocidad. El usuario deberá, regularmente, agregar dispositivos para manejar los estándares RS232C o RS422.

El SCI se configura y controla básicamente por cinco registros: BAUD, SCCR1, SCCR2, SCSR y SCDR, que se complementan con los registros PORTD, DDRD y el bit de modo de OR alambrada (DWOM) del puerto D en el registro SPCR.

Cuando el receptor o el transmisor se habilitan, la lógica del SCI toma el control de los alfileres asociados del puerto D; las direcciones de los datos de los alfileres RxD y TxD se sobreescriben como entrada y salida, respectivamente.

El bit DWOM modifica las funciones de los manejadores de los alfileres del puerto D, cuando esté siendo usado por los sistemas SCI o SPI.

El registro BAUD se utiliza para seleccionar la velocidad

de transmisión del SCI y contiene además dos bites para pruebas de fábrica.

BAUD $ 102B

7							0
TCLR	0	SCP1	SCP0	RCKB	SCR2	SCR1	SCR0

TCLR.- Borra la cadena de temporización de velocidad de transmisión (sólo de prueba).

RCKB.- Prueba el reloj de velocidad de transmisión del SCI (sólo de prueba).

SCP1, SCP0.- Bites de selección del preescalador serial. (Ver Tab. 4.11).

SCR2-SCR0.- Bites de selección de la velocidad del SCI.

Fig. 4.26 Registro BAUD.

SCR2	SCR1	SCR0	SALIDA DEL PREESCALADOR dividido por	VELOCIDAD MAYOR XTAL= 32.768 K	9600	4800
0	0	0	1	32.768 K	9600	4800
0	0	1	2	-	4800	2400
0	1	0	4	8.192 K	2400	1200
0	1	1	8	-	1200	600
1	0	0	16	2.048 K	600	300
1	0	1	32	-	300	-
1	1	0	64	512 K	-	-
1	1	1	128	-	-	-

Tab. 4.10 Selección de velocidad del SCI.

SCP1	SCP0	E dividido por	VELOCIDAD MAYOR XTAL=		
			2^{23}	8.0 MHz	4.0 MHz
0	0	1	131.07 K	-	-
0	1	3	-	-	-
1	0	4	32.768 K	-	-
1	1	13	-	9600	4800
		E=	2.1 MHz	2.0 MHz	1.0 MHz

Tab. 4.11 Selección del preescalador.

El registro SCCR1 incluye tres bites asociados con el formato opcional de datos de nueve bites: R8, T8 y M. Además contiene el bit WAKE que se usa para seleccionar uno de dos métodos para "despertar" al receptor. La circuitería lógica del receptor permite utilizar la función de "despertar" al receptor en sistemas con más de uno. Con esta función, un dispositivo transmisor dirige mensajes a un receptor individual o a un grupo de receptores a través de información direccionada con el octeto inicial de cada mensaje. Los receptores no direccionados activan

SCCR1 $ 102C

7							0
R8	T8	0	M	WAKE	0	0	0

R8.- Bit ocho de recepción.
T8.- Bit ocho de transmisión.
M =1 formato: 1 bit de inicio, 9 de datos, 1 de paro.
=0 formato: 1 bit de inicio, 8 de datos, 1 de paro.
WAKE =1 "despierta al receptor" por marca de dirección.
=0 "despierta al receptor" por línea desocupada.

Fig. 4.27 Registro SCCR1.

esta función, esto hace que los receptores "duerman" por el resto de los mensajes no deseados.

El registro SCCR2 contiene los controles principales del SCI. Los cuatro bites más significativos son los controles de habilitación de interrupciones locales que determinan cuándo se deberán encuestar las banderas de estado del SCI o cuándo se generarán interrupciones. Los bites TE y RE son los habilitadores de los subsistemas de transmisión y recepción, respectivamente. Levantar el bit RWU permite "dormir" al receptor, que se "despierta" por circuitería automáticamente, misma que borra este bit. El bit SBK permite generar caracteres de ruptura en la línea TxD.

SCCR2 $ 102D

7							0
TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK

TIE.- Habilitador de interrupción de transmisión.
TCIE.- Habilitador de interrupción de transmisión completa.
RIE.- Habilitador de interrupción de recepción.
ILIE.- Habilitador de interrupción de línea desocupada.
=1 habilita interrupciones.
=0 inhibe interrupciones.
TE.- Habilitador del transmisor.
RE.- Habilitador del receptor.
=1 enciende.
=0 apaga.
RWU =1 receptor en modo de "despertar".
=0 normal.
SBK.- Envío de caracteres de ruptura.

Fig. 4.28 Registro SCCR2.

El registro SCSR contiene dos banderas del estado de la

transmisión, TDRE y TC, y cinco relacionadas con la recepción: RFDR, OR, IDLE, NF y FE.

SCSR $ 102E

7							0
TDRE	TC	RDRF	IDLE	OR	NF	FE	0

TDRE.- Bandera de registro de transmisión de datos vacío.
TC.- Bandera de transmisión completa.
RDRF.- Bandera de registro de recepción de datos lleno.
IDLE.- Bandera de detección de línea desocupada.
OR.- Bandera de error de sobrelectura.
NF.- Bandera de error por ruido.
FE.- Bandera de error en la estructura de transferencia.

Fig. 4.29 Registro SCSR.

El registro SCDR está separado en dos: TDR que es un registro de sólo escritura para transmisión de datos y RDR de sólo lectura para la recepción. Cuando se lee por programa al SCDR, se accesa a RDR, y cuando se escribe en el SCDR, se accesa al TDR.

Transmisor del SCI

El transmisor del SCI utiliza un reloj interno para la salida serial de los datos a través del alfiler TxD. Para iniciar una transmisión debe estar encendido el bit TE y escribir el dato correspondiente en el registro SCDR.

La lógica de transmisión agrega un 0 como bit de inicio y un 1 como bit de paro al dato que se desea transmitir. El transmisor se puede configurar para enviar caracteres de ocho o nueve bites con M=0 o M=1, respectivamente. El transmisor cuenta con registros independientes para retener el dato y serializarlo,

lo que da opción de escribir un dato sobre TDR mientras otro se está serializando. Cuando la bandera TDRE se enciende, indica que el TDR está disponible para aceptar un nuevo dato y puede generar una interrupción opcionalmente si así lo indica la bandera TIE. Cuando el transmisor termina de enviar toda la información contenida en la cola de espera, se enciende la bandera TC y se puede generar una interrupción si se programó el bit TCIE. Además de los caracteres de datos, el transmisor es capaz de enviar caracteres de línea desocupada y de ruptura, los cuales son útiles en redes SCI multipunto.

Receptor del SCI

Se habilita por medio del bit RE y el receptor del SCI es responsable de sincronizar el flujo de datos seriales y recuperar los caracteres. La bandera RDRF se enciende cuando se recibe y se transfiere un caracter dentro del registro RDR y permite la generación opcional de una interrupción con la bandera local RIE. Cuando un caracter está listo para transferirse al receptor, pero si el caracter previo aún no ha sido leído, el nuevo caracter se pierde y se enciende la bandera OR para indicar este error; además, se genera una interrupción si el bit RIE está encendido. Se pueden detectar otros dos tipos de error con el estado de las banderas NF que indica la presencia de ruido durante la recepción de un caracter en el SCDR y FE que indica si existe un error en la estructura de transferencia del dato. Estas banderas no generan interrupciones.

Modos de datos de ocho y nueve bites

El bit M determina la longitud de los caracteres del SCI para transmitir y recibir. La configuración más común es un bit de inicio, ocho bites de datos y un bit de paro. Otra configuración es un bit de inicio, nueve bites de datos (el noveno corresponde al bit T8 en la transmisión o R8 para la recepción) y un bit de paro. El noveno bit se puede utilizar como

un bit de paridad (marca o espacio) al encender o apagar T8, entre otros usos.

4.11 Acumulador de pulsos

El acumulador de pulsos es un sistema basado en un temporizador/contador de ocho bites que se puede configurar para operar en dos modos básicos: contador de eventos y de acumulación de tiempo con disparo por compuerta. Este sistema se habilita o dehabilita por medio del bit de control PAEN, el bit PAMOD selecciona el modo de operación y el bit PEDGE controla la polaridad de las señales sobre el alfiler PAI que serán reconocidas por el sistema. Estos bites pertenecen al registro de control del acumulador de pulsos (PACTL).

Además, cuenta con dos fuentes de interrupción separadas: una se genera con la detección del nivel en el alfiler PAI y la otra se genera cuando el contador se satura (de $ FF a $ 00). Cada una de estas fuentes de interrupción tiene su propio bit de habilitación local y su propio vector de interrupción, por lo que no se requiere de encuestas para determinar la causa de la interrupción.

Los bites de habilitación de estas interrupciones son PAOVI, que determina si la bandera de interrupción por saturación del acumulador de pulsos (PAOVF) generará o no una interrupción y PAII, que hace lo mismo para la bandera de entrada al acumulador de pulsos (PAIF). Estos bites de habilitación pertenecen al registro de máscara de interrupción (TMSK2) y están sujetos a la máscara I del CCR.

Modo contador de eventos

En este modo, el contador de ocho bites se sincroniza para incrementar su valor en cada nivel activo del alfiler PAI. Este modo sirve para contabilizar ciclos de una señal de entrada,

unidades de tiempo, etc.

Modo de acumulación de tiempo con disparo por compuerta

Este modo cambia al acumulador de pulsos de contador a temporizador. Si el alfiler PAI es activo, el registro contador del acumulador de pulsos (PACNT) de ocho bites se incrementa cada 64 ciclos de reloj E. El bit PEDGE controla el nivel que inhibe la cuenta. Este modo se denomina de acumulación de tiempo con disparo por compuerta porque el registro PACNT se usa para acumular el tiempo total que el alfiler estuvo en nivel activo. Una de sus aplicaciones más comunes es la medición de ancho de pulso.

4.12 Convertidor analógico-digital (A/D)

El sistema A/D es un convertidor de aproximaciones sucesivas de ocho canales y ocho bites con una exactitud de ± 1/2 LSB sobre el rango completo de temperaturas de operación. Está formado por el convertidor, un multiplexor de entrada para seleccionar uno de 16 canales (que incluyen ocho canales asociados internamente con el MCU) y circuitería para configurar y controlar las actividades de conversión. Existen cuatro registros para resultados que cuentan con lógica de control para implementar secuencias de conversión automática, cuatro veces sobre un canal seleccionado o sobre cuatro canales una vez cada uno; estas secuencias se configuran para repetirse continuamente o detenerse después de un conjunto de cuatro conversiones. Se puede usar un oscilador interno RC para permitir la operación normal del A/D cuando se utilizan frecuencias del reloj del MCU muy bajas (menos de 750 KHz).

Los registros de resultados del A/D (ADR1-ADR4) son de sólo lectura y almacenan las conversiones de ocho bites. Después de una secuencia de conversión completa, se enciende la bandera de

conversiones completas (CCF) para indicar que los resultados son válidos. Los resultados de una nueva conversión se calculan en el A/D y se transfieren a los registros de resultados en una parte del ciclo del reloj donde las lecturas no tienen lugar, por consiguiente, no existe interferencia entre las lecturas por programa y las actualizaciones de resultados.

El convertidor A/D se puede configurar para trabajar en modo de canal múltiple (MULT=1) o de un solo canal (MULT=0). En el primer modo, se realizan conversiones sobre cada canal en el grupo de cuatro que se especifica por los bites de selección de canal CD y CC, según la Tab. 4.12; cada canal se asocia con un registro de resultado específico. En el segundo modo, el sistema se configura para llevar a cabo cuatro conversiones consecutivas sobre un solo canal que se especifica por los bites CD-CA. La bandera de estado CCF y los bites de control SCAN, MULT y CD-CA pertenecen al registro de estado y control del A/D (ADCTL).

ADCTL $ 1030

7							0
CCF	0	SCAN	MULT	CD	CC	CB	CA

CCF.- Bandera de conversiones completas (se enciende después de la cuarta conversion).
SCAN =1 conversiones continuas.
=0 cuatro conversiones y para.
MULT =1 convierte un grupo de cuatro canales.
=0 convierte un solo canal.
CD-CA.- Bites de selección de canal. (Ver Tab. 4.12).

Fig. 4.30 Registro ADCTL.

CD	CC	CB	CA	CANAL	RESULTADO EN ADRx
0	0	0	0	PE0	ADR1
0	0	0	1	PE1	ADR2
0	0	1	0	PE2	ADR3
0	0	1	1	PE3	ADR4
0	1	0	0	PE4 *	ADR1
0	1	0	1	PE5 *	ADR2
0	1	1	0	PE6 *	ADR3
0	1	1	1	PE7 *	ADR4

* No están disponibles en circuitos de 48 alfileres.

Tab. 4.12 Asignación de canales del sistema A/D.

Para iniciar una nueva secuencia de conversión, se escribe en el registro ADCTL y la bandera CCF se borra automáticamente. Por otra parte, para seleccionar la fuente de reloj interna RC se enciende el bit CSEL del registro OPTION; debido a que este reloj es asíncrono respecto al reloj E, se requiere un retardo de sincronización al final de cada secuencia de conversión para prevenir actualizaciones del registro de resultado en la misma parte del ciclo de reloj E donde se realiza una lectura.

La Fig. 4.31 muestra el diagrama de tiempos para una secuencia de cuatro conversiones A/D; el reloj E se utiliza como el reloj de conversión normalmente, como en este caso. De esta figura se observa que cuando el bit SCAN=1 las conversiones continúan repetidamente actualizando el contenido de los registros de resultado, de lo contrario, se realiza sólo un conjunto de cuatro conversiones para llenar los cuatro registros de resultado.

El sistema requiere una bomba de carga interna que dearrolla de 7 a 8 V para manejar la circuitería analógica del convertidor. Esta bomba se deshabilita al salir de un estado de reestablecimiento y se debe accionar por medio del bit de control

ADPU del registro OPTION; hay que generar un retardo después de encender este bit para permitir a la bomba y circuitos analógicos estabilizarse antes de utilizar el sistema.

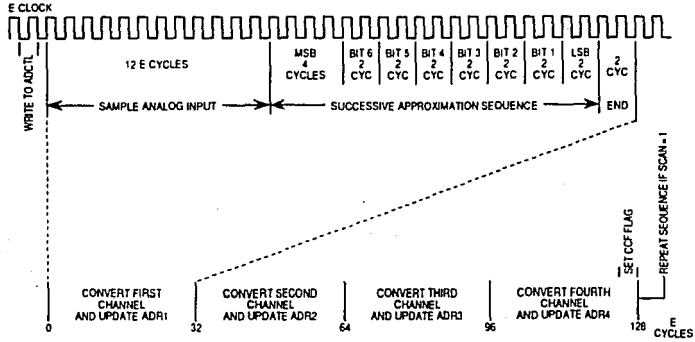


Fig. 4.31 Diagrama de tiempos para una secuencia de 4 conversiones A/D).

5. DESCRIPCIÓN DE LA CIRCUITERÍA

5.1 Circuitería del módulo 8031

El diseño del módulo está basado en el microcontrolador 8031 de INTEL, alrededor de éste se encuentran conectados varios dispositivos que permiten su funcionamiento como un sistema integrado bajo la norma de comunicación STD BUS. (Ver Capítulo 2). La circuitería del módulo en conjunto con un programa monitor permite al usuario la depuración de programas cargados en memoria RAM externa a través del puerto serie del mismo microcontrolador y la evaluación de prototipos desarrollados con el estándar mencionado.

Para su descripción, el módulo se divide en cuatro partes principales:

- Microcontrolador
- Decodificación de memoria
- Puertos de entrada/salida
- Adaptación al STD BUS

Microcontrolador

En esta parte se describen los circuitos de reloj y de reestablecimiento del microcontrolador, así como las señales de

interrupción del mismo.

El circuito de reestablecimiento consiste de un botón normalmente abierto que cuando se oprime, proporciona un nivel alto en el alfiler 9 (RESET); además genera la señal para el SYSRESET* del STD BUS.

El circuito de reloj emplea un cristal oscilador de 11.0592 MHz que permite obtener velocidades de transmisión serial estándar.

El microcontrolador maneja un canal multiplexado en el tiempo de direcciones/datos, por lo que se utiliza un circuito alimentador 74LS373 para separar direcciones y datos por medio de la señal de sincronización ALE proveniente del microcontrolador.

Las líneas T0 y T1 del 8031 son accesibles al usuario por medio de un conector de 10 alfileres de doble fila.

Para el manejo del trazado en el programa monitor se utiliza un decodificador de 3 a 8 74LS138 que controla un circuito 74LS74 para generar un nivel bajo en el alfiler 12 (INT0*), según se indica en el mapa de memoria de la Tab. 5.1.

Decodificación de memoria

El módulo opera con un solo espacio direccionable de memoria de 64 Koctetos por medio de una señal de habilitación de lectura común a ambas memorias, que se obtiene al conectar las señales PSEN* y RD* del microcontrolador a una compuerta AND 74LS08 de dos entradas.

La memoria externa consiste en una memoria de sólo lectura (ROM) de 8K x 8 con el circuito integrado 2764 y de una memoria de acceso aleatorio (RAM) de 8K x 8 con el circuito integrado 6264.

Para la habilitación de las memorias se utilizan las dos primeras salidas de un decodificador 3 a 8 74LS138, donde las líneas A15-A13 se utilizan para direccionar la memoria según el mapa de la Tab. 5.1.

DIRECCIÓN	DISPOSITIVO O SEÑAL
0000 H-1FFF H	ROM (8 K)
2000 H-3FFF H	RAM (8 K)
4000 H-5FFF H	MEMRQ* PRIMARIA (8 K), MEMEX BAJA
6000 H-7FFF H	MEMRQ* SECUNDARIA (8 K)
8000 H-9FFF H	IORQ* PRIMARIA (8 K), IOEXP BAJA
A000 H-BFFF H	IORQ* SECUNDARIA (8 K)
C000 H-DFFF H	NIVEL BAJO EN INT0*
E000 H-FFFF H	NIVEL ALTO EN INT0*

Tab. 5.1 Mapa de memoria del módulo 8031.

Puertos de entrada/salida

Los puertos utilizados pertenecen al microcontrolador y son: un puerto serie programable y un puerto paralelo bidireccional de ocho líneas que tiene acceso al exterior por medio de un conector de 10 alfileres de doble fila.

La comunicación serial sigue el estándar RS232C y para acoplar los niveles de voltaje usados por este estándar se utilizan los manejadores MC1488, MC1489 y un conector DB9 hembra para acoplamiento con el exterior.

Adaptación al STD BUS

Para la descripción de esta parte se mencionan cada una de las señales del STD-BUS requeridas para este sistema y cómo se implantaron. La explicación del conjunto de señales que forman este estándar aparece en el Capítulo 2.

Las direcciones A0-A15 se desacoplan a través de dos almacenadores 74LS244.

Las líneas de datos D0-D7 se desacoplan por medio de un transceptor bidireccional 74LS245 que se habilita por las señales de control MEMRQ* y RD*.

Las señales de control RD*, WR*, CLOCK*, STATUS1*, NMIRQ*, INTRQ* provienen del microcontrolador.

Las señales PBRESET*, SYSRESET* y MCSYNC* utilizan un circuito inversor 74LS04 para adecuarse a su nivel activo.

Las señales MEMEX, MEMRQ*, IOEXP e IORQ* se activan por medio del decodificador de memoria y de dos compuertas lógicas AND 74LS08.

PCO y PCI se conectan directamente debido a que no se requiere prioridad de salida ni de entrada.

Las señales RD*, WR*, CLOCK*, MCSYNC*, MEMEX, MEMRQ*, IOEXP e IORQ* se desacoplan a través de un almacenador 74LS244.

Los diagramas detallados de la circuitería se pueden consultar en el Apéndice A.

5.2 Circuitería del módulo MC68HC11

La circuitería del módulo basado en el microcontrolador MC68HC11 de MOTOROLA está adaptada también al estándar STD-BUS, consta del microcontrolador y de aquellos dispositivos conectados a él, que para su descripción se dividen en las siguientes partes:

- Microcontrolador
- Decodificación de memoria
- Puertos de entrada/salida

Microcontrolador

Este microcontrolador opera por omisión en el modo expandido pero puede programarse, si el usuario así lo requiere, para emular el modo sencillo por medio de la Unidad de Reemplazo de Puerto MC68HC24 (PRU) con sus líneas de control (IRQ*, E, R/W*, RESET*) conectadas directamente al MCU.

El circuito de reestablecimiento utiliza un botón normalmente abierto que cuando se oprime proporciona un nivel bajo en el alfiler 39 (RESET*) y además genera la señal SYSRESET* para el STD-BUS.

Para el circuito de reloj se emplea un cristal oscilador de 8 MHz con lo que se consigue una frecuencia de canal (reloj E) de 2 MHz.

El microcontrolador maneja un canal multiplexado en el tiempo de direcciones/datos, por lo que se utiliza un circuito alimentador 74HC373 para separar direcciones y datos por medio de la señal AS del microcontrolador.

Decodificación de memoria

El MC68HC11 maneja un solo espacio direccionable de memoria de 64 Koctetos.

La memoria externa consiste de una memoria de sólo lectura (ROM) de 8K x 8 con el circuito integrado 2764 y de una memoria de acceso aleatorio (RAM) de 8K x 8 con el circuito integrado 6264.

Las memoria externas ROM y RAM se ubican en la parte más alta del espacio direccionable (Ver Tab. 5.2).

Esta parte consiste principalmente de un decodificador 3 a 8 74HC138 que permite generar a través de sus dos últimas salidas las señales de habilitación para las memorias mencionadas, las direcciones A15-A13 se utilizan como líneas de selección y la señal E como línea de habilitación del decodificador.

Para decodificar al PRU y registros internos se debe cumplir que: a) las líneas A15-A12 igualen el contenido del registro INIT; b) la línea A11 conectada a CS* sea baja y; c) las líneas AD0-AD7 seleccionen un registro interno.

DIRECCIÓN	DISPOSITIVO O SEÑAL
$ 0000-$ 00FF	RAM INTERNA (256)
$ 0100-$ 0FFF	SIN USO
$ 1000-$17FFF	PRU + REGISTROS INTERNOS
$ 1800-$ 1FFF	SIN USO
$ 2000-$ 3FFF	MEMRQ* PRIMARIA (8 K), MEMEX BAJA
$ 4000-$ 5FFF	MEMRQ* SECUNADRIA (8 K)
$ 6000-$ 7FFF	IORQ* PRIMARIA (8 K), IOEXP BAJA
$ 8000-$ 9FFF	IORQ* SECUNDARIA (8 K)
$ B600-$ B7FF	EEPROM INTERNA (512)
$ B800-$ BFFF	SIN USO
$ C000-$ DFFF	RAM EXTERNA (8 K)
$ E000-$ FFFF	ROM EXTERNA (8 K)

Tab. 5.2 Mapa de memoria del módulo 68HC11.

Puertos de entrada/salida

El módulo cuenta con tres puertos del microcontrolador: el puerto A bidireccional de ocho líneas, el puerto E de entrada de cuatro líneas, el puerto D bidireccional de seis líneas y dos puertos soportados por el MC68HC24: los puertos B de salida y C bidireccional, ambos de ocho líneas.

El puerto A se desacopla de un conector de 20 alfileres por medio de un almacenador 74HC244.

Las líneas de entrada (puerto E) del convertidor analógico-digital (A/D) se desacoplan del exterior a través de amplificadores operacionales (circuito integrado LM324) configurados como seguidores no inversores. La referencia de voltaje del convertidor A/D consiste de un regulador de voltaje con diodo zener para obtener 5 V ± 0.001 en V_{RH}.

El puerto D soporta un puerto serial asíncrono que cumple con el estándar RS232C, por lo que se utilizan los manejadores MC1488, MC1489 para acoplar los niveles de voltaje y un conector DB9 hembra para el acceso de este puerto. Además, soporta un puerto serial síncrono que sigue la norma de comunicación RS422 y que a su vez usa los manejadores MC3486, MC3487 y un conector DB9 hembra para comunicación con el exterior. La circuitería para este puerto síncrono incluye también un almacenador 74HC244 y un brincador (puente) de tres alfileres, que permiten el manejo de las señales necesarias para operar en algunos de los modos que se seleccionan por programa (modo maestro o modo esclavo). Por omisión el puerto opera en el modo maestro, por lo que el brincador debe estar habilitando las líneas para este modo (con el puente entre los alfileres 2 y 3).

Los puertos B, C y las líneas STRA y STRB del MC68HC24 se accesan por medio de un conector de 20 alfileres de doble fila.

Adaptación al STD BUS

Esta parte de la circuitería utiliza compuertas lógicas AND, NAND, un decodificador 3 a 8, circuitos almacenadores y un transceptor bidireccional para llevar a cabo la adaptación de las señales requeridas en el sistema para cumplir con el estándar STD BUS.

Las direcciones A0-A15 se desacoplan a través de dos almacenadores 74HC244.

Las líneas de datos D0-D7 se desacoplan por medio de un transceptor bidireccional 74HC245 que se habilita por las señales de control MEMRQ* y R/W*.

Para obtener RD* y WR* a partir de la señal R/W* del microcontrolador se utiliza un arreglo de compuertas NAND 74HC00 de dos entradas.

Las señales STATUS0*, MCSYNC*, CLOCK*, NMIRQ* e INTRQ* provienen del microcontrolador.

La señal de control que genera STATUS1* se cambia a su nivel activo por medio de una compuerta NAND alambrada como un inversor.

La señal SYSRESET* proviene del circuito de reestablecimiento y PBRESET* se conecta al mismo circuito.

Las señales MEMEX, MEMRQ*, IOEXP e IORQ* se activan por medio del decodificador de memoria y de dos compuertas lógicas AND 74HC08.

Por último, PCO se conecta directamente a PCI debido a que no se requiere prioridad de salida ni de entrada.

Las señales RD*, WR*, STATUS0*, MCSYNC*, MEMEX, MEMRQ*, IOEXP e IORQ* se desacoplan a través de un almacenador 74LS244.

Los diagramas electrónicos del módulo se encuentran en el Apéndice D.

6. DESCRIPCIÓN GENERAL DEL PROGRAMA MONITOR

La siguiente descripción se refiere a las características del programa monitor y manejo de comandos de los módulos desarrollados con base en los microcontroladores 8031 y MC68HC11. A lo largo del texto, se hará referencia al programa monitor como si se tratara de uno solo, agregando únicamente las notas necesarias para diferenciarlos.

El programa monitor se encuentra residente en memoria externa ROM. Este programa permite al usuario interactuar con el módulo de desarrollo a través del puerto serie de una microcomputadora PC (Personal Computer) y el puerto serie del microcontrolador. Se requiere de un paquete de comunicaciones (Perfect Link, Procomm, Cross Talk, etc.) para llevar a cabo esta interacción.

El monitor está dividido en cinco bloques básicos:

a) Inicialización
b) Lectura de parámetros
c) Selección de comandos
d) Subrutinas
e) Rutinas de comando

a) Inicialización

FINALIDAD:

Inicializar los registros de control para la transmisión serial, manejo de interrupciones, apuntador de pila, variables internas y banderas.

Al término de cada comando, o después de desplegar un mensaje de error, el control del programa regresa a esta parte.

b) Lectura de parámetros

Se utiliza una fila en memoria RAM con un máximo de 16 octetos (que corresponden al comando M, con mayor longitud). El primer octeto siempre almacena al comando y los demás son para sus parámetros.

FINALIDAD:

Enviar el "prompt" del monitor (">").

Leer los comandos tecleados y sus parámetros. Inicia leyendo el primer caracter y lo almacena al principio de la fila. Continúa leyendo y almacenando los caracteres en la fila hasta encontrar un regreso de carro (CR). En cada lectura verifica si se trata de un retroceso (BS) para repetirla o continuar. Por cada caracter aceptado se incrementa un contador para conocer la longitud del comando leído.

c) Selección de comandos

En memoria ROM se encuentra la Tabla 1 con los códigos ASCII de cada comando, todos ellos de una sola letra, y el número de caracteres máximo por comando (para el monitor del MC68HC11, además se agregó un código de salto ($ 7E) y la dirección de la rutina correspondiente).

Se manejan dos tipos de comandos: aquéllos que contienen toda la información necesaria en sus parámetros para llevar a cabo su función y aquéllos que realizan una acción y después esperan por un caracter (ESPACIO o CR) para realizar otra.

FINALIDAD:

Seleccionar el comando, checar que no sobrepase el número máximo de caracteres leídos y saltar a la rutina del comando correspondiente. Comienza comparando el primer caracter leído en LECTURA DE PARÁMETROS con cada uno de los comandos ubicados en la Tabla 1 del programa monitor, cuando son iguales, checa si se ha excedido el número máximo de caracteres mediante el contador de longitud de comando, de ser así, envía el mensaje de error uno. En caso contrario, salta a la rutina de comando. En caso de que el caracter leído no sea un comando, envía el mensaje de error dos.

NÚMERO DE ERROR	MENSAJE
1	ERROR DE SINTAXIS
2	COMANDO INVÁLIDO
3	NUMERO INVÁLIDO
4	RANGO INVÁLIDO
5	DIFERENCIA EN PROGRAMA
6	ERROR DE LECTURA
7	ERROR: DIRECCIÓN DE ROM

Tab. 6.1 Mensajes de Error.

d) Subrutinas

En este bloque, se encuentran subrutinas que realizan diferentes funciones, tales como: conversión de códigos, validación de direcciones, de entrada/salida y de despliegue. Algunas de estas subrutinas están disponibles para el usuario y

se accesan a través de un salto a subrutina de acuerdo con las direcciones indicadas en la siguiente tabla.

ACCESO A SUBRUTINA	DIRECCIÓN EN MONITOR		FUNCIÓN
	8031	MC68HC11	
ASCII-HEXA	800 H	$ E800	Convierte A de código ASCII a hexadecimal.
HEXA-ASCII	803 H	$ E803	Convierte A de hexadecimal a código ASCII.
NUMASCII	806 H	$ E806	Convierte A de hexadecimal en 2 octetos ASCII y los envía por el puerto serie.
ASCIINUM	80C H	$ E80C	Convierte 2 octetos ASCII (direccionados por R0 en el 8031 y por IY en el MC68HC11) en un octeto hexadecimal y lo almacena en A.
LEE	80F H	$ E80F	Recibe un caracter ASCII por el puerto serie, lo almacena en A y hace eco.
ENVIA	815 H	$ E815	Envía el contenido ASCII de A por el puerto serie.
CR-LF	81B H	$ E81B	Envía un regreso de carro (CR) y un avance de línea (LF) por el puerto serie.
DESPMENS	81E H	$ E81E	Envía una cadena de caracteres ASCII por el puerto serie, según la longitud del mensaje apuntada por el DPTR en el 8031, o hasta encontrar un fin de texto ($ 04) en el MC68HC11, donde IX debe apuntar al principio del mensaje.

Tab. 6.2 Direcciones para salto a subrutinas del usuario.

SUBRUTINA ASCII-HEXA

FINALIDAD:

Convertir el código ASCII que se encuentra en el acumulador A a su respectivo valor hexadecimal, dejándolo en el mismo acumulador. Si el resultado no está dentro del rango [00 H,0F H], envía el mensaje de error tres. (Ver Tab. 6.1).

SUBRUTINA HEXA-ASCII

FINALIDAD:

Convertir el número hexadecimal que se encuentra en el acumulador A a su respectivo código ASCII, dejándolo en el mismo acumulador.

SUBRUTINA ASCIINUM

FINALIDAD:

Convertir dos octetos ASCII en un número hexadecimal, el cual se almacena en el acumulador A.

SUBRUTINA NUMASCII

FINALIDAD:

Convertir el número hexadecimal que se encuentra en el acumulador A a sus dos octetos ASCII correspondientes y enviarlos por el puerto serie.

SUBRUTINA DIR1

La dirección que se desea validar se encuentra en la fila generada por LECTURA DE PARÁMETROS.

FINALIDAD:

Convertir la primera dirección de código ASCII a hexadecimal y almacenarla en RAM interna.

SUBRUTINA DIR2

FINALIDAD:

Convertir la segunda dirección de código ASCII a hexadecimal y almacenarla en RAM interna. Compara ésta con la primera dirección, si la primera es mayor que la segunda, envía el mensaje de error cuatro. (Ver Tab. 6.1).

SUBRUTINA ENVIA

FINALIDAD:

Enviar el contenido del acumulador A por el puerto serie.

SUBRUTINA LEE

FINALIDAD:

Almacenar en el acumulador A el dato leído por el puerto serie.

SUBRUTINA LEER-DOS

FINALIDAD:

Leer dos caracteres ASCII, convertirlos en un número hexadecimal y almacenarlo en el acumulador A.

SUBRUTINA CR-LF

FINALIDAD:

Enviar un regreso de carro (CR) y un avance de línea (LF) por el puerto serie.

SUBRUTINA DESPMENS

El monitor del 8031 tiene una Tabla 2 de mensajes de error ubicada en memoria ROM, que contiene la longitud del mensaje y enseguida los códigos ASCII del mismo. También tiene una Tabla 3,

con el mismo formato para los nombres de los registros. El apuntador señala la dirección donde se encuentra la longitud del mensaje, para enviar caracter por caracter, tantas veces como lo indica dicha longitud.

En el caso del MC68HC11 se manejan las mismas tablas, con la diferencia de que en lugar de utilizar la longitud del mensaje para el despliegue, usa un código de fin de texto ($ 04) al final de cada mensaje.

FINALIDAD:

Enviar cadenas de caracteres a la pantalla por el puerto serie (mensajes de error, nombres de registros, pantalla de ayuda, etc).

SUBRUTINA DESPCONT

FINALIDAD:

Desplegar el contenido de la localidad indicada por el apuntador y respaldar su valor en un registro.

SUBRUTINA DESPLOC

FINALIDAD:

Desplegar la localidad y el contenido de la dirección señalada por el apuntador.

SUBRUTINA DESPREG

Los registros del usuario son actualizados en una fila de memoria RAM externa.

FINALIDAD:

Desplegar los registros del usuario y su contenido.

SUBRUTINA RESPALDA

FINALIDAD:

Almacenar el contenido actual de los registros en la fila de registros del usuario.

SUBRUTINA RECUPERA

FINALIDAD:

Almacenar el contenido actual de la fila de registros del usuario en los registros del programa del mismo.

SUBRUTINA CORRIMIENTO

FINALIDAD:

Convertir el contenido ASCII del acumulador A en hexadecimal para que este valor sea el nuevo cuarteto bajo de la localidad direccionada. El cuarteto bajo anterior pasa a ser el nuevo cuarteto alto.

e) Rutinas de comando

En este bloque se encuentra la programación necesaria para llevar a cabo la ejecución de cada comando por separado.

6.1 Sintaxis general de los comandos

COMANDO [<*parámetros*>](CR)

Cada línea de comando consta de:

Comando: (resaltado), siempre es de una sola letra mayúscula. El monitor no reconoce letras minúsculas.

Espacio: se utiliza para separar el comando de los parámetros y a los parámetros entre sí.

Parámetros: (en itálicas), los parámetros están especificados entre paréntesis angulares, cada línea puede tener desde cero hasta tres parámetros. Los corchetes son utilizados para indicar parámetros opcionales. El monitor maneja números y direcciones hexadecimales de cuatro caracteres.

CR: Debe presionarse para aceptar la línea de comando.

Los paréntesis indican un caracter especial. El monitor utiliza los caracteres especiales (ESPACIO) o (CR) para ejecutar acciones específicas dentro de algunos comandos.

6.2 Comandos del programa monitor

A: desplegar la pantalla de ayuda.

SINTAXIS: A(CR)

Este comando despliega la sintaxis y función que realizan cada uno de los comandos usados en el programa monitor, tal como se muestra en la Fig. 6.1.

B: borrar el contenido de un bloque de memoria.

SINTAXIS: B <*dir1*>[<*dir2*>](CR)

donde *dir1* y *dir2* son las direcciones inicial y final del bloque, respectivamente. Si *dir1* es una localidad de ROM, despliega el mansaje de error siete. (Ver Tab. 6.1).

Este comando pone ceros en las localidades desde *dir1* hasta *dir2*. Si se omite esta última, sólo pone a ceros *dir1*.

```
A(CR)                          AYUDA
F(CR)                          VELOCIDAD A 4800 BAUDS
G(CR)                          VELOCIDAD A 1200 BAUDS
L <num>(CR)                    LEE PROGRAMA
V <num>(CR)                    VERIFICA PROGRAMA
B <dir1>[ <dir2>](CR)          BORRA LOCALIDAD O BLOQUE
D <dir1>[ <dir2>](CR)          DESPLIEGA LOCALIDADES
C <dir1>[ <dir2>](CR)          CORRE PROGRAMA
O <dir1> <dir2>(CR)            CALCULA DESPLAZAMIENTOS RELATIVOS
I <dir1> <dir2>(CR)            IMPRIME PROGRAMA
M <dir1> <dir2> <dir3>(CR)     MUEVE BLOQUE DE MEMORIA
R(CR)                          DESPLIEGA REGISTROS
     (ESPACIO) sigue
     (CR)     termina
E <dir1>(CR)                   EXAMINA Y MODIFICA MEMORIA
     (ESPACIO) sigue
     (CR)     termina
T <dir1>(CR)                   TRAZA PROGRAMA
     (ESPACIO) sigue
     (CR)     termina
```

Fig. 6.1 Pantalla de ayuda.

C: correr un programa residente en memoria RAM.

SINTAXIS: C <*dir1*>[<*dir2*>](CR)

donde *dir1* es la dirección a partir de la cual inicia la ejecución del programa y *dir2* es la dirección usada para determinar el fin de corrida.

Este comando permite ejecutar un programa residente en memoria RAM, como si fuera una subrutina, desde *dir1* hasta *dir2*, sin incluir esta última. Si *dir2* se omite, el usuario debe colocar el código de retorno de subrutina correspondiente.

D: desplegar el contenido de un bloque de memoria.

SINTAXIS: D <*dir1*>[<*dir2*>](CR)

donde *dir1* y *dir2* son las direcciones inicial y final del bloque, respectivamente.

Este comando despliega el contenido hexadecimal de un bloque de memoria iniciando en *dir1* hasta *dir2*, si esta última no se indica, se despliegan cuatro renglones completos de 16 octetos.

E: examinar y modificar el contenido de una localidad de memoria.

SINTAXIS: E <*dir1*>(CR)

donde *dir1* es la dirección de la localidad que se desea examinar y modificar. Si es una localidad de ROM, envía el mensaje de error siete. (Ver Tab. 6.1).

Este comando permite leer y modificar el contenido de la memoria a partir de *dir1*.

Con (ESPACIO) muestra la siguiente localidad y con (CR) termina la operación.

Si el usuario requiere modificar el contenido de una localidad, debe proporcionar los caracteres hexadecimales correspondientes al valor deseado. Puede introducir caracteres las veces que sea necesario, el monitor sólo acepta los dos últimos.

F, G: cambiar la velocidad de transmisión del puerto serie.

SINTAXIS: F(CR) (cambia la velocidad a 4800)
G(CR) (cambia la velocidad a 1200)

La velocidad a la que opera el puerto serie por omisión es de 9600 bauds. Si se desea restituir la velocidad original se debe dar un reestablecimiento al módulo.

I: imprimir un programa.

SINTAXIS: I <*dir1*> <*dir2*>(CR)

donde *dir1* y *dir2* son las direcciones inicial y final del programa, respectivamente.

Este comando se utiliza para enviar un programa en bloques, desde *dir1* hasta *dir2*, según el formato de la Fig. 6.2.

TIPO	LONGITUD	DIRECCIÓN	DATOS	PARIDAD

TIPO	Tipo de bloque: S1 para cualquier bloque, S9 para el último. El monitor desecha otros tipos de bloque.
LONGITUD	Contador de pares de caracteres, sin incluir el tipo y la propia longitud.
DIRECCIÓN	Dirección de inicio bloque.
DATOS	Código de n datos a cargar o descargar.
PARIDAD	Octeto menos significativo del complemento a uno de la suma de: LONGITUD, DIRECCIÓN y DATOS.

Fig. 6.2a Formato para carga y descarga de programas del módulo 68HC11.

:	LONGITUD	DIRECCIÓN	NULO	DATOS	PARIDAD

:	Señala inicio de bloque.
LONGITUD	Contador de pares de datos.
DIRECCIÓN	Dirección de inicio bloque.
NULO	Caracter nulo (00 en ASCII).
DATOS	Código de n datos a cargar o descargar.
PARIDAD	Octeto menos significativo del complemento a dos de la suma de: LONGITUD, DIRECCIÓN y DATOS.

Fig. 6.2b Formato para carga y descarga de programas del módulo 8031.

L: leer un programa.

SINTAXIS: **L** <*num*>(CR)

donde *num* es un número hexadecimal de 16 bites.

Este comando permite leer un programa por bloques, de acuerdo con el formato de la Fig. 6.2 y cargarlo en memoria RAM a partir de la dirección del bloque más el desplazamiento indicado por *num*. Si la dirección resultante pertenece a una localidad de ROM, envía el mensaje de error siete. (Ver Tab. 6.1). Si los datos leidos no corresponden con el formato, envía el mensaje de error seis.

M: mover bloques de memoria.

SINTAXIS: **M** <*dir1*> <*dir2*> <*dir3*>(CR)

donde *dir1* y *dir2* son las direcciones inicial y final del bloque, respectivamente y *dir3* es la dirección inicial a donde se va a

mover el bloque. Si *dir3* es una localidad de ROM, envía el mensaje de error siete. (Ver Tab. 6.1).

Este comando mueve un bloque de memoria definido por *dir1* y *dir2*, a partir de la localidad indicada por *dir3*.

O: calcular desplazamientos relativos.

SINTAXIS: O <*dir1*> <*dir2*>(CR)

donde *dir1* y *dir2* son las direcciones inicial y final, respectivamente.

Este comando calcula desplazamientos relativos para auxiliar al usuario. Si la distancia entre *dir1* y *dir2* no se encuentra dentro del rango [-128,+127], envía el mensaje de error cuatro. (Ver Tab. 6.1).

R: examinar y modificar registros.

SINTAXIS: R(CR)

Este comando despliega el contenido de los registros principales del microcontrolador. Enseguida, los registros se despliegan uno a uno para que puedan ser modificados (igual que el comando E), inicia con el acumulador A y con (ESPACIO) muestra el siguiente registro o con (CR) termina la operación.

T: trazar un programa.

SINTAXIS: T <*dir1*>(CR)

donde *dir1* es la dirección a partir de la cual se comienza a trazar el programa.

Este comando permite trazar la instrucción indicada por

dir1. Con (ESPACIO) traza la siguiente instrucción y con (CR) termina la operación.

V: verificar un programa.

SINTAXIS: V <*num*>(CR)

donde *num* es un número hexadecimal de 16 bites.

Este comando compara un programa residente en memoria RAM contra uno que se lee. Es similar al comando L, excepto que el programa leido no se carga en memoria. En caso de haber diferencia entre los dos programas, envía el mensaje de error cinco. (Ver Tab. 6.1).

6.3 Vectores de interrupción y memoria interna del usuario

Las siguientes tablas muestran los vectores de interrupción, ubicados en memoria RAM, disponibles para el usuario. En la parte de inicialización, el programa monitor coloca un código de salto en el primer octeto del vector y en los dos siguientes, el usuario debe cargar la dirección de su rutina de servicio.

El programa monitor 8031 utiliza el banco 3 de registros internos, las localidades 68 H-7F H para manejo de variables y la localidad 20 H para manejo de banderas; por consiguiente, el usuario dispone de los bancos de registros 0, 1 y 2 y de las localidades 21 H-67 H de RAM interna.

De la misma manera, el programa monitor 68HC11 usa la memoria RAM interna a partir de la localidad $ 00 hasta la $ A7, dejando libres las localidades $ A8-$ FF al usuario.

FUENTE DE INTERRUPCIÓN	VECTOR
Acoplamiento de Comunicación Serial (SCI)	$ 006D-$ 006E
Acoplamiento Periférico Serial (SPI)	$ 0070-$ 0071
Entrada del Acumulador de Pulsos (PAI)	$ 0073-$ 0074
Saturación del Acumulador de Pulsos (PAOV)	$ 0076-$ 0077
Saturación del Temporizador (TO)	$ 0079-$ 007A
Captura por ent. 4/salida por comp. 5 (I4O5)	$ 007C-$ 007D
Salida por Comparación 4 (OC4)	$ 007F-$ 0080
Salida por Comparación 3 (OC3)	$ 0082-$ 0083
Salida por Comparación 2 (OC2)	$ 0085-$ 0086
Salida por Comparación 1 (OC1)	$ 0088-$ 0089
Captura por Entrada 3 (IC3)	$ 008B-$ 008C
Captura por Entrada 2 (IC2)	$ 008E-$ 008F
Captura por Entrada 1 (IC1)	$ 0091-$ 0092
Interrupción de Tiempo Real (RTI)	$ 0094-$ 0095
IRQ	$ 0097-$ 0098
XIRQ	$ 009A-$ 009B
Interrupción de Programa (SWI)	$ 009D-$ 009E
Código de Operación Ilegal (IOT)	$ 00AO-$ 00A1
Operación adecuada de la computadora (NOCOP)	$ 00A3-$ 00A4
Monitor del Reloj (CME)	$ 00A6-$ 00A7

Tab. 6.3 Vectores de interrupción del monitor MC68HC11.

FUENTE DE INTERRUPCIÓN	VECTOR
Interrupción Externa 0 (INT0)	3F91 H-3F92 H
Interrupción del Temporizador 0 (T0)	3F94 H-3F95 H
Interrupción Externa 1 (INT1)	3F97 H-3F98 H
Interrupción del Temporizador 1 (T1)	3F9A H-3F9B H

Tab. 6.4 Vectores de interrupción del monitor 8031.

7. CONCLUSIONES

Los módulos de desarrollo descritos en este trabajo han sido probados y operan satisfactoriamente. En breve, serán utilizados para depurar y probar algunos sistemas que están en desarrollo.

Cabe decir, que el programa monitor sirvió para la depuración del propio monitor, lo que fue realmente útil para la conclusión del mismo.

El programa monitor fue dividido en módulos para facilitar la modificación o la inserción de funciones, según las necesidades futuras de la Coordinación.

El circuito impreso del microcontrolador 68HC11 se elaboró con el circuito en su versión de 48 alfileres en doble fila, por estar éste disponible en la Coordinación, pero se piensa realizar otro utilizando el tipo PLCC de 52 alfileres para ahorrar espacio y aumentar las líneas del convertidor A/D.

8. BIBLIOGRAFÍA

1. DI GIACOMO, Joseph. Digital Bus Handbook. Mc. Graw-Hill Co. USA, 1990. pp. 8.1-8.33.

2. FOLTS, Harold y KARP, Harry. Compilation of Data Communications Standards. Mc. Graw-Hill Co. USA, 1978. pp. 701-726, 813-829.

3. GON WONG, Emma y ALVAREZ-ICAZA L., Luis. Controlador Universal Industrial. Informe del Instituto de Ingeniería, UNAM. México, D. F., 1989. pp. 15-18.

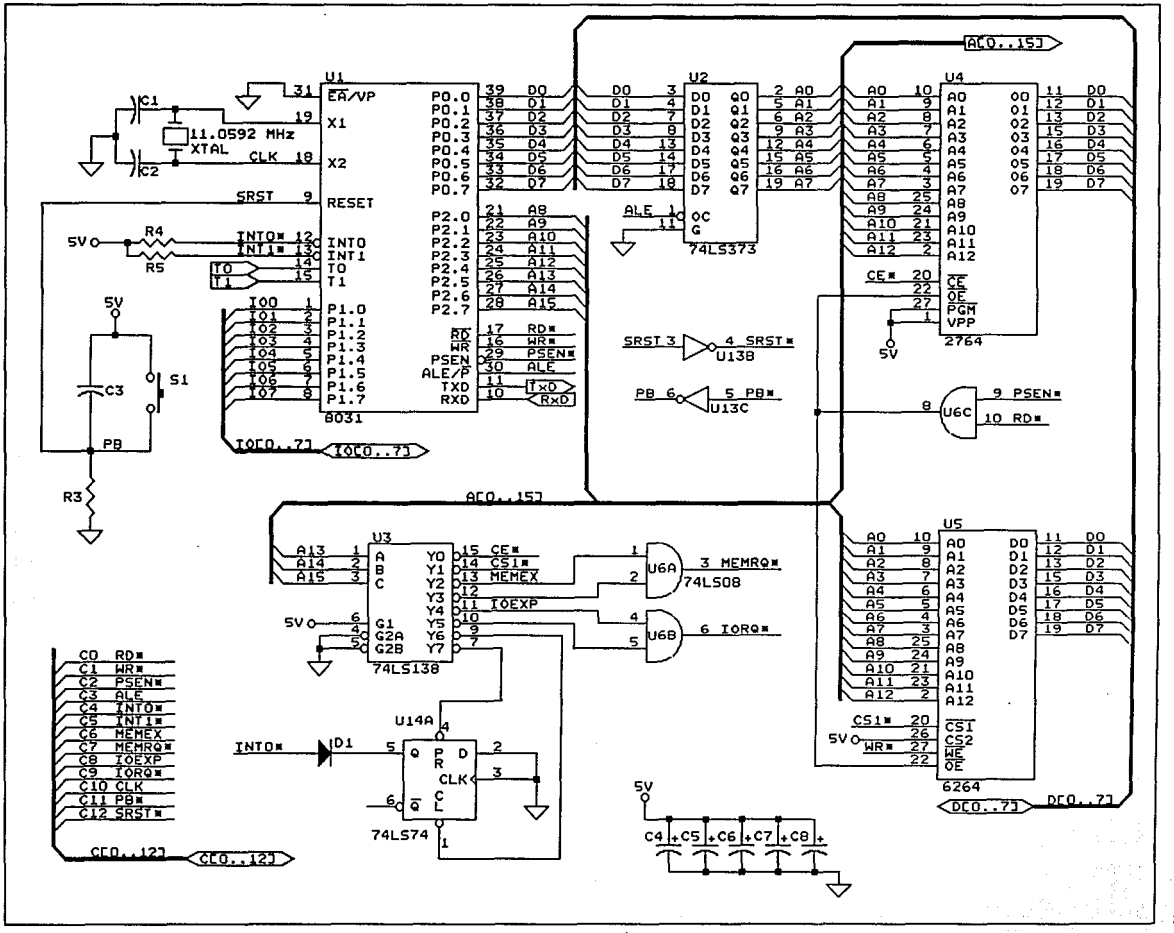
4. IEEE Inc. IEEE Standard for an 8-bit Microcomputer Bus System: STD BUS, IEEE Std 961-1987. The Institute of Electrical and Electronics Engineers Inc. USA, 1988.

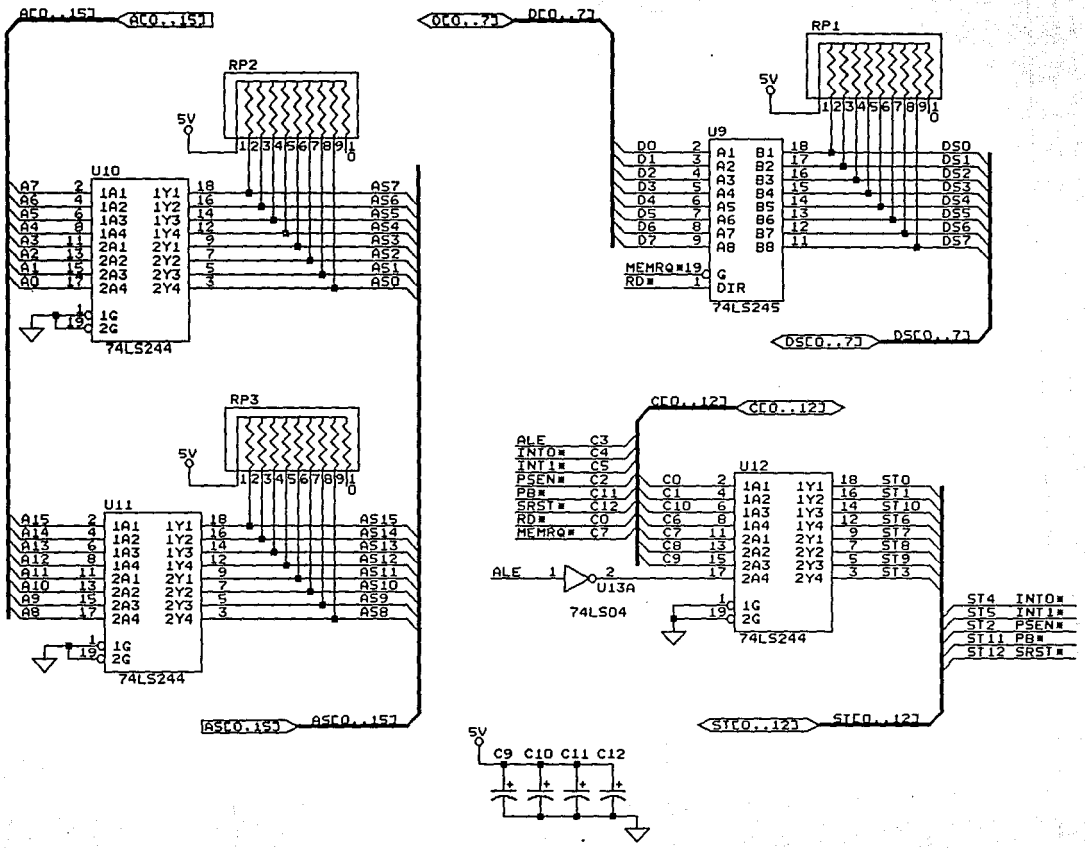
5. INTEL Co. Microcontroller Handbook. INTEL Co. USA, 1985. pp. 7.1, 10.60.

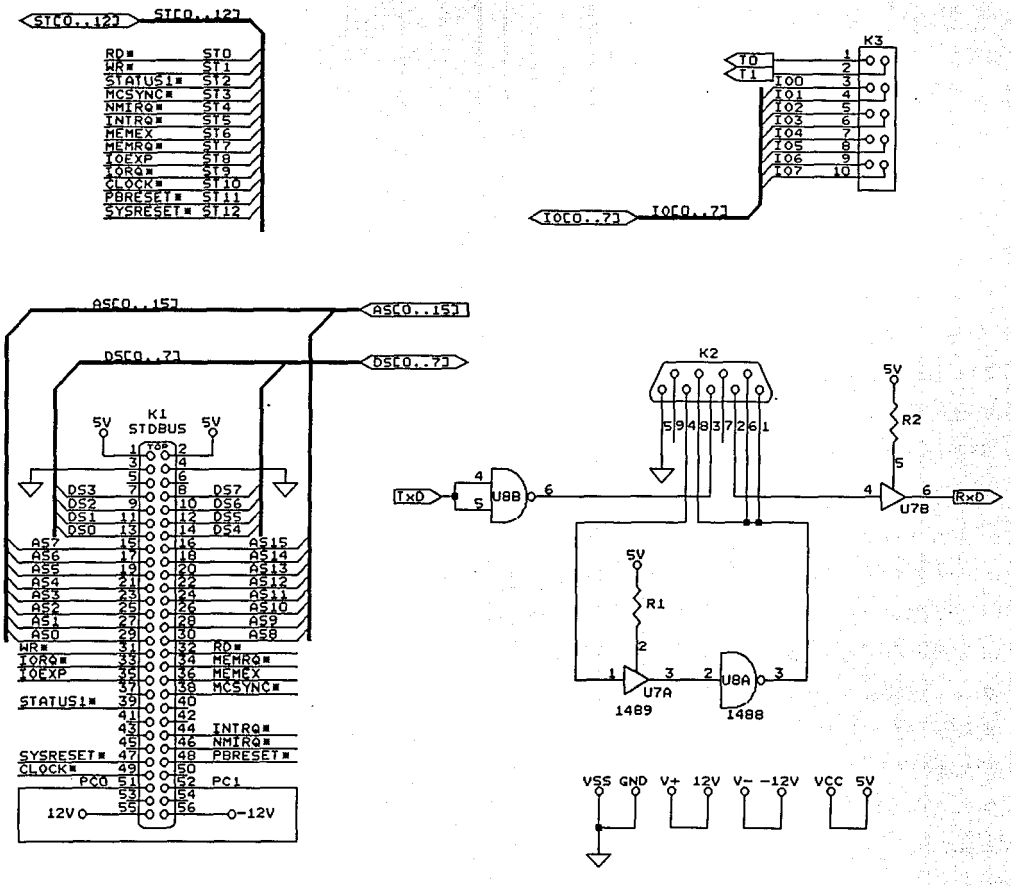
6. MOTOROLA Inc. M68HC11 Reference Manual. MOTOROLA Inc. USA, 1989. pp. 1.1, A.13.

APÉNDICE A

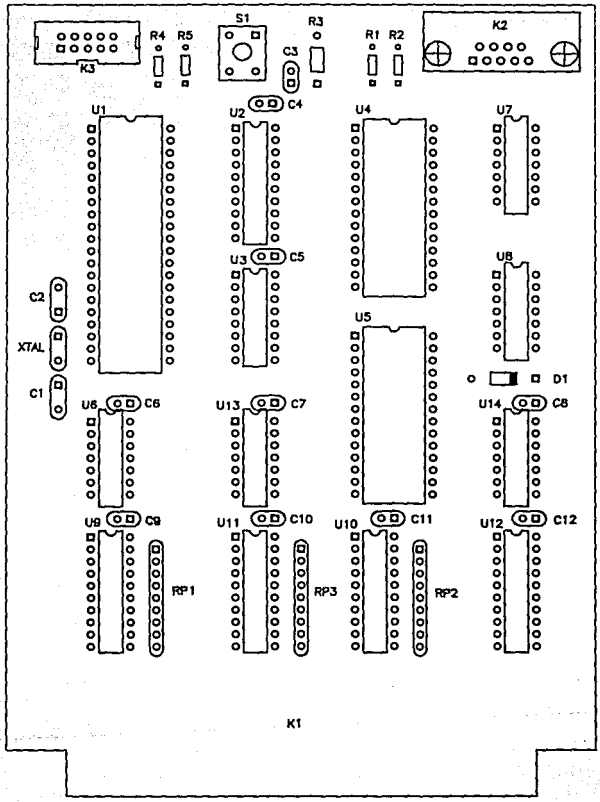
DIAGRAMAS ELECTRÓNICOS, DE DISPOSICIÓN, DEL CIRCUITO IMPRESO Y LISTA DE PARTES DEL 8031


U1
8031
EA/VP
X1
X2
CLK
RESET
INT0
INT1
T0
T1
P1.0
P1.1
P1.2
P1.3
P1.4
P1.5
P1.6
P1.7
P2.0
P2.1
P2.2
P2.3
P2.4
P2.5
P2.6
P2.7
RD
WR
PSEN
ALE/P
TXD
RXD
11.0592 MHz
XTAL
C1
C2
C3
R3
R4
R5
S1
SRST
PB
5V
IO[0..7]
A[0..15]
U2
74LS373
ALE
OC
G
U4
2764
CE
OE
PGM
VPP
U5
6264
CS1
CS2
WE
U3
74LS138
A13
A14
A15
G1
G2A
G2B
Y0
Y1
Y2
Y3
Y4
Y5
Y6
Y7
U6A
74LS08
U6B
U6C
3 MEMRQ
6 IORQ
IOEXP
MEMEX
U13B
U13C
SRST
PB
PSEN
RD
U14A
74LS74
INT0
D1
C0 RD
C1 WR
C2 PSEN
C3 ALE
C4 INT0
C5 INT1
C6 MEMEX
C7 MEMRQ
C8 IOEXP
C9 IORQ
C10 CLK
C11 PB
C12 SRST
C[0..12]
C4
C5
C6
C7
C8
D[0..7]

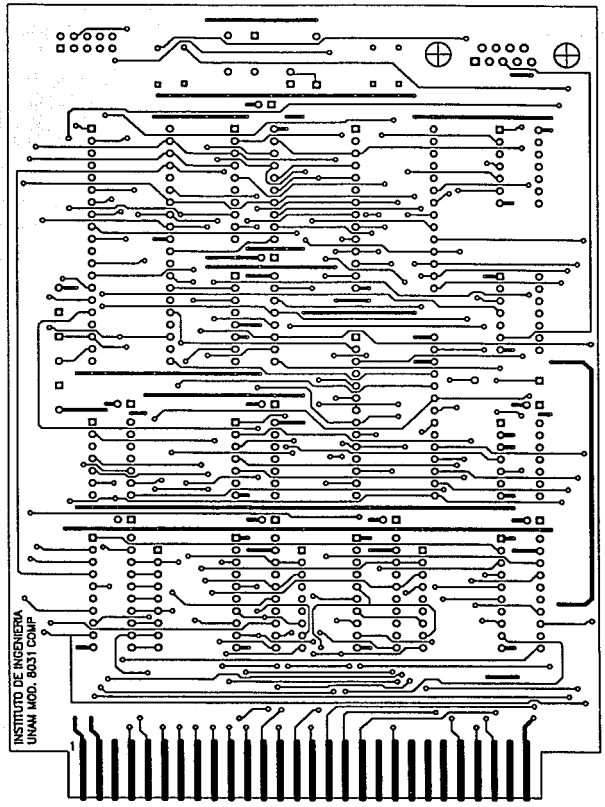

ACO..15]
ACO..15]
RP2
5V
1 2 3 4 5 6 7 8 9 10
U10
A7 2
A6 4
A5 6
A4 8
A3 11
A2 13
A1 15
A0 17
1A1 1Y1 18
1A2 1Y2 16
1A3 1Y3 14
1A4 1Y4 12
2A1 2Y1 9
2A2 2Y2 7
2A3 2Y3 5
2A4 2Y4 3
AS7
AS6
AS5
AS4
AS3
AS2
AS1
AS0
1
19
1G
2G
74LS244
RP3
5V
1 2 3 4 5 6 7 8 9 10
U11
A15 2
A14 4
A13 6
A12 8
A11 11
A10 13
A9 15
A8 17
1A1 1Y1 18
1A2 1Y2 16
1A3 1Y3 14
1A4 1Y4 12
2A1 2Y1 9
2A2 2Y2 7
2A3 2Y3 5
2A4 2Y4 3
AS15
AS14
AS13
AS12
AS11
AS10
AS9
AS8
1
19
1G
2G
74LS244
ASCO..15]
ASCO..15]
DCO..7]
DCO..7]
RP1
5V
1 2 3 4 5 6 7 8 9 10
U9
DO 2
D1 3
D2 4
D3 5
D4 6
D5 7
D6 8
D7 9
A1 B1 18
A2 B2 17
A3 B3 16
A4 B4 15
A5 B5 14
A6 B6 13
A7 B7 12
A8 B8 11
DS0
DS1
DS2
DS3
DS4
DS5
DS6
DS7
MEMRQ*19
RD* 1
G
DIR
74LS245
DSCO..7]
DSCO..7]
CCO..12]
CCO..12]
ALE C3
INTO* C4
INT1* C5
PSEN* C2
PB* C11
SRST* C12
RD* C0
MEMRQ* C7
U12
CO 2
C1 4
C10 6
C6 8
C7 11
C8 13
C9 15
1A1 1Y1 18
1A2 1Y2 16
1A3 1Y3 14
1A4 1Y4 12
2A1 2Y1 9
2A2 2Y2 7
2A3 2Y3 5
2A4 2Y4 3
ST0
ST1
ST10
ST6
ST7
ST8
ST9
ST3
ALE 1
2
U13A
74LS04
1
19
1G
2G
74LS244
ST4 INTO*
ST5 INT1*
ST2 PSEN*
ST11 PB*
ST12 SRST*
STCO..12]
STCO..12]
5V
C9 C10 C11 C12


ST[0..12]
ST[0..12]
RD*
WR*
STATUS1*
MCSYNC*
NMIRQ*
INTRQ*
MEMEX
MEMRQ*
IOEXP
IORQ*
CLOCK*
PBRESET*
SYSRESET*
ST0
ST1
ST2
ST3
ST4
ST5
ST6
ST7
ST8
ST9
ST10
ST11
ST12
K3
T0
T1
I00
I01
I02
I03
I04
I05
I06
I07
IO[0..7]
IO[0..7]
AS[0..15]
AS[0..15]
DS[0..7]
DS[0..7]
K1
STDBUS
5V
5V
TOP
DS3
DS2
DS1
DS0
DS7
DS6
DS5
DS4
AS7
AS6
AS5
AS4
AS3
AS2
AS1
AS0
AS15
AS14
AS13
AS12
AS11
AS10
AS9
AS8
WR*
IORQ*
IOEXP
STATUS1*
RD*
MEMRQ*
MEMEX
MCSYNC*
INTRQ*
NMIRQ*
PBRESET*
SYSRESET*
CLOCK*
PC0
PC1
12V
-12V
K2
5 9 4 8 3 7 2 6 1
5V
R2
TxD
U8B
U7B
RxD
5V
R1
U7A
1489
U8A
1488
VSS GND V+ 12V V- -12V VCC 5V

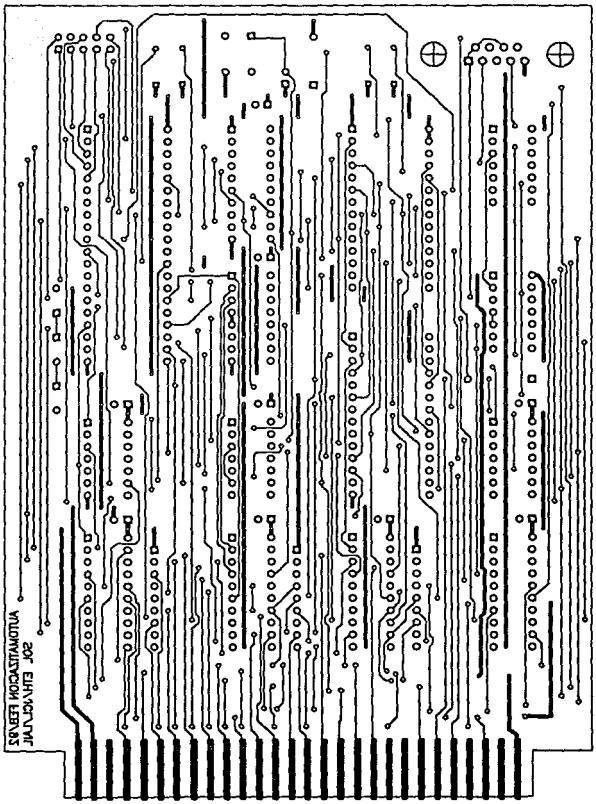
DISPOSICIÓN DE COMPONENTES



CIRCUITO IMPRESO DE LA CARA DE COMPONENTES



CIRCUITO IMPRESO DE LA CARA DE SOLDADURA



LISTA DE PARTES

No.	CANTIDAD	REFERENCIA	PARTE
1	1	U1	8031
2	1	U2	74LS373
3	1	U3	74LS138
4	1	U4	2764
5	1	U5	6264
6	1	U6	74LS08
7	1	U7	1489
8	1	U8	1488
9	1	U9	74LS245
10	3	U10-U12	74LS244
11	1	U13	74LS04
12	1	U14	74LS74
13	1	K1	PEINE 56 ALFILERES
14	1	K2	CONECTOR DB9 HEMBRA
15	1	K3	CONECTOR 10
16	1	XTAL	CRISTAL DE 11.0592 MHz
17	2	C1, C2	30 pF
18	1	C3	10 μF
19	9	C4-C12	0.1 μF
20	5	R1, R2, R4, R5	4.7 KΩ
21	1	R3	8.2 KΩ
22	3	RP1-RP3	4.7 KΩ
23	1	D1	DIODO 1N
24	1	S1	PUSH BUTTON

APÉNDICE B

LISTADO DEL PROGRAMA MONITOR 8031

```
                ;************************************
                ;*        CARACTERES ESPECIALES     *
                ;************************************
003E =          PROMPT  EQU '>'          ;PROMPT DEL MONITOR
0020 =          ESP     EQU 20H          ;ESPACIO
000D =          CR      EQU 0DH          ;REGRESO DE CARRO
000A =          LF      EQU 0AH          ;AVANCE DE LINEA
0008 =          BS      EQU 08H          ;RETROCESO

                ;************************************
                ;*              BANDERAS            *
                ;************************************
0000 =          VERIF   EQU 00H          ;BANDERA VERIFICA
0001 =          ECO     EQU 01H          ;BANDERA ECO
0003 =          DESP    EQU 03H          ;BANDERA DESPLIEGUE
0004 =          CORRE   EQU 04H          ;BANDERA CORRE

                ;************************************
                ;*          TABLAS EN ROM           *
                ;************************************
1C40 =          TAB1    EQU 1C40H        ;TABLA DE COMANDOS
1C5E =          TAB2    EQU TAB1+1EH     ;TABLA DE REGISTROS
1CAD =          TAB3    EQU TAB2+4FH     ;TABLA DE ERRORES
1CAD =          ERR1    EQU TAB3         ;ERROR DE SINTAXIS
1CBF =          ERR2    EQU ERR1+12H     ;COMANDO INVALIDO
1CD0 =          ERR3    EQU ERR2+11H     ;NUMERO INVALIDO
1CE0 =          ERR4    EQU ERR3+10H     ;RANGO INVALIDO
1CEF =          ERR5    EQU ERR4+0FH     ;DIFERENCIA EN PROGRAMA
1D06 =          ERR6    EQU ERR5+17H     ;ERROR DE LECTURA
1D17 =          ERR7    EQU ERR6+11H     ;ERROR: DIRECCION DE ROM
1D2F =          TAB4    EQU ERR7+18H     ;PANTALLA DE AYUDA

                ;************************************
                ;*   ESPACIOS RESERVADOS DE MEMORIA *
                ;************************************
0070 =          FILA    EQU 70H          ;FILA DE PARAMETROS EN RAM INT.
0072 =          FIL2    EQU FILA+02H     ;INICIO DE DIR. EN LA FILA
3F90 =          VECINT  EQU 3F90H        ;VECTORES DE INT. EN RAM EXT.
3F90 =          VINT0   EQU VECINT       ;VECTOR DE INT0
3F93 =          VT0     EQU VECINT+03H   ;VECTOR DE T0
3F96 =          VINT1   EQU VECINT+06H   ;VECTOR DE INT1
3F99 =          VT1     EQU VECINT+09H   ;VECTOR DE T1
3FA0 =          BUFLE   EQU 3FA0H        ;BUFFER DE LECT. Y ESC. EN RAM EXT.
3FE0 =          PROGCT  EQU 3FE0H        ;PROGRAMA PARA C Y T EN RAM EXT.
3FEA =          FILREG  EQU 3FEAH        ;FILA DE REGISTROS DEL
                                         ;USUARIO EN RAM EXT.

                ;************************************
                ;*   INICIO DE PROGRAMA EN ROM      *
                ;************************************
0000                    ORG 00H
0000 4112               AJMP INREG       ;SALTA A INICIALIZACION
                ;RUTINA DE SERVICIO DE INTERRUPCION DE INTO
```

```
                ;USADA PARA EL TRAZADO
0003                    ORG 03H
0003 023F90             LJMP VINTO
                ;RUTINA DE SERVICIO DE INTERRUPCION DE TO
000B                    ORG 0BH
000B 023F93             LJMP VTO
                ;RUTINA DE SERVICIO DE INTERRUPCION DE INT1
0013                    ORG 13H
0013 023F96             LJMP VINT1
                ;RUTINA DE SERVICIO DE INTERRUPCION DE T1
001B                    ORG 1BH
001B 023F99             LJMP VT1
                ;RUTINA DE SERVICIO DE INTERRUPCION DEL PUERTO SERIE
0023                    ORG 23H
0023 75A800             MOV IE,#00H     ;DESHABILITA INTERRUPCIONES
0026 E599               MOV A,SBUF
0028 300102             JNB ECO,PTO     ;SALTA SI LA BANDERA DE ECO
002B 11E4               ACALL ENVIA     ;ESTA APAGADA
002D 32          PTO:   RETI

                ;*************************************************
                ;*                SUBRUTINA CR-LF                *
                ;*  Envía un regreso de carro (CR) y un avance   *
                ;*  de línea (LF) por el puerto serie.           *
                ;*************************************************
002E C0E0       CRLF:   PUSH ACC        ;SALVA A
0030 740D               MOV A,#CR
0032 11E4               ACALL ENVIA     ;ENVIA UN REGRESO DE CARRO
0034 740A               MOV A,#LF
0036 11E4               ACALL ENVIA     ;ENVIA AVANCE DE LINEA
0038 D0E0               POP ACC
003A 22                 RET

                ;*************************************************
                ;*            SUBRUTINA HEXA-ASCII               *
                ;*  Convierte A de hexadecimal a código ASCII    *
                ;*  y almacena el resultado en A.                *
                ;*************************************************
003B 2430       HEAS:   ADD A,#30H
003D B43901             CJNE A,#39H,HEX ;SALTA SI A <> 39H
0040 22                 RET
0041 4002       HEX:    JC HEX1         ;SALTA SI A < 39H
0043 2407               ADD A,#07H
0045 22         HEX1:   RET

                ;*************************************************
                ;*            SUBRUTINA ASCII-HEXA               *
                ;*  Convierte A de código ASCII a hexadecimal.   *
                ;*  Si el resultado no está dentro del rango     *
                ;*  [00H, 0FH], envía el mensaje:   "NUMERO      *
                ;*  INVALIDO".                                   *
                ;*************************************************
0046 547F       ASHE:   ANL A,#7FH
```

```
0048 C3              CLR C
0049 9430            SUBB A,#30H
004B B41709          CJNE A,#17H,ASC2  ;SALTA SI A <> 17H
004E 901CD0    ASC1: MOV DPTR,#ERR3    ;"NUMERO INVALIDO"
0051 112E            ACALL CRLF
0053 11A3            ACALL MENS
0055 415E            AJMP INI
0057 50F5      ASC2: JNC ASC1          ;SALTA SI A > 17H
0059 B40A09          CJNE A,#0AH,ASC4  ;SALTA SI A <> 0AH
005C B41009    ASC3: CJNE A,#10H,ASC5  ;SALTA SI A <> 10H
005F 014E            AJMP ASC1
0061 C3        ASC6: CLR C
0062 9407            SUBB A,#07H
0064 22              RET
0065 50F5      ASC4: JNC ASC3          ;SALTA SI A > 0AH
0067 22              RET
0068 50F7      ASC5: JNC ASC6          ;SALTA SI A > 10H
006A 014E            AJMP ASC1

                ;***********************************************
                ;*              SUBRUTINA ASCIINUM             *
                ;*  Convierte dos octetos ASCII direccionados  *
                ;*  por R0 en un octeto hexadecimal y almacena *
                ;*  el resultado en A. Utiliza R3 para guardar *
                ;*  el cuarteto alto.                          *
                ;***********************************************
006C C0F0      ASNU: PUSH B            ;SALVA B
006E E6              MOV A,@R0         ;CARGA EL PRIMER CODIGO ASCII
006F 1146            ACALL ASHE        ;LO CONVIERTE EN HEXADECIMAL
0071 75F010          MOV B,#10H        ;CORRIMIENTO DEL CUARTETO BAJO
0074 A4              MUL AB            ;AL ALTO
0075 F51B            MOV 1BH,A         ;ALMACENA CUARTETO ALTO
0077 08              INC R0
0078 E6              MOV A,@R0         ;CARGA EL SEGUNDO CODIGO ASCII
0079 1146            ACALL ASHE        ;LO CONVIERTE EN HEXADECIMAL
007B 251B            ADD A,1BH         ;UNE CUARTETO ALTO CON BAJO
007D D0F0            POP B
007F 22              RET

                ;***********************************************
                ;*              SUBRUTINA NUMASCII             *
                ;*  Convierte el valor hexadecimal de A en los *
                ;*  dos octetos ASCII correspondientes y los   *
                ;*  envía por el puerto serie. Utiliza R3 para *
                ;*  almacenar el cuarteto bajo.                *
                ;***********************************************
0080 C0F0      NUAS: PUSH B            ;SALVA B
0082 F51B            MOV 1BH,A
0084 531B0F          ANL 1BH,#0FH      ;ALMACENA CUARTETO BAJO
0087 75F010          MOV B,#10H        ;CORRIMIENTO DEL CUARTETO ALTO
008A 84              DIV AB            ;AL BAJO EN A
008B 113B            ACALL HEAS        ;LO CONVIERTE EN ASCII
008D 11E4            ACALL ENVIA       ;ENVIA ASCII DEL CUARTETO ALTO
```

```
008F E51B            MOV A,1BH
0091 113B            ACALL HEAS      ;CONVIERTE CUARTETO BAJO EN ASCII
0093 11E4            ACALL ENVIA     ;LO ENVIA
0095 D0F0            POP B
0097 22              RET

                ;***********************************************
                ;*              SUBRUTINA VALIDADIR            *
                ;*  Convierte una dirección  ASCII (4 octetos), *
                ;*  apuntada por R0, en una dirección hexadecimal *
                ;*  (2 octetos). Almacena DIR alta y DIR baja en *
                ;*  las localidades apuntadas por R1.          *
                ;***********************************************
0098 116C       VALI:   ACALL ASNU      ;CONVIERTE DOS OCTETOS ASCII EN
009A F7                 MOV @R1,A       ;HEXADECIMAL Y ALMACENA DIR. ALTA
009B 08                 INC R0
009C 09                 INC R1
009D 116C               ACALL ASNU      ;CONVIERTE DOS OCTETOS ASCII EN
009F F7                 MOV @R1,A       ;HEXADECIMAL Y ALMACENA DIR. BAJA
00A0 08                 INC R0
00A1 09                 INC R1
00A2 22                 RET

                ;***********************************************
                ;*              SUBRUTINA DESPMENS             *
                ;*  Envía una cadena de caracteres ASCII por   *
                ;*  el puerto serie. En R2 carga la longitud   *
                ;*  la cadena.                                 *
                ;***********************************************
00A3 C002       MENS:   PUSH R2         ;SALVA R2
00A5 E4                 CLR A
00A6 93                 MOVC A,@A+DPTR
00A7 FA                 MOV R2,A        ;CARGA LONGITUD DE CADENA
00A8 0A                 INC R2
00A9 DA04       CR1:    DJNZ R2,CR2
00AB A3                 INC DPTR
00AC D002               POP R2
00AE 22                 RET
00AF A3         CR2:    INC DPTR        ;CICLO DE ENVIO DE CARACTERES
00B0 E4                 CLR A
00B1 93                 MOVC A,@A+DPTR
00B2 11E4               ACALL ENVIA
00B4 01A9               AJMP CR1

                ;***********************************************
                ;*              SUBRUTINA DESPCONT             *
                ;*  Despliega el contenido de la localidad     *
                ;*  apuntada por el DPTR y lo almacena en R4.  *
                ;***********************************************
00B6 7420       CONT:   MOV A,#ESP
00B8 11E4               ACALL ENVIA     ;ENVIA UN ESPACIO
00BA E0                 MOVX A,@DPTR    ;CARGA CONTENIDO EN A
00BB FC                 MOV R4,A        ;RESPALDA CONTENIDO EN R4, LO
```

```
00BC 1180              ACALL NUAS       ;CONVIERTE EN ASCII Y LO ENVIA
00BE 22                RET

                ;*********************************************
                ;*                SUBRUTINA DESPLOC              *
                ;*  Despliega la dirección apuntada por el       *
                ;*  DPTR y su contenido.                         *
                ;*********************************************
00BF E583       LOC:   MOV A,DPH        ;CARGA DIRECCION ALTA EN A
00C1 1180              ACALL NUAS       ;LA CONVIERTE EN ASCII Y LA ENVIA
00C3 E582              MOV A,DPL        ;CARGA DIRECCION BAJA EN A
00C5 1180              ACALL NUAS       ;LA CONVIERTE EN ASCII Y LA ENVIA
00C7 743A              MOV A,#3AH
00C9 11E4              ACALL ENVIA      ;ENVIA ":"
00CB 11B6              ACALL CONT       ;DESPLIEGA CONTENIDO DE LA DIR
00CD 22                RET

                ;*********************************************
                ;*                  SUBRUTINA LEE              *
                ;*  Habilita interrupción del puerto serie,    *
                ;*  espera por el caracter a ser leído y lo    *
                ;*  almacena en A.                             *
                ;*********************************************
00CE 75A890     LEE:   MOV IE,#90H      ;HABILITA INT DEL PTO. SERIE
00D1 3098FD            JNB RI,$         ;CONTINUA HASTA QUE SE ENCIENDA
00D4 C298              CLR RI
00D6 22                RET


                ;*************************************************
                ;*          SUBRUTINA LEE DOS OCTETOS            *
                ;*  Lee dos caracteres ASCII, los guarda en las  *
                ;*  localidades 6EH, 6FH y los convierte en un   *
                ;*  número hexadecimal que almacena en A.        *
                ;*************************************************
00D7 11CE       LEER2: ACALL LEE        ;LEE PRIMER CARACTER
00D9 F56E              MOV 6EH,A        ;LO ALMACENA EN 6EH
00DB 11CE              ACALL LEE        ;LEE SEGUNDO CARACTER
00DD F56F              MOV 6FH,A        ;LO ALMACENA EN 6FH
00DF 786E              MOV R0,#6EH      ;CONVIERTE LOS DOS CARACTERES
00E1 116C              ACALL ASNU       ;ASCII EN HEXADECIMAL
00E3 22                RET


                ;*********************************************
                ;*                SUBRUTINA ENVIA              *
                ;*  Envía contenido de A por el puerto serie.  *
                ;*********************************************
00E4 F599       ENVIA: MOV SBUF,A       ;ENVIA A POR EL PUERTO SERIE
00E6 3099FD            JNB TI,$         ;CONTINUA HASTA QUE SE ENCIENDA
00E9 C299              CLR TI
00EB 22                RET
```

```
                ;*************************************************
                ;*                  SUBRUTINA DIR                *
                ;*  Valida la primera dirección de la fila. La   *
                ;*  almacena en 6AH, 6BH y en el DPTR.           *
                ;*************************************************
00EC 7872       DIR:    MOV R0,#FIL2
00EE 796A               MOV R1,#6AH
00F0 1198               ACALL VALI      ;VALIDA DIR1
00F2 856A83             MOV DPH,6AH
00F5 856B82             MOV DPL,6BH
00F8 22                 RET

                ;*************************************************
                ;*                  SUBRUTINA DIR2               *
                ;*  Valida la segunda dirección de la fila. La   *
                ;*  almacena en R6, R7 y en 6CH, 6DH. Si esta    *
                ;*  dirección es menor que la primera, envía     *
                ;*  el mensaje: "RANGO INVALIDO".                *
                ;*************************************************
00F9 08         DIR2:   INC R0          ;INICIO DIR2 EN LA FILA
00FA 1198               ACALL VALI      ;VALIDA DIR2
00FC AF6D               MOV R7,6DH      ;ALMACENA DIR2 BAJA EN R7
00FE AE6C               MOV R6,6CH      ;ALMACENA DIR2 ALTA EN R6
0100 E56A               MOV A,6AH
0102 B56C07             CJNE A,6CH,SUB1 ;SALTA SI DIR1A <> DIR2A
0105 E56B               MOV A,6BH
0107 B56D0D             CJNE A,6DH,SUB2 ;SALTA SI DIR1B <> DIR2B
010A 800D               SJMP SUB3
010C 400B       SUB1:   JC SUB3         ;SALTA SI DIR1A < DIR2A
010E 901CE0     SUB4:   MOV DPTR,#ERR4  ;"RANGO INVALIDO"
0111 112E               ACALL CRLF
0113 11A3               ACALL MENS
0115 415E               AJMP INI
0117 50F5       SUB2:   JNC SUB4        ;SALTA SI DIR1B > DIR2B
0119 22         SUB3:   RET

                ;*************************************************
                ;*       SUBRUTINA DESPLIEGA REGISTROS           *
                ;*  Despliega los registros del usuario y su     *
                ;*  contenido. En 6AH, 6BH almacena la dirección *
                ;*  del nombre de registro que va a desplegar y  *
                ;*  en 68H, 69H la dirección de su contenido. La *
                ;*  localidad 6EH contiene el número máximo de   *
                ;*  registros a desplegar. En 6FH se contabilizan*
                ;*  los registros desplegados.                   *
                ;*************************************************
011A C083       REG:    PUSH DPH
011C C082               PUSH DPL
011E 901C5E             MOV DPTR,#TAB2  ;INICIO NOMBRES DE REGISTROS
0121 75683F             MOV 68H,#3FH
0124 7569EA             MOV 69H,#0EAH   ;INICIO FILA DE REGISTROS
```

```
0127 300305          JNB DESP,REG3
012A 756E07          MOV 6EH,#07H
012D 8003            SJMP REG2
012F 756E11    REG3: MOV 6EH,#11H
0132 D56E07    REG2: DJNZ 6EH,REG1
0135 D082            POP DPL
0137 D083            POP DPH
0139 112E            ACALL CRLF
013B 22              RET
013C 11A3      REG1: ACALL MENS       ;DESPLIEGA NOMBRE DEL REGISTRO
013E 85836A          MOV 6AH,DPH       ;GUARDA DIRECCION DEL SIGUIENTE
0141 85826B          MOV 6BH,DPL       ;NOMBRE DE REGISTRO
0144 856883          MOV DPH,68H
0147 856982          MOV DPL,69H       ;CARGA LA DIR. DEL REGISTRO
014A 11B6            ACALL CONT        ;DESPLIEGA CONTENIDO DEL REGISTRO
014C A3              INC DPTR
014D 858368          MOV 68H,DPH       ;GUARDA DIRECCION DEL SIGUIENTE
0150 858269          MOV 69H,DPL       ;REGISTRO
0153 856A83          MOV DPH,6AH       ;CARGA DIRECCION DEL SIGUIENTE
0156 856B82          MOV DPL,6BH       ;NOMBRE DE REGISTRO
0159 7420            MOV A,#ESP
015B 11E4            ACALL ENVIA       ;ENVIA "ESPACIO"
015D 056F            INC 6FH           ;CONTADOR DE REGISTROS DESPLEGADOS
015F E56F            MOV A,6FH
0161 B406CE          CJNE A,#06H,REG2  ;SALTA SI NO ES FIN DE RENGLON
0164 112E            ACALL CRLF
0166 756F00          MOV 6FH,#00H      ;BORRA CONTADOR
0169 80C7            SJMP REG2


                ;*****************************************
                ;*                                       *
                ;*           SUBRUTINA RESPALDA          *
                ;*  Almacena el contenido actual de los  *
                ;*  registros en la fila de registros del*
                ;*  usuario.                             *
                ;*                                       *
                ;*****************************************
016B 858368     RESP: MOV 68H,DPH
016E 858269           MOV 69H,DPL
0171 903FEA           MOV DPTR,#FILREG ;INICIO FILA DE REG. DEL USUARIO
0174 F0               MOVX @DPTR,A
0175 A3               INC DPTR
0176 E5F0             MOV A,B
0178 F0               MOVX @DPTR,A
0179 A3               INC DPTR
017A E5D0             MOV A,PSW
017C F0               MOVX @DPTR,A
017D A3               INC DPTR
017E E568             MOV A,68H
0180 F0               MOVX @DPTR,A
0181 A3               INC DPTR
0182 E569             MOV A,69H
0184 F0               MOVX @DPTR,A
0185 A3               INC DPTR
```

```
0186 E581              MOV A,SP
0188 F0                MOVX @DPTR,A
0189 A3                INC DPTR
018A E5A8              MOV A,IE
018C F0                MOVX @DPTR,A
018D A3                INC DPTR
018E E5B8              MOV A,IP
0190 F0                MOVX @DPTR,A
0191 A3                INC DPTR
0192 E589              MOV A,TMOD
0194 F0                MOVX @DPTR,A
0195 A3                INC DPTR
0196 E588              MOV A,TCON
0198 F0                MOVX @DPTR,A
0199 A3                INC DPTR
019A E598              MOV A,SCON
019C F0                MOVX @DPTR,A
019D A3                INC DPTR
019E E587              MOV A,PCON
01A0 F0                MOVX @DPTR,A
01A1 A3                INC DPTR
01A2 E58C              MOV A,TH0
01A4 F0                MOVX @DPTR,A
01A5 A3                INC DPTR
01A6 E58A              MOV A,TL0
01A8 F0                MOVX @DPTR,A
01A9 A3                INC DPTR
01AA E590              MOV A,P1
01AC F0                MOVX @DPTR,A
01AD A3                INC DPTR
01AE E599              MOV A,SBUF
01B0 F0                MOVX @DPTR,A
01B1 856883            MOV DPH,68H
01B4 856982            MOV DPL,69H
01B7 22                RET

                ;*************************************************
                ;*                SUBRUTINA RECUPERA             *
                ;*  Almacena el contenido actual de la fila de   *
                ;*  registros del usuario en los registros del   *
                ;*  programa del usuario.                        *
                ;*************************************************
01B8 903FEB     RECU:  MOV DPTR,#FILREG+1H    ;FILA DE REG. DEL USUARIO
01BB E0                MOVX A,@DPTR
01BC F5F0              MOV B,A
01BE A3                INC DPTR
01BF E0                MOVX A,@DPTR
01C0 F5D0              MOV PSW,A
01C2 A3                INC DPTR
01C3 E0                MOVX A,@DPTR
01C4 F568              MOV 68H,A
01C6 A3                INC DPTR
```

```
01C7 E0          MOVX A,@DPTR
01C8 F569        MOV 69H,A
01CA A3          INC DPTR
01CB A3          INC DPTR
01CC E0          MOVX A,@DPTR
01CD F5A8        MOV IE,A
01CF A3          INC DPTR
01D0 E0          MOVX A,@DPTR
01D1 F5B8        MOV IP,A
01D3 A3          INC DPTR
01D4 E0          MOVX A,@DPTR
01D5 F589        MOV TMOD,A
01D7 A3          INC DPTR
01D8 E0          MOVX A,@DPTR
01D9 F588        MOV TCON,A
01DB A3          INC DPTR
01DC E0          MOVX A,@DPTR
01DD F598        MOV SCON,A
01DF A3          INC DPTR
01E0 E0          MOVX A,@DPTR
01E1 F587        MOV PCON,A
01E3 A3          INC DPTR
01E4 E0          MOVX A,@DPTR
01E5 F58C        MOV TH0,A
01E7 A3          INC DPTR
01E8 E0          MOVX A,@DPTR
01E9 F58A        MOV TL0,A
01EB A3          INC DPTR
01EC E0          MOVX A,@DPTR
01ED F590        MOV P1,A
01EF A3          INC DPTR
01F0 E0          MOVX A,@DPTR
01F1 F599        MOV SBUF,A
01F3 903FEA      MOV DPTR,#FILREG
01F6 E0          MOVX A,@DPTR
01F7 856883      MOV DPH,68H
01FA 856982      MOV DPL,69H
01FD 22          RET

                 ;*****************************************************
                 ;*                SUBRUTINA CORRIMIENTO              *
                 ;*  A contiene el dato leído en ASCII. En R4 se      *
                 ;*  encuentra el contenido original almacenado       *
                 ;*  en la subrutina CONT correspondiente a la        *
                 ;*  localidad que se desea modificar. Convierte      *
                 ;*  A en hexadecimal. En R4, recorre el cuarteto     *
                 ;*  bajo al alto e intercambia los cuartetos         *
                 ;*  bajos de R4 y A.                                 *
                 ;*****************************************************
01FE C06A        CORRI:  PUSH 6AH          ;SALVA CONTENIDO DE 6AH
0200 1146                ACALL ASHE
0202 C0E0                PUSH ACC          ;SALVA EL DATO ACTUAL
```

```
0204 EC              MOV A,R4          ;CARGA DATO ORIGINAL
0205 23              RL A              ;CORRIMIENTO DEL CUARTETO
0206 23              RL A              ;BAJO AL CUARTETO ALTO
0207 23              RL A
0208 23              RL A
0209 796A            MOV R1,#6AH
020B D06A            POP 6AH           ;RECUPERA EL DATO ACTUAL
020D D7              XCHD A,@R1
020E FC              MOV R4,A          ;CARGA DATO ACTUALIZADO
020F D06A            POP 6AH
0211 22              RET


                ;*********************************************
                ; *** INICIALIZACION DE REGISTROS DE CONTROL ***
                ;*********************************************
0212 75B810     INREG:  MOV IP,#10H       ;PRIORIDAD ALTA PARA EL PTO. SERIE
0215 758920             MOV TMOD,#20H     ;TEMPORIZADOR 1: CONTADOR DE
                                          ;OCHO BITES CON AUTORECARGA
0218 758840             MOV TCON,#40H     ;ACTIVA EL TEMPORIZADOR 1
021B 759870             MOV SCON,#70H     ;MODO 1: OCHO BITES UART CON
                                          ;VELOCIDAD VARIABLE
021E 758700             MOV PCON,#00H     ;SMOD=0 PARA VELOCIDAD VARIABLE
0221 75B0FF             MOV P3,#0FFH      ;INTERRUPCIONES CON NIVEL BAJO,
                                          ;TxD Y RxD EN UART
0224 758DFD             MOV TH1,#0FDH     ;PARA VELOCIDAD DE 9600 BAUDS
0227 758B00             MOV TL1,#00H
022A 75D000             MOV PSW,#00H
022D 316B               ACALL RESP        ;INICIALIZA FILA REG DEL USUARIO
022F 75D018             MOV PSW,#18H      ;UTILIZA EL BANCO 3
0232 7402               MOV A,#02H        ;INICIALIZA VECTORES DE INTERRUP.
0234 903F90             MOV DPTR,#VINTO
0237 F0                 MOVX @DPTR,A
0238 A3                 INC DPTR
0239 7407               MOV A,#07H
023B F0                 MOVX @DPTR,A
023C A3                 INC DPTR
023D 7475               MOV A,#75H
023F F0                 MOVX @DPTR,A      ;RUTINA DE INT. PARA INTO EN 0775H
0240 A3                 INC DPTR
0241 7402               MOV A,#02H
0243 F0                 MOVX @DPTR,A
0244 A3                 INC DPTR
0245 7400               MOV A,#00H
0247 F0                 MOVX @DPTR,A
0248 A3                 INC DPTR
0249 F0                 MOVX @DPTR,A      ;RUTINA DE INT. PARA TO EN 0000H
024A A3                 INC DPTR
024B 7402               MOV A,#02H
024D F0                 MOVX @DPTR,A
024E A3                 INC DPTR
024F 7400               MOV A,#00H
0251 F0                 MOVX @DPTR,A
```

```
0252 A3                 INC DPTR
0253 F0                 MOVX @DPTR,A     ;RUTINA DE INT. PARA INT1 EN 0000H
0254 A3                 INC DPTR
0255 7402               MOV A,#02H
0257 F0                 MOVX @DPTR,A
0258 A3                 INC DPTR
0259 7400               MOV A,#00H
025B F0                 MOVX @DPTR,A
025C A3                 INC DPTR
025D F0                 MOVX @DPTR,A     ;RUTINA DE INT. PARA T1 EN 0000H
025E 758107     INI:    MOV SP,#07H      ;INICIALIZA APUNTADOR DE PILA
0261 7870               MOV R0,#70H
0263 7F11               MOV R7,#11H
0265 DF14       BORR:   DJNZ R7,LIMP
0267 D201               SETB ECO         ;ENCIENDE BANDERA DE ECO
0269 C200               CLR VERIF        ;APAGA BANDERA DE VERIFICA
026B C203               CLR DESP         ;APAGA BANDERA DESP EN TRAZADO
026D 7900               MOV R1,#00H
026F 7A00               MOV R2,#00H
0271 7B00               MOV R3,#00H
0273 7C00               MOV R4,#00H
0275 7D00               MOV R5,#00H
0277 7E00               MOV R6,#00H
0279 8005               SJMP PARAM
027B 7600       LIMP:   MOV @R0,#00H     ;CICLO DE BORRADO DE FILA DE
027D 08                 INC R0           ;PARAMETROS
027E 80E5               SJMP BORR

                ;*********************************************
                ;*              LECTURA DE PARAMETROS        *
                ;*  Lee caracteres y los almacena en la fila *
                ;*  de parámetros hasta encontrar un CR.     *
                ;*  Incrementa un contador (R2) por cada     *
                ;*  caracter leído, excepto CR y BS.         *
                ;*********************************************
0280 112E       PARAM:  ACALL CRLF
0282 743E               MOV A,#PROMPT
0284 11E4               ACALL ENVIA      ;ENVIA ">"
0286 7870               MOV R0,#FILA     ;INICIO FILA DE PARAMETROS
0288 11CE       UNO:    ACALL LEE
028A B40804             CJNE A,#BS,DOS   ;SALTA SI A <> BS
028D 18                 DEC R0
028E 1A                 DEC R2
028F 4188               AJMP UNO
0291 B40D02     DOS:    CJNE A,#CR,TRES ;SALTA SI A <> CR
0294 419B               AJMP SEL
0296 F6         TRES:   MOV @R0,A        ;ALMACENA DATO LEIDO EN FILA
0297 08                 INC R0
0298 0A                 INC R2           ;INCREMENTA CONT. DE CARACTERES
0299 4188               AJMP UNO
```

```
                ;*************************************************
                ;*              SELECCION DE COMANDOS            *
                ;*  Busca el comando leído en la Tabla 1, checa  *
                ;*  su longitud máxima y salta a la rutina de    *
                ;*  dicho comando. Si el primer caracter leído   *
                ;*  no es un comando, envía el mensaje: "COMANDO *
                ;*  INVALIDO". Si los caracteres leídos exceden  *
                ;*  la longitud máxima del comando, despliega el *
                ;*  mensaje: "ERROR DE SINTAXIS". R4 contiene el *
                ;*  offset para el salto a la rutina del comando *
                ;*  correspondiente. R3 es el contador del ciclo *
                ;*  de búsqueda de comando.                      *
                ;*************************************************
0300 =          COMAN   EQU 300H
029B 901C40     SEL:    MOV DPTR,#TAB1      ;INICIO TABLA DE COMANDOS
029E 7C00               MOV R4,#00H
02A0 7B0F               MOV R3,#0FH
02A2 DB05       CIC:    DJNZ R3,COM
02A4 901CBF             MOV DPTR,#ERR2      ;"COMANDO INVALIDO"
02A7 801B               SJMP SIETE
02A9 0C         COM:    INC R4              ;CICLO DE BUSQUEDA DE COMANDO
02AA 0C                 INC R4
02AB E4                 CLR A
02AC 93                 MOVC A,@A+DPTR      ;CARGA COMANDO DE TABLA
02AD B5700B             CJNE A,70H,CUATRO ;SALTA SI COMANDOS <>
02B0 A3                 INC DPTR
02B1 E4                 CLR A
02B2 93                 MOVC A,@A+DPTR      ;CARGA LONGITUD DE COMANDO
02B3 B51A09             CJNE A,1AH,CINCO    ;SALTA SI LONGITUDES <>
02B6 EC         SEIS:   MOV A,R4
02B7 900300             MOV DPTR,#COMAN
02BA 73                 JMP @A+DPTR         ;SALTA AL VECTOR DEL COMANDO
02BB A3         CUATRO: INC DPTR
02BC A3                 INC DPTR
02BD 41A2               AJMP CIC
02BF 50F5       CINCO:  JNC SEIS            ;SALTA SI A > R2
02C1 901CAD     OCHO:   MOV DPTR,#ERR1      ;"ERROR DE SINTAXIS"
02C4 112E       SIETE:  ACALL CRLF
02C6 11A3               ACALL MENS
02C8 415E               AJMP INI
0300                    ORG COMAN
                ; VECTORES DE COMANDOS
0300 00                 NOP
0301 00                 NOP
0302 611E               AJMP E
0304 617B               AJMP D
0306 A119               AJMP C
0308 A182               AJMP T
030A A1C5               AJMP R
030C 61C0               AJMP A
030E C12D               AJMP M
0310 C1E4               AJMP O
0312 E11F               AJMP BE
```

```
0314 61CD              AJMP L
0316 8177              AJMP I
0318 61CB              AJMP V
031A E159              AJMP F
031C E15E              AJMP G

                ;*****************************************************
                ;*                    RUTINA E                       *
                ;*  Despliega y permite modificar el contenido de    *
                ;*  la localidad apuntada por el DPTR. Con ESPACIO   *
                ;*  muestra la siguiente localidad y con CR termina. *
                ;*  R4 contiene el valor original de la localidad    *
                ;*  examinada cuando utiliza la subrutina LOC y      *
                ;*  contiene el valor actual después de realizar la  *
                ;*  subrutina CORRI.  En 6FH se contabilizan las     *
                ;*  localidades desplegadas.  R5 permite un máximo   *
                ;*  de dos caracteres en la ventana.                 *
                ;*  Si DIR1 pertenece a una localidad de ROM, envía  *
                ;*  el mensaje: "ERROR: DIRECCION DE ROM.            *
                ;*****************************************************
031E BA06A0     E:      CJNE R2,#06H,OCHO  ;SALTA SI NO HAY DIR1
0321 11EC               ACALL DIR
0323 E56A               MOV A,6AH
0325 B41F09             CJNE A,#1FH,E7     ;SALTA SI DIR1A <> 1FH
0328 901D17     E8:     MOV DPTR,#ERR7     ;"ERROR: DIRECCION DE ROM"
032B 112E               ACALL CRLF
032D 11A3               ACALL MENS
032F 415E               AJMP INI
0331 40F5       E7:     JC E8
0333 112E               ACALL CRLF
0335 C26F               CLR 6FH
0337 C201               CLR ECO            ;APAGA BANDERA DE ECO
0339 056F       E2:     INC 6FH            ;CONTADOR DE LOCS. DESPLEGADAS
033B E56F               MOV A,6FH
033D B40705             CJNE A,#07,E6      ;SALTA SI NO ES FIN DE RENGLON
0340 756F00             MOV 6FH,#00H
0343 112E               ACALL CRLF
0345 11BF       E6:     ACALL LOC
0347 11CE       E5:     ACALL LEE
0349 B42009             CJNE A,#ESP,E1     ;SALTA SI A <> ESP
034C 11E4               ACALL ENVIA        ;ECO
034E 7D00               MOV R5,#00H        ;BORRA CONTADOR DE VENTANA
0350 EC                 MOV A,R4           ;CARGA CONTENIDO ACTUAL
0351 F0                 MOVX @DPTR,A       ;REGRESA VALOR ACTUALIZADO
0352 A3                 INC DPTR
0353 80E4               SJMP E2
0355 B40D08     E1:     CJNE A,#CR,E3      ;SALTA SI A <> CR
0358 11E4               ACALL ENVIA        ;ECO
035A 7D00               MOV R5,#00H
035C EC                 MOV A,R4
035D F0                 MOVX @DPTR,A       ;CARGA DATO ACTUALIZADO EN FILA
035E 415E               AJMP INI
0360 0D         E3:     INC R5
```

```
0361 BD0311            CJNE R5,#03H,E4    ;SALTA SI CONTADOR <> 03H
0364 1D                DEC R5
0365 C0E0              PUSH ACC
0367 7408              MOV A,#BS
0369 11E4              ACALL ENVIA        ;ENVIA DOS VECES
036B 11E4              ACALL ENVIA        ;"RETROCESO"
036D D0E0              POP ACC
036F 31FE              ACALL CORRI
0371 1180              ACALL NUAS
0373 80D2              SJMP E5
0375 11E4      E4:     ACALL ENVIA        ;ECO
0377 31FE              ACALL CORRI
0379 80CC              SJMP E5

               ;*****************************************
               ;*                RUTINA D                 *
               ;*  Despliega el contenido de las localidades *
               ;*  desde DIR1 hasta DIR2. Si no existe DIR2, *
               ;*  se muestran cuatro renglones completos.   *
               ;*  R2 contiene el número de caracteres de la *
               ;*  fila de parámetros.                       *
               ;*****************************************
037B BA0607    D:      CJNE R2,#06H,D6 ;SALTA SI HAY DIR2
037E 11EC              ACALL DIR
0380 5382F0            ANL DPL,#0F0H   ;INICIO DE RENGLON A DESPLEGAR
0383 801D              SJMP D2
0385 BA0B36    D6:     CJNE R2,#0BH,D7 ;SALTA SI NO HAY DIR2
0388 11EC              ACALL DIR
038A 5382F0            ANL DPL,#0F0H
038D 11F9              ACALL DIR2
038F 112E      D1:     ACALL CRLF
0391 11BF              ACALL LOC
0393 A3        D5:     INC DPTR
0394 EE                MOV A,R6        ;CARGA DIR2 ALTA
0395 B5831A            CJNE A,DPH,D4   ;SALTA SI DIR1A <> DIR2A
0398 EF                MOV A,R7        ;CARGA DIR2 BAJA
0399 B58214            CJNE A,DPL,D8   ;SALTA SI DIR1B <> DIR2B
039C 11B6              ACALL CONT
039E 112E              ACALL CRLF
03A0 415E      D9:     AJMP INI
03A2 AE83      D2:     MOV R6,DPH
03A4 743F              MOV A,#3FH      ;PARA DESPLEGAR CUATRO RENGLONES
03A6 2582              ADD A,DPL
03A8 FF                MOV R7,A
03A9 4002              JC D3           ;SALTA SI HAY ACARRERO EN SUMA
03AB 80E2              SJMP D1
03AD 0E        D3:     INC R6
03AE 80DF              SJMP D1
03B0 40EE      D8:     JC D9
03B2 11B6      D4:     ACALL CONT
03B4 E582              MOV A,DPL
03B6 540F              ANL A,#0FH      ;DESPLIEGA 16 CONT. POR RENGLON
03B8 B40FD8            CJNE A,#0FH,D5  ;SALTA SI NO ES FIN DE RENGLON
```

```
03BB A3                 INC DPTR
03BC 80D1               SJMP D1
03BE 41C1       D7:     AJMP OCHO

                ;*************************************************
                ;*                    RUTINA A                   *
                ;*  Despliega la pantalla de ayuda donde se      *
                ;*  muestra la sintaxis y función que realiza    *
                ;*  cada comando.                                *
                ;*************************************************
03C0 901D2F     A:      MOV DPTR,#TAB4  ;INICIO PANTALLA DE AYUDA
03C3 11A3               ACALL MENS
03C5 11A3               ACALL MENS
03C7 11A3               ACALL MENS
03C9 415E               AJMP INI

                ;*************************************************
                ;*                    RUTINA V                   *
                ;*  Compara un programa de memoria contra uno    *
                ;*  que se lee. Enciende una bandera y realiza   *
                ;*  la rutina L. Si los datos comparados no      *
                ;*  coinciden, se envía el mensaje: "DIFERENCIA  *
                ;*  EN PROGRAMA".                                *
                ;*************************************************
03CB D200       V:      SETB VERIF      ;ENCIENDE BANDERA DE VERIFICA

                ;*************************************************
                ;*                    RUTINA L                   *
                ;*  Lee un programa por bloques y lo carga en    *
                ;*  la dirección de memoria del bloque mas el    *
                ;*  desplazamiento indicado. Cuando la bandera   *
                ;*  de verifica está encendida, sólo lo compara. *
                ;*  Si la paridad del bloque leído es diferente  *
                ;*  a la calculada, envía el mensaje: "ERROR DE  *
                ;*  LECTURA".  Si  DIR + NUM  no apunta a una    *
                ;*  localidad de RAM EXTERNA, envía el mensaje:  *
                ;*  "ERROR: DIRECCION DE ROM". La localidad 68H  *
                ;*  se utiliza como contador de datos, la 69H    *
                ;*  para el cálculo de paridad, 6EH almacena el  *
                ;*  dato a verificar; en 6AH, 6BH se almacena    *
                ;*  el número hexadecimal, en 6CH DIR alta y en  *
                ;*  6DH DIR baja. R5 se utiliza en los ciclos    *
                ;*  con el valor del contador de datos.          *
                ;*************************************************
03CD BA0646     L:      CJNE R2,#06H,L14  ;SALTA SI NO HAY NUM
03D0 C201               CLR ECO         ;APAGA BANDERA DE ECO
03D2 7872               MOV R0,#FIL2    ;INICIO DE NUM EN LA FILA
03D4 796A               MOV R1,#6AH
03D6 1198               ACALL VALI      ;ALMACENA NUMERO HEXADECIMAL
03D8 756900     L6:     MOV 69H,#00H    ;CALCULO DE PARIDAD EN 69H
03DB 11CE               ACALL LEE       ;LEE INICIO DE BLOQUE
03DD B43A51             CJNE A,#3AH,L1  ;SALTA SI A <> ":"
03E0 757F00             MOV 7FH,#00H
```

```
03E3 11D7              ACALL LEER2
03E5 F568              MOV 68H,A        ;ALMACENA CONTADOR DE DATOS
03E7 2569              ADD A,69H        ;CALCULA PARIDAD
03E9 F569              MOV 69H,A
03EB 11D7              ACALL LEER2
03ED F56C              MOV 6CH,A        ;ALMACENA DIRECCION ALTA
03EF 2569              ADD A,69H        ;CALCULA PARIDAD
03F1 F569              MOV 69H,A
03F3 11D7              ACALL LEER2
03F5 F56D              MOV 6DH,A        ;ALMACENA DIRECCION BAJA
03F7 2569              ADD A,69H        ;CALCULA PARIDAD
03F9 F569              MOV 69H,A
03FB 11D7              ACALL LEER2
03FD B4003A            CJNE A,#00H,L2   ;SALTA SI A <> 00H
0400 0568              INC 68H
0402 E56D              MOV A,6DH
0404 256B              ADD A,6BH
0406 F56D              MOV 6DH,A
0408 E56C              MOV A,6CH        ;SUMA NUMERO HEXADECIMAL MAS
040A 356A              ADDC A,6AH       ;DIRECCION Y GUARDA SUMA
040C F56C              MOV 6CH,A        ;EN 6CH, 6DH
040E B41F07            CJNE A,#1FH,L12 ;SALTA SI DIRA <> 1FH
0411 901D17    L13:    MOV DPTR,#ERR7   ;"ERROR: DIRECCION DE ROM"
0414 804D              SJMP L11
0416 41C1      L14:    AJMP OCHO
0418 40F7      L12:    JC L13           ;SALTA SI DIRA < 1FH
041A 903FA0            MOV DPTR,#BUFLE ;INICIO BUFFER DE LECT. Y ESC.
041D AD68              MOV R5,68H
041F DD1E      L7:     DJNZ R5,L3
0421 11D7              ACALL LEER2
0423 14                DEC A
0424 F4                CPL A
0425 B56912            CJNE A,69H,L2    ;SALTA SI HAY DIF. EN PARIDAD
0428 903FA0            MOV DPTR,#BUFLE
042B AD68              MOV R5,68H
042D DD1A      L4:     DJNZ R5,L5
042F 80A7              SJMP L6
0431 057F      L1:     INC 7FH
0433 E57F              MOV A,7FH
0435 B404A0            CJNE A,#04H,L6
0438 415E              AJMP INI
043A 901D06    L2:     MOV DPTR,#ERR6   ;"ERROR DE LECTURA"
043D 8024              SJMP L11
043F 11D7      L3:     ACALL LEER2      ;CICLO DE LECTURA DE DATOS
0441 F0                MOVX @DPTR,A
0442 2569              ADD A,69H        ;CALCULA PARIDAD
0444 F569              MOV 69H,A
0446 A3                INC DPTR
0447 80D6              SJMP L7
0449 E0        L5:     MOVX A,@DPTR     ;CICLO PARA ALMACENAR O VERIFICAR
044A A3                INC DPTR         ;PROGRAMA EN RAM
044B AE83              MOV R6,DPH       ;ALMACENA DIRECCION DEL SIG. DATO
044D AF82              MOV R7,DPL       ;DEL BUFFER
```

```
044F 856C83             MOV DPH,6CH
0452 856D82             MOV DPL,6DH      ;CARGA DIRECCION DEL PROGRAMA
0455 300011             JNB VERIF,L8     ;SALTA SI BANDERA VERIFICA APAGADA
0458 F56E               MOV 6EH,A        ;CARGA DATO DEL PROGRAMA
045A E0                 MOVX A,@DPTR     ;PARA VERIFICARLO
045B B56E02             CJNE A,6EH,L9    ;SALTA SI DATOS SON DIFERENTES
045E 800A               SJMP L10
0460 901CEF     L9:     MOV DPTR,#ERR5   ;"DIFERENCIA EN PROGRAMA"
0463 112E       L11:    ACALL CRLF
0465 11A3               ACALL MENS
0467 415E               AJMP INI
0469 F0         L8:     MOVX @DPTR,A     ;ALMACENA DATO EN RAM
046A A3         L10:    INC DPTR
046B 85836C             MOV 6CH,DPH      ;ALMACENA DIRECCION
046E 85826D             MOV 6DH,DPL      ;SIGUIENTE DE RAM
0471 8E83               MOV DPH,R6
0473 8F82               MOV DPL,R7
0475 80B6               SJMP L4

                ;*****************************************
                ;*                RUTINA I               *
                ;*  Envía un programa residente en RAM en *
                ;*  bloques a través del puerto serie. R2 se *
                ;*  utiliza como contador de datos, en 69H *
                ;*  se calcula la paridad del bloque, en 6EH *
                ;*  y 6FH se almacena la dirección del buffer *
                ;*  de LECTURA/ESCRITURA.                *
                ;*****************************************
0477 BA0B75     I:      CJNE R2,#0BH,I9 ;SALTA SI NO HAY DIR2
047A 11EC               ACALL DIR
047C 11F9               ACALL DIR2
047E 903FA0     I3:     MOV DPTR,#BUFLE ;INICIO BUFFER DE LECT. Y ESC.
0481 7A00               MOV R2,#00H      ;BORRA CONTADOR DE DATOS
0483 7569 00            MOV 69H,#00H     ;CALCULO DE PARIDAD EN 69H
0486 E56A               MOV A,6AH        ;CARGA DIR. ALTA
0488 F0                 MOVX @DPTR,A     ;ALMACENA DIR. ALTA EN BUFFER
0489 A3                 INC DPTR
048A 2569               ADD A,69H        ;CALCULA PARIDAD
048C F569               MOV 69H,A
048E E56B               MOV A,6BH        ;CARGA DIR. BAJA
0490 F0                 MOVX @DPTR,A     ;ALMACENA DIR. BAJA EN BUFFER
0491 A3                 INC DPTR
0492 2569               ADD A,69H        ;CALCULA PARIDAD
0494 F569               MOV 69H,A
0496 E4                 CLR A
0497 F0                 MOVX @DPTR,A     ;ALMACENA 00H EN BUFFER
0498 A3         I7:     INC DPTR
0499 85836E             MOV 6EH,DPH
049C 85826F             MOV 6FH,DPL      ;GUARDA DIR. SIGUIENTE DEL BUFFER
049F 856A83             MOV DPH,6AH
04A2 856B82             MOV DPL,6BH      ;CARGA DIRECCION DE PROGRAMA
04A5 E0                 MOVX A,@DPTR
04A6 856E83             MOV DPH,6EH
```

```
04A9 856F82          MOV DPL,6FH      ;CARGA DIRECCION DEL BUFFER
04AC F0              MOVX @DPTR,A
04AD 2569            ADD A,69H        ;CALCULA PARIDAD
04AF F569            MOV 69H,A
04B1 0A              INC R2
04B2 856A83          MOV DPH,6AH
04B5 856B82          MOV DPL,6BH      ;CARGA DIRECCION DE PROGRAMA
04B8 E582            MOV A,DPL
04BA 540F            ANL A,#0FH
04BC B40F38          CJNE A,#0FH,I1   ;SALTA SI NO ES FIN DE BLOQUE
04BF EA        I6:   MOV A,R2
04C0 2569            ADD A,69H        ;CALCULA PARIDAD
04C2 F4              CPL A
04C3 04              INC A            ;COMPLEMENTO A DOS
04C4 F569            MOV 69H,A
04C6 112E            ACALL CRLF       ;ENVIA "CR", "LF"
04C8 E4              CLR A
04C9 11E4            ACALL ENVIA      ;ENVIA "NULL"
04CB 743A            MOV A,#3AH
04CD 11E4            ACALL ENVIA      ;ENVIA ":"
04CF EA              MOV A,R2
04D0 1180            ACALL NUAS       ;ENVIA CONTADOR DE DATOS
04D2 0A              INC R2
04D3 0A              INC R2
04D4 0A              INC R2
04D5 0A              INC R2
04D6 903FA0          MOV DPTR,#BUFLE
04D9 DA16      I4:   DJNZ R2,I2
04DB E569            MOV A,69H
04DD 1180            ACALL NUAS       ;ENVIA PARIDAD
04DF 856A83          MOV DPH,6AH
04E2 856B82          MOV DPL,6BH      ;CARGA DIR. DE PROGRAMA
04E5 EE              MOV A,R6
04E6 B58327          CJNE A,DPH,I8    ;SALTA SI DIR1A <> DIR2A
04E9 EF              MOV A,R7
04EA B58223          CJNE A,DPL,I8    ;SALTA SI DIR1B <> DIR2B
04ED 415E            AJMP INI
04EF 41C1      I9:   AJMP OCHO
04F1 E0        I2:   MOVX A,@DPTR     ;CICLO DE ENVIO DE DATOS
04F2 1180            ACALL NUAS
04F4 A3              INC DPTR
04F5 80E2            SJMP I4
04F7 EE        I1:   MOV A,R6
04F8 B58306          CJNE A,DPH,I5    ;SALTA SI DIR1A <> DIR2A
04FB EF              MOV A,R7
04FC B58202          CJNE A,DPL,I5    ;SALTA SI DIR1B <> DIR2B
04FF 80BE            SJMP I6
0501 A3        I5:   INC DPTR
0502 85836A          MOV 6AH,DPH
0505 85826B          MOV 6BH,DPL      ;GUARDA DIR. SIGUIENTE DE PROGRAMA
0508 856E83          MOV DPH,6EH
050B 856F82          MOV DPL,6FH      ;CARGA DIR. SIGUIENTE DEL BUFFER
050E 8088            SJMP I7
```

```
0510 A3          I8:    INC DPTR
0511 85836A             MOV 6AH,DPH
0514 85826B             MOV 6BH,DPL      ;CARGA DIR. SIGUIENTE DE PROGRAMA
0517 817E               AJMP I3

                ;*************************************************
                ;*                    RUTINA C                   *
                ;*  Corre un programa residente en RAM como si    *
                ;*  fuera una subrutina.  En R2 se almacena el    *
                ;*  contenido de DIR2 temporalmente y se          *
                ;*  sustituye por un retorno de subrutina (22H).  *
                ;*  Si DIR1 pertenece a una localidad de ROM,     *
                ;*  envía el mensaje: "ERROR: DIRECCION DE ROM".  *
                ;*************************************************
0519 BA0616     C:     CJNE R2,#06H,C1   ;SALTA SI HAY DIR2
051C C204               CLR CORRE         ;APAGA BANDERA DE CORRE
051E 11EC               ACALL DIR
0520 E56A               MOV A,6AH
0522 B41F09             CJNE A,#1FH,C2    ;SALTA SI DIR1A <> 1FH
0525 901D17     C3:     MOV DPTR,#ERR7    ;"ERROR: DIRECCION DE ROM"
0528 112E               ACALL CRLF
052A 11A3               ACALL MENS
052C 415E               AJMP INI
052E 40F5       C2:     JC C3             ;SALTA SI DIR1A < 1FH
0530 801B               SJMP C6
0532 BA0B4B     C1:     CJNE R2,#0BH,C4   ;SALTA SI NO HAY DIR2
0535 D204               SETB CORRE        ;ENCIENDE BANDERA DE CORRE
0537 11EC               ACALL DIR
0539 E56A               MOV A,6AH
053B B41F02             CJNE A,#1FH,C5    ;SALTA SI DIR1A <> 1FH
053E A125               AJMP C3
0540 40E3       C5:     JC C3             ;SALTA SI DIR1A < 1FH
0542 11F9               ACALL DIR2
0544 8E83               MOV DPH,R6
0546 8F82               MOV DPL,R7        ;CARGA DIR2 EN EL DPTR
0548 E0                 MOVX A,@DPTR
0549 FA                 MOV R2,A          ;ALMACENA CONTENIDO DE DIR2 EN R2
054A 7422               MOV A,#22H
054C F0                 MOVX @DPTR,A      ;LO INTERCAMBIA POR UN RET
054D 903FE0     C6:     MOV DPTR,#PROGCT  ;INICIO PROGRAMA DE CORRIDA
0550 7412               MOV A,#12H
0552 F0                 MOVX @DPTR,A
0553 A3                 INC DPTR
0554 E56A               MOV A,6AH
0556 F0                 MOVX @DPTR,A
0557 A3                 INC DPTR
0558 E56B               MOV A,6BH
055A F0                 MOVX @DPTR,A      ;CARGA "LCALL DIR1"
055B A3                 INC DPTR
055C 7402               MOV A,#02H
055E F0                 MOVX @DPTR,A
055F A3                 INC DPTR
0560 7405               MOV A,#05H
```

```
0562 F0                 MOVX @DPTR,A
0563 A3                 INC DPTR
0564 7470               MOV A,#70H
0566 F0                 MOVX @DPTR,A        ;CARGA "LJMP 570H"
0567 75D000             MOV PSW,#00H        ;BANCO 0 PARA EL USUARIO
056A 31B8               ACALL RECU
056C 023FE0             LJMP PROGCT
0570                    ORG 570H
0570 316B               ACALL RESP
0572 75D018             MOV PSW,#18H        ;BANCO 3
0575 300406             JNB CORRE,C7        SALTA SI BANDERA APAGADA
0578 8E83               MOV DPH,R6
057A 8F82               MOV DPL,R7          ;CARGA DIR2
057C EA                 MOV A,R2            ;REGRESA EL CONTENIDO ORIGINAL
057D F0                 MOVX @DPTR,A        ;DE DIR2
057E 415E       C7:     AJMP INI
0580 41C1       C4:     AJMP OCHO

                ;*************************************************
                ;*                    RUTINA T                   *
                ;*  Traza la instrucción indicada por DIR1. Con  *
                ;*  ESPACIO, traza la siguiente instrucción y    *
                ;*  con CR termina el trazado. R2 contiene el    *
                ;*  número de caracteres de la fila.  R5 se usa  *
                ;*  como contador de instrucciones no trazadas.  *
                ;*************************************************
0582 112E       T:      ACALL CRLF
0584 D203               SETB DESP           ;ENCIENDE BANDERA
0586 BA0608             CJNE R2,#06H,TDIE   ;SALTA SI NO HAY DIR1
0589 11EC               ACALL DIR
058B 11BF               ACALL LOC
058D 112E               ACALL CRLF
058F 8002               SJMP TTRE
0591 41C1       TDIE:   AJMP OCHO
0593 903F91     TTRE:   MOV DPTR,#VINTO+01H
0596 7407               MOV A,#07H
0598 F0                 MOVX @DPTR,A
0599 A3                 INC DPTR
059A 7475               MOV A,#75H
059C F0                 MOVX @DPTR,A        ;VECTOR DE INTERRUPCION 775H
059D 903FE0             MOV DPTR,#PROGCT    ;INICIO PROGRAMA DE TRAZADO
05A0 7412               MOV A,#12H
05A2 F0                 MOVX @DPTR,A
05A3 A3                 INC DPTR
05A4 E56A               MOV A,6AH
05A6 F0                 MOVX @DPTR,A
05A7 A3                 INC DPTR
05A8 E56B               MOV A,6BH
05AA F0                 MOVX @DPTR,A        ;CARGA "LCALL DIR1"
05AB A3                 INC DPTR
05AC 7402               MOV A,#02H
05AE F0                 MOVX @DPTR,A
05AF A3                 INC DPTR
```

```
05B0 7407          MOV A,#07H
05B2 F0            MOVX @DPTR,A
05B3 A3            INC DPTR
05B4 74D0          MOV A,#0D0H
05B6 F0            MOVX @DPTR,A          ;CARGA "LJMP 7D0H"
05B7 7D00          MOV R5,#00H
05B9 75A881        MOV IE,#81H           ;HABILITA INT0*
05BC 90C000        MOV DPTR,#0C000H      ;ENVIA PULSO BAJO A INT0*
05BF 743F          MOV A,#3FH
05C1 F0            MOVX @DPTR,A
05C2 023FE0        LJMP PROGCT

                ;*****************************************************
                ;*                      RUTINA R                     *
                ;*  Despliega todos los registros y su contenido,    *
                ;*  los muestra de uno en uno para que puedan ser    *
                ;*  modificados.  Con ESPACIO despliega el registro  *
                ;*  siguiente o con CR termina.  R5 permite un       *
                ;*  máximo de dos caracteres en la ventana. En 6AH,  *
                ;*  6BH almacena la dirección del nombre del         *
                ;*  registro que va a desplegar y en 68H, 69H la     *
                ;*  dirección de su contenido.                       *
                ;*****************************************************
05C5 112E       R:      ACALL CRLF
05C7 C201               CLR ECO
05C9 311A               ACALL REG         ;DESPLIEGA REGISTROS DEL USUARIO
05CB 901C5E             MOV DPTR,#TAB2    ;INICIO NOMBRES DE REGISTROS
05CE 75683F             MOV 68H,#3FH      ;INICIO FILA DE REGISTROS
05D1 7569EA             MOV 69H,#0EAH
05D4 E582       RTRE:   MOV A,DPL
05D6 B4AD02             CJNE A,#0ADH,RCIN  ;SALTA SI NO ES ULTIMO REGISTRO
05D9 415E               AJMP INI
05DB 11A3       RCIN:   ACALL MENS        ;DESPLIEGA NOMBRE DEL REGISTRO
05DD 85836A             MOV 6AH,DPH       ;GUARDA DIRECCION DEL SIGUIENTE
05E0 85826B             MOV 6BH,DPL       ;NOMBRE DE REGISTRO
05E3 856883             MOV DPH,68H
05E6 856982             MOV DPL,69H       ;CARGA LA DIR. DEL REGISTRO
05E9 11B6               ACALL CONT
05EB 11CE       RCUA:   ACALL LEE
05ED B42017             CJNE A,#ESP,RUNO  ;SALTA SI A <> ESP
05F0 11E4               ACALL ENVIA       ;ECO
05F2 7D00               MOV R5,#00H
05F4 EC                 MOV A,R4
05F5 F0                 MOVX @DPTR,A      ;CARGA DATO ACTUALIZADO EN FILA
05F6 112E               ACALL CRLF
05F8 A3                 INC DPTR          ;APUNTA AL SIGUIENTE REGISTRO
05F9 858368             MOV 68H,DPH       ;GUARDA DIR. DEL SIGUIENTE REGISTRO
05FC 858269             MOV 69H,DPL
05FF 856A83             MOV DPH,6AH       ;RECUPERA DIR. DEL SIGUIENTE
0602 856B82             MOV DPL,6BH       ;NOMBRE DE REGISTRO
0605 80CD               SJMP RTRE
0607 B40D08     RUNO:   CJNE A,#CR,RDOS ;SALTA SI A <> CR
060A 11E4               ACALL ENVIA       ;ECO
```

```
060C 7D00              MOV R5,#00H
060E EC                MOV A,R4
060F F0                MOVX @DPTR,A     ;CARGA DATO ACTUALIZADO EN FILA
0610 415E              AJMP INI
0612 0D        RDOS:   INC R5           ;CONTADOR DE VENTANA
0613 BD0311            CJNE R5,#03H,RSEI
0616 1D                DEC R5
0617 C0E0              PUSH ACC
0619 7408              MOV A,#BS
061B 11E4              ACALL ENVIA      ;ENVIA DOS VECES
061D 11E4              ACALL ENVIA      ;"RETROCESO"
061F D0E0              POP ACC
0621 31FE              ACALL CORRI
0623 1180              ACALL NUAS
0625 80C4              SJMP RCUA
0627 11E4      RSEI:   ACALL ENVIA      ;ECO
0629 31FE              ACALL CORRI
062B 80BE              SJMP RCUA

               ;*****************************************
               ;*                RUTINA M               *
               ;*  Mueve un bloque de memoria definido por *
               ;*  DIR1 hasta DIR2 a partir de la localidad *
               ;*  indicada por DIR3. Si DIR3 pertenece a  *
               ;*  una localidad de ROM, envía el mensaje: *
               ;*  "ERROR: DIRECCION DE ROM".             *
               ;*****************************************
062D BA1015    M:      CJNE R2,#10H,M5 ;SALTA SI NO HAY DIR3
0630 11EC              ACALL DIR
0632 11F9              ACALL DIR2
0634 08                INC R0
0635 1198              ACALL VALI       ;ALMACENA DIR3 EN 6EH, 6FH
0637 E56E              MOV A,6EH
0639 B41F0B            CJNE A,#1FH,M3   ;SALTA SI DIR3A <> 1FH
063C 901D17    M4:     MOV DPTR,#ERR7   ;"ERROR: DIRECCION DE ROM"
063F 112E              ACALL CRLF
0641 11A3              ACALL MENS
0643 415E              AJMP INI
0645 41C1      M5:     AJMP OCHO
0647 40F3      M3:     JC M4            ;SALTA SI DIR3A < 1FH
0649 E56A              MOV A,6AH
064B B56E07            CJNE A,6EH,M8    ;SALTA SI DIR1A <> DIR3A
064E E56B              MOV A,6BH
0650 B56F02            CJNE A,6FH,M8
0653 415E              AJMP INI
0655 505B      M8:     JNC M9           ;SALTA SI DIR1 > DIR3
0657 EF                MOV A,R7
0658 C3                CLR C
0659 9582              SUBB A,DPL
065B FF                MOV R7,A         ;R7 = DIR2B - DIR1B
065C EE                MOV A,R6
065D 9583              SUBB A,DPH
065F FE                MOV R6,A         ;R6 = DIR2A - DIR1A
```

```
0660 EF             MOV A,R7
0661 256F           ADD A,6FH
0663 F56F           MOV 6FH,A       ;6FH = DIR3B + (DIR2B - DIR1B)
0665 EE             MOV A,R6
0666 356E           ADDC A,6EH
0668 F56E           MOV 6EH,A       ;6EH = DIR3A + (DIR2A - DIR1A)
066A 856C83         MOV DPH,6CH
066D 856D82         MOV DPL,6DH     ;CARGA DIR3 EN DPTR
0670 E583     M1:   MOV A,DPH
0672 B56A0F         CJNE A,6AH,M2   ;SALTA SI DIR1A <> DIR2A
0675 E582           MOV A,DPL
0677 B56B0A         CJNE A,6BH,M2   ;SALTA SI DIR1B <> DIR2B
067A E0             MOVX A,@DPTR    ;CARGA ULTIMO DATO
067B 856E83         MOV DPH,6EH
067E 856F82         MOV DPL,6FH     ;CARGA DIR3 FINAL EN DPTR
0681 F0             MOVX @DPTR,A    ;ALMACENA ULTIMO DATO
0682 415E           AJMP INI
0684 E0       M2:   MOVX A,@DPTR    ;CICLO DE TRANSFERENCIA DE DATOS
0685 1582           DEC DPL
0687 AA82           MOV R2,DPL
0689 BAFF02         CJNE R2,#0FFH,M6  ;SALTA SI NO HAY ACARREO
068C 1583           DEC DPH
068E 85836C   M6:   MOV 6CH,DPH
0691 85826D         MOV 6DH,DPL     ;GUARDA DIR2 DECREMENTADA
0694 856E83         MOV DPH,6EH
0697 856F82         MOV DPL,6FH     ;CARGA DIR3 EN DPTR
069A F0             MOVX @DPTR,A    ;ALMACENA DATO
069B 1582           DEC DPL
069D AA82           MOV R2,DPL
069F BAFF02         CJNE R2,#0FFH,M7  ;SALTA SI NO HAY ACARREO
06A2 1583           DEC DPH
06A4 85836E   M7:   MOV 6EH,DPH
06A7 85826F         MOV 6FH,DPL     ;GUARDA DIR3 DECREMENTADA
06AA 856C83         MOV DPH,6CH
06AD 856D82         MOV DPL,6DH     ;CARGA DIR2 DECREMENTADA EN DPTR
06B0 80BE           SJMP M1
06B2 E583     M9:   MOV A,DPH
06B4 B51E0F         CJNE A,1EH,M10  ;SALTA SI DIR1A <> DIR2A
06B7 E582           MOV A,DPL
06B9 B51F0A         CJNE A,1FH,M10  ;SALTA SI DIR1B <> DIR2B
06BC E0             MOVX A,@DPTR    ;CARGA ULTIMO DATO
06BD 856E83         MOV DPH,6EH
06C0 856F82         MOV DPL,6FH     ;CARGA DIR3 FINAL EN DPTR
06C3 F0             MOVX @DPTR,A    ;ALMACENA ULTIMO DATO
06C4 415E           AJMP INI
06C6 E0       M10:  MOVX A,@DPTR    ;CICLO DE TRANSFERENCIA DE DATOS
06C7 A3             INC DPTR
06C8 85836A         MOV 6AH,DPH
06CB 85826B         MOV 6BH,DPL     ;ALMACENA DIR1
06CE 856E83         MOV DPH,6EH
06D1 856F82         MOV DPL,6FH     ;CARGA DIR3
06D4 F0             MOVX @DPTR,A
06D5 A3             INC DPTR
```

```
06D6 85836E             MOV 6EH,DPH
06D9 85826F             MOV 6FH,DPL      ;ALAMACENA DIR3 INCREMENTADA
06DC 856A83             MOV DPH,6AH
06DF 856B82             MOV DPL,6BH      ;CARGA DIR1 INCREMENTADA
06E2 80CE               SJMP M9

                ;***********************************************
                ;*                  RUTINA O                   *
                ;*  Calcula desplazamientos relativos. Si el   *
                ;*  resultado no está dentro de [-128, +127],  *
                ;*  despliega el mensaje: "RANGO INVALIDO".    *
                ;***********************************************
06E4 BA0B1F     O:      CJNE R2,#0BH,O6  ;SALTA SI NO HAY DIR2
06E7 11EC               ACALL DIR
06E9 08                 INC RO
06EA 1198               ACALL VALI
06EC AE6C               MOV R6,6CH
06EE AF6D               MOV R7,6DH
06F0 C3                 CLR C
06F1 EF                 MOV A,R7
06F2 9582               SUBB A,DPL
06F4 FF                 MOV R7,A         ;R7 = DIR2B - DIR1B
06F5 EE                 MOV A,R6
06F6 9583               SUBB A,DPH
06F8 FE                 MOV R6,A         ;R6 = DIR2A - DIR1A
06F9 BEFF17             CJNE R6,#0FFH,O5 ;SALTA SI ES POSITIVO
06FC BF8009             CJNE R7,#80H,O4  ;VERIFICA RANGO NEGATIVO
06FF EF         O1:     MOV A,R7
0700 112E               ACALL CRLF
0702 1180               ACALL NUAS       ;DESPLIEGA RESULTADO
0704 415E               AJMP INI
0706 41C1       O6:     AJMP OCHO
0708 50F5       O4:     JNC O1           ;SALTA SI R7 > 80H
070A 901CE0     O3:     MOV DPTR,#ERR4   ;"RANGO INVALIDO"
070D 112E               ACALL CRLF
070F 11A3               ACALL MENS
0711 415E               AJMP INI
0713 BE00F4     O5:     CJNE R6,#00H,O3  ;SALTA SI ESTA FUERA DE RANGO
0716 BF7F02             CJNE R7,#7FH,O2
0719 80E4               SJMP O1
071B 40E2       O2:     JC O1            ;SALTA SI R7 < 7FH
071D 80EB               SJMP O3

                ;***********************************************
                ;*                  RUTINA B                   *
                ;*  Borra el contenido de un bloque de memoria.*
                ;*  Si no existe DIR2, borra DIR1 solamente. R2*
                ;*  contiene el número de caracteres de la fila*
                ;*  de parámetros.  Si DIR1 pertenece a una    *
                ;*  localidad de ROM, envía el mensaje: "ERROR:*
                ;*  DIRECCION DE ROM".                         *
                ;***********************************************
071F BA0614     BE:     CJNE R2,#06H,B4 ;SALTA SI HAY DIR2
```

```
0722 11EC              ACALL DIR
0724 E56A              MOV A,6AH
0726 B41F09            CJNE A,#1FH,B7  ;SALTA SI DIR1A <> 1FH
0729 901D17     B8:    MOV DPTR,#ERR7  ;"ERROR: DIRECCION DE ROM"
072C 112E              ACALL CRLF
072E 11A3              ACALL MENS
0730 415E              AJMP INI
0732 40F5       B7:    JC B8
0734 8018              SJMP B2
0736 BA0B1E     B4:    CJNE R2,#0BH,B5 ;SALTA SI NO HAY DIR2
0739 11EC              ACALL DIR
073B E56A              MOV A,6AH
073D B41F02            CJNE A,#1FH,B9  ;SALTA SI DIR1A <> 1FH
0740 80E7              SJMP B8
0742 40E5       B9:    JC B8
0744 11F9              ACALL DIR2
0746 EE         B3:    MOV A,R6
0747 B58308            CJNE A,DPH,B1   ;SALTA SI DIR1A <> DIR2A
074A EF                MOV A,R7
074B B58204            CJNE A,DPL,B1   ;SALTA SI DIR1B <> DIR2B
074E E4         B2:    CLR A
074F F0                MOVX @DPTR,A    ;BORRA LA LOCALIDAD APUNTADA
0750 415E              AJMP INI
0752 E4         B1:    CLR A           ;CICLO DE BORRADO DE BLOQUE
0753 F0                MOVX @DPTR,A
0754 A3                INC DPTR
0755 80EF              SJMP B3
0757 41C1       B5:    AJMP OCHO

                ;************************************************
                ;*                   RUTINA F                   *
                ;*  Carga un FAH en el registro TH1 para operar *
                ;*  a una velocidad de 4800 bauds.              *
                ;************************************************
0759 758DFA     F:     MOV TH1,#0FAH
075C 415E              AJMP INI

                ;************************************************
                ;*                   RUTINA G                   *
                ;*  Carga un E8H en el registro TH1 para operar *
                ;*  a una velocidad de 1200 bauds.              *
                ;************************************************
075E 758DE8     G:     MOV TH1,#0E8H
0761 415E              AJMP INI


                ;************************************************
                ;* RUTINA DE SERVICIO DE INTERRUPCION DE TRAZADO *
                ;************************************************
0775                   ORG 775H
0775 C0D0              PUSH PSW
0777 75D018            MOV PSW,#18H
077A 0D                INC R5          ;CONT. DE INSTRUC. NO TRAZADAS
```

```
077B BD0426             CJNE R5,#04H,TOCH
077E 1D                 DEC R5
077F D0D0               POP PSW
0781 316B               ACALL RESP      ;RESPALDA REGISTROS DEL USUARIO
0783 75D018             MOV PSW,#18H
0786 311A               ACALL REG       ;DESPLIEGA REGISTROS
0788 A981               MOV R1,SP
078A 8783               MOV DPH,@R1
078C 19                 DEC R1
078D 8782               MOV DPL,@R1     ;CARGA DIRECCION DE INST. A TRAZAR
078F 11CE       TCUA:   ACALL LEE
0791 B42009             CJNE A,#ESP,TDOS  ;SALTA SI A <> ESP
0794 11BF               ACALL LOC         ;DESPLIEGA DIR. Y CODIGO DE OP.
0796 112E               ACALL CRLF
0798 75A881             MOV IE,#81H       ;HABILITA INTO*
079B 8009               SJMP TCIN
079D B40DEF     TDOS:   CJNE A,#CR,TCUA   ;SALTA SI A <> CR
07A0 758109             MOV SP,#09H       ;REGRESO A "LJMP 7D0H"
07A3 32                 RETI
07A4 D0D0       TOCH:   POP PSW
07A6 90E000     TCIN:   MOV DPTR,#0E000H  ;ENVIA NIVEL ALTO
07A9 743F               MOV A,#3FH
07AB F0                 MOVX @DPTR,A
07AC 7583C0             MOV DPH,#0C0H     ;ENVIA NIVEL BAJO
07AF 743F               MOV A,#3FH
07B1 F0                 MOVX @DPTR,A
07B2 31B8               ACALL RECU
07B4 43A881             ORL IE,#81H       ;HABILITA INT. EXTERNA 0
07B7 32                 RETI
07D0                    ORG 7D0H
07D0 90E000             MOV DPTR,#0E000H  ;ENVIA NIVEL ALTO Y DESHABILITA
07D3 743F               MOV A,#3FH        ;INTO* POR HARDWARE
07D5 F0                 MOVX @DPTR,A
07D6 415E               AJMP INI

                ;****************************************
                ;*    ACCESO DEL USUARIO A SUBRUTINAS   *
                ;****************************************
0800                    ORG 0800H
0800 020046             LJMP ASHE
0803 02003B             LJMP HEAS
0806 12002E             LCALL CRLF
0809 020080             LJMP NUAS
080C 02006C             LJMP ASNU
080F 12002E             LCALL CRLF
0812 0200CE             LJMP LEE
0815 12002E             LCALL CRLF
0818 0200E4             LJMP ENVIA
081B 02002E             LJMP CRLF
081E 12002E             LCALL CRLF
0821 0200A3             LJMP MENS
```

```
                ;*****************************************
                ;*          TABLA 1: TABLA DE COMANDOS          *
                ;*****************************************
1C40                    ORG TAB1
1C40 45                 DB 'E'
1C41 06                 DB 06H
1C42 44                 DB 'D'
1C43 0B                 DB 0BH
1C44 43                 DB 'C'
1C45 0B                 DB 0BH
1C46 54                 DB 'T'
1C47 06                 DB 06H
1C48 52                 DB 'R'
1C49 01                 DB 01H
1C4A 41                 DB 'A'
1C4B 01                 DB 01H
1C4C 4D                 DB 'M'
1C4D 10                 DB 10H
1C4E 4F                 DB 'O'
1C4F 0B                 DB 0BH
1C50 42                 DB 'B'
1C51 0B                 DB 0BH
1C52 4C                 DB 'L'
1C53 06                 DB 06H
1C54 49                 DB 'I'
1C55 0B                 DB 0BH
1C56 56                 DB 'V'
1C57 06                 DB 06H
1C58 46                 DB 'F'
1C59 01                 DB 01H
1C5A 47                 DB 'G'
1C5B 01                 DB 01H

                ;*****************************************
                ;*        TABLA 2: NOMBRES DE REGISTROS         *
                ;*****************************************
1C5E                    ORG TAB2
1C5E 04                 DB 04H
1C5F 41 43 43           DB 'ACC:'
1C62 3A
1C63 02                 DB 02H
1C64 42 3A              DB 'B:'
1C66 04                 DB 04H
1C67 50 53 57           DB 'PSW:'
1C6A 3A
1C6B 04                 DB 04H
1C6C 44 50 48           DB 'DPH:'
1C6F 3A
1C70 04                 DB 04H
1C71 44 50 4C           DB 'DPL:'
1C74 3A
1C75 03                 DB 03H
1C76 53 50 3A           DB 'SP:'
```

```
1C79 03                 DB 03H
1C7A 49 45 3A           DB 'IE:'
1C7D 03                 DB 03H
1C7E 49 50 3A           DB 'IP:'
1C81 05                 DB 05H
1C82 54 4D 4F           DB 'TMOD:'
1C85 44 3A
1C87 05                 DB 05H
1C88 54 43 4F           DB 'TCON:'
1C8B 4E 3A
1C8D 05                 DB 05H
1C8E 53 43 4F           DB 'SCON:'
1C91 4E 3A
1C93 05                 DB 05H
1C94 50 43 4F           DB 'PCON:'
1C97 4E 3A
1C99 04                 DB 04H
1C9A 54 48 30           DB 'TH0:'
1C9D 3A
1C9E 04                 DB 04H
1C9F 54 4C 30           DB 'TL0:'
1CA2 3A
1CA3 03                 DB 03H
1CA4 50 31 3A           DB 'P1:'
1CA7 05                 DB 05H
1CA8 53 42 55           DB 'SBUF:'
1CAB 46 3A

                ;*****************************************
                ;*        TABLA 3: TABLA DE ERRORES      *
                ;*****************************************
1CAD                    ORG TAB3
1CAD 11                 DB 11H
1CAE 45 52 52           DB 'ERROR DE SINTAXIS'
1CB1 4F 52 20 44 45 20 53 49 4E 54
1CBB 41 58 49 53
1CBF 10                 DB 10H
1CC0 43 4F 4D           DB 'COMANDO INVALIDO'
1CC3 41 4E 44 4F 20 49 4E 56 41 4C
1CCD 49 44 4F
1CD0 0F                 DB 0FH
1CD1 4E 55 4D           DB 'NUMERO INVALIDO'
1CD4 45 52 4F 20 49 4E 56 41 4C 49
1CDE 44 4F
1CE0 0E                 DB 0EH
1CE1 52 41 4E           DB 'RANGO INVALIDO'
1CE4 47 4F 20 49 4E 56 41 4C 49 44
1CEE 4F
1CEF 16                 DB 16H
1CF0 44 49 46           DB 'DIFERENCIA EN PROGRAMA'
1CF3 45 52 45 4E 43 49 41 20 45 4E
1CFD 20 50 52 4F 47 52 41 4D 41
1D06 10                 DB 10H
```

```
1D07 45 52 52          DB 'ERROR DE LECTURA'
1D0A 4F 52 20 44 45 20 4C 45 43 54
1D14 55 52 41
1D17 17                DB 17H
1D18 45 52 52          DB 'ERROR: DIRECCION DE ROM'
1D1B 4F 52 3A 20 44 49 52 45 43 43
1D25 49 4F 4E 20 44 45 20 52 4F 4D

                ;*********************************************
                ;*          TABLA 4: PANTALLA DE AYUDA       *
                ;*********************************************
1D2F                   ORG TAB4
1D2F D4                DB 0D4H
1D30 0D 0A             DB CR,LF
1D32 41 28 43          DB 'A(CR)                                AYUDA'
1D35 52 29 09 09 09 20 20 20 20 41
1D3F 59 55 44 41
1D43 0D 0A             DB CR,LF
1D45 46 28 43          DB 'F(CR)                                VELOCIDAD A 4800
BAUDS'
1D48 52 29 09 09 09 20 20 20 20 56
1D52 45 4C 4F 43 49 44 41 44 20 41
1D5C 20 34 38 30 30 20 42 41 55 44
1D66 53
1D67 0D 0A             DB CR,LF
1D69 47 28 43          DB 'G(CR)                                VELOCIDAD A 1200
BAUDS'
1D6C 52 29 09 09 09 20 20 20 20 56
1D76 45 4C 4F 43 49 44 41 44 20 41
1D80 20 31 32 30 30 20 42 41 55 44
1D8A 53
1D8B 0D 0A             DB CR,LF
1D8D 4C 20 3C          DB 'L <num>(CR)                  LEE PROGRAMA'
1D90 6E 75 6D 3E 28 43 52 29 09 09
1D9A 20 20 20 20 4C 45 45 20 50 52
1DA4 4F 47 52 41 4D 41
1DAA 0D 0A             DB CR,LF
1DAC 56 20 3C          DB 'V <num>(CR)                  VERIFICA PROGRAMA'
1DAF 6E 75 6D 3E 28 43 52 29 09 09
1DB9 20 20 20 20 56 45 52 49 46 49
1DC3 43 41 20 50 52 4F 47 52 41 4D
1DCD 41
1DCE 0D 0A             DB CR,LF
1DD0 42 20 3C          DB 'B <dir1>[ <dir2>](CR)                BORRA LOCALIDAD O
BLOQUE'
1DD3 64 69 72 31 3E 5B 20 3C 64 69
1DDD 72 32 3E 5D 28 43 52 29 09 20
1DE7 20 20 20 42 4F 52 52 41 20 4C
1DF1 4F 43 41 4C 49 44 41 44 20 4F
1DFB 20 42 4C 4F 51 55 45
1E02 F8                DB 0F8H
1E03 0D 0A             DB CR,LF
1E05 44 20 3C          DB 'D <dir1>[ <dir2>](CR)                DESPLIEGA
```

```
LOCALIDADES'
 1E08 64 69 72 31 3E 5B 20 3C 64 69
 1E12 72 32 3E 5D 28 43 52 29 09 20
 1E1C 20 20 20 44 45 53 50 4C 49 45
 1E26 47 41 20 4C 4F 43 41 4C 49 44
 1E30 41 44 45 53
 1E34 0D 0A              DB CR,LF
 1E36 43 20 3C           DB 'C <dir1>[ <dir2>](CR)          CORRE PROGRAMA'
 1E39 64 69 72 31 3E 5B 20 3C 64 69
 1E43 72 32 3E 5D 28 43 52 29 09 20
 1E4D 20 20 20 43 4F 52 52 45 20 50
 1E57 52 4F 47 52 41 4D 41
 1E5E 0D 0A              DB CR,LF
 1E60 4F 20 3C           DB 'O <dir1> <dir2>(CR)     CALCULA DESPLAZAMIENTOS
RELATIVOS'
 1E63 64 69 72 31 3E 20 3C 64 69 72
 1E6D 32 3E 28 43 52 29 09 20 20 20
 1E77 20 43 41 4C 43 55 4C 41 20 44
 1E81 45 53 50 4C 41 5A 41 4D 49 45
 1E8B 4E 54 4F 53 20 52 45 4C 41 54
 1E95 49 56 4F 53
 1E99 0D 0A              DB CR,LF
 1E9B 49 20 3C           DB 'I <dir1> <dir2>(CR)     IMPRIME PROGRAMA'
 1E9E 64 69 72 31 3E 20 3C 64 69 72
 1EA8 32 3E 28 43 52 29 09 20 20 20
 1EB2 20 49 4D 50 52 49 4D 45 20 50
 1EBC 52 4F 47 52 41 4D 41
 1EC3 0D 0A              DB CR,LF
 1EC5 4D 20 3C           DB 'M <dir1> <dir2> <dir3>(CR)  MUEVE BLOQUE DE
MEMORIA'
 1EC8 64 69 72 31 3E 20 3C 64 69 72
 1ED2 32 3E 20 3C 64 69 72 33 3E 28
 1EDC 43 52 29 20 20 4D 55 45 56 45
 1EE6 20 42 4C 4F 51 55 45 20 44 45
 1EF0 20 4D 45 4D 4F 52 49 41
 1EF8 F5                 DB 0F5H
 1EF9 0D 0A              DB CR,LF
 1EFB 52 28 43           DB 'R(CR)                          DESPLIEGA
REGISTROS'
 1EFE 52 29 09 09 09 20 20 20 20 44
 1F08 45 53 50 4C 49 45 47 41 20 52
 1F12 45 47 49 53 54 52 4F 53
 1F1A 0D 0A              DB CR,LF
 1F1C 20 20 20           DB '     (ESPACIO) sigue'
 1F1F 20 20 28 45 53 50 41 43 49 4F
 1F29 29 20 73 69 67 75 65
 1F30 0D 0A              DB CR,LF
 1F32 20 20 20           DB '      (CR)    termina'
 1F35 20 20 28 43 52 29 20 20 20 20
 1F3F 74 65 72 6D 69 6E 61
 1F46 0D 0A              DB CR,LF
 1F48 45 20 3C           DB 'E <dir1>(CR)                   EXAMINA Y MODIFICA
MEMORIA'
```

```
1F4B 64 69 72 31 3E 28 43 52 29 09
1F55 09 20 20 20 20 45 58 41 4D 49
1F5F 4E 41 20 59 20 4D 4F 44 49 46
1F69 49 43 41 20 4D 45 4D 4F 52 49
1F73 41
1F74 0D 0A              DB CR,LF
1F76 20 20 20           DB '      (ESPACIO) sigue'
1F79 20 20 28 45 53 50 41 43 49 4F
1F83 29 20 73 69 67 75 65
1F8A 0D 0A              DB CR,LF
1F8C 20 20 20           DB '      (CR)     termina'
1F8F 20 20 28 43 52 29 20 20 20 20
1F99 74 65 72 6D 69 6E 61
1FA0 0D 0A              DB CR,LF
1FA2 54 20 3C           DB 'T <dir1>(CR)                    TRAZA PROGRAMA'
1FA5 64 69 72 31 3E 28 43 52 29 09
1FAF 09 20 20 20 20 54 52 41 5A 41
1FB9 20 50 52 4F 47 52 41 4D 41
1FC2 0D 0A              DB CR,LF
1FC4 20 20 20           DB '      (ESPACIO) sigue'
1FC7 20 20 28 45 53 50 41 43 49 4F
1FD1 29 20 73 69 67 75 65
1FD8 0D 0A              DB CR,LF
1FDA 20 20 20           DB '      (CR)     termina'
1FDD 20 20 28 43 52 29 20 20 20 20
1FE7 74 65 72 6D 69 6E 61
0000                    END
```

ETIQUETA VALOR

```
A . . . . . L 03C0
ASC1. . . . . L 004E
ASC2. . . . . L 0057
ASC3. . . . . L 005C
ASC4. . . . . L 0065
ASC5. . . . . L 0068
ASC6. . . . . L 0061
ASHE. . . . . L 0046
ASNU. . . . . L 006C
B1. . . . . L 0752
B2. . . . . L 074E
B3. . . . . L 0746
B4. . . . . L 0736
B5. . . . . L 0757
B7. . . . . L 0732
B8. . . . . L 0729
B9. . . . . L 0742
BE. . . . . L 071F
BORR. . . . . L 0265
BS. . . . . I 0008
BUFLE . . . . . I 3FA0
C . . . . . L 0519
C1. . . . . L 0532
```

```
C2. . . . . L 052E
C3. . . . . L 0525
C4. . . . . L 0580
C5. . . . . L 0540
C6. . . . . L 054D
C7. . . . . L 057E
CIC . . . . . L 02A2
CINCO . . . . . L 02BF
COM . . . . . L 02A9
COMAN . . . . . I 0300
CONT. . . . . L 00B6
CORRE . . . . . I 0004
CORRI . . . . . L 01FE
CR. . . . . I 000D
CR1 . . . . . L 00A9
CR2 . . . . . L 00AF
CRLF. . . . . L 002E
CUATRO. . . . . L 02BB
D . . . . . L 037B
D1. . . . . L 038F
D2. . . . . L 03A2
D3. . . . . L 03AD
D4. . . . . L 03B2
D5. . . . . L 0393
D6. . . . . L 0385
D7. . . . . L 03BE
D8. . . . . L 03B0
D9. . . . . L 03A0
DESP. . . . . I 0003
DIR . . . . . L 00EC
DIR2. . . . . L 00F9
DOS . . . . . L 0291
E . . . . . L 031E
E1. . . . . L 0355
E2. . . . . L 0339
E3. . . . . L 0360
E4. . . . . L 0375
E5. . . . . L 0347
E6. . . . . L 0345
E7. . . . . L 0331
E8. . . . . L 0328
ECO . . . . . I 0001
ENVIA . . . . . L 00E4
ERR1. . . . . I 1CAD
ERR2. . . . . I 1CBF
ERR3. . . . . I 1CD0
ERR4. . . . . I 1CE0
ERR5. . . . . I 1CEF
ERR6. . . . . I 1D06
ERR7. . . . . I 1D17
ESP . . . . . I 0020
F . . . . . L 0759
FIL2. . . . . I 0072
```

```
FILA. . . . . I 0070
FILREG. . . . . I 3FEA
G . . . . . L 075E
HEAS. . . . . L 003B
HEX . . . . . L 0041
HEX1. . . . . L 0045
I . . . . . L 0477
I1. . . . . L 04F7
I2. . . . . L 04F1
I3. . . . . L 047E
I4. . . . . L 04D9
I5. . . . . L 0501
I6. . . . . L 04BF
I7. . . . . L 0498
I8. . . . . L 0510
I9. . . . . L 04EF
INI . . . . . L 025E
INREG . . . . . L 0212
L . . . . . L 03CD
L1. . . . . L 0431
L10 . . . . . L 046A
L11 . . . . . L 0463
L12 . . . . . L 0418
L13 . . . . . L 0411
L14 . . . . . L 0416
L2. . . . . L 043A
L3. . . . . L 043F
L4. . . . . L 042D
L5. . . . . L 0449
L6. . . . . L 03D8
L7. . . . . L 041F
L8. . . . . L 0469
L9. . . . . L 0460
LEE . . . . . L 00CE
LEER2 . . . . . L 00D7
LF. . . . . I 000A
LIMP. . . . . L 027B
LOC . . . . . L 00BF
M . . . . . L 062D
M1. . . . . L 0670
M10 . . . . . L 06C6
M2. . . . . L 0684
M3. . . . . L 0647
M4. . . . . L 063C
M5. . . . . L 0645
M6. . . . . L 068E
M7. . . . . L 06A4
M8. . . . . L 0655
M9. . . . . L 06B2
MENS. . . . . L 00A3
NUAS. . . . . L 0080
O . . . . . L 06E4
O1. . . . . L 06FF
```

```
02. . . . . L 071B
03. . . . . L 070A
04. . . . . L 0708
05. . . . . L 0713
06. . . . . L 0706
OCHO. . . . . L 02C1
PARAM . . . . . L 0280
PROGCT. . . . . I 3FE0
PROMPT. . . . . I 003E
PTO . . . . . L 002D
R . . . . . L 05C5
RCIN. . . . . L 05DB
RCUA. . . . . L 05EB
RDOS. . . . . L 0612
RECU. . . . . L 01B8
REG . . . . . L 011A
REG1. . . . . L 013C
REG2. . . . . L 0132
REG3. . . . . L 012F
RESP. . . . . L 016B
RSEI. . . . . L 0627
RTRE. . . . . L 05D4
RUNO. . . . . L 0607
SALTA . . . . . U 0000
SEIS. . . . . L 02B6
SEL . . . . . L 029B
SIETE . . . . . L 02C4
SUB1. . . . . L 010C
SUB2. . . . . L 0117
SUB3. . . . . L 0119
SUB4. . . . . L 010E
T . . . . . L 0582
TAB1. . . . . I 1C40
TAB2. . . . . I 1C5E
TAB3. . . . . I 1CAD
TAB4. . . . . I 1D2F
TCIN. . . . . L 07A6
TCUA. . . . . L 078F
TDIE. . . . . L 0591
TDOS. . . . . L 079D
TOCH. . . . . L 07A4
TRES. . . . . L 0296
TTRE. . . . . L 0593
UNO . . . . . L 0288
V . . . . . L 03CB
VALI. . . . . L 0098
VECINT. . . . . I 3F90
VERIF . . . . . I 0000
VINTO . . . . . I 3F90
VINT1 . . . . . I 3F96
VTO . . . . . I 3F93
VT1 . . . . . I 3F99
```

APÉNDICE C

JUEGO DE INSTRUCCIONES DEL 8031

Interrupt Response Time: To finish execution of current instruction, respond to the interrupt request, push the PC and to vector to the first instruction of the interrupt service program requires 38 to 81 oscillator periods (3 to 7μs @ 12 MHz).

INSTRUCTIONS THAT AFFECT FLAG SETTINGS*

INSTRUCTION	FLAG C	FLAG OV	FLAG AC	INSTRUCTION	FLAG C	FLAG OV	FLAG AC
ADD	X	X	X	CLR C	O		
ADDC	X	X	X	CPL C	X		
SUBB	X	X	X	ANL C,bit	X		
MUL	O	X		ANL C,/bit	X		
DIV	O	X		ORL C,bit	X		
DA	X			ORL C,/bit	X		
RRC	X			MOV C,bit	X		
RLC	X			CJNE	X		
SETB C	1						

*Note that operations on SFR byte address 208 or bit addresses 209-215 (i.e., the PSW or bits in the PSW) will also affect flag settings.

Notes on instruction set and addressing modes:

- **Rn** — Register R7-R0 of the currently selected Register Bank.
- **direct** — 8-bit internal data location's address. This could be an Internal Data RAM location (0-127) or a SFR [i.e., I/O port, control register, status register, etc. (128-255)].
- **@Ri** — 8-bit internal data RAM location (0-255) addressed indirectly through register R1 or R0.
- **#data** — 8-bit constant included in instruction.
- **#data 16** — 16-bit constant included in instruction.
- **addr 16** — 16-bit destination address. Used by LCALL & LJMP. A branch can be anywhere within the 64K-byte Program Memory address space.
- **addr 11** — 11-bit destination address. Used by ACALL & AJMP. The branch will be within the same 2K-byte page of program memory as the first byte of the following instruction.
- **rel** — Signed (two's complement) 8-bit offset byte. Used by SJMP and all conditional jumps. Range is -128 to +127 bytes relative to first byte of the following instruction.
- **bit** — Direct Addressed bit in Internal Data RAM or Special Function Register.
- ***** — New operation not provided by 8048AH/8049AH.

ARITHMETIC OPERATIONS

Mnemonic		Description	Byte	Oscillator Period
ADD	A,Rn	Add register to Accumulator	1	12
ADD	A,direct	Add direct byte to Accumulator	2	12
ADD	A,@Ri	Add indirect RAM to Accumulator	1	12
ADD	A,#data	Add immediate data to Accumulator	2	12
ADDC	A,Rn	Add register to Accumulator with Carry	1	12
ADDC	A,direct	Add direct byte to Accumulator with Carry	2	12
ADDC	A,@Ri	Add indirect RAM to Accumulator with Carry	1	12
ADDC	A,#data	Add immediate data to Acc with Carry	2	12
SUBB	A,Rn	Subtract register from Acc with borrow	1	12
SUBB	A,direct	Subtract direct byte from Acc with borrow	2	12

ARITHMETIC OPERATIONS Cont.

Mnemonic		Description	Byte	Oscillator Period
SUBB	A,@Ri	Subtract indirect RAM from Acc with borrow	1	12
SUBB	A,#data	Subtract immediate data from Acc with borrow	2	12
INC	A	Increment Accumulator	1	12
INC	Rn	Increment register	1	12
INC	direct	Increment direct byte	2	12
INC	@Ri	Increment indirect RAM	1	12
DEC	A	Decrement Accumulator	1	12
DEC	Rn	Decrement Register	1	12
DEC	direct	Decrement direct byte	2	12
DEC	@Ri	Decrement indirect RAM	1	12
INC	DPTR	Increment Data Pointer	1	24
MUL	AB	Multiply A & B	1	48
DIV	AB	Divide A by B	1	48
DA	A	Decimal Adjust Accumulator	1	12

LOGICAL OPERATIONS

Mnemonic		Description	Byte	Oscillator Period
ANL	A,Rn	AND register to Accumulator	1	12
ANL	A,direct	AND direct byte to Accumulator	2	12
ANL	A,@Ri	AND indirect RAM to Accumulator	1	12
ANL	A,#data	AND immediate data to Accumulator	2	12
ANL	direct,A	AND Accumulator to direct byte	2	12
ANL	direct,#data	AND immediate data to direct byte	3	24
ORL	A,Rn	OR register to Accumulator	1	12
ORL	A,direct	OR direct byte to Accumulator	2	12
ORL	A,@Ri	OR indirect RAM to Accumulator	1	12
ORL	A,#data	OR immediate data to Accumulator	2	12
ORL	direct,A	OR Accumulator to direct byte	2	12
ORL	direct,#data	OR immediate data to direct byte	3	24
XRL	A,Rn	Exclusive-OR register to Accumulator	1	12
XRL	A,direct	Exclusive-OR direct byte to Accumulator	2	12

LOGICAL OPERATIONS Cont.

Mnemonic		Description	Byte	Oscillator Period
XRL	A,@Ri	Exclusive-OR indirect RAM to Accumulator	1	12
XRL	A,#data	Exclusive-OR immediate data to Accumulator	2	12
XRL	direct,A	Exclusive-OR Accumulator to direct byte	2	12
XRL	direct,#data	Exclusive-OR immediate data to direct byte	3	24
CLR	A	Clear Accumulator	1	12
CPL	A	Complement Accumulator	1	12
RL	A	Rotate Accumulator Left	1	12
RLC	A	Rotate Accumulator Left through the Carry	1	12
RR	A	Rotate Accumulator Right	1	12
RRC	A	Rotate Accumulator Right through the Carry	1	12
SWAP	A	Swap nibbles within the Accumulator	1	12

8051 Instruction Set Summary (Continued)

DATA TRANSFER

Mnemonic		Description	Byte	Oscillator Period
MOV	A,Rn	Move register to Accumulator	1	12
MOV	A,direct	Move direct byte to Accumulator	2	12
MOV	A,@Ri	Move indirect RAM to Accumulator	1	12
MOV	A,#data	Move immediate data to Accumulator	2	12
MOV	Rn,A	Move Accumulator to register	1	12
MOV	Rn,direct	Move direct byte to register	2	24
MOV	Rn,#data	Move immediate data to register	2	12
MOV	direct,A	Move Accumulator to direct byte	2	12
MOV	direct,Rn	Move register to direct byte	2	24
MOV	direct,direct	Move direct byte to direct	3	24
MOV	direct,@Ri	Move indirect RAM to direct byte	2	24
MOV	direct,#data	Move immediate data to direct byte	3	24
MOV	@Ri,A	Move Accumulator to indirect RAM	1	12
MOV	@Ri,direct	Move direct byte to indirect RAM	2	24
MOV	@Ri,#data	Move immediate data to indirect RAM	2	12

DATA TRANSFER Cont.

Mnemonic		Description	Byte	Oscillator Period
MOV	DPTR,#data16	Load Data Pointer with a 16-bit constant	3	24
MOVC	A,@A+DPTR	Move Code byte relative to DPTR to Acc	1	24
MOVC	A,@A+PC	Move Code byte relative to PC to Acc	1	24
MOVX	A,@Ri	Move External RAM (8-bit addr) to Acc	1	24
MOVX	A,@DPTR	Move External RAM (16-bit addr) to Acc	1	24
MOVX	@Ri,A	Move Acc to External RAM (8-bit addr)	1	24
MOVX	@DPTR,A	Move Acc to External RAM (16-bit addr)	1	24
PUSH	direct	Push direct byte onto stack	2	24
POP	direct	Pop direct byte from stack	2	24
XCH	A,Rn	Exchange register with Accumulator	1	12
XCH	A,direct	Exchange direct byte with Accumulator	2	12
XCH	A,@Ri	Exchange indirect RAM with Accumulator	1	12
XCHD	A,@Ri	Exchange low-order Digit indirect RAM with Acc	1	12

8051 Instruction Set Summary (Continued)

BOOLEAN VARIABLE MANIPULATION

Mnemonic		Description	Byte	Oscillator Period
CLR	C	Clear Carry	1	12
CLR	bit	Clear direct bit	2	12
SETB	C	Set Carry	1	12
SETB	bit	Set direct bit	2	12
CPL	C	Complement Carry	1	12
CPL	bit	Complement direct bit	2	12
ANL	C,bit	AND direct bit to Carry	2	24
ANL	C,/bit	AND complement of direct bit to Carry	2	24
ORL	C,bit	OR direct bit to Carry	2	24
ORL	C,/bit	OR complement of direct bit to Carry	2	24
MOV	C,bit	Move direct bit to Carry	2	12
MOV	bit,C	Move Carry to direct bit	2	24
JC	rel	Jump if Carry is set	2	24
JNC	rel	Jump if Carry not set	2	24
JB	bit,rel	Jump if direct Bit is set	3	24
JNB	bit,rel	Jump if direct Bit is Not set	3	24
JBC	bit,rel	Jump if direct Bit is set & clear bit	3	24

PROGRAM BRANCHING

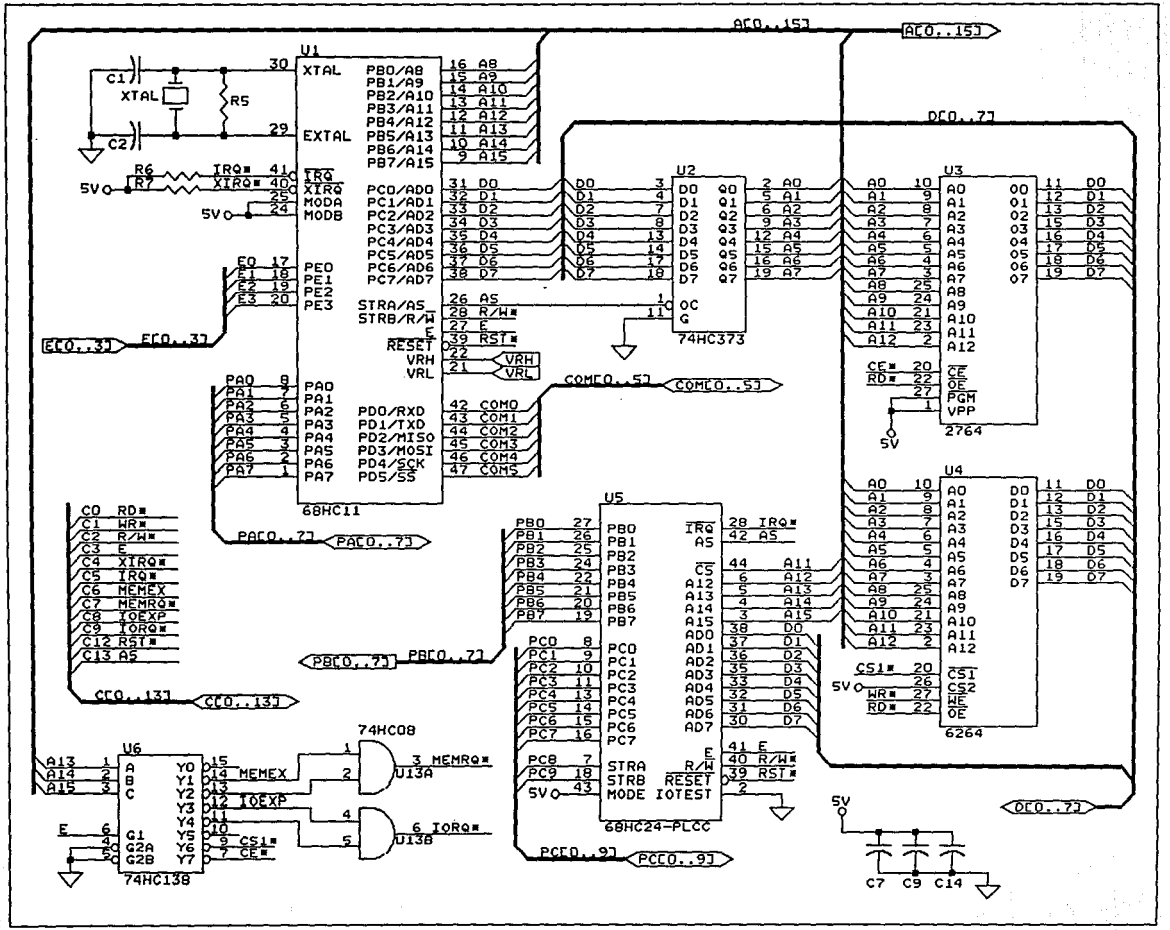
Mnemonic		Description	Byte	Oscillator Period
ACALL	addr11	Absolute Subroutine Call	2	24
LCALL	addr16	Long Subroutine Call	3	24
RET		Return from Subroutine	1	24

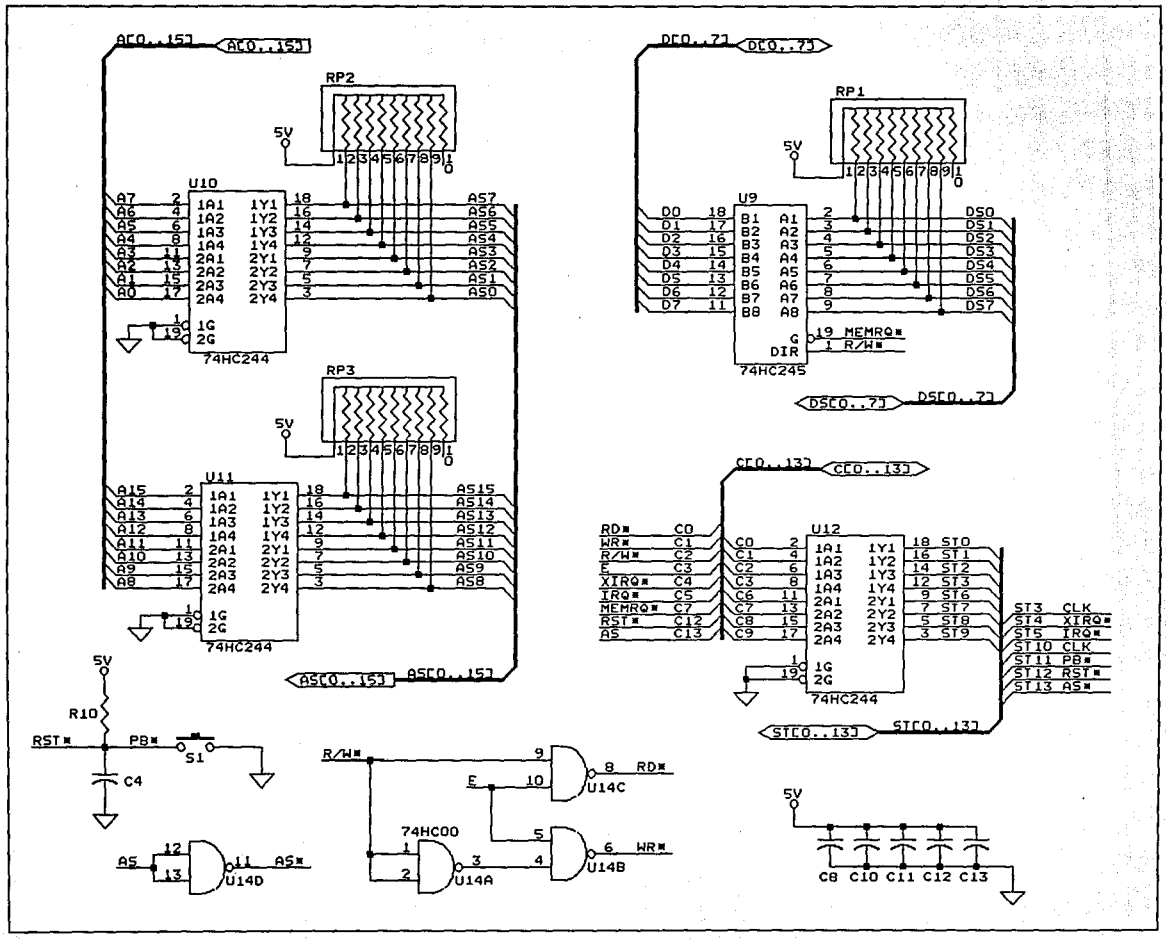
PROGRAM BRANCHING Cont.

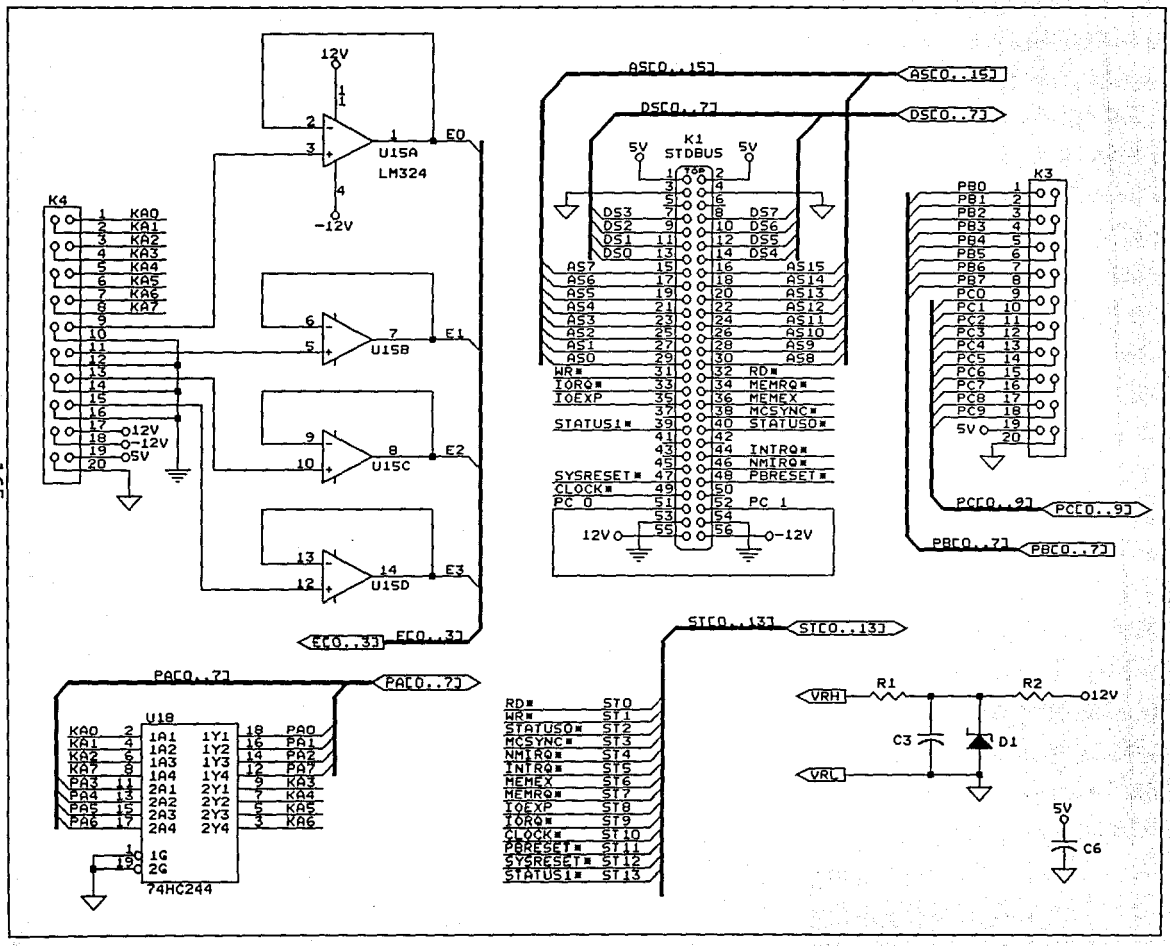
Mnemonic		Description	Byte	Oscillator Period
RETI		Return from interrupt	1	24
AJMP	addr11	Absolute Jump	2	24
LJMP	addr16	Long Jump	3	24
SJMP	rel	Short Jump (relative addr)	2	24
JMP	@A+DPTR	Jump indirect relative to the DPTR	1	24
JZ	rel	Jump if Accumulator is Zero	2	24
JNZ	rel	Jump if Accumulator is Not Zero	2	24
CJNE	A,direct,rel	Compare direct byte to Acc and Jump if Not Equal	3	24
CJNE	A,#data,rel	Compare immediate to Acc and Jump if Not Equal	3	24
CJNE	Rn,#data,rel	Compare immediate to register and Jump if Not Equal	3	24
CJNE	@Ri,#data,rel	Compare immediate to indirect and Jump if Not Equal	3	24
DJNZ	Rn,rel	Decrement register and Jump if Not Zero	2	24
DJNZ	direct,rel	Decrement direct byte and Jump if Not Zero	3	24
NOP		No Operation	1	12

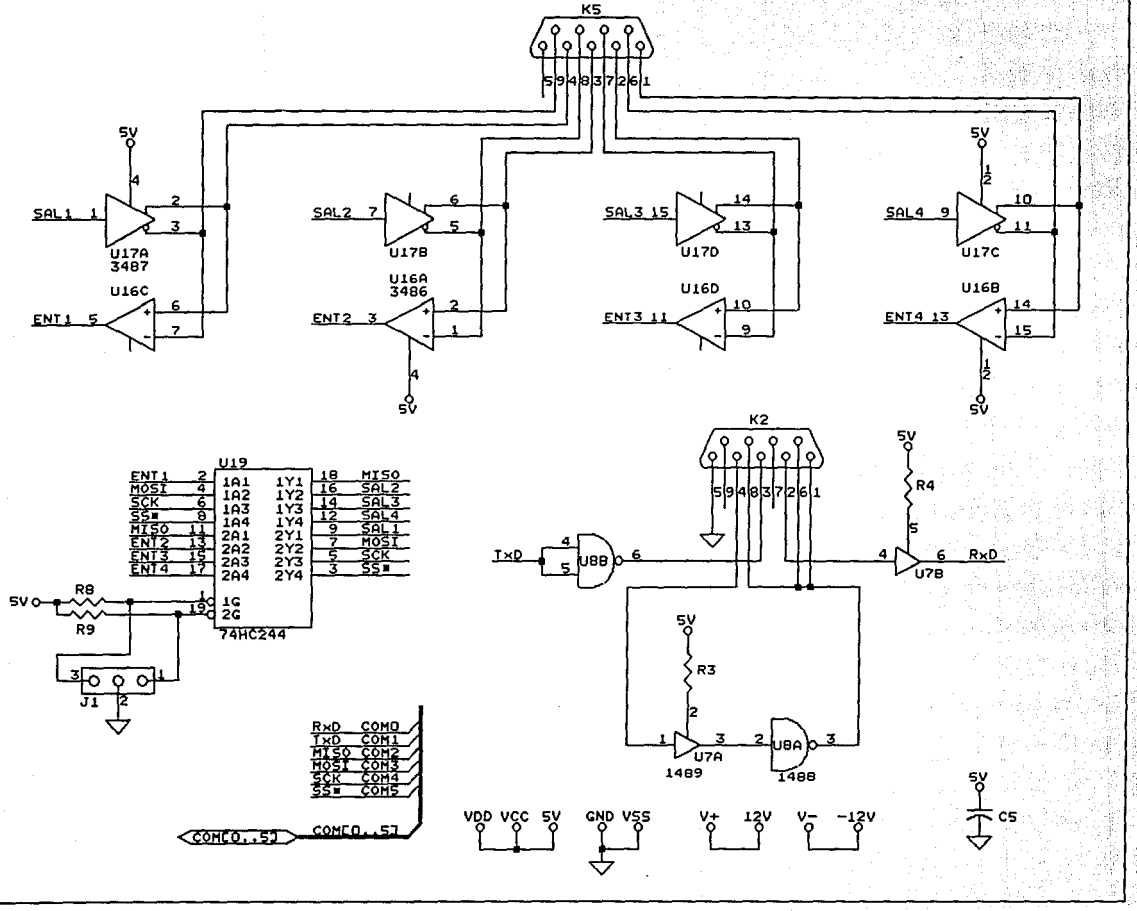
APÉNDICE D

DIAGRAMAS ELECTRÓNICOS, DE DISPOSICIÓN, DEL CIRCUITO IMPRESO Y LISTA DE PARTES DEL MC68HC11

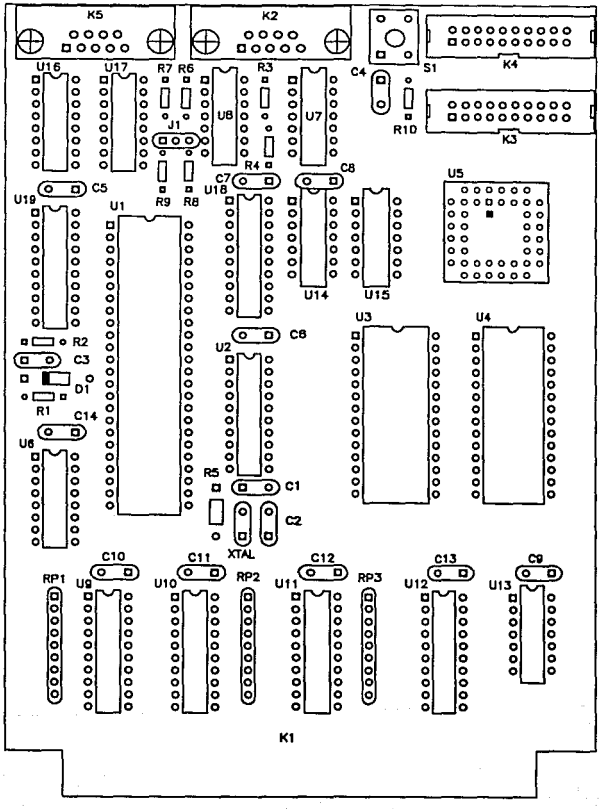

U1
68HC11
XTAL
EXTAL
MODA
MODB
XIRQ
IRQ
RESET
VRH
VRL
PE0
PE1
PE2
PE3
STRA/AS
STRB/R/W
E
PA0
PA1
PA2
PA3
PA4
PA5
PA6
PA7
PB0/A8
PB1/A9
PB2/A10
PB3/A11
PB4/A12
PB5/A13
PB6/A14
PB7/A15
PC0/AD0
PC1/AD1
PC2/AD2
PC3/AD3
PC4/AD4
PC5/AD5
PC6/AD6
PC7/AD7
PD0/RXD
PD1/TXD
PD2/MISO
PD3/MOSI
PD4/SCK
PD5/SS
C1
C2
R5
R6
R7
5V
IRQ*
XIRQ*
A0..15]
D0..7]
E0..3]
COM0..5]
PA0..7]
PB0..7]
C0..13]
C0 RD*
C1 WR*
C2 R/W*
C3 E
C4 XIRQ*
C5 IRQ*
C6 MEMEX
C7 MEMRQ*
C8 IOEXP
C9 IORQ*
C12 RST*
C13 A5
U2
74HC373
U3
2764
U4
6264
U5
68HC24-PLCC
PC0..9]
U6
74HC138
74HC08
U13A
U13B
MEMEX
IOEXP
IORQ*
CS1*
CE*
C7
C9
C14


AC0..15
AC0..15
RP2
5V
U10
74HC244
AS0..15
AS0..15
RP3
U11
74HC244
DC0..7
DC0..7
RP1
U9
74HC245
DS0..7
DS0..7
CC0..13
CC0..13
U12
74HC244
ST0..13
ST0..13
R10
RST*
PB*
S1
C4
R/W*
E
74HC00
U14A
U14B
U14C
U14D
RD*
WR*
AS
MEMRQ*
IRQ*
C8 C10 C11 C12 C13

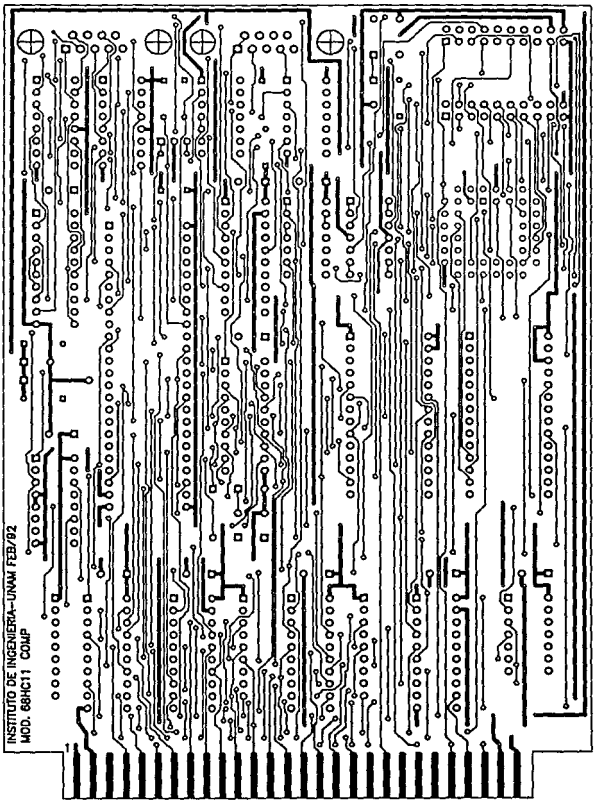

12V
U15A
LM324
-12V
E0
K4
KA0
KA1
KA2
KA3
KA4
KA5
KA6
KA7
12V
-12V
5V
U15B
E1
U15C
E2
U15D
E3
EC0..3]
EC0..3]
PA0..7]
PA[0..7]
U18
74HC244
ST0
ST1
ST2
ST3
ST4
ST5
ST6
ST7
ST8
ST9
ST10
ST11
ST12
ST13
AS[0..15]
DS[0..7]
K1
STDBUS
DS3
DS2
DS1
DS0
DS7
DS6
DS5
DS4
AS7
AS6
AS5
AS4
AS3
AS2
AS1
AS0
AS15
AS14
AS13
AS12
AS11
AS10
AS9
AS8
WR*
IORQ*
IOEXP
RD*
MEMRQ*
MEMEX
MCSYNC*
STATUS0*
STATUS1*
INTRQ*
NMIRQ*
SYSRESET*
CLOCK*
PC 0
PBRESET*
PC 1
12V
-12V
K3
PB0
PB1
PB2
PB3
PB4
PB5
PB6
PB7
PC0
PC1
PC2
PC3
PC4
PC5
PC6
PC7
PC8
PC9
5V
PC[0..9]
PB[0..7]
ST[0..13]
VRH
VRL
R1
R2
C3
D1
C6


K5
5 9 4 8 3 7 2 6 1
5V
SAL1 1
U17A
3487
SAL2 7
U17B
SAL3 15
U17D
SAL4 9
U17C
U16C
ENT1 5
U16A
3486
ENT2 3
U16D
ENT3 11
U16B
ENT4 13
U19
ENT1 2
MOSI 4
SCK 6
SS■ 8
MISO 11
ENT2 13
ENT3 15
ENT4 17
1A1 1A2 1A3 1A4 2A1 2A2 2A3 2A4
1G 2G
1Y1 1Y2 1Y3 1Y4 2Y1 2Y2 2Y3 2Y4
18 MISO
16 SAL2
14 SAL3
12 SAL4
9 SAL1
7 MOSI
5 SCK
3 SS■
74HC244
R8
R9
J1
K2
R4
TxD
U8B
U7B
RxD
R3
U7A
1489
U8A
1488
RxD COM0
TxD COM1
MISO COM2
MOSI COM3
SCK COM4
SS■ COM5
COM[0..5]
VDD VCC 5V
GND VSS
V+ 12V
V- -12V
C5

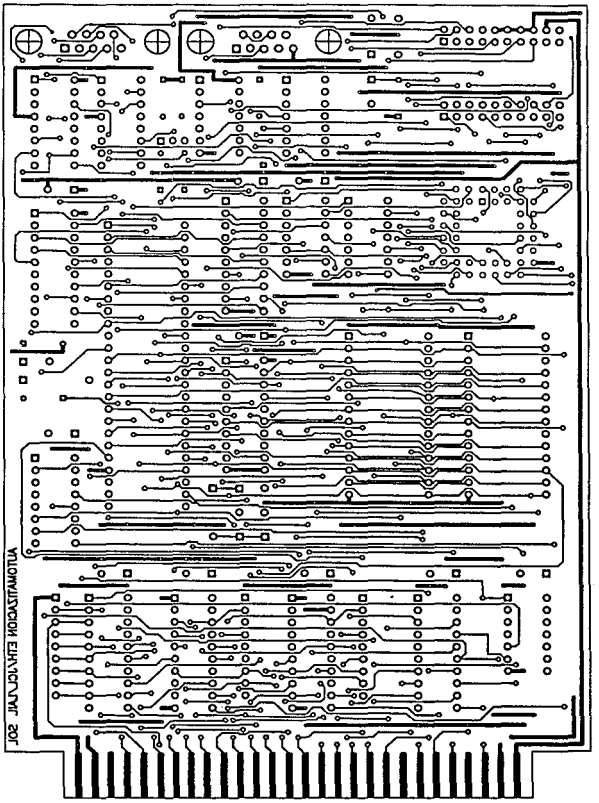
DISPOSICIÓN DE COMPONENTES



CIRCUITO IMPRESO DE LA CARA DE COMPONENTES



CIRCUITO IMPRESO DE LA CARA DE SOLDADURA



LISTA DE PARTES

No.	CANTIDAD	REFERENCIA	PARTE
1	1	U1	68HC11
2	1	U2	74HC373
3	1	U3	2764
4	1	U4	6264
5	1	U5	68HC24
6	1	U6	74HC138
7	1	U7	1489
8	1	U8	1488
9	1	U9	74HC245
10	5	U10-U12, U18, U19	74HC244
11	1	U13	74HC08
12	1	U14	74HC00
13	1	U15	LM324
14	1	U16	3486
15	1	U17	3487
16	1	K1	PEINE 56 ALFILERES
17	2	K2, K5	CONECTOR DB9 HEMBRA
18	1	K3	CONECTOR 20
19	1	K4	CONECTOR 20
20	1	XTAL	CRISTAL DE 8 MHz
21	2	C1, C2	18 pF
22	2	C3, C4	1 μF
23	10	C5-C14	0.1 μF
24	1	R1	1 KΩ
25	1	R2	1.2 KΩ
26	6	R3, R4, R6-R9	4.7 KΩ
27	1	R5	10 MΩ
28	1	R10	8.2 KΩ
29	3	RP1-RP3	4.7 KΩ
30	1	D1	DIODO ZENER 5 V
31	1	S1	PUSH BUTTON
32	1	J1	BRINCADOR DE 3

APÉNDICE E

LISTADO DEL PROGRAMA MONITOR MC68HC11

```
                        ************************************
                        *     REGISTROS DEL MC68HC11       *
                        ************************************
1000                    INIREG  EQU $1000
1009                    DDRD    EQU INIREG+$09
100E                    TCNT    EQU INIREG+$0E
101E                    TI405   EQU INIREG+$1E
1022                    TMSK1   EQU INIREG+$22
1023                    TFLG1   EQU INIREG+$23
102B                    BAUD    EQU INIREG+$2B
102C                    SCCR1   EQU INIREG+$2C
102D                    SCCR2   EQU INIREG+$2D
102E                    SCSR    EQU INIREG+$2E
102F                    SCDR    EQU INIREG+$2F
103C                    HPRIO   EQU INIREG+$3C
103D                    INIT    EQU INIREG+$3D


                        ************************************
                        *       UBICACION DE MEMORIA       *
                        ************************************
E000                    INIROM  EQU $E000
FFD6                    FINROM  EQU INIROM+$1FD6
0000                    INIRAM  EQU $0000


                        ************************************
                        *       CARACTERES ESPECIALES      *
                        ************************************
003E                    PROMPT  EQU '>'         PROMPT DEL MONITOR
0020                    ESP     EQU $20         ESPACIO
000D                    CR      EQU $0D         REGRESO DE CARRO
000A                    LF      EQU $0A         AVANCE DE LINEA
0008                    BS      EQU $08         RETROCESO
0004                    EOT     EQU $04         FIN DE TEXTO


                        ************************************
                        *          TABLAS EN ROM           *
                        ************************************
F024                    TAB1    EQU INIROM+$1024    TABLA DE COMANDOS
F06A                    FT1     EQU TAB1+$46        FIN DE TABLA 1
F06A                    TAB2    EQU FT1             TABLA DE REGISTROS
F083                    FT2     EQU TAB2+$19        FIN DE TABLA 2
F083                    TAB3    EQU FT2             TABLA DE ERRORES
F083                    ERR1    EQU TAB3            ERROR DE SINTAXIS
F095                    ERR2    EQU ERR1+$12        COMANDO INVALIDO
F0A6                    ERR3    EQU ERR2+$11        NUMERO INVALIDO
F0B6                    ERR4    EQU ERR3+$10        RANGO INVALIDO
F0C5                    ERR5    EQU ERR4+$0F        DIFERENCIA EN PROGRAMA
F0DC                    ERR6    EQU ERR5+$17        ERROR DE LECTURA
F0ED                    ERR7    EQU ERR6+$11        ERROR: DIRECCION DE ROM
F105                    TAB4    EQU ERR7+$18        PANTALLA DE AYUDA
```

```
                        ***********************************
                        *      REGISTROS DEL MONITOR      *
                        ***********************************
0000                            ORG INIRAM
0000                    DIRT    RMB 1
0001                    DIR1A   RMB 1   DIR1 ALTA
0002                    DIR1B   RMB 1   DIR1 BAJA
0003                    DIR2A   RMB 1   DIR2 ALTA
0004                    DIR2B   RMB 1   DIR2 BAJA
0005                    DIR3A   RMB 1   DIR3 ALTA
0006                    DIR3B   RMB 1   DIR3 BAJA
0007                    CONTE   RMB 1   CONTENIDO DE UNA LOCALIDAD
0008                    LONG    RMB 1   LONGITUD DE COMANDO Y CONT. DE DATOS
0009                    CICLO   RMB 1   CONTADOR DE CICLO
000A                    VENT    RMB 1   CONTADOR DE VENTANA
000B                    ELEM    RMB 1   CONTADOR DE ELEMENTOS DESPLEGADOS
000C                    TEMP    RMB 1   REGISTRO TEMPORAL
000D                    LECT    RMB 1   CONTADOR DE LECTURAS

                        ***********************************
                        *             BANDERAS            *
                        ***********************************
000E                    VERIF   RMB 1           BANDERA VERIFICA
000F                    ECO     RMB 1           BANDERA ECO
0010                    BLANCO  RMB 1           BANDERA BLANCO
0011                    DOBLE   RMB 1           BANDERA DOBLE
0012                    PRIM    RMB 1           BANDERA PRIMERA

                        ***************************************
                        *   ESPACIOS RESERVADOS DE MEMORIA    *
                        ***************************************
0013                    FILA    RMB 16          FILA DE PARAMETROS
0015                    FIL1    EQU FILA+$02    INICIO DE DIR1 EN LA FILA
001A                    FIL2    EQU FILA+$07    INICIO DE DIR2 EN LA FILA
001F                    FIL3    EQU FILA+$0C    INICIO DE DIR3 EN LA FILA
0023                    FILREG  RMB 9           FILA DE REGISTROS DEL USUARIO
002C                    BUFLE   RMB 64          BUFFER DE LECTURA/ESCRITURA

                        ****************************
                        *  VECTORES DE INTERRUPCION  *
                        ****************************
006C                    VSCI    RMB 3   SCI
006F                    VSPI    RMB 3   SPI
0072                    VPAI    RMB 3   NIVEL DE ENTRADA AL ACUM. DE PULSOS
0075                    VPAOV   RMB 3   SATURACION DEL ACUM. DE PULSOS
0078                    VTO     RMB 3   SATURACION DEL TEMPORIZADOR
007B                    VI4O5   RMB 3   CAPT. POR ENT. 4/SAL. POR COMP. 5
007E                    VOC4    RMB 3   SALIDA POR COMPARACION 4
0081                    VOC3    RMB 3   SALIDA POR COMPARACION 3
0084                    VOC2    RMB 3   SALIDA POR COMPARACION 2
0087                    VOC1    RMB 3   SALIDA POR COMPARACION 1
008A                    VIC3    RMB 3   CAPTURA POR ENTRADA 3
008D                    VIC2    RMB 3   CAPTURA POR ENTRADA 2
0090                    VIC1    RMB 3   CAPTURA POR ENTRADA 1
```

```
0093                    VRTI    RMB 3   INTERRUPCION DE TIEMPO REAL
0096                    VIRQ    RMB 3   INTERRUPCION ENMASCARABLE
0099                    VXIRQ   RMB 3   INTERRUPCION NO ENMASCARABLE
009C                    VSWI    RMB 3   INTERRUPCION POR PROGRAMA
009F                    VIOT    RMB 3   DETECCION DE CODIGO ILEGAL
00A2                    VNOCOP  RMB 3   FALLA DEL COP
00A5                    VCME    RMB 3   FALLA DEL RELOJ


                        ********************************
                        *    INICIO DE PROGRAMA EN ROM    *
                        ********************************
E000                            ORG INIROM

                        ************************************************
                        *                SUBRUTINA ASCII-HEXA          *
                        *  Convierte A de código ASCII a hexadecimal.  *
                        *  Si el resultado no está dentro del rango    *
                        *  [$ 00, $ 0F], envía el mensaje:   "NUMERO   *
                        *  INVALIDO".                                  *
                        ************************************************
E000 84 7F              ASHE:   ANDA #$7F
E002 80 30                      SUBA #$30
E004 81 17                      CMPA #$17
E006 25 0B                      BLO ASC2        SALTA SI A < $ 17
E008 18 CE F0 A6        ASC1:   LDY #ERR3       "NUMERO INVALIDO"
E00C 8D 4F                      BSR CRLF
E00E 8D 58                      BSR MENS
E010 7E E1 82                   JMP INI
E013 81 0A              ASC2:   CMPA #$0A
E015 24 01                      BHS ASC3        SALTA SI A >= $ 0A
E017 39                         RTS
E018 81 10              ASC3:   CMPA #$10
E01A 23 EC                      BLS ASC1        SALTA SI A <= $ 10
E01C 80 07                      SUBA #$07
E01E 39                         RTS

                        ************************************************
                        *              SUBRUTINA ASCIINUM              *
                        *  Convierte dos octetos ASCII direccionados   *
                        *  por IY en un octeto hexadecimal y almacena  *
                        *  el resultado en A.  Utiliza B para guardar  *
                        *  el cuarteto alto.                           *
                        ************************************************
E01F 37                 ASNU:   PSHB            SALVA B
E020 18 A6 00                   LDAA 0,Y        CARGA EL PRIMER CODIGO ASCII
E023 8D DB                      BSR ASHE        LO CONVIERTE EN HEXADECIMAL
E025 48                         LSLA            CORRIMIENTO DEL CUARTETO BAJO
E026 48                         LSLA            AL ALTO EN A
E027 48                         LSLA
E028 48                         LSLA
E029 16                         TAB             ALMACENA CUARTETO ALTO
E02A 18 08                      INY
E02C 18 A6 00                   LDAA 0,Y        CARGA EL SEGUNDO CODIGO ASCII
```

```
E02F 8D CF              BSR ASHE        LO CONVIERTE EN HEXADECIMAL
E031 1B                 ABA             UNE CUARTETO ALTO CON BAJO
E032 33                 PULB
E033 39                 RTS

                        ****************************************
                        *          SUBRUTINA HEXA-ASCII        *
                        *  Convierte A de hexadecimal a código ASCII  *
                        *  y almacena el resultado en A.       *
                        ****************************************
E034 8B 30              HEAS:   ADDA #$30
E036 81 39                      CMPA #$39
E038 23 02                      BLS HEX1        SALTA SI A <= $ 39
E03A 8B 07                      ADDA #$07
E03C 39                 HEX1:   RTS

                        ****************************************
                        *            SUBRUTINA NUMASCII        *
                        *  Convierte el valor hexadecimal de A en sus  *
                        *  dos octetos ASCII correspondientes y los  *
                        *  envía por el puerto serie.  Utiliza B para  *
                        *  almacenar el cuarteto bajo.         *
                        ****************************************
E03D 37                 NUAS:   PSHB            SALVA B
E03E 16                         TAB
E03F C4 0F                      ANDB #$0F       ALMACENA CUARTETO BAJO
E041 44                         LSRA            CORRIMIENTO DEL CUARTETO ALTO
E042 44                         LSRA            AL BAJO EN A
E043 44                         LSRA
E044 44                         LSRA
E045 8D ED                      BSR HEAS        LO CONVIERTE EN ASCII
E047 8D 07                      BSR ENVIA       ENVIA ASCII DEL CUARTETO ALTO
E049 17                         TBA
E04A 8D E8                      BSR HEAS        CONVIERTE CUARTETO BAJO EN
                                                ASCII
E04C 8D 02                      BSR ENVIA       LO ENVIA
E04E 33                         PULB
E04F 39                         RTS

                        ****************************************
                        *              SUBRUTINA ENVIA         *
                        *  Envía contenido de A por el puerto serie.  *
                        ****************************************
E050 3C                 ENVIA:  PSHX            SALVA IX
E051 CE 10 2E                   LDX #SCSR
E054 1F 00 80 FC                BRCLR 0,X $80 *     TDRE=0
E058 B7 10 2F                   STAA SCDR       ENVIA CONTENIDO DE A
E05B 38                         PULX
E05C 39                         RTS
                        ****************************************
                        *              SUBRUTINA CR-LF         *
                        *  Envía un regreso de carro (CR) y un avance  *
                        *  de línea (LF) por el puerto serie.  *
                        ****************************************
```

```
E05D 36          CRLF:   PSHA            SALVA A
E05E 86 0D               LDAA #CR
E060 8D EE               BSR ENVIA       ENVIA UN REGRESO DE CARRO
E062 86 0A               LDAA #LF
E064 8D EA               BSR ENVIA       ENVIA AVANCE DE LINEA
E066 32                  PULA
E067 39                  RTS

                 ********************************************
                 *              SUBRUTINA DESPMENS          *
                 *  Envía una cadena de caracteres ASCII por *
                 *  el puerto serie, direccionada por IY,    *
                 *  hasta encontrar el fin de texto ($ 04).  *
                 ********************************************

E068 18 A6 00    MENS:   LDAA 0,Y
E06B 18 08               INY             APUNTA AL SIGUIENTE DATO
E06D 81 04               CMPA #EOT
E06F 27 04               BEQ MEN1        SALTA SI ES FIN DE TEXTO
E071 8D DD               BSR ENVIA
E073 20 F3               BRA MENS
E075 39          MEN1:   RTS

                 ********************************************
                 *              SUBRUTINA DESPCONT          *
                 *  Despliega el contenido de la localidad  *
                 *  apuntada por IX y lo almacena en CONTE  *
                 ********************************************

E076 7D 00 10    CONT:   TST BLANCO
E079 27 04               BEQ CONT1       SALTA SI BANDERA APAGADA
E07B 86 20               LDAA #ESP
E07D 8D D1               BSR ENVIA       ENVIA UN ESPACIO
E07F A6 00       CONT1:  LDAA 0,X        CARGA CONTENIDO EN A
E081 97 07               STAA CONTE      RESPALDA CONTENIDO EN CONTE
E083 8D B8               BSR NUAS        CONVIERTE EN ASCII Y ENVIA
E085 39                  RTS

                 ********************************************
                 *              SUBRUTINA DESPLOC           *
                 *  Despliega la direccion apuntada por IX y *
                 *  su contenido.                           *
                 ********************************************

E086 3C          LOC:    PSHX
E087 32                  PULA            CARGA DIRECCION ALTA EN A
E088 8D B3               BSR NUAS        CONVIERTE EN ASCII Y ENVIA
E08A 32                  PULA            CARGA DIRECCION BAJA EN A
E08B 8D B0               BSR NUAS        CONVIERTE EN ASCII Y ENVIA
E08D 86 3A               LDAA #$3A
E08F 8D BF               BSR ENVIA       ENVIA ":"
E091 8D E3               BSR CONT        DESPLIEGA CONTENIDO DE LA DIR.
E093 39                  RTS
```

```
                        ********************************************
                        *                                          *
                        *              SUBRUTINA LEE               *
                        *  Espera por el caracter a ser leído, lo  *
                        *  almacena en A y lo envía si la bandera  *
                        *  de ECO está encendida.                  *
                        ********************************************
E094 3C                 LEE:    PSHX                SALVA IX
E095 CE 10 2E                   LDX #SCSR
E098 1F 00 20 FC                BRCLR 0,X $20 *     RDRF=0
E09C B6 10 2F                   LDAA SCDR
E09F 84 7F                      ANDA #$7F
E0A1 7D 00 0F                   TST ECO             PRUEBA BANDERA DE ECO
E0A4 27 02                      BEQ LEE1            SALTA SI ESTA APAGADA
E0A6 8D A8                      BSR ENVIA           ECO
E0A8 38                 LEE1:   PULX
E0A9 39                         RTS

                        ********************************************
                        *        SUBRUTINA LEE DOS OCTETOS         *
                        *  Lee dos caracteres ASCII, los guarda en las *
                        *  dos primeras localidades de la fila y los  *
                        *  convierte en un número hexadecimal que     *
                        *  almacena en A.                             *
                        ********************************************
E0AA 18 3C              LEER2:  PSHY                SALVA IY
E0AC 8D E6                      BSR LEE             LEE PRIMER CARACTER
E0AE 97 13                      STAA FILA           LO ALMACENA EN INICIO DE FILA
E0B0 8D E2                      BSR LEE             LEE SEGUNDO CARACTER
E0B2 97 14                      STAA FILA+1         LO ALMACENA EN FILA+1
E0B4 18 CE 00 13                LDY #FILA           CONVIERTE LOS DOS CARACTERES
E0B8 BD E0 1F                   JSR ASNU            ASCII EN HEXADECIMAL
E0BB 18 38                      PULY
E0BD 39                         RTS

                        ********************************************
                        *                SUBRUTINA DIR1            *
                        *  Valida la primera dirección de la fila, la  *
                        *  almacena en DIR1A, DIR1B  y en IX.        *
                        ********************************************
E0BE 18 CE 00 15        DIR1:   LDY #FIL1           INICIO DIR1 EN LA FILA
E0C2 BD E0 1F                   JSR ASNU            VALIDA LA PARTE ALTA
E0C5 97 01                      STAA DIR1A          ALMACENA DIR1 ALTA
E0C7 18 08                      INY
E0C9 BD E0 1F                   JSR ASNU            VALIDA LA PARTE BAJA
E0CC 97 02                      STAA DIR1B          ALMACENA DIR1 BAJA
E0CE DE 01                      LDX DIR1A           ALMACENA DIR1 EN IX
E0D0 39                         RTS
                        ********************************************
                        *                SUBRUTINA DIR2            *
                        *  Valida la segunda dirección de la fila, la  *
                        *  almacena en DIR2A, DIR2B. Si esta dirección *
                        *  es menor que la primera, envía el mensaje:  *
                        *  "RANGO INVALIDO".                           *
                        ********************************************
```

```
E0D1 18 CE 00 1A        DIR2:   LDY #FIL2       INICIO DIR2 EN LA FILA
E0D5 BD E0 1F                   JSR ASNU        VALIDA LA PARTE ALTA
E0D8 97 03                      STAA DIR2A      ALMACENA DIR2 ALTA
E0DA 18 08                      INY
E0DC BD E0 1F                   JSR ASNU        VALIDA LA PARTE BAJA
E0DF 97 04                      STAA DIR2B      ALAMACENA DIR2 BAJA
E0E1 9C 03                      CPX DIR2A
E0E3 22 01                      BHI DDOS        SALTA SI DIR1 > DIR2
E0E5 39                 DTRE:   RTS
E0E6 18 CE F0 B6        DDOS:   LDY #ERR4       "RANGO INVALIDO"
E0EA BD E0 5D                   JSR CRLF
E0ED BD E0 68                   JSR MENS
E0F0 7E E1 82                   JMP INI

                        ********************************************
                        *          SUBRUTINA DESPLIEGA REGISTROS   *
                        *  Despliega los registros del usuario y su*
                        *  contenido. Almacena la dirección del nombre*
                        *  de registro que va a desplegar en IY y la*
                        *  dirección de su contenido en IX.        *
                        ********************************************

E0F3 18 CE F0 6A        REG:    LDY #TAB2       INICIO NOMBRES DE REGISTROS
E0F7 CE 00 23                   LDX #FILREG     INICIO FILA DE REGISTROS
E0FA BD E0 68           REG1:   JSR MENS        ENVIA NOMBRE DEL REGISTRO
E0FD BD E0 76                   JSR CONT        ENVIA CONTENIDO DEL REGISTRO
E100 8C 00 26                   CPX #FILREG+3
E103 25 07                      BLO REG2
E105 7F 00 10                   CLR BLANCO      APAGA BANDERA
E108 08                         INX
E109 BD E0 76                   JSR CONT        ENVIA CONTENIDO SIN ESPACIO
E10C 86 20              REG2:   LDAA #ESP
E10E BD E0 50                   JSR ENVIA       ENVIA "ESPACIO"
E111 08                         INX
E112 7C 00 10                   INC BLANCO      ENCIENDE BANDERA
E115 18 8C F0 83                CPY #FT2
E119 25 DF                      BLO REG1        SALTA SI NO ES FIN DE TABLA 2
E11B BD E0 5D                   JSR CRLF
E11E 39                         RTS

                        ********************************************
                        *          SUBRUTINA RESPALDA              *
                        *  Almacena el contenido actual de los     *
                        *  registros en la fila de registros del   *
                        *  usuario.                                *
                        ********************************************

E11F 36                 RESP:   PSHA
E120 DD 23                      STD FILREG      ALMACENA A Y B EN FILA
E122 07                         TPA
E123 97 25                      STAA FILREG+2   ALMACENA CCR EN FILA
E125 DF 26                      STX FILREG+3    ALMACENA IX EN FILA
E127 18 DF 28                   STY FILREG+5    ALMACENA IY EN FILA
E12A 9F 2A                      STS FILREG+7    ALMACENA SP EN FILA
E12C 32                         PULA
E12D 39                         RTS
```

```
                        ********************************************
                        *                                          *
                        *            SUBRUTINA RECUPERA            *
                        *  Almacena el contenido actual de la fila de *
                        *  registros del usuario en los registros del *
                        *  programa del mismo.                     *
                        ********************************************
E12E 96 25              RECU:   LDAA FILREG+2   CARGA CCR DE FILA
E130 06                         TAP
E131 DC 23                      LDD FILREG      CARGA A Y B DE FILA
E133 DE 26                      LDX FILREG+3    CARGA IX DE FILA
E135 18 DE 28                   LDY FILREG+5    CARGA IY DE FILA
E138 39                         RTS


                        ********************************************
                        *                                          *
                        *          SUBRUTINA CORRIMIENTO           *
                        *  Convierte el dato leído en A en hexadecimal. *
                        *  En CONTE se encuentra el contenido original *
                        *  de la localidad que se desea modificar.  En *
                        *  el registro B se hace un corrimiento del *
                        *  cuarteto bajo al alto y en D se recorren *
                        *  todos los cuartetos para dejar finalmente el *
                        *  dato actual en A y en CONTE.            *
                        ********************************************
E139 37                 CORRI:  PSHB            SALVA B
E13A BD E0 00                   JSR ASHE
E13D 16                         TAB             ALMACENA DATO ACTUAL EN B
E13E 96 07                      LDAA CONTE      CARGA DATO ORIGINAL
E140 58                         ASLB            CORRIMIENTO DEL CUARTETO
E141 58                         ASLB            BAJO AL CUARTETO ALTO
E142 58                         ASLB
E143 58                         ASLB
E144 05                         ASLD
E145 05                         ASLD
E146 05                         ASLD
E147 05                         ASLD
E148 97 07                      STAA CONTE      CARGA DATO ACTUALIZADO
E14A 33                         PULB
E14B 39                         RTS

                        ********************************************
                        *       SUBRUTINA CORRIMIENTO DOBLE        *
                        ********************************************
E14C 18 3C              CORR2:  PSHY            SALVA IY
E14E BD E0 00                   JSR ASHE
E151 36                         PSHA            SALVA DATO ACTUAL EN PILA
E152 DC 07                      LDD CONTE       CARGA DATO ORIGINAL EN D
E154 05                         ASLD
E155 05                         ASLD
E156 05                         ASLD
E157 05                         ASLD            RECORRE UN CUARTETO A LA IZQ.
E158 18 8F                      XGDY            DATO RECORRIDO EN IY
E15A 33                         PULB            RECUPERA DATO ACTUAL EN B
E15B 18 3A                      ABY             AGREGA DATO ACTUAL A IY
```

```
E15D 18 8F                      XGDY            DATO ACTUALIZADO EN D
E15F DD 07                      STD CONTE       DATO ACTUAL EN CONTE Y LONG
E161 18 38                      PULY
E163 39                         RTS

                        ************************************************
                        *** INICIALIZACION DE REGISTROS DE CONTROL ***
                        ************************************************
E164 86 25              INREG:  LDAA #$25
E166 B7 10 3C                   STAA HPRIO      MODO EXPANDIDO MULTIPLEXADO
E169 86 30                      LDAA #$30
E16B B7 10 2B                   STAA BAUD       PARA VELOCIDAD DE 9600 BAUDS
E16E 86 3A                      LDAA #$3A
E170 B7 10 09                   STAA DDRD       SCI, SPI EN MODO MAESTRO
E173 CE 00 6C                   LDX #VSCI
E176 86 7E                      LDAA #$7E       INICIALIZA VECTORES
E178 A7 00              VEC1:   STAA 0,X        DEL USUARIO CON
E17A 08                         INX             CODIGO DE JMP
E17B 08                         INX
E17C 08                         INX
E17D 8C 00 A8                   CPX #VSCI+60
E180 25 F6                      BLO VEC1        SALTA SI NO ES ULTIMO VECTOR
E182 8E 00 D0           INI:    LDS #$00D0      INICIALIZA APUNTADOR DE PILA
E185 86 01                      LDAA #$01
E187 B7 10 3D                   STAA INIT       RAM EN $0000
E18A 7F 10 2C                   CLR SCCR1       MODO UART, 8 BITES
E18D 86 0C                      LDAA #$0C
E18F B7 10 2D                   STAA SCCR2      ACTIVA TE Y RE
E192 7F 00 10                   CLR BLANCO
E195 7C 00 10                   INC BLANCO      ENCIENDE BANDERA
E198 7F 00 0F                   CLR ECO
E19B 7C 00 0F                   INC ECO         ENCIENDE BANDERA
E19E 7F 00 0E                   CLR VERIF       APAGA BANDERA
E1A1 CE 00 13                   LDX #FILA
E1A4 6F 00              BORRA:  CLR 0,X         BORRA FILA DE PARAMETROS
E1A6 08                         INX
E1A7 8C 00 23                   CPX #FILA+16
E1AA 25 F8                      BLO BORRA

                        ************************************************
                        *              LECTURA DE PARAMETROS           *
                        *  Lee caracteres y los almacena en la fila    *
                        *  de parámetros hasta encontrar un CR.        *
                        *  Incrementa un contador (LONG) por cada      *
                        *  caracter leído, excepto si se trata de CR   *
                        *  o BS.                                       *
                        ************************************************
E1AC 7F 00 08                   CLR LONG
E1AF BD E0 5D                   JSR CRLF
E1B2 86 3E                      LDAA #PROMPT
E1B4 BD E0 50                   JSR ENVIA       ENVIA ">"
E1B7 18 CE 00 13                LDY #FILA       INICIO FILA DE PARAMETROS
E1BB BD E0 94           UNO:    JSR LEE
E1BE 81 08                      CMPA #BS
```

```
E1C0 26 07              BNE DOS         SALTA SI A <> BS
E1C2 18 09              DEY
E1C4 7A 00 08           DEC LONG
E1C7 20 F2              BRA UNO
E1C9 81 0D      DOS:    CMPA #CR
E1CB 27 0A              BEQ SEL         SALTA SI A = CR
E1CD 18 A7 00           STAA 0,Y        ALMACENA DATO LEIDO EN FILA
E1D0 18 08              INY
E1D2 7C 00 08           INC LONG        INCREMENTA CONT. DE CARACTERES
E1D5 20 E4              BRA UNO
```

```
*****************************************************
*                SELECCION DE COMANDOS              *
*  Busca el comando leído en la Tabla 1, checa      *
*  su longitud máxima y salta a la rutina de        *
*  dicho comando. Si el primer caracter leído       *
*  no es un comando, envía el mensaje: "COMANDO     *
*  INVALIDO". Si los caracteres leídos exceden      *
*  la longitud máxima del comando, despliega el     *
*  mensaje: "ERROR DE SINTAXIS".                    *
*****************************************************
```

```
E1D7 18 CE F0 24 SEL:   LDY #TAB1       INICIO TABLA DE COMANDOS
E1DB C6 05              LDAB #$05       INCREMENTA CUATRO LOCALIDADES
E1DD 18 A6 00   TRES:   LDAA 0,Y
E1E0 91 13              CMPA FILA
E1E2 26 10              BNE CUATRO      SALTA SI COMANDOS <>
E1E4 18 A6 01           LDAA 1,Y
E1E7 91 08              CMPA LONG
E1E9 2C 06              BGE SEIS        SALTA SI A >= LONG
E1EB 18 CE F0 83        LDY #ERR1       "ERROR DE SINTAXIS"
E1EF 20 0F              BRA CINCO
E1F1 18 6E 02   SEIS:   JMP 2,Y         SALTA A LA RUTINA
E1F4 18 3A      CUATRO: ABY             APUNTA AL SIGUIENTE COMANDO
E1F6 18 8C F0 6A        CPY #FT1
E1FA 25 E1              BLO TRES        SALTA SI NO ES FIN DE TABLA 1
E1FC 18 CE F0 95        LDY #ERR2       "COMANDO INVALIDO"
E200 BD E0 5D   CINCO:  JSR CRLF
E203 BD E0 68           JSR MENS
E206 7E E1 82           JMP INI
```

```
*****************************************************
*                      RUTINA E                     *
*  Despliega y permite modificar el contenido de la *
*  localidad apuntada por IX. Con ESPACIO muestra   *
*  la siguiente localidad y con CR termina. Si DIR1 *
*  pertenece a una localidad de ROM, despliega el   *
*  mensaje de error: "ERROR: DIRECCION DE ROM. En   *
*  la variable CONTE se almacena el valor original  *
*  de la localidad examinada cuando utiliza LOC, y  *
*  contiene el valor actual después de realizar la  *
*  subrutina CORRI. En ELEM se contabilizan las     *
*  localidades desplegadas y VENT permite un máximo *
*  de dos caracteres en la ventana.                 *
*****************************************************
```

```
E209 96 08      RUTE:   LDAA LONG
```

```
E20B 81 06                  CMPA #$06
E20D 27 06                  BEQ E7          SALTA SI HAY DIR1
E20F 18 CE F0 83            LDY #ERR1       "ERROR DE SINTAXIS"
E213 20 0C                  BRA E8
E215 BD E0 BE         E7:   JSR DIR1
E218 8C E0 00               CPX #INIROM
E21B 25 0D                  BLO E9          SALTA SI NO ES DIR. DE ROM
E21D 18 CE F0 ED            LDY #ERR7       "ERROR: DIRECCION DE ROM"
E221 BD E0 5D         E8:   JSR CRLF
E224 BD E0 68               JSR MENS
E227 7E E1 82               JMP INI
E22A BD E0 5D         E9:   JSR CRLF
E22D 7F 00 0B               CLR ELEM        BORRA CONTADOR
E230 7F 00 0F               CLR ECO         APAGA BANDERA DE ECO
E233 7C 00 0B         E2:   INC ELEM        CONTADOR DE LOCS. DEPLEGADAS
E236 96 0B                  LDAA ELEM
E238 81 07                  CMPA #$07
E23A 25 06                  BLO E6          SALTA SI NO ES FIN DE RENGLON
E23C 7F 00 0B               CLR ELEM
E23F BD E0 5D               JSR CRLF
E242 BD E0 86         E6:   JSR LOC
E245 BD E0 94         E5:   JSR LEE
E248 81 20                  CMPA #ESP
E24A 26 0D                  BNE E1          SALTA SI A <> ESP
E24C BD E0 50               JSR ENVIA       ECO
E24F 7F 00 0A               CLR VENT        BORRA CONTADOR DE VENTANA
E252 96 07                  LDAA CONTE      CARGA CONTENIDO ACTUAL
E254 A7 00                  STAA 0,X        REGRESA VALOR ACTUALIZADO
E256 08                     INX
E257 20 DA                  BRA E2
E259 81 0D            E1:   CMPA #CR
E25B 26 0D                  BNE E3          SALTA SI A <> CR
E25D BD E0 50               JSR ENVIA       ECO
E260 7F 00 0A               CLR VENT
E263 96 07                  LDAA CONTE
E265 A7 00                  STAA 0,X        CARGA DATO ACTUALIZADO
E267 7E E1 82               JMP INI
E26A 7C 00 0A         E3:   INC VENT
E26D 36                     PSHA            SALVA A
E26E 96 0A                  LDAA VENT
E270 81 03                  CMPA #$03
E272 25 14                  BLO E4          SALTA SI CONTADOR < $ 03
E274 7A 00 0A               DEC VENT
E277 86 08                  LDAA #BS
E279 BD E0 50               JSR ENVIA       ENVIA "RETROCESO"
E27C BD E0 50               JSR ENVIA
E27F 32                     PULA
E280 BD E1 39               JSR CORRI
E283 BD E0 3D               JSR NUAS
E286 20 BD                  BRA E5
E288 32               E4:   PULA
E289 BD E0 50               JSR ENVIA       ECO
E28C BD E1 39               JSR CORRI
E28F 20 B4                  BRA E5
```

```
                        ********************************************
                        *                RUTINA D                  *
                        *  Despliega el contenido de las localidades *
                        *  desde DIR1 hasta DIR2. Si no existe DIR2  *
                        *  se muestran cuatro renglones completos.   *
                        ********************************************
E291 96 08              RUTD:   LDAA LONG
E293 81 06                      CMPA #$06
E295 26 0B                      BNE D5          SALTA SI HAY DIR2
E297 BD E0 BE                   JSR DIR1
E29A 84 F0                      ANDA #$F0       INICIO DE RENGLON A DESPLEGAR
E29C 97 02                      STAA DIR1B
E29E DE 01                      LDX DIR1A
E2A0 20 35                      BRA D2
E2A2 81 0B              D5:     CMPA #$0B
E2A4 27 0D                      BEQ D6          SALTA SI HAY DIR2
E2A6 18 CE F0 83                LDY #ERR1       "ERROR DE SINTAXIS"
E2AA BD E0 5D                   JSR CRLF
E2AD BD E0 68                   JSR MENS
E2B0 7E E1 82                   JMP INI
E2B3 BD E0 BE           D6:     JSR DIR1
E2B6 84 F0                      ANDA #$F0
E2B8 97 02                      STAA DIR1B
E2BA DE 01                      LDX DIR1A
E2BC BD E0 D1                   JSR DIR2
E2BF BD E0 5D           D1:     JSR CRLF
E2C2 BD E0 86                   JSR LOC
E2C5 08                 D3:     INX
E2C6 DF 01                      STX DIR1A       ALMACENA DIR1 SIGUIENTE
E2C8 9C 03                      CPX DIR2A
E2CA 25 14                      BLO D4          SALTA SI DIR1 < DIR2
E2CC 22 06                      BHI D7          SALTA SI DIR1 > DIR2
E2CE BD E0 76                   JSR CONT        DESPLIEGA ULTIMA LOCALIDAD
E2D1 BD E0 5D                   JSR CRLF
E2D4 7E E1 82           D7:     JMP INI
E2D7 CC 00 3F           D2:     LDD #$003F      PARA DESPLEGAR 4 RENGLONES
E2DA D3 01                      ADDD DIR1A      D = DIR1 + $ 003F
E2DC DD 03                      STD DIR2A       DIR2 = D
E2DE 20 DF                      BRA D1
E2E0 BD E0 76           D4:     JSR CONT
E2E3 12 02 0F 02                BRSET DIR1B $0F D8
E2E7 20 DC                      BRA D3          SALTA SI NO ES FIN DE RENGLON
E2E9 08                 D8:     INX
E2EA 20 D3                      BRA D1

                        ********************************************
                        *                RUTINA A                  *
                        *  Despliega la pantalla de ayuda donde se   *
                        *  muestra la sintaxis y función de cada     *
                        *  comando.                                  *
                        ********************************************
E2EC 18 CE F1 05        RUTA:   LDY #TAB4       INICIO PANTALLA DE AYUDA
E2F0 BD E0 68                   JSR MENS
E2F3 7E E1 82                   JMP INI
```

```
                        ********************************************
                        *                  RUTINA V                *
                        *  Compara un programa de memoria contra uno *
                        *  que se lee. Enciende una bandera y realiza *
                        *  la rutina L. Si los datos comparados no  *
                        *  coinciden, se envía el mensaje: "DIFERENCIA *
                        *  EN PROGRAMA".                           *
                        ********************************************

E2F6 7F 00 0E           RUTV:   CLR VERIF
E2F9 7C 00 0E                   INC VERIF       ENCIENDE BANDERA DE VERIFICA

                        ********************************************
                        *                  RUTINA L                *
                        *  Lee un programa por bloques y lo carga en *
                        *  la dirección de memoria del bloque más el *
                        *  desplazamiento indicado. Cuando la bandera *
                        *  de verifica está encendida, sólo lo compara. *
                        *  Si la paridad del bloque leído es diferente *
                        *  a la calculada, envía el mensaje: "ERROR DE *
                        *  LECTURA". Si DIR + NUM pertenece a una  *
                        *  localidad de ROM, envía el mensaje: "ERROR: *
                        *  DIRECCION DE ROM". LONG se utiliza como *
                        *  contador de datos y B para el cálculo de *
                        *  paridad. El número hexadecimal se almacena *
                        *  en DIR2A, DIR2B y la dirección del bloque *
                        *  en DIR1A, DIR1B. CICLO se inicializa con el *
                        *  valor del contador de datos.            *
                        ********************************************

E2FC 96 08              RUTL:   LDAA LONG
E2FE 81 06                      CMPA #$06
E300 27 07                      BEQ L15         SALTA SI HAY NUM
E302 18 CE F0 83                LDY #ERR1       "ERROR DE SINTAXIS"
E306 7E E3 E6                   JMP L10
E309 7F 00 0F           L15:    CLR ECO         APAGA BANDERA DE ECO
E30C 7F 00 0D                   CLR LECT        BORRA CONTADOR
E30F 18 CE 00 15                LDY #FIL1       INICIO NUM EN LA FILA
E313 BD E0 1F                   JSR ASNU        VALIDA LA PARTE ALTA DE NUM
E316 97 03                      STAA DIR2A
E318 18 08                      INY
E31A BD E0 1F                   JSR ASNU        VALIDA LA PARTE BAJA DE NUM
E31D 97 04                      STAA DIR2B
E31F 5F                 L4:     CLRB            CALCULO DE PARIDAD EN B
E320 BD E0 94                   JSR LEE         LEE INICIO DE BLOQUE
E323 81 53                      CMPA #$53
E325 26 2F                      BNE L2          SALTA SI A <> "S"
E327 7F 00 0D                   CLR LECT
E32A BD E0 94                   JSR LEE         LEE TIPO DE BLOQUE
E32D 81 31                      CMPA #$31
E32F 27 31                      BEQ L1          SALTA SI ES BLOQUE TIPO 1
E331 81 39                      CMPA #$39
E333 27 12                      BEQ L9          SALTA SI ES BLOQUE TIPO 9
E335 BD E0 AA                   JSR LEER2       LEE CONTADOR DE DATOS
E338 97 08                      STAA LONG       ALMACENA CONTADOR DE DATOS
E33A BD E0 AA           L3:     JSR LEER2       ELIMINA BLOQUES QUE
```

```
E33D 7A 00 08            DEC LONG        NO SEAN TIPO 1 6 9
E340 7D 00 08            TST LONG
E343 26 F5               BNE L3
E345 20 D8               BRA L4
E347 BD E0 AA      L9:   JSR LEER2       LEE Y DESECHA: CONTADOR
E34A BD E0 AA            JSR LEER2       DIR. ALTA
E34D BD E0 AA            JSR LEER2       DIR. BAJA
E350 BD E0 AA            JSR LEER2       PARIDAD
E353 7E E1 82            JMP INI
E356 7C 00 0D      L2:   INC LECT
E359 96 0D               LDAA LECT
E35B 81 04               CMPA #$04
E35D 26 C0               BNE L4
E35F 7E E1 82            JMP INI
E362 CE 00 01      L1:   LDX #DIR1A
E365 BD E0 AA            JSR LEER2       LEE CONTADOR DE DATOS
E368 97 08               STAA LONG       ALMACENA CONTADOR DE DATOS
E36A 1B                  ABA             CALCULA PARIDAD
E36B 16                  TAB
E36C BD E0 AA            JSR LEER2
E36F 97 01               STAA DIR1A      ALMACENA DIRECCION ALTA
E371 1B                  ABA             CALCULA PARIDAD
E372 16                  TAB
E373 7A 00 08            DEC LONG
E376 BD E0 AA            JSR LEER2
E379 97 02               STAA DIR1B      ALMACENA DIRECCION BAJA
E37B 1B                  ABA             CALCULA PARIDAD
E37C 16                  TAB
E37D 7A 00 08            DEC LONG
E380 37                  PSHB            SALVA PARIDAD
E381 DC 03               LDD DIR2A
E383 E3 00               ADDD 0,X        DIR + NUM
E385 1A 83 E0 00         CPD #INIROM
E389 25 06               BLO L5          SALTA SI DIR+NUM NO ES DE ROM
E38B 18 CE F0 ED         LDY #ERR7       "ERROR: DIRECCION DE ROM"
E38F 20 55               BRA L10
E391 8F            L5:   XGDX            ALMACENA DIR+NUM EN IX
E392 33                  PULB            RECUPERA PARIDAD
E393 18 CE 00 2C         LDY #BUFLE      INICIO BUFFER DE LECT. Y ESC.
E397 96 08               LDAA LONG
E399 97 09               STAA CICLO
E39B 7A 00 09      L7:   DEC CICLO
E39E 7D 00 09            TST CICLO
E3A1 27 0C               BEQ L6          SALTA A FIN DE CICLO
E3A3 BD E0 AA            JSR LEER2       CICLO DE LECTURA DE DATOS
E3A6 18 A7 00            STAA 0,Y        ALMACENA DATO EN BUFFER
E3A9 18 08               INY             APUNTA AL SIGUIENTE DATO
E3AB 1B                  ABA             CALCULA PARIDAD
E3AC 16                  TAB
E3AD 20 EC               BRA L7
E3AF 53            L6:   COMB            COMPLEMENTO A UNO
E3B0 BD E0 AA            JSR LEER2
E3B3 11                  CBA
E3B4 27 06               BEQ L8          SALTA SI PARIDAD CORRECTA
```

```
E3B6 18 CE F0 DC          LDY #ERR6       "ERROR DE LECTURA"
E3BA 20 2A                BRA L10
E3BC 18 CE 00 2C    L8:   LDY #BUFLE
E3C0 96 08                LDAA LONG
E3C2 97 09                STAA CICLO
E3C4 7A 00 09       L13:  DEC CICLO
E3C7 7D 00 09             TST CICLO
E3CA 26 03                BNE L14
E3CC 7E E3 1F             JMP L4          SALTA SI ES FIN DE CICLO
E3CF 18 A6 00       L14:  LDAA 0,Y        CARGA DATO DEL BUFFER
E3D2 7D 00 0E             TST VERIF       PRUEBA BANDERA DE VERIFICA Y
E3D5 26 07                BNE L12         SALTA SI ESTA ENCENDIDA
E3D7 A7 00                STAA 0,X        ALMACENA DATO EN MEMORIA
E3D9 08             L11:  INX
E3DA 18 08                INY
E3DC 20 E6                BRA L13
E3DE A1 00          L12:  CMPA 0,X
E3E0 27 F7                BEQ L11         SALTA SI DATOS IGUALES
E3E2 18 CE F0 C5          LDY #ERR5       "DIFERENCIA EN PROGRAMA"
E3E6 BD E0 5D       L10:  JSR CRLF
E3E9 BD E0 68             JSR MENS
E3EC 7E E1 82             JMP INI

                    ********************************************
                    *                RUTINA I                  *
                    *  Envía un programa residente en memoria  *
                    *  en bloques a través del puerto serie.   *
                    *  LONG se utiliza como contador de datos  *
                    *  y B para el cálculo de paridad. En IY   *
                    *  se almacena la dirección del buffer de  *
                    *  LECTURA/ESCRITURA.                      *
                    ********************************************

E3EF 96 08          RUTI: LDAA LONG
E3F1 81 0B                CMPA #$0B
E3F3 27 0D                BEQ I7            SALTA SI HAY DIR2
E3F5 18 CE F0 83          LDY #ERR1         "ERROR DE SINTAXIS"
E3F9 BD E0 5D             JSR CRLF
E3FC BD E0 68             JSR MENS
E3FF 7E E1 82             JMP INI
E402 BD E0 BE       I7:   JSR DIR1
E405 BD E0 D1             JSR DIR2
E408 18 CE 00 2C    I3:   LDY #BUFLE        BUFFER DE LECT. Y ESC.
E40C 7F 00 08             CLR LONG          BORRA CONTADOR DE DATOS
E40F 5F                   CLRB              CALCULO DE PARIDAD EN B
E410 DF 01                STX DIR1A
E412 96 01                LDAA DIR1A        CARGA DIR. ALTA
E414 8D 79                BSR ALM           ALMACENA DIR. ALTA EN BUFFER
E416 96 02                LDAA DIR1B        CARGA DIR. BAJA
E418 8D 75                BSR ALM           ALMACENA DIR. BAJA EN BUFFER
E41A A6 00          I6:   LDAA 0,X          CARGA DATO DE MEMORIA
E41C 8D 71                BSR ALM           ALMACENA DATO EN BUFFER
E41E 12 02 0F 09          BRSET DIR1B $0F I1
E422 9C 03                CPX DIR2A
E424 27 05                BEQ I1            SALTA SI DIR1 = DIR2
```

```
E426 08                        INX
E427 DF 01                     STX DIR1A
E429 20 EF                     BRA I6
E42B 7C 00 08          I1:     INC LONG
E42E DB 08                     ADDB LONG          CALCULA PARIDAD
E430 53                        COMB               COMPLEMENTO A UNO
E431 BD E0 5D                  JSR CRLF           ENVIA "CR", "LF"
E434 4F                        CLRA
E435 BD E0 50                  JSR ENVIA          ENVIA "NULL"
E438 86 53                     LDAA #$53
E43A BD E0 50                  JSR ENVIA          ENVIA "S"
E43D 86 31                     LDAA #$31
E43F BD E0 50                  JSR ENVIA          ENVIA TIPO DE BLOQUE "1"
E442 96 08                     LDAA LONG
E444 BD E0 3D                  JSR NUAS           ENVIA CONTADOR DE DATOS
E447 18 CE 00 2C               LDY #BUFLE
E44B 7A 00 08          I4:     DEC LONG
E44E 7D 00 08                  TST LONG
E451 27 0A                     BEQ I2             SALTA SI ES FIN DE CICLO
E453 18 A6 00                  LDAA 0,Y
E456 BD E0 3D                  JSR NUAS           ENVIA DATO
E459 18 08                     INY
E45B 20 EE                     BRA I4
E45D 17                I2:     TBA
E45E BD E0 3D                  JSR NUAS
E461 9C 03                     CPX DIR2A
E463 26 26                     BNE I5
E465 BD E0 5D                  JSR CRLF           ENVIA "CR", "LF"
E468 4F                        CLRA
E469 BD E0 50                  JSR ENVIA          ENVIA "NULL"
E46C 86 53                     LDAA #$53
E46E BD E0 50                  JSR ENVIA          ENVIA "S"
E471 86 39                     LDAA #$39
E473 BD E0 50                  JSR ENVIA          ENVIA TIPO DE BLOQUE "9"
E476 86 03                     LDAA #$03
E478 BD E0 3D                  JSR NUAS           ENVIA CONTADOR DE DATOS
E47B 4F                        CLRA
E47C BD E0 3D                  JSR NUAS           ENVIA DIR. ALTA
E47F 4F                        CLRA
E480 BD E0 3D                  JSR NUAS           ENVIA DIR. BAJA
E483 86 FC                     LDAA #$FC
E485 BD E0 3D                  JSR NUAS           ENVIA PARIDAD
E488 7E E1 82                  JMP INI
E48B 08                I5:     INX
E48C 7E E4 08                  JMP I3
                       * SUBRUTINA ALMACENA *
E48F 18 A7 00          ALM:    STAA 0,Y
E492 18 08                     INY
E494 1B                        ABA
E495 16                        TAB
E496 7C 00 08                  INC LONG
E499 39                        RTS
```

```
                    ********************************************
                    *                  RUTINA C                *
                    *  Corre un programa residente en RAM como si  *
                    *  fuera una subrutina.  Si DIR1 pertenece a una *
                    *  localidad de ROM, envía el mensaje de error:  *
                    *  "ERROR: DIRECCION DE ROM. En TEMP se almacena *
                    *  el contenido de DIR2 temporalmente y se     *
                    *  sustituye por un retorno de subrutina ($ 39).  *
                    *  Si no se proporciona DIR2, el usuario deberá  *
                    *  colocar el retorno de subrutina para marcar   *
                    *  el fin de programa.                         *
                    ********************************************
E49A 96 25          RUTC:   LDAA FILREG+2
E49C 8A 10                  ORAA #$10
E49E 97 25                  STAA FILREG+2
E4A0 96 08                  LDAA LONG
E4A2 81 06                  CMPA #$06
E4A4 26 15                  BNE C4          SALTA SI HAY DIR2
E4A6 BD E0 BE               JSR DIR1
E4A9 8C E0 00               CPX #INIROM
E4AC 24 1F                  BHS C5          SALTA SI ES DIRECCION DE ROM
E4AE 3C                     PSHX            SALVA DIR1
E4AF BD E1 2E               JSR RECU
E4B2 38                     PULX
E4B3 AD 00                  JSR 0,X         SALTA A SUBRUTINA EN "DIR1"
E4B5 BD E1 1F               JSR RESP
E4B8 7E E1 82               JMP INI
E4BB 81 0B          C4:     CMPA #$0B
E4BD 27 06                  BEQ C1          SALTA SI HAY DIR2
E4BF 18 CE F0 83            LDY #ERR1       "ERROR DE SINTAXIS"
E4C3 20 0C                  BRA C3
E4C5 BD E0 BE       C1:     JSR DIR1
E4C8 8C E0 00               CPX #INIROM
E4CB 25 0D                  BLO C2          SALTA SI NO ES DIR. DE ROM
E4CD 18 CE F0 ED    C5:     LDY #ERR7       "ERROR: DIRECCION DE ROM"
E4D1 BD E0 5D       C3:     JSR CRLF
E4D4 BD E0 68               JSR MENS
E4D7 7E E1 82               JMP INI
E4DA BD E0 D1       C2:     JSR DIR2
E4DD 18 DE 03               LDY DIR2A
E4E0 18 A6 00               LDAA 0,Y
E4E3 97 0C                  STAA TEMP       ALMACENA CONTENIDO DE DIR2
E4E5 86 39                  LDAA #$39
E4E7 18 A7 00               STAA 0,Y        CARGA $ 39 EN DIR2
E4EA 3C                     PSHX            SALVA DIR1
E4EB BD E1 2E               JSR RECU
E4EE 38                     PULX
E4EF AD 00                  JSR 0,X         SALTA A SUBRUTINA EN "DIR1"
E4F1 BD E1 1F               JSR RESP
E4F4 96 0C                  LDAA TEMP
E4F6 18 DE 03               LDY DIR2A
E4F9 18 A7 00               STAA 0,Y        REGRESA CONT. ORIGINAL A DIR2
E4FC 7E E1 82               JMP INI
```

```
                        ****************************************
                        *  Rutina de servicio de interrupción de *
                        *  trazado.                              *
                        ****************************************
E4FF 9F 2A      RUTRA:  STS FILREG+7    ALMACENA SP EN FILA
E501 32                 PULA
E502 97 25              STAA FILREG+2   ALMACENA CCR EN FILA
E504 33                 PULB
E505 32                 PULA
E506 DD 23              STD FILREG      ALMACENA A Y B EN FILA
E508 38                 PULX
E509 DF 26              STX FILREG+3    ALMACENA IX EN FILA
E50B 18 38              PULY
E50D 18 DF 28           STY FILREG+5    ALMACENA IY EN FILA
E510 38                 PULX
E511 DF 01              STX DIR1A       ALMACENA PC EN DIR1A
E513 BD E0 F3           JSR REG
E516 CE E5 4F           LDX #TRAZA
E519 3C                 PSHX            SALVA DIR DE RETORNO
E51A 18 3C              PSHY            SALVA IY
E51C DE 01              LDX DIR1A
E51E 3C                 PSHX            SALVA IX
E51F 36                 PSHA            SALVA A
E520 37                 PSHB            SALVA B
E521 07                 TPA
E522 36                 PSHA            SALVA CCR
E523 CE 10 22           LDX #$1022
E526 1D 01 F4           BCLR 1,X $F4    APAGA OCSF
E529 3B                 RTI

                        ****************************************************
                        *                    RUTINA T                      *
                        *  Traza la instrucción indicada por DIR1. Con     *
                        *  ESPACIO, traza la siguiente instrucción y       *
                        *  con CR termina el trazado.                      *
                        ****************************************************
E52A 96 08      RUTT:   LDAA LONG
E52C 81 06              CMPA #$06
E52E 27 0D              BEQ T1          SALTA SI HAY DIR1
E530 18 CE F0 83        LDY #ERR1       "ERROR DE SINTAXIS"
E534 BD E0 5D           JSR CRLF
E537 BD E0 68           JSR MENS
E53A 7E E1 82           JMP INI
E53D 7F 00 12   T1:     CLR PRIM
E540 7C 00 12           INC PRIM        ENCIENDE BANDERA
E543 86 7E              LDAA #$7E       ALMACENA CODIGO DE
E545 97 00              STAA DIRT       JMP EN DIRT
E547 CE E4 FF           LDX #RUTRA
E54A DF 7C              STX VI4O5+1
E54C BD E0 BE           JSR DIR1
E54F 0F         TRAZA:  SEI
E550 7F 10 22           CLR TMSK1
E553 96 25              LDAA FILREG+2
E555 84 EF              ANDA #$EF
```

```
E557 97 25                  STAA FILREG+2
E559 7D 00 12               TST PRIM
E55C 26 0E                  BNE T3          SALTA SI ES LA PRIMERA
E55E BD E0 94         T2:   JSR LEE
E561 81 20                  CMPA #ESP
E563 27 07                  BEQ T3
E565 81 0D                  CMPA #CR
E567 26 F5                  BNE T2
E569 7E E1 82               JMP INI
E56C 7F 00 12         T3:   CLR PRIM        BORRA BANDERA
E56F BD E0 5D               JSR CRLF
E572 BD E0 86               JSR LOC
E575 BD E0 5D               JSR CRLF
E578 C6 41                  LDAB #$41
E57A FE 10 0E               LDX TCNT
E57D 3A                     ABX
E57E FF 10 1E               STX TI405
E581 CE 10 22               LDX #$1022
E584 1D 01 F4               BCLR 1,X $F4    APAGA OC5F
E587 1C 00 0B               BSET 0,X $0B    ENCIENDA OC5I
E58A 0E                     CLI
E58B BD E1 2E               JSR RECU
E58E 7E 00 00               JMP DIRT        SALTA A RAM
```

```
*****************************************************
*                      RUTINA R                     *
*  Despliega todos los registros y su contenido     *
*  y los muestra de uno en uno para que se          *
*  puedan modificar.  Con ESPACIO despliega el      *
*  siguiente registro o con CR termina. VENT        *
*  permite un máximo de dos caracteres en la        *
*  ventana. IY almacena la dirección del nombre     *
*  de registro que va a desplegar e IX tiene la     *
*  dirección de su contenido.                       *
*****************************************************
```

```
E591 BD E0 5D         RUTR: JSR CRLF
E594 7F 00 0F               CLR ECO         APAGA BANDERA
E597 7F 00 11               CLR DOBLE       APAGA BANDERA
E59A BD E0 F3               JSR REG         DESPLIEGA REGS. DEL USUARIO
E59D 18 CE F0 6A            LDY #TAB2       INICIO NOMBRES DE REGISTROS
E5A1 CE 00 23               LDX #FILREG     INICIO FILA DE REGISTROS
E5A4 8C 00 26         R3:   CPX #FILREG+3
E5A7 25 03                  BLO R7
E5A9 7C 00 11               INC DOBLE
E5AC 18 8C F0 83      R7:   CPY #FT2
E5B0 25 03                  BLO R5          SALTA SI NO ES FIN DE TABLA 2
E5B2 7E E1 82               JMP INI
E5B5 BD E0 68         R5:   JSR MENS        DESPLIEGA NOMBRE DEL REGISTRO
E5B8 BD E0 76               JSR CONT        DESPLIEGA CONT. DEL REGISTRO
E5BB 7D 00 11               TST DOBLE
E5BE 27 12                  BEQ R4          SALTA SI ES REG. DE 1 OCTETO
E5C0 D6 07                  LDAB CONTE      ALMACENA PARTE ALTA
E5C2 08                     INX             APUNTA A LA PARTE BAJA
E5C3 7F 00 10               CLR BLANCO
```

```
E5C6 BD E0 76            JSR CONT        DESPLIEGA PARTE BAJA
E5C9 96 07               LDAA CONTE
E5CB 97 08               STAA LONG       ALMACENA PARTE BAJA EN LONG
E5CD D7 07               STAB CONTE      ALMACENA PARTE ALTA EN CONTE
E5CF 7C 00 10            INC BLANCO      ENCIENDE BANDERA
E5D2 BD E0 94      R4:   JSR LEE
E5D5 81 20               CMPA #ESP
E5D7 26 21               BNE R1          SALTA SI A <> ESP
E5D9 BD E0 50            JSR ENVIA       ECO
E5DC 7F 00 0A            CLR VENT        BORRA CONTADOR DE VENTANA
E5DF 7D 00 11            TST DOBLE
E5E2 26 06               BNE R8          SALTA SI ES REG. DE 2 OCTETOS
E5E4 96 07               LDAA CONTE
E5E6 A7 00               STAA 0,X        CARGA DATO ACTUALIZADO EN FILA
E5E8 20 0A               BRA R9
E5EA 96 08         R8:   LDAA LONG
E5EC A7 00               STAA 0,X        PARTE BAJA ACTUAL EN FILA
E5EE 09                  DEX
E5EF 96 07               LDAA CONTE
E5F1 A7 00               STAA 0,X        PARTE ALTA ACTUAL EN FILA
E5F3 08                  INX
E5F4 BD E0 5D      R9:   JSR CRLF
E5F7 08                  INX             APUNTA AL SIGUIENTE REGISTRO
E5F8 20 AA               BRA R3
E5FA 81 0D         R1:   CMPA #CR
E5FC 26 17               BNE R2          SALTA SI A <> CR
E5FE BD E0 50            JSR ENVIA       ECO
E601 7F 00 0A            CLR VENT
E604 7D 00 11            TST DOBLE
E607 27 05               BEQ R10         SALTA SI ES REG. DE 1 OCTETO
E609 96 08               LDAA LONG
E60B A7 00               STAA 0,X        PARTE BAJA ACTUAL EN FILA
E60D 09                  DEX             APUNTA A PARTE ALTA EN FILA
E60E 96 07         R10:  LDAA CONTE
E610 A7 00               STAA 0,X        CARGA DATO ACTUALIZADO EN FILA
E612 7E E1 82            JMP INI
E615 7C 00 0A      R2:   INC VENT
E618 36                  PSHA            SALVA A
E619 7D 00 11            TST DOBLE
E61C 27 04               BEQ R11         SALTA SI ES REG. DE 1 OCTETO
E61E C6 05               LDAB #05        AMPLIA VENTANA A 4 DIGITOS
E620 20 02               BRA R12
E622 C6 03         R11:  LDAB #03        PARA VENTANA DE 2 DIGITOS
E624 96 0A         R12:  LDAA VENT
E626 11                  CBA
E627 25 2D               BLO R6          SALTA SI VENT < B
E629 7A 00 0A            DEC VENT
E62C 86 08               LDAA #BS
E62E BD E0 50            JSR ENVIA       ENVIA "RETROCESO"
E631 BD E0 50            JSR ENVIA
E634 7D 00 11            TST DOBLE
E637 27 13               BEQ R13         SALTA SI ES REG. DE 1 OCTETO
E639 BD E0 50            JSR ENVIA       ENVIA "RETROCESO"
E63C BD E0 50            JSR ENVIA
```

```
E63F 32                  PULA
E640 BD E1 4C            JSR CORR2
E643 BD E0 3D            JSR NUAS        ENVIA PARTE ALTA
E646 17                  TBA
E647 BD E0 3D            JSR NUAS        ENVIA PARTE BAJA
E64A 20 86               BRA R4
E64C 32            R13:  PULA
E64D BD E1 39            JSR CORRI
E650 BD E0 3D            JSR NUAS
E653 7E E5 D2            JMP R4
E656 32            R6:   PULA
E657 BD E0 50            JSR ENVIA       ECO
E65A 7D 00 11            TST DOBLE
E65D 27 06               BEQ R14         SALTA SI ES REG. DE 1 OCTETO
E65F BD E1 4C            JSR CORR2
E662 7E E5 D2            JMP R4
E665 BD E1 39      R14:  JSR CORRI
E668 7E E5 D2            JMP R4

                         ****************************************
                         *                RUTINA M              *
                         *  Mueve un bloque de memoria definido por *
                         *  DIR1 hasta DIR2 a partir de la localidad *
                         *  indicada por DIR3. Si DIR3 pertenece a  *
                         *  una localidad de ROM, envía el mensaje: *
                         *  "ERROR: DIRECCION DE ROM".            *
                         ****************************************

E66B 96 08         RUTM: LDAA LONG
E66D 81 10               CMPA #$10
E66F 27 06               BEQ M4          SALTA SI HAY DIR3
E671 18 CE F0 83         LDY #ERR1       "ERROR DE SINTAXIS"
E675 20 17               BRA M5
E677 BD E0 BE      M4:   JSR DIR1
E67A BD E0 D1            JSR DIR2
E67D 18 CE 00 1F         LDY #FIL3
E681 BD E0 1F            JSR ASNU
E684 97 05               STAA DIR3A
E686 81 E0               CMPA #$E0
E688 25 0D               BLO M3          SALTA SI NO ES DIR. DE ROM
E68A 18 CE F0 ED         LDY #ERR7       "ERROR: DIRECCION DE ROM"
E68E BD E0 5D      M5:   JSR CRLF
E691 BD E0 68            JSR MENS
E694 7E E1 82            JMP INI
E697 18 08         M3:   INY
E699 BD E0 1F            JSR ASNU
E69C 97 06               STAA DIR3B
E69E 18 DE 05            LDY DIR3A       CARGA DIR3 EN IY
E6A1 9C 05               CPX DIR3A
E6A3 25 16               BLO M6          SALTA SI DIR1 < DIR3
E6A5 22 03               BHI M1          SALTA SI DIR1 > DIR3
E6A7 7E E1 82            JMP INI
E6AA A6 00         M1:   LDAA 0,X        TRANSFIERE DATO
E6AC 18 A7 00            STAA 0,Y
E6AF 9C 03               CPX DIR2A
```

```
E6B1 26 03                  BNE M2          SALTA SI DIR1 <> DIR2
E6B3 7E E1 82               JMP INI
E6B6 08              M2:    INX
E6B7 18 08                  INY
E6B9 20 EF                  BRA M1
E6BB DC 03           M6:    LDD DIR2A       D = DIR2
E6BD 93 01                  SUBD DIR1A      D = DIR2 - DIR1
E6BF D3 05                  ADDD DIR3A      D = (DIR2 - DIR1) + DIR3
E6C1 18 8F                  XGDY            IY = D
E6C3 DE 03                  LDX DIR2A       IX = DIR2
E6C5 A6 00           M8:    LDAA 0,X        TRANSFIERE DATO
E6C7 18 A7 00               STAA 0,Y
E6CA 9C 01                  CPX DIR1A
E6CC 26 03                  BNE M7          SALTA SI DIR2 <> DIR1
E6CE 7E E1 82               JMP INI
E6D1 09              M7:    DEX
E6D2 18 09                  DEY
E6D4 20 EF                  BRA M8

                    ********************************************
                    *                RUTINA O                  *
                    *  Calcula desplazamientos relativos. Si el *
                    *  resultado no está dentro de [-128, +127], *
                    *  despliega el mensaje: "RANGO INVALIDO".   *
                    ********************************************
E6D6 96 08           RUTO:  LDAA LONG
E6D8 81 0B                  CMPA #$0B
E6DA 27 06                  BEQ O2          SALTA SI HAY DIR2
E6DC 18 CE F0 83            LDY #ERR1       "ERROR DE SINTAXIS"
E6E0 20 31                  BRA O3
E6E2 BD E0 BE        O2:    JSR DIR1
E6E5 18 CE 00 1A            LDY #FIL2
E6E9 BD E0 1F               JSR ASNU
E6EC 97 03                  STAA DIR2A
E6EE 18 08                  INY
E6F0 BD E0 1F               JSR ASNU
E6F3 97 04                  STAA DIR2B
E6F5 DC 03                  LDD DIR2A       D = DIR2
E6F7 93 01                  SUBD DIR1A      D = DIR2 - DIR1
E6F9 1A 83 00 7F            CPD #$007F
E6FD 2E 10                  BGT O1          SALTA SI D > $ 007F
E6FF 1A 83 FF 80            CPD #$FF80
E703 2D 0A                  BLT O1          SALTA SI D < $ FF80
E705 17                     TBA
E706 BD E0 5D               JSR CRLF
E709 BD E0 3D               JSR NUAS        DESPLIEGA RESULTADO
E70C 7E E1 82               JMP INI
E70F 18 CE F0 B6     O1:    LDY #ERR4       "RANGO INVALIDO"
E713 BD E0 5D        O3:    JSR CRLF
E716 BD E0 68               JSR MENS
E719 7E E1 82               JMP INI
```

```
                        ********************************************
                        *                                          *
                        *                RUTINA B                  *
                        *  Borra el contenido de un bloque de memoria. *
                        *  Si no existe DIR2 borra DIR1 solamente.     *
                        ********************************************

E71C 96 08              RUTB:   LDAA LONG
E71E 81 06                      CMPA #$06
E720 27 11                      BEQ B4          SALTA SI HAY DIR1
E722 81 0B                      CMPA #$0B
E724 27 1C                      BEQ B5          SALTA SI HAY DIR2
E726 18 CE F0 83                LDY #ERR1       "ERROR DE SINTAXIS"
E72A BD E0 5D           B8:     JSR CRLF
E72D BD E0 68                   JSR MENS
E730 7E E1 82                   JMP INI
E733 BD E0 BE           B4:     JSR DIR1
E736 8C E0 00                   CPX #INIROM
E739 25 02                      BLO B2          SALTA SI NO ES DIR. DE ROM
E73B 20 0D                      BRA B6
E73D 6F 00              B2:     CLR 0,X         BORRA UNA LOCALIDAD
E73F 7E E1 82                   JMP INI
E742 BD E0 BE           B5:     JSR DIR1
E745 8C E0 00                   CPX #INIROM
E748 25 06                      BLO B7          SALTA SI NO ES DIR. DE ROM
E74A 18 CE F0 ED        B6:     LDY #ERR7       "ERROR: DIRECCION DE ROM"
E74E 20 DA                      BRA B8
E750 BD E0 D1           B7:     JSR DIR2
E753 6F 00              B1:     CLR 0,X         BORRA LOCALIDAD
E755 9C 03                      CPX DIR2A
E757 26 03                      BNE B3          SALTA SI DIR1 <> DIR2
E759 7E E1 82                   JMP INI
E75C 08                 B3:     INX
E75D 20 F4                      BRA B1

                        ********************************************
                        *                RUTINA F                  *
                        *  Carga un $ 31 en el registro BAUD para  *
                        *  operar a una velocidad de 4800 bauds.   *
                        ********************************************

E75F 86 31              RUTF:   LDAA #$31
E761 B7 10 2B                   STAA BAUD
E764 7E E1 82                   JMP INI

                        ********************************************
                        *                RUTINA G                  *
                        *  Carga un $ 33 en el registro BAUD para  *
                        *  operar a una velocidad de 1200 bauds.   *
                        ********************************************

E767 86 33              RUTG:   LDAA #$33
E769 B7 10 2B                   STAA BAUD
E76C 7E E1 82                   JMP INI
```

```
                        ****************************************
                        *    ACCESO DEL USUARIO A SUBRUTINAS   *
                        ****************************************
E800                            ORG $E800
E800 7E E0 00                   JMP ASHE
E803 7E E0 34                   JMP HEAS
E806 BD E0 5D                   JSR CRLF
E809 7E E0 3D                   JMP NUAS
E80C 7E E0 1F                   JMP ASNU
E80F BD E0 5D                   JSR CRLF
E812 7E E0 94                   JMP LEE
E815 BD E0 5D                   JSR CRLF
E818 7E E0 50                   JMP ENVIA
E81B 7E E0 5D                   JMP CRLF
E81E BD E0 5D                   JSR CRLF
E821 7E E0 68                   JMP MENS

                        ********************************************
                        *          TABLA 1: TABLA DE COMANDOS      *
                        ********************************************
F024                            ORG TAB1
F024 45                         FCC 'E'
F025 06                         FCB $6
F026 7E                         FCB $7E
F027 E2 09                      FDB RUTE
F029 44                         FCC 'D'
F02A 0B                         FCB $B
F02B 7E                         FCB $7E
F02C E2 91                      FDB RUTD
F02E 43                         FCC 'C'
F02F 0B                         FCB $B
F030 7E                         FCB $7E
F031 E4 9A                      FDB RUTC
F033 54                         FCC 'T'
F034 06                         FCB $6
F035 7E                         FCB $7E
F036 E5 2A                      FDB RUTT
F038 52                         FCC 'R'
F039 01                         FCB $1
F03A 7E                         FCB $7E
F03B E5 91                      FDB RUTR
F03D 41                         FCC 'A'
F03E 01                         FCB $1
F03F 7E                         FCB $7E
F040 E2 EC                      FDB RUTA
F042 4D                         FCC 'M'
F043 10                         FCB $10
F044 7E                         FCB $7E
F045 E6 6B                      FDB RUTM
F047 4F                         FCC 'O'
F048 0B                         FCB $B
F049 7E                         FCB $7E
F04A E6 D6                      FDB RUTO
F04C 42                         FCC 'B'
```

```
F04D 0B                         FCB $B
F04E 7E                         FCB $7E
F04F E7 1C                      FDB RUTB
F051 4C                         FCC 'L'
F052 06                         FCB $6
F053 7E                         FCB $7E
F054 E2 FC                      FDB RUTL
F056 49                         FCC 'I'
F057 0B                         FCB $B
F058 7E                         FCB $7E
F059 E3 EF                      FDB RUTI
F05B 56                         FCC 'V'
F05C 06                         FCB $6
F05D 7E                         FCB $7E
F05E E2 F6                      FDB RUTV
F060 46                         FCC 'F'
F061 01                         FCB $1
F062 7E                         FCB $7E
F063 E7 5F                      FDB RUTF
F065 47                         FCC 'G'
F066 01                         FCB $1
F067 7E                         FCB $7E
F068 E7 67                      FDB RUTG

                        ********************************************
                        *          TABLA 2: NOMBRES DE REGISTROS   *
                        ********************************************
F06A                            ORG TAB2
F06A 41 43 43 3A                FCC 'ACC:'
F06E 04                         FCB EOT
F06F 42 3A                      FCC 'B:'
F071 04                         FCB EOT
F072 43 43 52 3A                FCC 'CCR:'
F076 04                         FCB EOT
F077 49 58 3A                   FCC 'IX:'
F07A 04                         FCB EOT
F07B 49 59 3A                   FCC 'IY:'
F07E 04                         FCB EOT
F07F 53 50 3A                   FCC 'SP:'
F082 04                         FCB EOT

                        ********************************************
                        *          TABLA 3: TABLA DE ERRORES       *
                        ********************************************
F083                            ORG TAB3
F083 45 52 52 4F 52 20          FCC 'ERROR DE SINTAXIS'
     44 45 20 53 49 4E
     54 41 58 49 53
F094 04                         FCB EOT
F095 43 4F 4D 41 4E 44          FCC 'COMANDO INVALIDO'
     4F 20 49 4E 56 41
     4C 49 44 4F
F0A5 04                         FCB EOT
F0A6 4E 55 4D 45 52 4F          FCC 'NUMERO INVALIDO'
```

```
     20 49 4E 56 41 4C
     49 44 4F
F0B5 04                         FCB EOT
F0B6 52 41 4E 47 4F 20          FCC 'RANGO INVALIDO'
     49 4E 56 41 4C 49
     44 4F
F0C4 04                         FCB EOT
F0C5 44 49 46 45 52 45          FCC 'DIFERENCIA EN PROGRAMA'
     4E 43 49 41 20 45
     4E 20 50 52 4F 47
     52 41 4D 41
F0DB 04                         FCB EOT
F0DC 45 52 52 4F 52 20          FCC 'ERROR DE LECTURA'
     44 45 20 4C 45 43
     54 55 52 41
F0EC 04                         FCB EOT
F0ED 45 52 52 4F 52 3A          FCC 'ERROR: DIRECCION DE ROM'
     20 44 49 52 45 43
     43 49 4F 4E 20 44
     45 20 52 4F 4D
F104 04                         FCB EOT

                        ********************************************
                        *          TABLA 4: PANTALLA DE AYUDA      *
                        ********************************************
F105                            ORG TAB4
F105 0D 0A                      FCB CR,LF
F107 41 28 43 52 29 09          FCC 'A(CR)                          AYUDA'
     09 09 20 20 20 20
     41 59 55 44 41
F118 0D 0A                      FCB CR,LF
F11A 46 28 43 52 29 09          FCC 'F(CR)                          VELOCIDAD
A 4800 BAUDS'
     09 09 20 20 20 20
     56 45 4C 4F 43 49
     44 41 44 20 41 20
     34 38 30 30 20 42
     41 55 44 53
F13C 0D 0A                      FCB CR,LF
F13E 47 28 43 52 29 09          FCC 'G(CR)                          VELOCIDAD
A 1200 BAUDS'
     09 09 20 20 20 20
     56 45 4C 4F 43 49
     44 41 44 20 41 20
     31 32 30 30 20 42
     41 55 44 53
F160 0D 0A                      FCB CR,LF
F162 4C 20 3C 6E 75 6D          FCC 'L <num>(CR)                    LEE
PROGRAMA'
     3E 28 43 52 29 09
     09 20 20 20 20 4C
     45 45 20 50 52 4F
     47 52 41 4D 41
F17F 0D 0A                      FCB CR,LF
```

```
F181 56 20 3C 6E 75 6D          FCC 'V <num>(CR)                VERIFICA
PROGRAMA'
     3E 28 43 52 29 09
     09 20 20 20 20 56
     45 52 49 46 49 43
     41 20 50 52 4F 47
     52 41 4D 41
F1A3 0D 0A                      FCB CR,LF
F1A5 42 20 3C 64 69 72          FCC 'B <dir1>[ <dir2>](CR)      BORRA
LOCALIDAD O BLOQUE'
     31 3E 5B 20 3C 64
     69 72 32 3E 5D 28
     43 52 29 09 20 20
     20 20 42 4F 52 52
     41 20 4C 4F 43 41
     4C 49 44 41 44 20
     4F 20 42 4C 4F 51
     55 45
F1D7 0D 0A                      FCB CR,LF
F1D9 44 20 3C 64 69 72          FCC 'D <dir1>[ <dir2>](CR)      DESPLIEGA
LOCALIDADES'
     31 3E 5B 20 3C 64
     69 72 32 3E 5D 28
     43 52 29 09 20 20
     20 20 44 45 53 50
     4C 49 45 47 41 20
     4C 4F 43 41 4C 49
     44 41 44 45 53
F208 0D 0A                      FCB CR,LF
F20A 43 20 3C 64 69 72          FCC 'C <dir1>[ <dir2>](CR)      CORRE
PROGRAMA'
     31 3E 5B 20 3C 64
     69 72 32 3E 5D 28
     43 52 29 09 20 20
     20 20 43 4F 52 52
     45 20 50 52 4F 47
     52 41 4D 41
F232 0D 0A                      FCB CR,LF
F234 4F 20 3C 64 69 72          FCC 'O <dir1> <dir2>(CR)        CALCULA
DESPLAZAMIENTOS RELATIVOS'
     31 3E 20 3C 64 69
     72 32 3E 28 43 52
     29 09 20 20 20 20
     43 41 4C 43 55 4C
     41 20 44 45 53 50
     4C 41 5A 41 4D 49
     45 4E 54 4F 53 20
     52 45 4C 41 54 49
     56 4F 53
F26D 0D 0A                      FCB CR,LF
F26F 49 20 3C 64 69 72          FCC 'I <dir1> <dir2>(CR)        IMPRIME
PROGRAMA'
     31 3E 20 3C 64 69
     72 32 3E 28 43 52
```

```
     29 09 20 20 20 20
     49 4D 50 52 49 4D
     45 20 50 52 4F 47
     52 41 4D 41
F297 0D 0A                    FCB CR,LF
F299 4D 20 3C 64 69 72        FCC 'M <dir1> <dir2> <dir3>(CR)  MUEVE BLOQUE
DE MEMORIA'
     31 3E 20 3C 64 69
     72 32 3E 20 3C 64
     69 72 33 3E 28 43
     52 29 20 20 4D 55
     45 56 45 20 42 4C
     4F 51 55 45 20 44
     45 20 4D 45 4D 4F
     52 49 41
F2CC 0D 0A                    FCB CR,LF
F2CE 52 28 43 52 29 09        FCC 'R(CR)                       DESPLIEGA
REGISTROS'
     09 09 20 20 20 20
     44 45 53 50 4C 49
     45 47 41 20 52 45
     47 49 53 54 52 4F
     53
F2ED 0D 0A                    FCB CR,LF
F2EF 20 20 20 20 20 28        FCC '     (ESPACIO) sigue'
     45 53 50 41 43 49
     4F 29 20 73 69 67
     75 65
F303 0D 0A                    FCB CR,LF
F305 20 20 20 20 20 28        FCC '     (CR)    termina'
     43 52 29 20 20 20
     20 74 65 72 6D 69
     6E 61
F319 0D 0A                    FCB CR,LF
F31B 45 20 3C 64 69 72        FCC 'E <dir1>(CR)                EXAMINA Y
MODIFICA MEMORIA'
     31 3E 28 43 52 29
     09 09 20 20 20 20
     45 58 41 4D 49 4E
     41 20 59 20 4D 4F
     44 49 46 49 43 41
     20 4D 45 4D 4F 52
     49 41
F347 0D 0A                    FCB CR,LF
F349 20 20 20 20 20 28        FCC '     (ESPACIO) sigue'
     45 53 50 41 43 49
     4F 29 20 73 69 67
     75 65
F35D 0D 0A                    FCB CR,LF
F35F 20 20 20 20 20 28        FCC '     (CR)    termina'
     43 52 29 20 20 20
     20 74 65 72 6D 69
     6E 61
F373 0D 0A                    FCB CR,LF
```

```
F375 54 20 3C 64 69 72          FCC 'T <dir1>(CR)                    TRAZA
PROGRAMA'
     31 3E 28 43 52 29
     09 09 20 20 20 20
     54 52 41 5A 41 20
     50 52 4F 47 52 41
     4D 41
F395 0D 0A                      FCB CR,LF
F397 20 20 20 20 20 28          FCC '     (ESPACIO) sigue'
     45 53 50 41 43 49
     4F 29 20 73 69 67
     75 65
F3AB 0D 0A                      FCB CR,LF
F3AD 20 20 20 20 20 28          FCC '     (CR)    termina'
     43 52 29 20 20 20
     20 74 65 72 6D 69
     6E 61
F3C1 04                         FCB EOT

                        ********************************************
                        *    RESET Y VECTORES DE INTERRUPCION DEL CPU  *
                        ********************************************

FFD6                            ORG FINROM
FFD6 00 6C                      FDB VSCI
FFD8 00 6F                      FDB VSPI
FFDA 00 72                      FDB VPAI
FFDC 00 75                      FDB VPAOV
FFDE 00 78                      FDB VTO
FFE0 00 7B                      FDB VI405
FFE2 00 7E                      FDB VOC4
FFE4 00 81                      FDB VOC3
FFE6 00 84                      FDB VOC2
FFE8 00 87                      FDB VOC1
FFEA 00 8A                      FDB VIC3
FFEC 00 8D                      FDB VIC2
FFEE 00 90                      FDB VIC1
FFF0 00 93                      FDB VRTI
FFF2 00 96                      FDB VIRQ
FFF4 00 99                      FDB VXIRQ
FFF6 00 9C                      FDB VSWI
FFF8 00 9F                      FDB VIOT
FFFA 00 A2                      FDB VNOCOP
FFFC 00 A5                      FDB VCME
FFFE E1 64                      FDB INREG
                                END
```

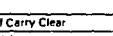
APÉNDICE F

JUEGO DE INSTRUCCIONES Y REGISTROS DEL MC68HC11

INSTRUCTIONS, ADDRESSING MODES, AND EXECUTION TIMES

Source Form(s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal) Opcode	Operand(s)	Bytes	Cycle	Condition Codes S	X	H	I	N	Z	V	C
ABA	Add Accumulators	A + B → A	INH	1B		1	2	-	-	↕	-	↕	↕	↕	↕
ABX	Add B to X	IX + 00 B → IX	INH	3A		1	3	-	-	-	-	-	-	-	-
ABY	Add B to Y	IY + 00 B → IY	INH	18 3A		2	4	-	-	-	-	-	-	-	-
ADCA (opr)	Add with Carry to A	A + M + C → A	A IMM	89	ii	2	2	-	-	↕	-	↕	↕	↕	↕
			A DIR	99	dd	2	3								
			A EXT	B9	hh ll	3	4								
			A IND,X	A9	ff	2	4								
			A IND,Y	18 A9	ff	3	5								
ADCB (opr)	Add with Carry to B	B + M + C → B	B IMM	C9	ii	2	2	-	-	↕	-	↕	↕	↕	↕
			B DIR	D9	dd	2	3								
			B EXT	F9	hh ll	3	4								
			B IND,X	E9	ff	2	4								
			B IND,Y	18 E9	ff	3	5								

Source Form(s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal) Opcode	Operand(s)	Bytes	Cycle	Condition Codes S	X	H	I	N	Z	V	C
ADDA (opr)	Add Memory to A	A + M → A	A IMM	8B	ii	2	2	-	-	↕	-	↕	↕	↕	↕
			A DIR	9B	dd	2	3								
			A EXT	BB	hh ll	3	4								
			A IND,X	AB	ff	2	4								
			A IND,Y	18 AB	ff	3	5								
ADDB (opr)	Add Memory to B	B + M → B	B IMM	CB	ii	2	2	-	-	↕	-	↕	↕	↕	↕
			B DIR	DB	dd	2	3								
			B EXT	FB	hh ll	3	4								
			B IND,X	EB	ff	2	4								
			B IND,Y	18 EB	ff	3	5								
ADDD (opr)	Add 16-Bit to D	D + M:M + 1 → D	IMM	C3	jj kk	3	4	-	-	-	-	↕	↕	↕	↕
			DIR	D3	dd	2	5								
			EXT	F3	hh ll	3	6								
			IND,X	E3	ff	2	6								
			IND, Y	18 E3	ff	3	7								
ANDA (opr)	AND A with Memory	A•M → A	A IMM	84	ii	2	2	-	-	-	-	↕	↕	0	-
			A DIR	94	dd	2	3								
			A EXT	B4	hh ll	3	4								
			A IND,X	A4	ff	2	4								
			A IND,Y	18 A4	ff	3	5								

ANDB (opr)	AND B with Memory	B•M → B	B IMM	C4	ii	2	2	-	-	-	-	↕	↕	0	-
			B DIR	D4	dd	2	3								
			B EXT	F4	hh ll	3	4								
			B IND,X	E4	ff	2	4								
			B IND,Y	18 E4	ff	3	5								
ASL (opr)	Arithmetic Shift Left		EXT	78	hh ll	3	6	-	-	-	-	↕	↕	↕	↕
			IND,X	68	ff	2	6								
		C b7 b0	IND,Y	18 68	ff	3	7								
ASLA			A INH	48		1	2								
ASLB			B INH	58		1	2								
ASLD	Arithmetic Shift Left Double		INH	05		1	3	-	-	-	-	↕	↕	↕	↕
		C b15 b0													
ASR (opr)	Arithmetic Shift Right		EXT	77	hh ll	3	6	-	-	-	-	↕	↕	↕	↕
			IND,X	67	ff	2	6								
		b7 b0 C	IND,Y	18 67	ff	3	7								
ASRA			A INH	47		1	2								
ASRB			B INH	57		1	2								
BCC (rel)	Branch if Carry Clear	? C = 0	REL	24	rr	2	3	-	-	-	-	-	-	-	-
BCLR (opr) (msk)	Clear Bit(s)	M•($\overline{mm}$) → M	DIR	15	dd mm	3	6	-	-	-	-	↕	↕	0	-
			IND,X	1D	ff mm	3	7								
			IND,Y	18 1D	ff mm	4	8								
BCS (rel)	Branch if Carry Set	? C = 1	REL	25	rr	2	3	-	-	-	-	-	-	-	-
BEQ (rel)	Branch if = Zero	? Z = 1	REL	27	rr	2	3	-	-	-	-	-	-	-	-

Source Form(s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal) Opcode	Operand(s)	Bytes	Cycle	Condition Codes S	X	H	I	N	Z	V	C
BGE (rel)	Branch if ≥ Zero	? N ⊕ V = 0	REL	2C	rr	2	3	·	·	·	·	·	·	·	·
BGT (rel)	Branch if > Zero	? Z + (N ⊕ V) = 0	REL	2E	rr	2	3	·	·	·	·	·	·	·	·
BHI (rel)	Branch if Higher	? C + Z = 0	REL	22	rr	2	3	·	·	·	·	·	·	·	·
BHS (rel)	Branch if Higher or Same	? C = 0	REL	24	rr	2	3	·	·	·	·	·	·	·	·
BITA (opr)	Bit(s) Test A with Memory	A•M	A IMM	85	ii	2	2	·	·	·	·	↕	↕	0	·
			A DIR	95	dd	2	3								
			A EXT	B5	hh ll	3	4								
			A IND,X	A5	ff	2	4								
			A IND,Y	18 A5	ff	3	5								
BITB (opr)	Bit(s) Test B with Memory	B•M	B IMM	C5	ii	2	2	·	·	·	·	↕	↕	0	·
			B DIR	D5	dd	2	3								
			B EXT	F5	hh ll	3	4								
			B IND,X	E5	ff	2	4								
			B IND,Y	18 E5	ff	3	5								
BLE (rel)	Branch if ≤ Zero	? Z + (N ⊕ V) = 1	REL	2F	rr	2	3	·	·	·	·	·	·	·	·
BLO (rel)	Branch if Lower	? C = 1	REL	25	rr	2	3	·	·	·	·	·	·	·	·
BLS (rel)	Branch if Lower or Same	? C + Z = 1	REL	23	rr	2	3	·	·	·	·	·	·	·	·

BLT (rel)	Branch if < Zero	? N ⊕ V = 1	REL	2D	rr	2	3	·	·	·	·	·	·	·	·
BMI (rel)	Branch if Minus	? N = 1	REL	2B	rr	2	3	·	·	·	·	·	·	·	·
BNE (rel)	Branch if Not = Zero	? Z = 0	REL	26	rr	2	3	·	·	·	·	·	·	·	·
BPL (rel)	Branch if Plus	? N = 0	REL	2A	rr	2	3	·	·	·	·	·	·	·	·
BRA (rel)	Branch Always	? 1 = 1	REL	20	rr	2	3	·	·	·	·	·	·	·	·
BRCLR(opr) (msk) (rel)	Branch if Bit(s) Clear	? M• mm = 0	DIR	13	dd mm rr	4	6	·	·	·	·	·	·	·	·
			IND,X	1F	ff mm rr	4	7								
			IND,Y	18 1F	ff mm rr	5	8								
BRN (rel)	Branch Never	? 1 = 0	REL	21	rr	2	3	·	·	·	·	·	·	·	·
BRSET(opr) (msk) (rel)	Branch if Bit(s) Set	? ($\overline{M}$)•mm = 0	DIR	12	dd mm rr	4	6	·	·	·	·	·	·	·	·
			IND,X	1E	ff mm rr	4	7								
			IND,Y	18 1E	ff mm rr	5	8								
BSET(opr) (msk)	Set Bit(s)	M + mm → M	DIR	14	dd mm	3	6	·	·	·	·	↕	↕	0	·
			IND,X	1C	ff mm	3	7								
			IND,Y	18 1C	ff mm	4	8								
BSR (rel)	Branch to Subroutine	See Special Ops	REL	8D	rr	2	6	·	·	·	·	·	·	·	·
BVC (rel)	Branch if Overflow Clear	? V = 0	REL	28	rr	2	3	·	·	·	·	·	·	·	·
BVS (rel)	Branch if Overflow Set	? V = 1	REL	29	rr	2	3	·	·	·	·	·	·	·	·
CBA	Compare A to B	A − B	INH	11		1	2	·	·	·	·	↕	↕	↕	↕
CLC	Clear Carry Bit	0 → C	INH	0C		1	2	·	·	·	·	·	·	·	0
CLI	Clear Interrupt Mask	0 → I	INH	0E		1	2	·	·	·	0	·	·	·	·

Source Form(s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal) Opcode	Operand(s)	Bytes	Cycle	Condition Codes S	X	H	I	N	Z	V	C
CLR (opr)	Clear Memory Byte	0 → M	EXT	7F	hh ll	3	6	·	·	·	·	0	1	0	0
			IND,X	6F	ff	2	6								
			IND,Y	18 6F	ff	3	7								
CLRA	Clear Accumulator A	0 → A	A INH	4F		1	2	·	·	·	·	0	1	0	0
CLRB	Clear Accumulator B	0 → B	B INH	5F		1	2	·	·	·	·	0	1	0	0
CLV	Clear Overflow Flag	0 → V	INH	0A		1	2	·	·	·	·	·	·	0	·
CMPA (opr)	Compare A to Memory	A − M	A IMM	81	ii	2	2	·	·	·	·	↕	↕	↕	↕
			A DIR	91	dd	2	3								
			A EXT	B1	hh ll	3	4								
			A IND,X	A1	ff	2	4								
			A IND,Y	18 A1	ff	3	5								
CMPB (opr)	Compare B to Memory	B − M	B IMM	C1	ii	2	2	·	·	·	·	↕	↕	↕	↕
			B DIR	D1	dd	2	3								
			B EXT	F1	hh ll	3	4								
			B IND,X	E1	ff	2	4								
			B IND,Y	18 E1	ff	3	5								
COM (opr)	1's Complement Memory Byte	$FF − M → M	EXT	73	hh ll	3	6	·	·	·	·	↕	↕	0	1
			IND,X	63	ff	2	6								
			IND,Y	18 63	ff	3	7								

COMA	1's Complement A	$FF - A → A	A INH	43		1	2	·	·	·	·	↕	↕	0	↕
COMB	1's Complement B	$FF - B → B	B INH	53		1	2	·	·	·	·	↕	↕	0	↕
CPD (opr)	Compare D to Memory 16-Bit	D - M:M + 1	IMM	1A 83	jj kk	4	5	·	·	·	·	↕	↕	↕	↕
			DIR	1A 93	dd	3	6								
			EXT	1A B3	hh ll	4	7								
			IND,X	1A A3	ff	3	7								
			IND,Y	CD A3	ff	3	7								
CPX (opr)	Compare X to Memory 16-Bit	IX - M:M + 1	IMM	8C	jj kk	3	4	·	·	·	·	↕	↕	↕	↕
			DIR	9C	dd	2	5								
			EXT	BC	hh ll	3	6								
			IND,X	AC	ff	2	6								
			IND,Y	CD AC	ff	3	7								
CPY (opr)	Compare Y to Memory 16-Bit	IY - M:M + 1	IMM	18 8C	jj kk	4	5	·	·	·	·	↕	↕	↕	↕
			DIR	18 9C	dd	3	6								
			EXT	18 BC	hh ll	4	7								
			IND,X	1A AC	ff	3	7								
			IND,Y	18 AC	ff	3	7								
DAA	Decimal Adjust A	Adjust Sum to BCD	INH	19		1	2	·	·	·	·	↕	↕	↕	↕
DEC (opr)	Decrement Memory Byte	M - 1 → M	EXT	7A	hh ll	3	6	·	·	·	·	↕	↕	↕	·
			IND,X	6A	ff	2	6								
			IND,Y	18 6A	ff	3	7								
DECA	Decrement Accumulator A	A - 1 → A	A INH	4A		1	2	·	·	·	·	↕	↕	↕	·
DECB	Decrement Accumulator B	B - 1 → B	B INH	5A		1	2	·	·	·	·	↕	↕	↕	·
DES	Decrement Stack Pointer	SP - 1 → SP	INH	34		1	3	·	·	·	·	·	·	·	·

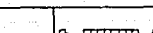
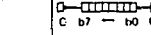
Source Form(s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal) Opcode	Operand(s)	Bytes	Cycle	S	X	H	I	N	Z	V	C
DEX	Decrement Index Register X	IX - 1 → IX	INH	09		1	3	·	·	·	·	·	↕	·	·
DEY	Decrement Index Register Y	IY - 1 → IY	INH	18 09		2	4	·	·	·	·	·	↕	·	·
EORA (opr)	Exclusive OR A with Memory	A ⊕ M → A	A IMM	88	ii	2	2	·	·	·	·	↕	↕	0	·
			A DIR	98	dd	2	3								
			A EXT	B8	hh ll	3	4								
			A IND,X	A8	ff	2	4								
			A IND,Y	18 A8	ff	3	5								
EORB (opr)	Exclusive OR B with Memory	B ⊕ M → B	B IMM	C8	ii	2	2	·	·	·	·	↕	↕	0	·
			B DIR	D8	dd	2	3								
			B EXT	F8	hh ll	3	4								
			B IND,X	E8	ff	2	4								
			B IND,Y	18 E8	ff	3	5								
FDIV	Fractional Divide 16 by 16	D/IX → IX; r → D	INH	03		1	41	·	·	·	·	·	↕	↕	↕
IDIV	Integer Divide 16 by 16	D/IX → IX; r → D	INH	02		1	41	·	·	·	·	·	↕	0	↕
INC (opr)	Increment Memory Byte	M + 1 → M	EXT	7C	hh ll	3	6	·	·	·	·	↕	↕	↕	·
			IND,X	6C	ff	2	6								
			IND,Y	18 6C	ff	3	7								
INCA	Increment Accumulator A	A + 1 → A	A INH	4C		1	2	·	·	·	·	↕	↕	↕	·
INCB	Increment Accumulator B	B + 1 → B	B INH	5C		1	2	·	·	·	·	↕	↕	↕	·

INS	Increment Stack Pointer	SP + 1 → SP	INH	31		1	3	·	·	·	·	·	·	·	·
INX	Increment Index Register X	IX + 1 → IX	INH	08		1	3	·	·	·	·	·	↕	·	·
INY	Increment Index Register Y	IY + 1 → IY	INH	18 08		2	4	·	·	·	·	·	↕	·	·
JMP (opr)	Jump	See Special Ops	EXT	7E	hh ll	3	3	·	·	·	·	·	·	·	·
			IND,X	6E	ff	2	3								
			IND,Y	18 6E	ff	3	4								
JSR (opr)	Jump to Subroutine	See Special Ops	DIR	9D	dd	2	5	·	·	·	·	·	·	·	·
			EXT	BD	hh ll	3	6								
			IND,X	AD	ff	2	6								
			IND,Y	18 AD	ff	3	7								
LDAA (opr)	Load Accumulator A	M → A	A IMM	86	ii	2	2	·	·	·	·	↕	↕	0	·
			A DIR	96	dd	2	3								
			A EXT	B6	hh ll	3	4								
			A IND,X	A6	ff	2	4								
			A IND,Y	18 A6	ff	3	5								
LDAB (opr)	Load Accumulator B	M → B	B IMM	C6	ii	2	2	·	·	·	·	↕	↕	0	·
			B DIR	D6	dd	2	3								
			B EXT	F6	hh ll	3	4								
			B IND,X	E6	ff	2	4								
			B IND,Y	18 E6	ff	3	5								

Source Form(s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal) Opcode	Operand(s)	Bytes	Cycle	Condition Codes S	X	H	I	N	Z	V	C
LDD (opr)	Load Double Accumulator D	M → A, M + 1 → B	IMM	CC	jj kk	3	3	·	·	·	·	↕	↕	0	·
			DIR	DC	dd	2	4								
			EXT	FC	hh ll	3	5								
			IND,X	EC	ff	2	5								
			IND,Y	18 EC	ff	3	6								
LDS (opr)	Load Stack Pointer	M:M + 1 → SP	IMM	8E	jj kk	3	3	·	·	·	·	↕	↕	0	·
			DIR	9E	dd	2	4								
			EXT	BE	hh ll	3	5								
			IND,X	AE	ff	2	5								
			IND,Y	18 AE	ff	3	6								
LDX (opr)	Load Index Register X	M:M + 1 → IX	IMM	CE	jj kk	3	3	·	·	·	·	↕	↕	0	·
			DIR	DE	dd	2	4								
			EXT	FE	hh ll	3	5								
			IND,X	EE	ff	2	5								
			IND,Y	CD EE	ff	3	6								
LDY (opr)	Load Index Register Y	M:M + 1 → IY	IMM	18 CE	jj kk	4	4	·	·	·	·	↕	↕	0	·
			DIR	18 DE	dd	3	5								
			EXT	18 FE	hh ll	4	6								
			IND,X	1A EE	ff	3	6								
			IND,Y	18 EE	ff	3	6								

LSL (opr)	Logical Shift Left	←  C b7 b0	EXT	78	hh ll	3	6	·	·	·	·	↕	↕	↕	↕
			IND,X	68	ff	2	6								
			IND,Y	18 68	ff	3	7								
LSLA			A INH	48		1	2								
LSLB			B INH	58		1	2								
LSLD	Logical Shift Left Double	 C b15 b0	INH	05		1	3	·	·	·	·	↕	↕	↕	↕
LSR (opr)	Logical Shift Right	→  b7 b0 C	EXT	74	hh ll	3	6	·	·	·	·	0	↕	↕	↕
			IND,X	64	ff	2	6								
			IND,Y	18 64	ff	3	7								
LSRA			A INH	44		1	2								
LSRB			B INH	54		1	2								
LSRD	Logical Shift Right Double	 b15 b0 C	INH	04		1	3	·	·	·	·	0	↕	↕	↕
MUL	Multiply B by B	A x B → D	INH	3D		1	10	·	·	·	·	·	·	·	↕
NEG (opr)	2's Complement Memory Byte	0 - M → M	EXT	70	hh ll	3	6	·	·	·	·	↕	↕	↕	↕
			IND,X	60	ff	2	6								
			IND,Y	18 60	ff	3	7								
NEGA	2's Complement A	0 - A → A	A INH	40		1	2	·	·	·	·	↕	↕	↕	↕
NEGB	2's Complement B	0 - B → B	B INH	50		1	2	·	·	·	·	↕	↕	↕	↕
NOP	No Operation	No Operation	INH	01		1	2	·	·	·	·	·	·	·	·

Source Form(s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal) Opcode	Operand(s)	Bytes	Cycle	Condition Codes S	X	H	I	N	Z	V	C
ORAA (opr)	OR Accumulator A (inclusive)	A + M → A	A IMM	8A	ii	2	2	·	·	·	·	↕	↕	0	·
			A DIR	9A	dd	2	3								
			A EXT	BA	hh ll	3	4								
			A IND,X	AA	ff	2	4								
			A IND,Y	18 AA	ff	3	5								
ORAB (opr)	OR Accumulator B (inclusive)	B + M → B	B IMM	CA	ii	2	2	·	·	·	·	↕	↕	0	·
			B DIR	DA	dd	2	3								
			B EXT	FA	hh ll	3	4								
			B IND,X	EA	ff	2	4								
			B IND,Y	18 EA	ff	3	5								
PSHA	Push A onto Stack	A → Stk, SP = SP - 1	A INH	36		1	3	·	·	·	·	·	·	·	·
PSHB	Push B onto Stack	B → Stk, SP = SP - 1	B INH	37		1	3	·	·	·	·	·	·	·	·
PSHX	Push X onto Stack (Lo First)	IX → Stk, SP = SP - 2	INH	3C		1	4	·	·	·	·	·	·	·	·
PSHY	Push Y onto Stack (Lo First)	IY → Stk, SP = SP - 2	INH	18 3C		2	5	·	·	·	·	·	·	·	·
PULA	Pull A from Stack	SP = SP + 1, A ← Stk	A INH	32		1	4	·	·	·	·	·	·	·	·
PULB	Pull B from Stack	SP = SP + 1, B ← Stk	B INH	33		1	4	·	·	·	·	·	·	·	·
PULX	Pull X from Stack (Hi First)	SP = SP + 2, IX ← Stk	INH	38		1	5	·	·	·	·	·	·	·	·
PULY	Pull Y from Stack (Hi First)	SP = SP + 2, IY ← Stk	INH	18 38		2	6	·	·	·	·	·	·	·	·

ROL (opr)	Rotate Left		EXT	79	hh ll	3	6	·	·	·	·	·	↕	↕	·	↕	↕
			IND,X	69	ff	2	6										
			IND,Y	18 69	ff	3	7										
ROLA			A INH	49		1	2										
ROLB			B INH	59		1	2										
ROR (opr)	Rotate Right		EXT	76	hh ll	3	6	·	·	·	·	·	↕	↕	↕	↕	↕
			IND,X	66	ff	2	6										
			IND,Y	18 66	ff	3	7										
RORA			A INH	46		1	2										
RORB			B INH	56		1	2										
RTI	Return from Interrupt	See Special Ops	INH	3B		1	12	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕
RTS	Return from Subroutine	See Special Ops	INH	39		1	5	·	·	·	·	·	·	·	·	·	·
SBA	Subtract B from A	A - B → A	INH	10		1	2	·	·	·	·	·	↕	↕	↕	↕	↕
SBCA (opr)	Subtract with Carry from A	A - M - C → A	A IMM	82	ii	2	2	·	·	·	·	·	↕	↕	↕	↕	↕
			A DIR	92	dd	2	3										
			A EXT	B2	hh ll	3	4										
			A IND,X	A2	ff	2	4										
			A IND,Y	18 A2	ff	3	5										
SBCB (opr)	Subtract with Carry from B	B - M - C → B	B IMM	C2	ii	2	2	·	·	·	·	·	↕	↕	↕	↕	↕
			B DIR	D2	dd	2	3										
			B EXT	F2	hh ll	3	4										
			B IND,X	E2	ff	2	4										
			B IND,Y	18 E2	ff	3	5										

Source Form(s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal) Opcode	Operand(s)	Bytes	Cycle	Condition Codes S	X	H	I	N	Z	V	C
SEC	Set Carry	1 → C	INH	0D		1	2	·	·	·	·	·	·	·	↕
SEI	Set Interrupt Mask	1 → I	INH	0F		1	2	·	·	·	↕	·	·	·	·
SEV	Set Overflow Flag	1 → V	INH	0B		1	2	·	·	·	·	·	·	↕	·
STAA (opr)	Store Accumulator A	A → M	A DIR	97	dd	2	3	·	·	·	·	↕	↕	0	·
			A EXT	B7	hh ll	3	4								
			A IND,X	A7	ff	2	4								
			A IND,Y	18 A7	ff	3	5								
STAB (opr)	Store Accumulator B	B → M	B DIR	D7	dd	2	3	·	·	·	·	↕	↕	0	·
			B EXT	F7	hh ll	3	4								
			B IND,X	E7	ff	2	4								
			B IND,Y	18 E7	ff	3	5								
STD (opr)	Store Accumulator D	A → M, B → M + 1	DIR	DD	dd	2	4	·	·	·	·	↕	↕	0	·
			EXT	FD	hh ll	3	5								
			IND,X	ED	ff	2	5								
			IND,Y	18 ED	ff	3	6								
STOP	Stop Internal Clocks		INH	CF		1	2	·	·	·	·	·	·	·	·
STS (opr)	Store Stack Pointer	SP → M:M + 1	DIR	9F	dd	2	4	·	·	·	·	↕	↕	0	·
			EXT	BF	hh ll	3	5								
			IND,X	AF	ff	2	5								
			IND,Y	18 AF	ff	3	6								
STX (opr)	Store Index Register X	IX → M:M + 1	DIR	DF	dd	2	4	·	·	·	·	↕	↕	0	·
			EXT	FF	hh ll	3	5								
			IND,X	EF	ff	2	5								
			IND,Y	CD EF	ff	3	6								
STY (opr)	Store Index Register Y	IY → M:M + 1	DIR	18 DF	dd	3	5	·	·	·	·	↕	↕	0	·
			EXT	18 FF	hh ll	4	6								
			IND,X	1A EF	ff	3	6								
			IND,Y	18 EF	ff	3	6								
SUBA (opr)	Subtract Memory from A	A - M → A	A IMM	80	ii	2	2	·	·	·	·	↕	↕	↕	↕
			A DIR	90	dd	2	3								
			A EXT	B0	hh ll	3	4								
			A IND,X	A0	ff	2	4								
			A IND,Y	18 A0	ff	3	5								
SUBB (opr)	Subtract Memory from B	B - M → B	B IMM	C0	ii	2	2	·	·	·	·	↕	↕	↕	↕
			B DIR	D0	dd	2	3								
			B EXT	F0	hh ll	3	4								
			B IND,X	E0	ff	2	4								
			B IND,Y	18 E0	ff	3	5								
SUBD (opr)	Subtract Memory from D	D - M:M + 1 → D	IMM	83	jj kk	3	4	·	·	·	·	↕	↕	↕	↕
			DIR	93	dd	2	5								
			EXT	B3	hh ll	3	6								
			IND,X	A3	ff	2	6								
			IND,Y	18 A3	ff	3	7								
SWI	Software Interrupt	See Special Ops	INH	3F		1	14	·	·	·	↕	·	·	·	·

Source Form(s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal) Opcode	Operand(s)	Bytes	Cycle	Condition Codes S	X	H	I	N	Z	V	C
TAB	Transfer A to B	A → B	INH	16		1	2	-	-	-	-	↕	↕	0	-
TAP	Transfer A to CC Register	A → CCR	INH	06		1	2	↕	↕	↕	↕	↕	↕	↕	↕
TBA	Transfer B to A	B → A	INH	17		1	2	-	-	-	-	↕	↕	0	-
TEST	TEST (Only in Test Mode)	Address Bus Counts	INH	00		1	*	-	-	-	-	-	-	-	-
TPA	Transfer CC Register to A	CCR → A	INH	07		1	2	-	-	-	-	-	-	-	-
TST (opr)	Test for Zero or Minus	M - 0	EXT	7D	hh ll	3	6	-	-	-	-	↕	↕	0	0
			IND,X	6D	ff	2	6								
			IND,Y	18 6D	ff	3	7								
TSTA		A - 0	A INH	4D		1	2	-	-	-	-	↕	↕	0	0
TSTB		B - 0	B INH	5D		1	2	-	-	-	-	↕	↕	0	0
TSX	Transfer Stack Pointer to X	SP + 1 → IX	INH	30		1	3	-	-	-	-	-	-	-	-
TSY	Transfer Stack Pointer to Y	SP + 1 → IY	INH	18 30		2	4	-	-	-	-	-	-	-	-
TXS	Transfer X to Stack Pointer	IX - 1 → SP	INH	35		1	3	-	-	-	-	-	-	-	-
TYS	Transfer Y to Stack Pointer	IY - 1 → SP	INH	18 35		2	4	-	-	-	-	-	-	-	-
WAI	Wait for Interrupt	Stack Regs & WAIT	INH	3E		1	**	-	-	-	-	-	-	-	-
XGDX	Exchange D with X	IX → D, D → IX	INH	8F		1	3	-	-	-	-	-	-	-	-
XGDY	Exchange D with Y	IY → D, D → IY	INH	18 8F		2	4	-	-	-	-	-	-	-	-

NOTES:

Cycle:

* = Infinity or until reset occurs

** = 12 cycles are used beginning with the opcode fetch. A wait state is entered which remains in effect for an integer number of MPU E-clock cycle (n) until an interrupt is recognized. Finally, two additional cycles are used to fetch the appropriate interrupt vector (total = 14 + n).

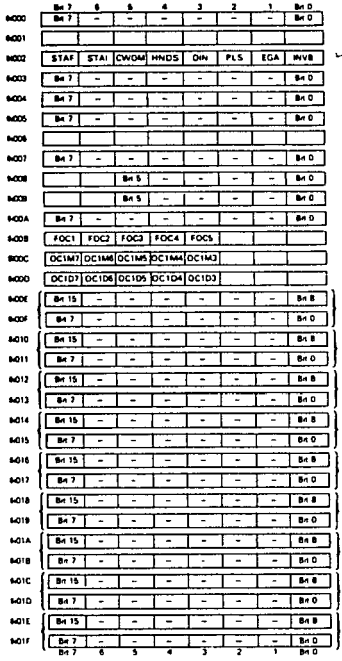
Operands:

dd = 8-bit direct address $0000-$00FF. (High byte assumed to be $00.)
ff = 8-bit positive offset $00 (0) to $FF (255) added to index.
hh = High order byte of 16-bit extended address.
ii = One byte of immediate data.
jj = High order byte of 16-bit immediate data.
kk = Low order byte of 16-bit immediate data.
ll = Low order byte of 16-bit extended address.
mm = 8-bit mask (set bits to be affected).
rr = Signed relative offset $80 (- 128) to $7F (+ 127). Offset relative to the address following the machine code offset byte.

Condition Codes:

– Bit not changed
0 Always cleared (logic 0).
1 Always set (logic 1).
↕ Bit cleared or set depending on operation.
↓ Bit may be cleared, cannot become set.

REGISTER AND CONTROL BIT SUMMARY



REGISTER AND CONTROL BIT SUMMARY