MSX88

Manual de usuario

1. INTRODUCCIÓN. 3
2. CONJUNTO DE HERRAMIENTAS DEL ENTORNO MSX88. 3
2.1. ASM88. 3
2.2. LINK88 4
2.3. MSX88. 4
3. DESCRIPCIÓN DEL MSX88. 6
3.1. BLOQUES CONSTITUTIVOS. 6
3.2. PANTALLAS Y CONFIGURACIONES. 6
3.3. MODOS DE FUNCIONAMIENTO 9
4. CPU SX88. 9
4.1. ARQUITECTURA. 9
4.2. LENGUAJE MÁQUINA. 10
4.2.1. MODOS DE DIRECCIONAMIENTO. 10
4.2.2. TIPOS DE INSTRUCCIONES. 10
4.2.3. CÓDIGO MAQUINA DE LAS INSTRUCCIONES. 11
4.2.4. INSTRUCCIONES DE TRANSFERENCIA. 12
4.2.5. INSTRUCCIONES ARITMÉTICO-LÓGICAS 12
4.2.6. INSTRUCCIONES DE COMPARACIÓN. 18
4.2.7. INSTRUCCIONES DE INCREMENTO/DECREMENTO. 19
4.2.8. INSTRUCCIONES DE MANEJO DE LA PILA. 20
4.2.9. INSTRUCCIONES DE CAMBIO DE FLUJO DE PROGRAMA. 20
4.2.10. INSTRUCCIONES DE GESTIÓN DE LAS INTERRUPCIONES. 21
4.2.11. INSTRUCCIONES DE CONTROL. 22
4.2.12. INSTRUCCIONES DE ENTRADA/SALIDA. 22
5. MEMORIA 23
5.1. ORGANIZACIÓN. 23
6. LAS INTERRUPCIONES DEL SX88 23
6.1. TABLA DE VECTORES DE INTERRUPCIÓN. 24
7. PERIFERIA. 24
7.1. PERIFÉRICOS INTERNOS. 24
7.2. PERIFÉRICOS EXTERNOS. 24
7.3. PIO. 25
7.3.1. Selección de registros. 25
7.3.2. Conexión en el sistema. 25

7.4. HAND. 26
7.4.1. Selección de registros. 26
7.4.2. Conexión en el sistema. 27
7.5. TIMER. 27
7.5.1. Selección de registros. 27
7.5.2. Conexión en el sistema. 28
7.6. CONTROLADOR DE INTERRUPCIONES (PIC). 28
7.6.1. Selección de registros. 29
7.6.2. Conexión en el sistema. 30
7.7. CONTROLADOR DE DMA (CDMA). 30
7.7.1. Selección de registros. 31
7.7.2. Conexión en el sistema. 32
7.7.3. BARRA DE LEDS. 33
7.7.4. Conexión en el sistema. 33
7.8. BARRA DE MICROCONMUTADORES. 33
7.8.1. Conexión en el sistema. 33
7.9. IMPRESORA. 33
7.9.1. Interfaz Centronics simplificado. 33
7.9.2. Conexión en el sistema. 33
8. PROGRAMA MONITOR. 34
8.1. COMANDOS DEL MONITOR. 35
8.1.1. DESCRIPCIÓN DE LOS COMANDOS. 35
9. TECLAS O SECUENCIAS DE FUNCIÓN. 40
10. REFINAMIENTO DE LAS INSTRUCCIONES. 41
10.1. INSTRUCCIONES DE TRANSFERENCIA. 41
10.2. INSTRUCCIONES ARITMÉTICAS, Y LÓGICAS DE DOS OPERANDOS. 43
10.3. INSTRUCCIONES DE COMPARACIÓN. 47
10.4. INSTRUCCIONES DE INCREMENTO, DECREMENTO, Y LÓGICAS DE UN OPERANDO. 47
10.5. INSTRUCCIONES DE MANEJO DE PILA. 48
10.6. INSTRUCCIONES DE CAMBIO DE FLUJO DEL PROGRAMA. 49
10.7. INSTRUCCIONES ASOCIADAS A SUBRUTINAS. 50
10.8. INSTRUCCIONES DE GESTIÓN DE LAS INTERRUPCIONES. 50
10.9. INSTRUCCIONES DE ENTRADA/SALIDA. 51
10.10. INSTRUCCIONES DE CONTROL 53

Condiciones de utilización del programa.

El programa que describe esta documentación, MSX88 Versión 3.0, se puede utilizar por cualquier centro de enseñanza público del Estado Español, y por usuarios particulares siempre y cuando se haga sin ánimo de lucro, conforme reza en la autorización de distribución otorgada al GATE-UPM.

Cualquier otro tipo de utilización debe ser autorizada por su autor.

El programa se distribuye como es. El autor no se responsabiliza de cualquier problema que pueda surgir en el ordenador o en el software que éste contenga como consecuencia directa o indirecta de la ejecución de msx88 .

Los usuarios que deseen darse de alta para recibir información periódica de publicaciones y documentación relativa a nuevas versiones y productos MSX88 , deben hacerlo enviando sus datos a la dirección de correo electrónico "msx88@diatel.upm.es" o al número de FAX +34 91-336 78 17. a nombre de Rubén de Diego Martínez.

Igualmente, los profesores que estén utilizando, o deseen utilizar msx88, podrán solicitar soporte -en cualquier aspecto- sobre msx88 en las direcciones de correo electrónico o FAX arriba indicados. No se soportarán dudas o problemas de personas que no acrediten su condición de profesor.

El autor se reserva el derecho de realizar cualquier tipo de modificación en el software y en la documentación sin previo aviso.

Toda la documentación y ultimas versiones que se pongan a disposición pública estarán accesibles vía ftp en los hosts:

- verne.diatel.upm.es (versiones más actualizadas)
- ftp.diatel.upm.es

La OFFICIAL HOME PAGE de msx88 en internet es:

HTTP://VERNE.DIATEL.UPM.ES/MSX88

En esta URL se encontrará toda la información de msx88 relativa a últimas versiones, manuales, evolución de msx88…

El autor se reserva el derecho a modificar las especificaciones y las condiciones de utilización cuando crea conveniente.

1. INTRODUCCIÓN.

El presente documento, no pretende ser más que una guía que proporcione una visión general del entorno didáctico MSX88, facilitando al usuario, el apoyo necesario tanto en la primera toma de contacto con el mismo, como en las sucesivas sesiones de trabajo, a través de las cuales irá progresando.

La presente documentación es una documentación preliminar que está siendo corregida y actualizada. En la documentación definitiva se incluirán varios ejemplos de utilización y una guía didáctica de utilización del MSX88.

El autor desea agradecer a: Ernesto J. Sánchez Gil, Jesús Dueñas Palomino, Jose´Moro Narvaez, Angel Pinardo Anguita, Juan M. Marrón Mariñez, Carlos Portasany Sánchez, Mario del Pozo Baselga, Justo Martínez Fernández, Eloy Rodriguez Villa, José Luis del Amo Muñoz, Jorge Ruiz Cazorla , Mª de la Palma García Hernández y Carlos la Torre Moncayo, programadores del MSX88 desde sus primeras versiones. El autor también desea agradecer todas las colaboraciones anónimas en forma de sugerencias y buenos consejos de todos los alumnos y profesores usuarios del MSX88 que han permitido pulir los fallos de este programa.

En primer lugar, se presentará brevemente el conjunto de herramientas que conforman el entorno, para a continuación pasar a describir el núcleo de la aplicación, distinguiendo sus bloques constitutivos.

2. CONJUNTO DE HERRAMIENTAS DEL ENTORNO MSX88.

Actualmente, el kit de aprendizaje MSX88, está compuesto por las siguientes herramientas:

- ASM88: Ensamblador para la CPU SX88.
- LINK88: Programa montador para el MSX88
- MSX88: Emulador del sistema microcomputador, cuya CPU es SX88.

2.1. ASM88.

La CPU SX88, como cualquier otra, tiene su propio ensamblador, éste es el ASM88., que es capaz de ensamblar cualquier programa fuente escrito en lenguaje de ensamble del SX88, generando dos ficheros a partir del primero; un fichero objeto con extensión .OBJ, destinado a ser procesado únicamente por el montador LINK88, y un fichero listado con extensión .LST, cuya generación es opcional.

El ASM88 es un ensamblador cruzado, es decir, se ejecuta en una máquina distinta - el PC bajo MS-DOS- , de en la que correrán los programas que ensambla - el MSX88-. El código que produce es no reubicable, se ha de elegir una ubicación fija para los programas y datos dentro del espacio de memoria definido en el MSX88, de esta manera siempre se tiene un control total sobre lo que se está haciendo.

Su invocación se realiza desde el DOS, tecleando;

- ASM88 [nom_prog _fuente] [nom_prog_lst] <CR>

siendo;

[nom_progm_fuente], el nombre del fichero fuente a ensamblar , pudiéndose omitir la extensión siempre que ésta sea .ASM, ya que el ensamblador busca por defecto un fichero con extensión .ASM. El hecho de colocar nom_prog_fuente entre corchetes ([]),

indica que éste es opcional, ya que de no ser suministrado en la invocación, será el propio ensamblador quien lo solicitará.

[nom_prog_lst]; el nombre del fichero listado sin extensión, siendo éste al igual que el anterior, opcional. Si en la línea de comando, no se proporcionan ni éste ni el anterior, ambos serán solicitados por el ensamblador, de proporcionarse sólo el primero, el fichero listado no se generará.

2.2. LINK88

LINK88 es el nombre del programa montador de los programas objetos producidos por el ASM88., que al igual que el último, es un montador cruzado.

No tiene funciones de enlazador, pues el tamaño de los programas a ejecutar en el MSX88, hace que carezca de sentido la realización de una programación modular. Pero aún así, se mantiene esta herramienta dentro del entorno, para mantener la coherencia con el resto de los entornos de desarrollo software existentes, consiguiendo de esta manera, que el modo de trabajo con el MSX88 sea lo más parecido al de con cualquier sistema real.

A pesar de que los ficheros objetos y ejecutables tendrán un aspecto muy similar. el MSX88, no es capaz de cargar los programas si estos no han sido procesados antes por LINK88.

Para invocar a este programa montador, se tecleará desde el DOS;

- *LINK88 [nom_prog_obj] <CR>*

siendo *[nom_progm_obj],* el nombre del fichero objeto, que se quiere montar, sin extensión., debiendo ser ésta *.OBJ.* Los corchetes (*[]),* indican que éste es opcional, ya que de no ser suministrado en la invocación, será el propio ensamblador quien lo solicitará.

2.3. MSX88.

MSX88 es el programa que constituye el núcleo del entorno de herramientas. Es un emulador de un sistema digital basado en microprocesador, cuya CPU denominada SX88, se puede ver como un 8088 simplificado.

Desde sus comienzos, en los que tan sólo contaba de CPU y memoria, MSX88 ha sufrido una notable evolución, pasando actualmente a incorporar un bloque de periferia, admitiendo diversos conexionados y configuraciones. Aún así, lo que se ha conservado como una constante a lo largo de las diversas versiones de este emulador, es la equivalencia, en la medida de lo posible, de sus elementos con CPUs y periféricos de la familia añadir, nunca quitar u olvidar, a lo aprendido sobre el MSX88.

Para arrancar el programa, tan sólo es necesario teclear desde el DOS;

- *MSX88 <CR>*

Tras ésto, en la pantalla aparecerá una carátula, que desaparecerá pasados unos instantes, para dar paso a la pantalla principal o *pantalla 0* (figura 1). En ella, se muestra la CPU, el bloque de memoria, una ventana de comunicación con el usuario, que se comporta como una pantalla de un ordenador, y un bloque de periferia con el que se pretende hacer referencia al conjunto de periféricos que forman parte de este sistema microprocesador *software*.

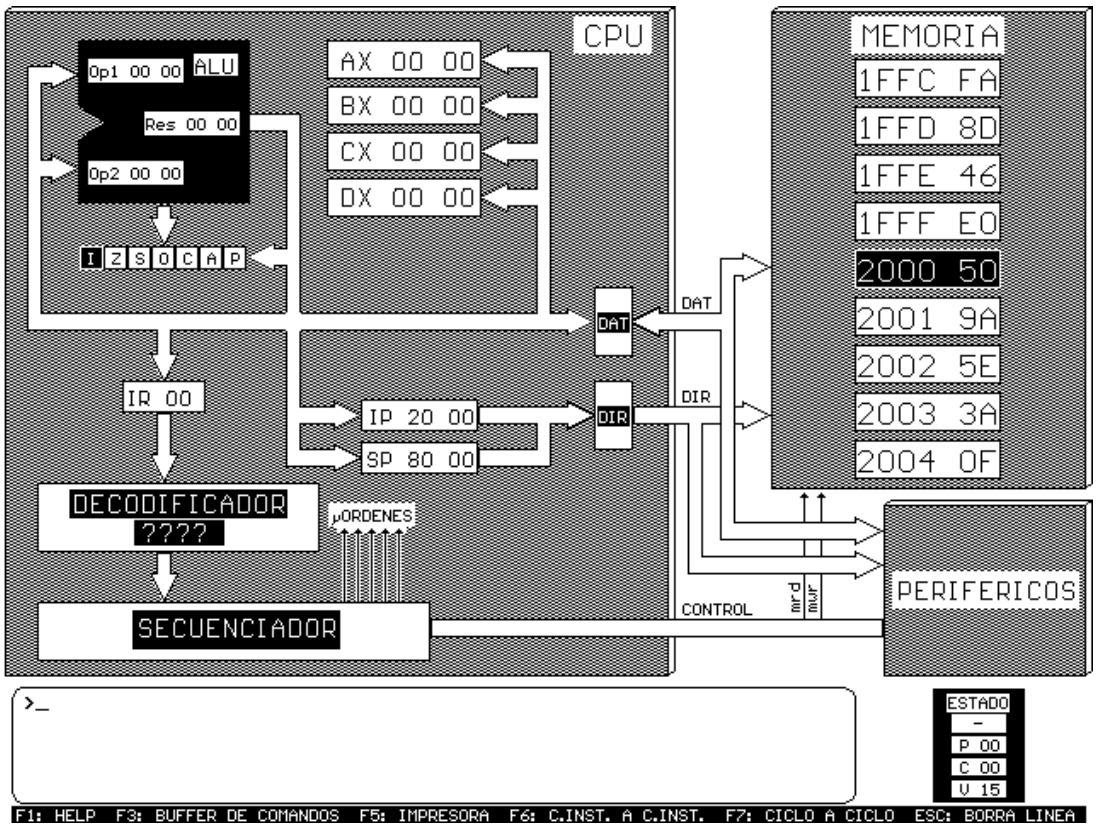

Figura - 1 : Pantalla principal del MSX88.

Los Periféricos que posee el MSX88 son:

- **PIO**: Su comportamiento es similar al Intel 8255 programado en modo 0.
- **Periférico de Handshaking:** Similar al modo 2 del Intel 8255.
- **Controlador de Interrupciones(PIC)**: Pretende asemejarse al Intel 8259.
- **Barra de Leds.**
- **Barra de Microconmutadores:** Accionables desde el teclado.
- **Controlador de Acceso Directo a Memoria (CDMA):** Semejante al Intel 8237.
- **Impresora:** Se conecta a diversos dispositivos a través de un Interfaz Centronics, que deberá ser programado por el usuario.

Como se puede advertir, el MSX88, además de poseer una serie de periféricos propios, compartirá un conjunto de periféricos con el PC donde esté corriendo en un determinado momento, siendo éstos:

- **Teclado:** De donde recibirá las órdenes procedentes del usuario.
- **Pantalla:** A través de la que se presentan las diferentes pantallas del programa, permitiendo al usuario llevar a cabo el seguimiento de la simulación.
- **Unidades de Disco:** Desde donde el MSX88 podrá cargar programas ejecutables especialmente creados, desde el DOS, para el MSX88, y será aquí, donde almacenará el contenido de parte o de la totalidad de la memoria, cuando el usuario así lo solicite.

3. DESCRIPCIÓN DEL MSX88.

3.1. BLOQUES CONSTITUTIVOS.

En esencia, se puede decir, que MSX88 está constituido por los siguientes cuatro grandes bloques:

- **CPU SX88:** Versión simplificada de la CPU 8088 de Intel.
- **Memoria:** En total existen 64 Kbytes.
- **Periferia:** Múltiples conexiones son posibles entre ésta y los dos bloques anteriores.
- **Programa monitor:** Pequeño sistema operativo de que dispone MSX88.

3.2. PANTALLAS Y CONFIGURACIONES.

Las distintas formas de conectar el conjunto CPU-Memoria con los periféricos, determinan las diferentes configuraciones posibles en MSX88. A su vez, cada uno de éstos modos de conexionado, se muestra al usuario a través de dos pantallas; ***pantalla 0*** y ***pantalla 1.*** La primera de ellas, es la pantalla principal que incialmente se muestra tras arrancar el programa (figura-1), siendo común a todas las configuraciones. En ésta, se representa la CPU mostrando toda su arquitectura interna, la memoria, y el bloque de periferia que a la vez que evidencia su presencia, oculta su complejidad, junto con buses que constituyen los caminos de comunicación entre éstos bloques. Se concibe como una pantalla básica orientada a los usuarios que se inician en el tema, en la que los objetivos de estudio se centrarán en el funcionamiento básico de la CPU.

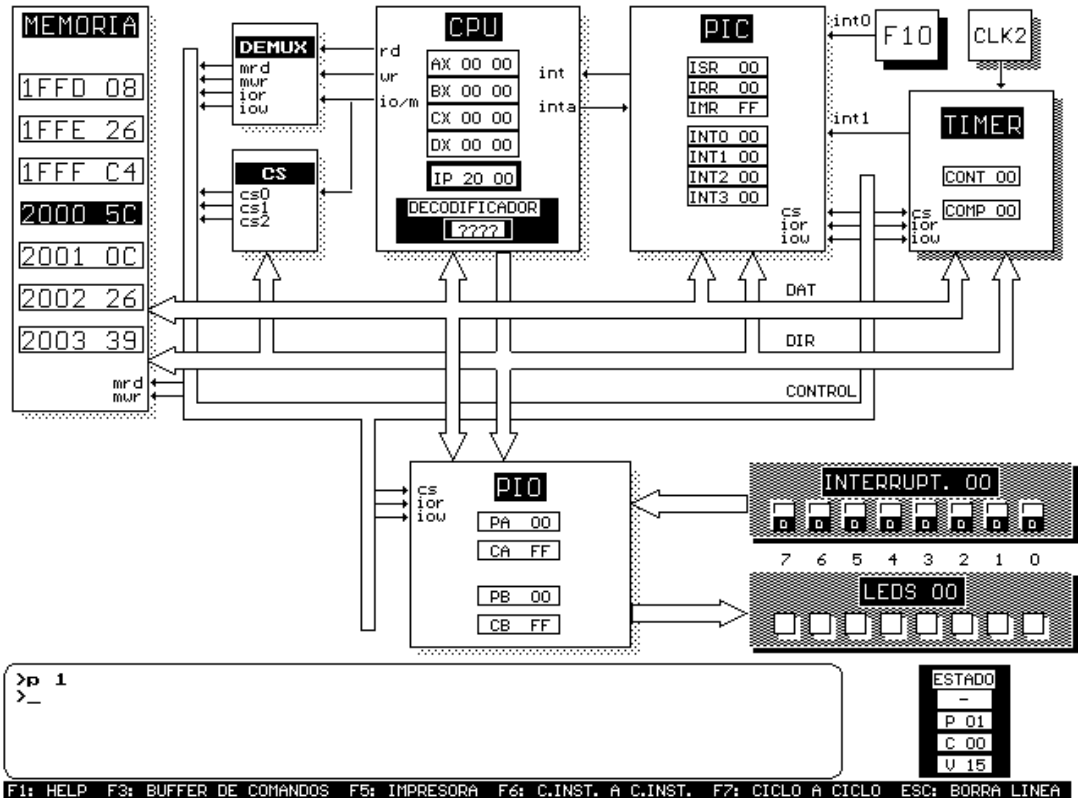

Figura 2: Conexionado básico en MSX88.

La *pantalla 1* (figuras 2 a 5) se centra en el estudio de los periféricos. Ésta a diferencia de la anterior, varía en función de la configuración en la que se encuentre el emulador en un determinado momento. Aún así, existen una serie de elementos, situados en la parte superior, que se mantendrán constantes a lo largo de todas las configuraciones; se trata del conjunto CPU-Memoria, representado a la izquierda, y de los periféricos PIC y Timer, que se visualizan en la parte derecha. El resto de los elementos que se visualicen en esta pantalla, vendrán determinados por la configuración o modo de conexionado seleccionado por el usuario.

Se establecen cuatro modos de conexionado en MSX88:

- ***Conexionado 0:*** Es el más simple de todos ellos (figura 2), pues no contempla los elementos relacionados con las interrupciones. A los elementos constantes situados en la parte superior, que serán invariantes para el resto de conexionados, incorpora el PIO conectado a una barra de Leds y a otra de microconmuta-dores.
- ***Conexionado 1:*** Este, (figura 3) sustituye la barra de Leds y microconmutadores de la configuración anterior por una impresora con un Interfaz Centronics, que habrá de ser implementado por el usuario.

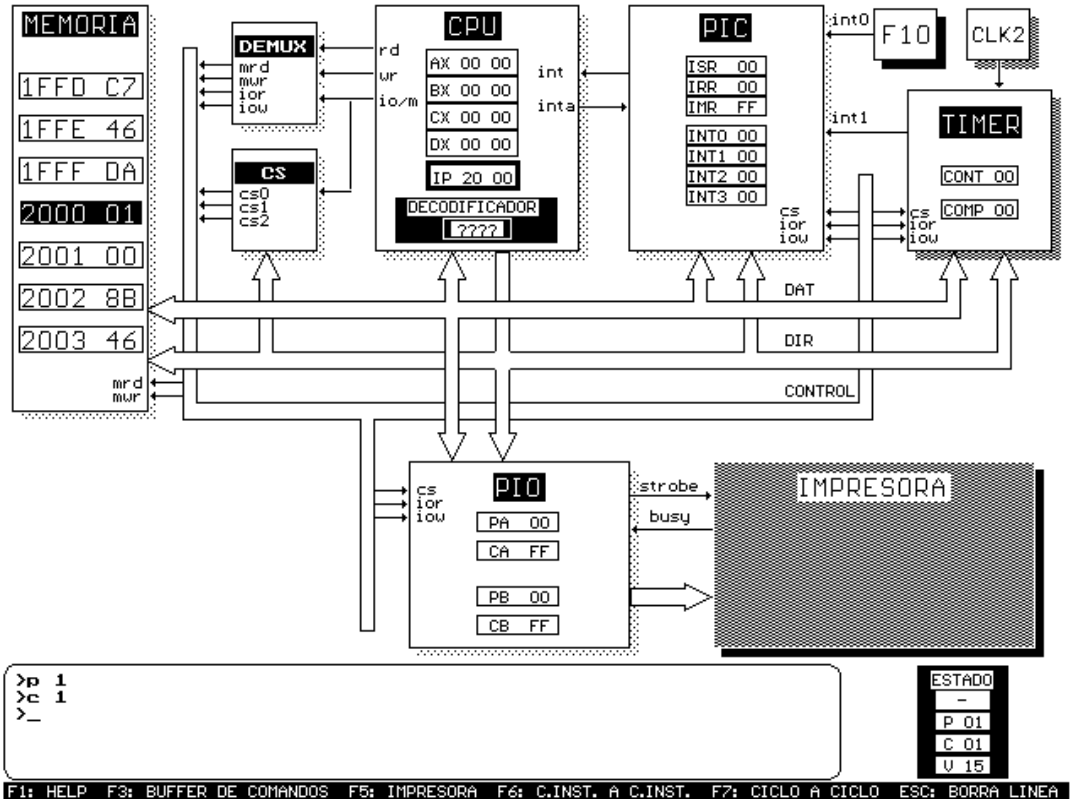

Figura 3: Conexionado 1 de la periferia en MSX88.

- ***Conexionado 2:*** Esta configuración (figura 4) es idéntica a la anterior con la salvedad de haber sustituido el PIO por un dispositivo que hace handshaking, implementando el Interfaz Centronics necesario para conectarse a la impresora. La relación entre éste y la CPU se lleva a cabo bien a través de interrupciones, o bien por consulta de estado.
- ***Conexionado 3:*** Es la configuración más compleja de todas (figura 5), pudiéndose ver como un *conexionado 2*, al que se le ha incorporado un CDMA conectado al Hand y a los buses del sistema.

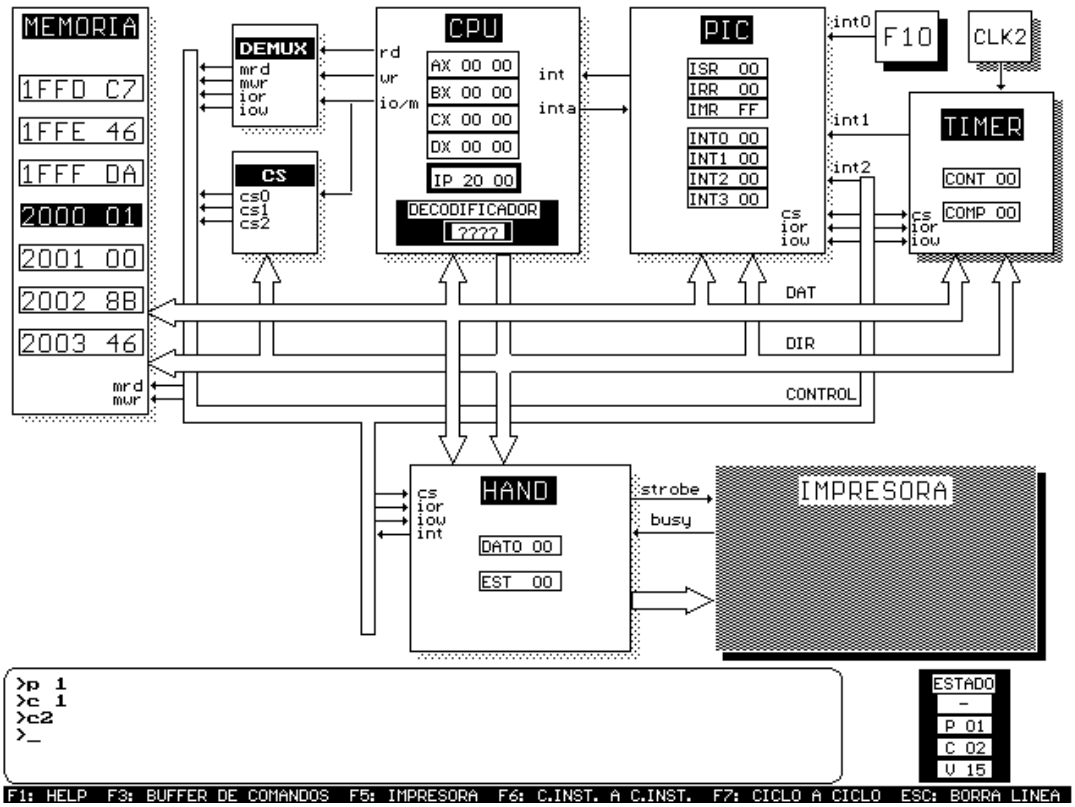

Figura 4: Conexionado 2 de la periferia en MSX88.

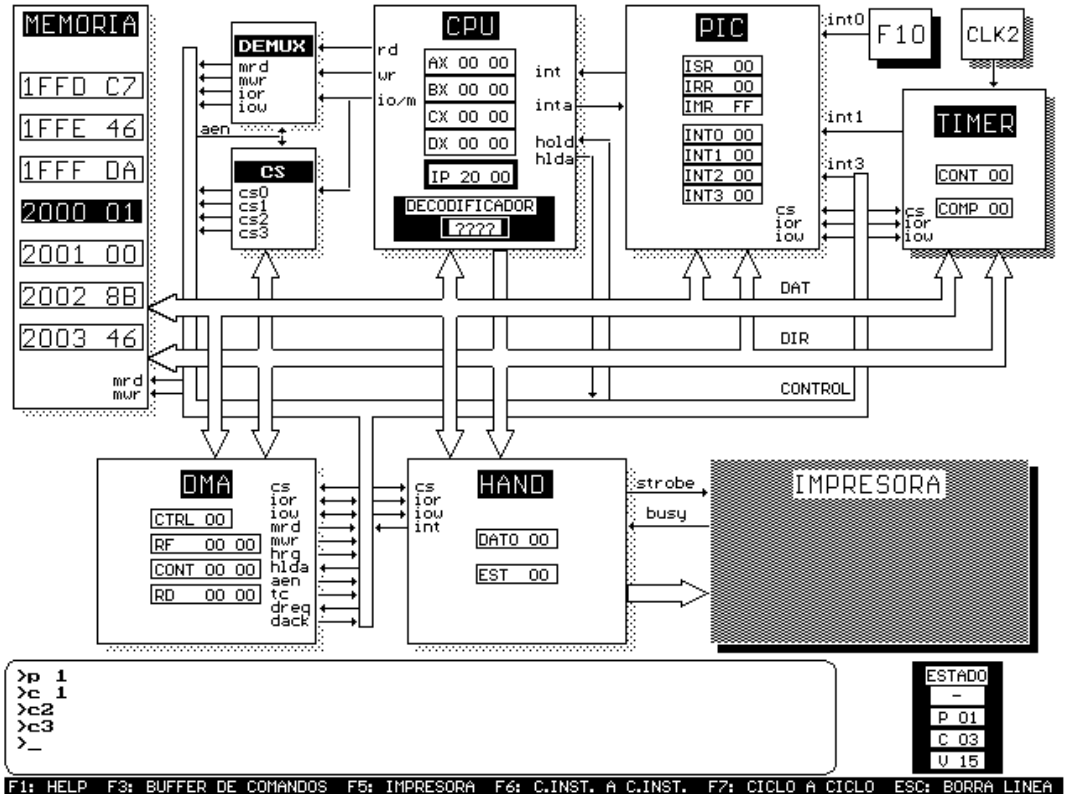

Figura 5: Conexionado 3 de la periferia en MSX88.

3.3. MODOS DE FUNCIONAMIENTO.

MSX88 ofrece al usuario múltiples formas de analizar la evolución dinámica de los distintos elementos que conforman el sistema microprocesador simulado, durante la ejecución de un programa, pensado para la CPU SX88,. o parte de éste (instrucciones, ciclos de instrucción , ciclos de máquina), dando lugar a los diversos modos de funcionamiento que presenta esta herramienta. Estos son:

- ***Modo Rápido:*** En este modo no se representarán en pantalla más que las variaciones que se produzcan en los periféricos externos (leds, microconmutadores e impresora). El resto de los elementos permanecerán en el estado anterior al comienzo de la ejecución del programa, hasta que éste concluya, momento en el que se actualizarán con el estado alcanzado al finalizar dicha ejecución.
- ***Modo Lento:*** En este modo se mostrará la evolución del estado de todos y cada uno de los elementos presentes en el sistema microcomputador, incluyendo los flujos de información de los buses y las señales existentes, que se rellenan de forma continua con colores distintos a los que tienen cuando no están activados, pudiéndose variar esta velocidad de activación a petición del usuario.
- ***Modo Registros:*** Se puede ver como un *modo rápido* en el que además se visualizarán los cambios que experimenten los registros y las posiciones de memoria, durante la ejecución de un programa.
- ***Modo Líneas:*** Se define como un *modo registros*, al que se une la activación y desactivación de las líneas de protocolo asociadas a la periferia.
- ***Modo Periféricos:*** Se trata de un *modo registros* en el que además se visualizan las señales relacionadas con la periferia, junto con las escrituras y lecturas de puertos.

Junto a estos cinco modos, existe la posibilidad de omitir o no la representación del ciclo de fetch (ciclo de búsqueda del código de operación) de cada una de las instrucciones que se ejecutan. En caso de omitirse, dicha representación se reduce a un único acceso a memoria.

4. CPU SX88.

4.1. ARQUITECTURA.

La arquitectura de esta CPU pretende ser una simplificación de la del 8088, siendo sus rasgos arquitecturales más destacados los siguientes:

- Arquitectura interna de 16 bits y externa de 8 bits.
- Bus de direcciones de 16 bits, lo que permite el direccionamiento de 2^{16}= 64 Kbytes de memoria principal.
- Bus de control integrado por las líneas: INT, INTA, NMI, RW, RD, IO/M, HOLD, HLDA.
- Mantiene el mismo esquema de interrupciones que el 8086/88: Líneas NMI e INT, y las interrupciones software INT xx. La línea NMI no está asociada a evento alguno, y la línea INT se conecta a un PIC.
- Registros de uso general AX; BX; CX y DX, de 16 bits, pudiéndose tratar también como registros de 8 bits(AH, AL, BH, BL...).

- ALU de 16 bits capaz de ejecutar las operaciones: ADC, SUB, AND, OR, XOR, NOT, INC y DEC.
- Registro de indicadores con los flags de: Cero, Paridad, Paridad Auxiliar, Signo, Overflow, e indicador de interrupciones**.**
- Registros contador de programa (IP) y puntero de pila (SP) de 16 bits.
- Soporta los siguientes modos de direccionamiento: Dato Inmediato, Registro, Relativo a contador de programa , Relativo a pila, Directo e Indirecto basado en el registro BX.
- Existen además dos registros internos de almacenamiento temporal de 16 bits que aparecen en pantalla únicamente cuando son necesarios.

4.2. LENGUAJE MÁQUINA.

El set de instrucciones que ofrece esta CPU es una parte real del repertorio de instrucciones del microprocesador 8086/88, que recoge solamente las instrucciones más generales y usuales.

4.2.1. MODOS DE DIRECCIONAMIENTO.

Las instrucciones del SX88 están codificadas con cero, uno o dos operandos (registro, memoria, dato inmediato). Las operaciones se hacen entre registros, registros y memoria, datos inmediatos y registros, o entre datos inmediatos y memoria, pero nunca entre memoria y memoria. Los modos de direccionamiento de los operandos indican la manera de calcular la dirección real de éstos.

Los siguientes tipos de direccionamiento son utilizados en las instrucciones de esta CPU para referenciar a los operandos involucrados en la instrucción:

- **Direccionamiento inmediato.**

  El operando es un dato inmediato contenido en la instrucción, con lo que el tamaño de la instrucción viene condicionado por el tamaño del operando.

- **Direccionamiento directo.**

  En la instrucción se indica la dirección real de memoria en la que está contenido el operando. Si el operando ocupa varias posiciones de memoria consecutivas, en la instrucción figurará la dirección más baja.

- **Direccionamiento indirecto a través del registro BX.**

  La dirección de memoria donde está el operando viene determinada por el contenido del registro BX.

- **Direccionamiento relativo a contador de programa**.

  Este tipo de direccionamiento se utiliza en las instrucciones de salto condicional, en las que la dirección de salto se obtiene sumando al registro IP el desplazamiento contenido en la propia instrucción.

4.2.2. TIPOS DE INSTRUCCIONES.

Desde un punto de vista funcional se pueden distinguir los siguientes tipos de instrucciones:

- **Instrucciones de transferencia de datos:** MOV.
- **Instrucciones aritmético-lógicas:**
  - **Instrucciones aritméticas:** ADD, ADC, SUB, SBB.
  - **Instrucciones lógicas:** AND, OR, XOR, NEG, NOT.
- **Instrucciones de comparación:** CMP.
- **Instrucciones de incremento/decremento:** INC, DEC.
- **Instrucciones de manejo de la Pila:** PUSH, POP, PUSHF, POPF.
- **Instrucciones de cambio de flujo de programa:**
  - **Instrucciones de salto incondicional:** JMP.
  - **Instrucciones de salto condicional:** JZ, JNZ, JS, JNS, JC, JNC, JO, JNO.
  - **Instrucciones asociadas a subrutinas:** CALL, RET.
- **Instrucciones de gestión de las Interrupciones:** INT, IRET, STI, CLI.
- **Instrucciones de control:** NOP, HLT.
- **Instrucciones de entrada/salida:** IN, OUT.

4.2.3. CÓDIGO MAQUINA DE LAS INSTRUCCIONES.

A continuación, se muestra el código máquina de cada una de las instrucciones de la CPU SX88. En su codificación, se han utilizado las siguientes abreviaturas:

- **W:** Indica el tamaño de los operandos:

W	SIGNIFICADO
0	operandos de 8 bits
1	operandos de 16 bits

r R	r R	r R	W = 0	W = 1
0	0	0	AL	AX
0	0	1	CL	CX
0	1	0	DL	DX
0	1	1	BL	BX
1	0	0	AH	SP
1	0	1	CH	- -
1	1	0	DH	- -
1	1	1	BH	--

- **rrr, RRR:** Referencian a los registros (**registro, REGISTRO,** respectivamente) implicados en la instrucción, de acuerdo a la siguiente tabla:

4.2.4. INSTRUCCIONES DE TRANSFERENCIA.

MOV memoria, registro

· *Direccionamiento directo:*

1000100W	00rrr110	LSB dir	MSB dir

· *Direccionamiento indirecto:*

1000100W	00rrr111

MOV registro, memoria

Direccionamiento directo:

1000101W	00rrr110	LSB dir	MSB dir

· *Direccionamiento indirecto:*

1000101W	00rrr111

MOV memoria, dato inmediato

· *Direccionamiento directo:*

1100011W	00000110	LSB dir	MSB dir	LSB dato	MSB dato

Direccionamiento indirecto:

1100011W	000000131	LSB dato	MSB dato

MOV registro, dato inmediato

1011Wrrr	LSB dato	MSB dato

MOV registro, REGISTRO

1000101W	11rrrRRR

4.2.5. INSTRUCCIONES ARITMÉTICO-LÓGICAS.

INSTRUCCIONES ARITMÉTICAS.

ADD memoria, registro

· *Direccionamiento directo:*

0000000W	00rrr110	LSB dir	MSB dir

· *Direccionamiento indirecto:*

0000000W	00rrr111

ADD registro, memoria

· *Direccionamiento directo:*

0000001W	00rrr110	LSB dir	MSB dir

· *Direccionamiento indirecto:*

0000001W	00rrr111

ADD memoria, dato inmediato

· *Direccionamiento directo:*

1000000W	00000110	LSB dir	MSB dir	LSB dato	MSB dato

· *Direccionamiento indirecto:*

1000000W	00000111	LSB dato	MSB dato

ADD registro, dato inmediato

1000000W	11000rrr	LSB dato

ADD registro, REGISTRO

0000001W	11rrrRRR

ADC memoria, registro

· *Direccionamiento directo:*

0001000W	00rrr110	LSB dir	MSB dir

· *Direccionamiento indirecto:*

0001000W	00rrr111

ADC registro, memoria

· *Direccionamiento directo:*

0001001W	00rrr110	LSB dir	MSB dir

· *Direccionamiento indirecto:*

0001001W	00rrr111

ADC memoria, dato inmediato

· *Direccionamiento directo:*

1000000W	00010110	LSB dir	MSB dir	LSB dato	MSB dato

· *Direccionamiento indirecto:*

1000000W	00010111	LSB dato	MSB dato

ADC registro, dato inmediato

1000000W	11010rrr	LSB dato	MSB dato

ADC registro, REGISTRO

0001001W	11rrrRRR

SUB memoria, registro

· *Direccionamiento directo:*

0010100W	00rrr110	LSB dir	MSB dir

· *Direccionamiento indirecto:*

0010100W	00rrr111

SUB registro, memoria

· *Direccionamiento directo:*

0010101W	00rrr110	LSB dir	MSB dir

· *Direccionamiento indirecto:*

0010101W	00rrr111

SUB memoria, dato inmediato

· *Direccionamiento directo:*

1000000W	00101110	LSB dir	MSB dir	LSB dato	MSB dato

· *Direccionamiento indirecto:*

1000000W	00101111	LSB dato	MSB dato

SUB registro, dato inmediato

1000000W	11101rrr	LSB dato	MSB dato

SUB registro, REGISTRO

0010101W	11rrrRRR

SBB memoria, registro

· *Direccionamiento directo:*

0001100W	00rrr110	LSB dir	MSB dir

· *Direccionamiento indirecto:*

0001100W	00rrr111

SBB registro, memoria

· *Direccionamiento directo:*

0001101W	00rrr110	LSB dir	MSB dir

· *Direccionamiento indirecto:*

0001101W	00rrr111

SBB memoria, dato inmediato

· *Direccionamiento directo:*

1000000W	00011110	LSB dir	MSB dir	LSB dato	MSB dato

· *Direccionamiento indirecto:*

1000000W	00011111	LSB dato	MSB dato

SBB registro, dato inmediato

1000000W	11011rrr	LSB dato	MSB dato

SBB registro, REGISTRO

0001101W	11rrrRRR

· INSTRUCCIONES LÓGICAS.

AND memoria, registro

· *Direccionamiento directo:*

0010000W	00rrr110	LSB dir	MSB dir

· *Direccionamiento indirecto:*

0010000W	00rrr111

AND registro, memoria

· *Direccionamiento directo:*

0010001W	00rrr110	LSB dir	MSB dir

· *Direccionamiento indirecto:*

0010001W	00rrr111

AND memoria, dato inmediato

· *Direccionamiento directo:*

1000000W	00100110	LSB dir	MSB dir	LSB dato	MSB dato

· *Direccionamiento indirecto:*

1000000W	00100111	LSB dato	MSB dato

AND registro, dato inmediato

1000000W	11100rrr	LSB dato	MSB dato

AND registro, REGISTRO

0010001W	11rrrRRR

OR memoria, registro

· *Direccionamiento directo:*

0000100W	00rrr110	LSB dir	MSB dir

· *Direccionamiento indirecto:*

0000100W	00rrr111

OR registro, memoria

· *Direccionamiento directo:*

0000101W	00rrr110	LSB dir	MSB dir

· *Direccionamiento indirecto:*

0000101W	00rrr111

OR memoria, dato inmediato

· *Direccionamiento directo:*

1000000W	00001110	LSB dir	MSB dir	LSB dato	MSB dato

· *Direccionamiento indirecto:*

1000000W	00001111	LSB dato	MSB dato

OR registro, dato inmediato

1000000W	11001rrr	LSB dato	MSB dato

OR registro, REGISTRO

0000101W	11rrrRRR

XOR memora, registro

· *Direccionamiento directo:*

0011000W	00rrr110	LSB dir	MSB dir

· *Direccionamiento indirecto:*

0011000W	00rrr111

XOR registro, memoria

· *Direccionamiento directo:*

0011001W	00rrr110	LSB dir	MSB dir

· *Direccionamiento indirecto:*

0011001W	00rrr111

XOR memoria, dato inmediato

· *Direccionamiento directo:*

1000000W	00110110	LSB dir	MSB dir	LSB dato	MSB dato

· *Direccionamiento indirecto:*

1000000W	00110111	LSB dato	MSB dato

XOR registro, dato inmediato

1000000W	11110rrr	LSB dato	MSB dato

XOR registro, REGISTRO

0011001W	11rrrRRR

NEG memoria

· *Direccionamiento directo:*

1111011W	00011110	LSB dir	MSB dir

· *Direccionamiento indirecto:*

1111011W	00011110	LSB dir	MSB dir

NEG registro

1111011W	00011111

NOT memoria

· *Direccionamiento directo:*

1111011W	00010110	LSB dir	MSB dir

· *Direccionamiento indirecto:*

1111011W	00010111

NOT registro

1111011W	11010rrr

4.2.6. INSTRUCCIONES DE COMPARACIÓN.

CMP memoria, registro

· *Direccionamiento directo:*

0011100W	00rrr110	LSB dir	MSB dir

· *Direccionamiento indirecto:*

0011100W	00rrr111

CMP registro, memoria

· *Direccionamiento directo:*

0011101W	00rrr110	LSB dir	MSB dir

· *Direccionamiento indirecto:*

0011101W	00rrr111

CMP memoria, dato inmediato

· *Direccionamiento directo:*

1000000W	00111110	LSB dir	MSB dir	LSB dato	MSB dato

· *Direccionamiento indirecto:*

1000000W	00111111	LSB dato	MSB dato

CMP registro, dato inmediato

1000000W	11111rrr	LSB dato	MSB dato

CMP registro, REGISTRO

0011101W	11rrrRRR

4.2.7. INSTRUCCIONES DE INCREMENTO/DECREMENTO.

INC memoria

· *Direccionamiento directo:*

1111111W	00000110	LSB dir	MSB dir

· *Direccionamiento indirecto:*

1111111W	00000111

INC registro

1111111W	11000rrr

DEC memoria

· *Direccionamiento directo:*

1111111W	00001110	LSB dir	MSB dir

· *Direccionamiento indirecto:*

1111111W	00001111

DEC registro

1111111W	11001rrr

4.2.8. INSTRUCCIONES DE MANEJO DE LA PILA.

PUSH registro

01010rrr

POP registro

01011rrr

PUSHF

10011100

POPF

10011101

4.2.9. INSTRUCCIONES DE CAMBIO DE FLUJO DE PROGRAMA.

· INSTRUCCIÓN DE SALTO INCONDICIONAL.

JMP dirección

11101001	LSB dir	MSB dir

· INSTRUCCIONES DE SALTO CONDICIONAL.

JZ desplazamiento

01110100	desplazam.

JNZ desplazamiento

01110101	desplazam.

JS desplazamiento

01111000	desplazam.

JNS desplazamiento

01111001	desplazam.

JC desplazamiento

01110010	desplazam.

JNC desplazamiento

01110011	desplazam.

JO desplazamiento

01110000	desplazam.

JNO desplazamiento

01110001	desplazam.

INSTRUCCIONES ASOCIADAS A SUBRUTINAS.

CALL dirección

11101000	LSB dir	MSB dir

RET

11000011

4.2.10.INSTRUCCIONES DE GESTIÓN DE LAS INTERRUPCIONES.

INT número

11001101	número

IRET

11001111

STI

11111011

CLI

11111010

4.2.11.INSTRUCCIONES DE CONTROL.

NOP

10011000

HLT

11110100

4.2.12.INSTRUCCIONES DE ENTRADA/SALIDA.

IN AL, dirección

11100100	dirección

IN AX, dirección

11100101	dirección

OUT dirección , AL

11100110	dirección

OUT dirección, AX

11100111	dirección

IN AL, DX

11101100

IN AX, DX

11101101

OUT DX, AL

11101110

OUT DX, AX

11101111

5. MEMORIA.

La memoria del MSX88 se representa en pantalla como un bloque en cuyo interior existe una ventana de 9 posiciones. La posición central de ésta, representa el octeto que está siendo direccionado en cada momento de la ejecución de un programa.

Además, en la pantalla principal, es posible visualizar una segunda ventana idéntica a la descrita, concibiéndose ésta última como un área de datos, que a diferencia de la primera, durante la ejecución de un programa, permanecerá fijada en torno a la posición especificada por el usuario.

5.1. ORGANIZACIÓN.

El MSX88 posee un total de 64 Kbytes direccionados linealmente.El usuario tiene a su disposición, un total de 32 Kbytes (desde la posición 0000H hasta la 7FFFH), estando los 32 Kbytes de memoria alta ocupados por el programa monitor (sistema operativo). Por ello, las escrituras por encima de la dirección 7FFFH pueden dar lugar a comportamientos anómalos de la aplicación.

En el primer Kbyte se encuentra la tabla de vectores de interrupción, tal como ocurre en el microprocesador en el que se inspira.

6. LAS INTERRUPCIONES DEL SX88

Las interrupciones de la CPU SX88 son las mismas que las del 8086/88, siendo éstas las siguientes:

- **Hardware:** Líneas INT y NMI. Conectadas a distintos periféricos en función de la configuración.
- **Software**: Instrucción INT xx.

El proceso de atención de una interrupción es idéntico al usado por la CPU 8086/88, siendo todas vectorizadas, excepto NMI que es de salto indirecto.

6.1. TABLA DE VECTORES DE INTERRUPCIÓN.

La tabla de vectores de interrupción es el nexo de unión entre un tipo de interrupción (0...255) y el procedimiento que ha sido designado para atenderla. Cada interrupción vectorizada tiene un código que la identifica para la CPU. Existen por lo tanto 256 entradas en la tabla, una para cada tipo de interrupción.

Cada entrada de la tabla es una doble palabra (4 bytes) que contiene la dirección del procedimiento que va a dar servicio al tipo de interrupción correspondiente. La palabra (16 bits) situada en la dirección más alta de memoria estará siempre a cero debido a que la memoria del MSX88 sólo consta de un segmento de 64 Kbytes, mientras que la segunda es donde el usuario deberá cargar la dirección lógica (en el SX88 coincide con la dirección física) de comienzo de la rutina de tratamiento de la interrupción que corresponda.

Las 256 posibles interrupciones, correspondientes a los códigos del 0 al 0FFH, están organizadas en la siguiente forma:

- Cuatro interrupciones software que ocupan las posiciones correspondientes a los tipos 0, 3, 6, y 7.:
  - · TIPO 0: Finaliza la ejecución de un programa.
  - · TIPO 3: Sirve para poner puntos de parada después de la ejecución de instrucciones concretas.
  - · TIPO 6: Espera para leer un carácter del teclado y lo almacena en la posición de memoria cuya dirección se indica en el registro BX.
  - · TIPO 7: Escribe en la pantalla de comandos un bloque de datos. La dirección de comienzo del bloque se deberá cargar en el registro BX ,y el número de datos que componen el bloque en el registro AL.
- 251 interrupciones libres para ser utilizadas por el usuario.

7. PERIFERIA.

Como todo sistema microcomputador, MSX88 posee una serie de periféricos propios, pudiéndose establecer una clara distinción entre *periféricos internos*, y *periféricos externos*.

7.1. PERIFÉRICOS INTERNOS.

Dentro de este grupo se distinguen los siguientes:

- **PIO:** Puertos paralelo de entrada/salida.
- **HAND-SHAKE:** Periférico de Handshaking.
- **PIC:** Controlador de interrupciones.
- **TIMER:** Contador de eventos.
- **CDMA:** Controlador de Acceso directo a memoria.

7.2. PERIFÉRICOS EXTERNOS.

Conjunto de dispositivos integrado por la pantalla, las unidades de disco y el teclado del ordenador donde se ejecuta MSX88, y además por los siguientes:

- **Barra de LEDS.**
- **Barra de Microconmutadores.**
- **Impresora.**

7.3. PIO.

Es una interfaz de entrada/salida de propósito general, que presenta dos puertos paralelo de 8 bits, denominados A y B, pudiéndose programar sus bits individualmente como entrada o salida. Tras un reset, ambos puertos son configurados como entradas.

Posee cuatro registros internos de ocho bits:

· Dos de datos: **PA** y **PB**, destinados a contener el dato presente en los puertos A y B, respectivamente.

· Dos de control: **CA** y **CB**, que permiten programar individualmente los bits de los puertos A y B, respectivamente. La programación, consiste en realizar una operación de escritura sobre el bit correspondiente al que se programa. Escribiendo un **0** el bit quedará configurado como ***salida***, mientras que un **1** lo configura como ***entrada***.

7.3.1. Selección de registros.

El acceso a los registros, tanto en lectura como en escritura, se hace de acuerdo a la siguiente tabla:

A1	A0	Registro
0	0	PA
0	1	PB
1	0	CA
1	1	CB

Para lograr un acceso se debe activar también la señal CS del PIO. El acceso será de escritura o de lectura en función del estado de las señales IOW e IOR..

7.3.2. Conexión en el sistema.

En el sistema, los registros del PIO se sitúan a partir de la dirección **30H**, y la conexión de sus líneas es la que sigue:

- CS: conectada a la salida CS2 de la lógica de selección de la periferia.
- IOR , IOW: se conectan a las líneas del mismo nombre de la lógica de generación de las señales de lectura/escritura.
- D0...D7: conectados a los bits correspondientes del bus de datos del sistema.

- A0¸, A1: conectadas a los bits menos significativos del bus de direcciones (A0 y A1¸, respectivamente).
- PA0...PA7, PB0...PB7: tienen dos posibles conexiones en función de la configuración en que se encuentre el PIO:
  - Configuración 0 (figura 2): se conectan a los correspondientes bits de los buses de los microinterruptores (PA0...PA7), y de los leds (PB0...PB7).
  - Configuración 1 (figura 3): se conectan a un bus Centronics de impresora, correspondiéndose con las siguientes líneas del interfaz:
    · PA0: línea busy .
    · PA1: línea strobe.
    · PB0...PB7: líneas de datos.

7.4. HAND.

Interfaz de periferia que cumple la temporización especificada en el interfaz Centronics, no admitiendo modo de programación alguno. Se comunica con la CPU por consulta de estado o a través de interrupciones, en función del BIT 7 de su registro de estado.

Internamente posee dos registros de 8 bits:

- **DATO:** Registro de datos. Una operación de escritura sobre éste permite sacar un dato a las líneas D0...D7, mientras que una lectura del mismo proporciona el último dato sacado por las líneas D0...D7.
- **EST:** Registro de estado. Su formato es el que se muestra a continuación:

INT	X	X	X	X	X	STR	BUSY

El significado de los bits es el que sigue:

- En lectura:
  · BIT 0: 0: Línea BUSY desactivada.
  1: Línea BUSY activada.
  · BIT 1: 0: Línea STROBE desactivada
  1: Línea STROBE activada
  · BITS 2...6: No tienen sentido.
  · BIT 7: 0: No se activará la línea INT.
  1: Se activará la línea INT cuando la línea BUSY no esté activada.
- En escritura:
  · BIT 0...6: Su contenido es indiferente.
  · BIT 7: 0: Inhibe la activación de la línea INT.
  1: La línea INT se activará al desactivarse la línea BUSY.

7.4.1. Selección de registros.

El acceso a los registros, tanto en lectura como en escritura, se hace de acuerdo a la siguiente tabla:

A0	DATO
0	Registro
1	EST

Para lograr un acceso se debe activar también al señal CS del HAND. El acceso será de escritura o de lectura en función del estado de las señales IOW e IOR..

7.4.2. Conexión en el sistema.

En el sistema, los registros del HAND se sitúan a partir de la dirección **40H,** siendo la conexión de sus líneas la que sigue:

- CS: conectada a la salida CS2 de la lógica de selección de la periferia.
- IOR , IOW: se conectan a las líneas del mismo nombre de la lógica de generación de las señales de lectura/escritura.
- D0...D7: conectados a los bits correspondientes del bus de datos del sistema.
- A0: conectadas al bit menos significativo del bus de direcciones (A0).
- STROBE, BUSY, P0...P7: se conectan a las líneas correspondientes del Interfaz Centronics de la impresora..
- INT: su conexión varía en función de la configuración en que se halle el HAND:.
    - Configuración 2 (figura 4): se conecta a la entrada INT2 del Controlador de Interrupciones (PIC).
    - Configuración 3 (figura 5): se encuentra conectada a la entrada DREQ del CDMA.

7.5. TIMER.

Contador de eventos, que realiza una cuenta ascendente los pulsos de la señal aplicada a su entrada INT, restaurándose el valor inicial de cuenta al final de la misma.

Posee dos registros internos de ocho bits

- **COMP:** Registro de comparación, que determina el módulo de la cuenta del TIMER.
- **CONT:** Registro contador, que muestra la cuenta de los pulsos de la señal aplicada a la entrada INT del periférico. La coincidencia de su valor con el del registro anterior provoca la activación de la salida OUT.

7.5.1. Selección de registros.

El acceso a los registros, tanto en lectura como en escritura, se hace de acuerdo a la siguiente tabla:

A0	Registro
0	CONT
1	COMP

Para lograr un acceso, se debe activar también la señal CS del TIMER El tipo de acceso (escritura o lectura) lo determina el estado de las señales IOW e IOR..

7.5.2. Conexión en el sistema.

En el sistema, los registros de este dispositivo se sitúan a partir de la dirección **10H.** Su conexión tiene lugar en todas y cada una de las configuraciones, en el modo que a continuación se expone:

- CS: esta línea se conecta a la salida CS1 de la lógica de selección de la periferia.
- IOR , IOW: conectadas a las líneas del mismo nombre de la lógica de generación de las señales de lectura/escritura.
- D0...D7: conectados a los bits correspondientes del bus de datos del sistema.
- A0: línea de selección que se conecta al bit menos significativo del bus de direcciones (A0).
- IN: a esta entrada se le aplica un reloj de frecuencia 1 Hz.
- OUT: se conecta a la entrada de interrupción INT1 del PIC, por lo que provocará una interrupción al activarse cuando los valores de los dos registros internos del dispositivo coincidan.

7.6. CONTROLADOR DE INTERRUPCIONES (PIC).

El controlador de interrupciones puede manejar hasta ocho peticiones de interrupción independientes al mismo tiempo, numeradas de la 0 (INT0) a la 7 (INT7), de las cuales seleccionará una única para presentarla a la entrada de interrupción INT de la CPU.

Debido a la independencia existente entre las entradas de interrupción, cabe la posibilidad de que se presenten varias al mismo tiempo, por lo que el controlador, es capaz de mantener memorizadas las interrupciones secundarias mientras el procesador da servicio a la más prioritaria. Si más de una petición de interrupción se producen exactamente al mismo tiempo entonces el PIC las pasa a la CPU en un orden de prioridad, donde la petición por la entrada 0 tiene la prioridad más alta y la de la 7 la menor.

Una consecuencia muy importante de este esquema es que la CPU debe indicarle al controlador cuando ha completado el servicio de cada interrupción. Por lo tanto al final de la rutina de servicio de la interrupción habrá que mandar al registro de comandos del controlador el comando de final de la interrupción (EOI) representado por el valor 20H.

Sus registros internos, todos ellos de 8 bits, son:

- **ISR:** Registro de Interrupción en Servicio. Indica cual es la interrupción que está siendo atendida, mediante la puesta a 1 del bit asociado a esa entrada de interrupción (bit 0 se asocia a la entrada INT0 ... bit 7 se asocia a la entrada INT7).

- **IRR:** Registro de petición de interrupción. Almacena las interrupciones demandadas hasta el momento. Así, al activarse una entrada de interrupción el bit correspondiente se pone a 1, tornándose a 0 cuando ésta pasa a ser atendida (bits 0..7 se asocian a las entradas INT0...INT7, respectivamente).
- **IMR:** Registro de Máscara de Interrupciones. Permite el enmascaramiento selectivo de cada una de las entradas de interrupción mediante la puesta a 1 de su bit asociado (bit 0 se asocia a INT0,... bit 7 se asocia a INT7). Tras un reset (los bits de este registro quedarán establecidos a cero) todas las entradas de interrupción quedarán desenmascaradas.
- **INT0, ...,INT7:** Cada uno de estos registros contiene el valor del vector de interrupción correspondiente a la entrada del mismo nombre.
- **EOI:** Registro de comandos de sólo escritura, donde la CPU deberá mandar el comando de final de interrupción (EOI), representado por el valor 20H, al final de la rutina de tratamiento de la interrupción. El efecto de está escritura es la puesta a cero del bit del registro ISR correspondiente a la interrupción.

7.6.1. Selección de registros.

El acceso, tanto de lectura como de escritura, a los registros del Controlador de Interrupciones, se hace en función de la siguiente combinación de señales de dirección:

A3	A2	A1	A0	Registro
0	0	0	0	EOI
0	0	0	1	IMR
0	0	1	0	IRR
0	0	1	1	ISR
0	1	0	0	INT0
0	1	0	1	INT1
0	1	1	0	INT2
0	1	1	1	INT3
1	0	0	0	INT4
1	0	0	1	INT5
1	0	1	0	INT6
1	0	1	1	INT7

Para lograr un acceso se debe activar también al señal CS del controlador. El acceso será de escritura o de lectura en función del estado de las señales IOW e IOR.

.

7.6.2. Conexión en el sistema.

En el sistema, los registros del PIC se sitúan a partir de la dirección **20H.** Se halla presente en todas y cada una de las configuraciones, siendo la conexión de sus líneas, la siguiente:

- CS: conectada a la salida CS3 de la lógica de selección de la periferia.
- IOR , IOW: se conectan a las líneas del mismo nombre de la lógica de generación de las señales de lectura/escritura.
- DO...D7 conectados a los bits correspondientes del bus de datos del sistema.
- A0, A1, A2, A3: conectadas a los bits menos significativos del bus de direcciones (A0, A1, A2, A3, respectivamente).
- INT, INTA: se conectan a las líneas de la CPU del mismo nombre.
- INT0: se conecta a la tecla F10 que produce una interrupción al ser pulsada.
- INT1: se conecta a la línea OUT del TIMER.
- INT2: en la configuración 2 (figura 4) se conecta a la línea INT del HAND.
- INT3: en la configuración 3 (figura 5) se conecta a la salida TC del CDMA, que se activa para indicar el fin de transferencia.
- INT4...INT7: estas entradas no se utilizan en el sistema.

7.7. CONTROLADOR DE DMA (CDMA).

Ofrece la posibilidad de realizar *transferencias de datos de 8 bits memoria-memoria* o *memoria-periférico y a la inversa*, sin intervención directa de la CPU y en robo de ciclo (la CPU le ha de ceder los buses con el protocolo HRQ/HLDA), siendo el *tamaño máximo del bloque* a transferir de *64Kbytes*.

Posee *un* único *canal* de DMA, pudiendo realizar *transferencias* en *modo bloque* (una vez iniciada la transferencia, transmite datos sin parar hasta que se agota el bloque a transferir) , o bajo *demanda* del periférico al que se encuentre conectado.

Internamente ofrece los siguientes registros:

- **CTRL:** Registro de control. Su formato es el que se muestra a continuación:

TC	X	X	X	MT	ST	TT	STOP

El significado de sus bits varía en función de la operación (lectura o escritura) que sobre él se realice. Así:

- En escritura:
  - C0:
    0: No tiene sentido
    1: Al escribirlo la CPU detiene momentáneamente la transferencia en curso.
  - C1 (TT=Tipo Transferencia):

0: Transferencia Periférico -Memoria, o a la inversa.

1: Transferencia Memoria-Memoria.

- C2 (ST = Sentido Transferencia): Sólo tiene sentido si C1=0. Entonces:

0: Sentido transferencia Periférico-Memoria.

1: Sentido transferencia Memoria-Periférico.

- C3 (MT = Modo Transferencia):

0: Transferencia por demanda.

1: Transferencia en modo bloque.

- C4...C7: No usados. No importa su contenido.

- En lectura:

En general se lee el último valor que se ha escrito en cada uno de los bits salvo para los siguientes:

- C0:

0: Transferencia en curso.

1: Transferencia detenida por la CPU temporalmente.

- C7(TC = Terminal Count):

0: Transferencia no finalizada.

1: Transferencia ya finalizada.

- **RF:** Registro de direcciones Fuente. En transferencias memoria-periférico o a la inversa, se carga en él la dirección del bloque de memoria a transferir o recibir. En transferencias memoria-memoria actúa tal como su nombre indica.
- **RD:** Registro de direcciones Destino. Sólo tiene sentido su utilización en transferencias memoria-memoria, actuando tal como su nombre indica.
- **CONT:** Registro Contador. Indica el número de octetos a transferir.

Además de éstos, dispone de más registros ocultos, necesarios para su funcionamiento.

Una vez cargados los valores adecuados en los registros, para lograr el arranque de la transferencia programada, es necesario forzar los bits A3, A2 y A1 del controlador a 1 a la vez que se activa CS, sin importar el valor de las líneas IOR e IOW, acción que, además pondrá a 0 el bit C0. Si se desea detener momentáneamente la transferencia, será necesario poner C0 a 1, pero su rearranque no se logrará poniendo C0 a 0, sino a través de la condición CS=A0=A1=A2.

7.7.1. Selección de registros.

El acceso, tanto de lectura como de escritura, a los registros del Controlador de DMA, se ha ce en función de la siguiente combinación de señales de direcciones:

A2	A1	A0	Registro
0	0	0	RFL
0	0	1	RFH
0	1	0	CONTL
0	1	1	CONTH
1	0	0	RDL
1	0	1	RDH
1	1	0	CTRL
1	1	1	ARRANQUE

Para lograr un acceso se debe activar también al señal CS del controlador. El acceso será de escritura o de lectura en función del estado de las señales IOW e IOR..

7.7.2. Conexión en el sistema.

En el sistema, los registros del CDMA se sitúan a partir de la dirección **50H.** Únicamente se halla presente en la configuración 3 (figura 5), siendo la conexión de sus líneas la que sigue:

- CS: conectada a la salida CS3 de la lógica de selección de la periferia.
- IOR , IOW: se conectan a las líneas del mismo nombre de la lógica de generación de las señales de lectura/escritura.
- MRD, MWR: se conectan a las líneas de lectura y escritura de la memoria.
- AEN: se aplica a la lógica de selección de la periferia y de generación de las señales de lectura/escritura.
- TC: su activación se realiza cuando el registro contador alcanza el valor cero, momento en el que provocará una interrupción por hallarse conectada a la entrada de interrupción INT3 del Controlador de Interrupciones (PIC).
- DREQ: esta entrada es puesta por la línea INT del HAND, para solicitar una transferencia.
- DACK: se conecta a la línea CS del HAND para indicarle que su petición de DMA se está completando.
- HRQ: se aplica a la entrada HOLD de la CPU para realizar la petición de Bus.
- HLDA: está entrada es puesta por la línea HLDA de la CPU, a través de la cual el Controlador recibe el reconocimiento a la petición de Bus
- DO...D7 conectados a los bits correspondientes del bus de datos del sistema.
- A0, A1, A2: conectadas a los bits menos significativos del bus de direcciones (A0, A1, A2, respectivamente).

7.7.3. BARRA DE LEDS.

Matriz formada por ocho diodos luminiscentes, que se mantienen encendidos o apagados en función del valor de su entrada, activándose al tomar ésta el valor 1, y desactivándose cuando ésta a 0.

7.7.4. Conexión en el sistema.

La conexión de este dispositivo al sistema se lleva a cabo únicamente en la configuración 0 (figura 2) a través de un bus de datos; que agrupa ocho líneas, correspondiéndose cada una de ellas con la entrada de un LED concreto, conectado al puerto PB del PIO.

7.8. BARRA DE MICROCONMUTADORES.

Periférico que consta de ocho conmutadores de dos posiciones. La salida de cada uno de ellos reflejará su estado en cada momento (1 interruptor cerrado, 0 interruptor abierto).

7.8.1. Conexión en el sistema.

La conexión de este dispositivo al sistema se lleva a cabo únicamente en la configuración 0 (figura 2) a través de un bus de datos; que agrupa ocho líneas, correspondiéndose cada una de ellas con la salida de un microconmutador concreto, conectado al puerto PA del PIO.

7.9. IMPRESORA.

Impresora ASCII de 20 columnas, que posee un buffer de tamaño cinco caracteres.

Su interfaz de conexión se corresponde con un Interfaz Centronics gestionado únicamente por las líneas *strobe* y *busy*.

7.9.1. Interfaz Centronics simplificado.

- **Descripción de señales.**

- **STROBE:** Señal de validación de dato en las líneas de datos, indicándole a la impresora que la información en las líneas de datos es estable.

- **DATA1...DATA8:** Lineas de datos. Estas señales mantienen los bits del dato en paralelo desde el primero o menos significativo (DATA1) hasta el octavo o más significativo (DATA8).

- **BUSY:** Indica que la impresora no puede recibir datos por una de las siguientes causas:

  · Está entrando información.

  · La impresora está imprimiendo.

  · El buffer está lleno.

  · Ha ocurrido un error.

7.9.2. Conexión en el sistema.

La conexión de la impresora en el sistema, varía en función de la configuración. Así:

- Configuración 1 (figura 3): En esta configuración la impresora se encuentra conectada al PIO en el siguiente modo:
    - BUSY, STROBE: conectadas a los dos bits menos significativos (bit 0, bit 1, respectivamente) del puerto PA:
    - DATA1...DATA8: conectadas a los bits del correspondiente peso del puerto PB.
- Configuraciones 2 y 3 (figuras 4 y 5): En este caso la conexión se lleva a cabo con el HAND:
    - BUSY, STROBE: se conectan a los dos bits menos significativos (bit 0, bit 1, respectivamente) del registro de estado del HAND.
    - DATA1...DATA8: conectadas a los bits del correspondiente peso del registro de datos del HAND.

8. PROGRAMA MONITOR.

Como cualquier ordenador real, el MSX88 tiene un sistema operativo (S.O.), cargado en la parte alta de la memoria, que se encarga de realizar diversas tareas. Obviamente, este sistema operativo es mínimo y por ello adopta la denominación de *monitor*.

Ofrece la posibilidad de ejecutar diversos comandos y funciones, e incluso soporta algunas utilidades, equivalentes a las *llamadas al sistema* presentes en cualquier S.O., que permiten a un programa de usuario realizar operaciones de entrada/salida sobre el teclado y la pantalla de usuario.

Los comandos del monitor son los equivalentes a los de cualquier programa depurador, adaptados a la filosofía MSX88. Las funciones, en esencia, son comandos asociados a teclas de función, o a combinaciones de éstas.

En todo momento facilita al usuario el uso del simulador, proporcionándole información de estado y de ayuda.

La información de estado la muestra a través de la ventana de estado que se encuentra presente en todas y cada una de las pantallas de la aplicación, dando información de la pantalla y conexionado seleccionados, así como de la velocidad y modo de ejecución.

En cuanto a la información de ayuda, ésta se manifiesta mediante dos vías; una barra de información situada en la parte inferior de cada pantalla, y un *help* a varios niveles.

La barra presente en la parte baja de cada pantalla muestra las teclas o secuencias de función. Su contenido variará para mostrar las funcionalidades más útiles en cada momento (bajo control del monitor, bajo control de un programa de usuario, en la ejecución de ciertos comandos ...).

El *help* que ofrece el monitor consta de varios niveles de ayuda presentados a través de una serie de ventanas desencadenadas en cascada. La información de cada nivel se recoge en una ventana que da acceso al nivel de ayuda inmediatamente más interno por el mero hecho de realizar una selección en ella. En la figura 6 se muestra el aspecto de la pantalla principal al mostrar uno de los niveles del *help*.

8.1. COMANDOS DEL MONITOR.

LISTA DE COMANDOS POR ORDEN ALFABÉTICO.

- + : Modo Rápido.
- - : Modo Lento.
- B : Borrar.
- C : Configuración.
- D : Dump (Visualizar).
- E : Enter.
- F6 : Ciclo instrucción a ciclo instrucción.
- F7 : Ciclo a ciclo.
- G : Go.
- H : Help.
- I : Input.
- L : Load.
- M : Microswitch.
- O : Output.
- P : Pantalla.
- Q : Quit.
- R : Register.
- RESET.

8.1.1. DESCRIPCIÓN DE LOS COMANDOS.

Como se ha dicho, los comandos del programa monitor son similares a los de cualquier programa depurador. Su invocación se llevará a cabo a través de la pantalla de usuario, tecleando una línea de comando que consta de un mnemónico (una o varias letras) indicando el comando de que se trata, seguido de una serie de parámetros, en caso de poseerlos. Cada línea de comando deberá terminarse con un retorno de carro(<CR>), siendo indiferente tanto la utilización de letras mayúsculas o minúsculas, como la inserción de espacios en blanco al comienzo o final de la línea.

En lo que sigue, se describen brevemente cada uno de los comandos indicando su formato y la acción que realizan.

- **+ : Rápido.**

  - **Descripción:**

    Activa el modo de ejecución rápido.

- **Formatos:**

+<CR> : Selecciona el Modo Rápido.

+R<CR> : Selecciona el Modo Registros.

+L<CR> : Selecciona el Modo Líneas.

+P<CR> : Selecciona el Modo Periféricos.

+F<CR> : Restaura la visualización del ciclo de fetch, o la reduce a un único acceso a memoria.

- **- : Lento.**

- **Descripción:**

Selecciona el modo de ejecución lento.

- **Formato:**

-<CR>

- **B : Borrar.**

- **Descripción:**

Borra la pantalla de comandos o la hoja de la impresora.

- **Formatos:**

B<CR> : Borra la pantalla de comandos.

B I<CR> : Borra la hoja de la impresora.

- **C : Configuración.**

- **Descripción:**

Establece la configuración .

- **Formato:**

C <nº configuración><CR>

donde:

· <nº configuración> debe ser ; 0,1,2,ó3.

- **D : Dump.**

- **Descripción:**

Permite visualizar el contenido de la memoria.

- **Formatos:**

D<dirección><CR> : Visualiza <dirección > en la posición central de la memoria.

DD <dirección><CR> :Visualiza <dirección> en la posición central del bloque derecho de la memoria de la pantalla 0.

D D-<CR> : Borra el bloque derecho de memoria de la pantalla 0.

- **E : Enter.**

  - **Descripción:**

    Permite modificar el contenido de la memoria almacenando los bytes que se le indican a partir de <dirección>.

  - **Formato:**

    E <dirección> <byte1> [<byte2>[...[<byteN>]]]1

- **F6 : Ciclo instrucción a ciclo instrucción.**

  - **Descripción:**

    Determina la información que se presentará en la pantalla de comandos en la próxima ejecución *"ciclo instrucción a ciclo instrucción"*.

  - **Formatos:**

    F6+<CR> : Sólo se presenta el número del ciclo de instrucción que se ejecuta.

    F6-<CR> : Se presentará el número y nombre del ciclo de instrucción en ejecución.

- **F7 : Ciclo a ciclo.**

  - **Descripción:**

    Establece la información que se suministrará a través de la pantalla de comandos en la próxima ejecución *"ciclo a ciclo"*.

  - **Formatos:**

    F7+<CR> : Se presentará el número de ciclo máquina que se ejecuta.

    F7-<CR> : Además del número, se mostrará un mnemónico que indica la acción que realiza la CPU en el ciclo máquina.

- **G : Go.**

  - **Descripción:**

    Lanza la ejecución de un programa cargado en memoria a partir de la dirección apuntada por el registro IP y coloca un punto de ruptura en cada dirección de memoria especificada en los parámetros.

  - **Formato:**

    G [<dirección1>[<dirección2>]]<CR>

1 Los corchetes ([ ]) indican opcionalidad en los parámetros.

- **H : Help.**

  - **Descripción:**

    Proporciona un *help* de varios niveles de ayuda presentados sucesivamente a través de ventanas desencadenadas en cascada, cada una de las cuales se abandona pulsando la tecla <*ESC*>.

  - **Formatos:**

    H<CR>

    HELP<CR>

- **I : Input.**

  - **Descripción:**

    Lee y visualiza el contenido de un puerto de entrada.

  - **Formato:**

    I <puerto><CR>

    donde:

    · <puerto> es la dirección del puerto expresada como un número hexadecimal comprendido entre 00H y FFH.

- **L : Load.**

  - **Descripción:**

    Almacena el contenido de un fichero en la memoria.

  - **Formato:**

    L <nom_fich><CR>

    donde:

    · <nom_fich> es el nombre del fichero cuya extensión puede ser:

    - ".EJE": fichero ejecutable obtenido a partir de las herramientas ASM88 y LINK88.

    - ".MEM": fichero obtenido a partir del comando S"(*save*).

    - sin extensión: se entiende que es un fichero ejecutable(".EJE").

- **M : Microswitch.**

  - **Descripción:**

    Permite modificar y obtener el valor de los microinterruptores.

  - **Formato:**

    M [<número>]<CR>

    donde:

· <número> puede ser:

- un dígito de 0 a 7 indicando el número de microinterruptor a conmutar.

- un dato hexadecimal de dos cifras indicando el dato que será representado en los microinterruptores.

- inexistente: en este caso el comando proporciona el valor representado por los microconmutadores.

- **O : Output.**

- **Descripción:**

Envía un byte a un puerto de salida.

- **Formato:**

O <puerto> <byte><CR>

donde:

· <puerto>: especifica la dirección del puerto expresada como un número hexadecimal comprendido entre 00H y FFH.

· <byte>: valor del byte a ser enviado al puerto.

- **P : Pantalla.**

- **Descripción:**

Establece la pantalla de trabajo.

- **Formato:**

P <nº pantalla><CR>

donde:

· <nº pantalla> debe ser 0,ó, 1.

- **Q : Quit.**

- **Descripción:**

Permite abandonar la sesión de trabajo devolviendo el control al S.O.

- **Formato:**

Q<CR>

- **R : Register.**

- **Descripción:**

Permite modificar el contenido de los registros.

- **Formato:**

R <registro> <dato>

donde:

· <registro>: es el registro cuyo contenido se modifica, siendo uno de los siguientes:

AL, AH, AX, BL, BH, BX, CL, CH, CX, DL, DH, DX, IP, SP, RF.

· <dato>: valor que adoptará el registro.

- **RESET**.

  - **Descripción:**

    Inicializa todos los periféricos.

  - **Formato:**

    RESET<CR>

9. TECLAS O SECUENCIAS DE FUNCIÓN.

Con este nombre se conocen a una serie de teclas o combinación de éstas, cuya pulsación lleva asociada la ejecución de determinados comandos.

Dependiendo de si se está bajo control del monitor (modo edición) o de un programa de usuario(modo ejecución), la definición de éstas variará. Así:

- En ***modo edición*** se tienen definidas las siguientes:

  - **F1:** Es una alternativa al comando **"H"**(*help*), que provoca la presentación de ventanas de ayuda a varios niveles.

  - **F3**: Presenta el contenido del buffer de comandos en la pantalla de usuario.

  - **F5:** Muestra y oculta el papel de la impresora.

  - **F6:** Se realiza una *ejecución ciclo de instrucción a ciclo de instrucción*, al ritmo de pulsación de esta tecla, a partir de la dirección especificada por el registro IP (contador de programa)

  - **F7:** Provoca una *ejecución paso a paso* a partir de la dirección apuntada por el registro IP.

  - **ESC:** Borra la línea actual de la pantalla de comandos.

  - **Secuencias de teclas Alt:** Todas ellas tienen en común el proporcionar un acceso directo a ciertos elementos del sistema microprocesador. Una vez que se logra acceder a los mismos, el movimiento a su través se consigue del mismo modo que en un editor de texto.

    - **Alt-D:** Es una alternativa al comando **"D"** (*dump*), que permite explorar el contenido de todas y cada una de las posiciones de memoria, proporcionando acceso directo a la misma.

    - **Alt-E:** Se puede ver como una alternativa al comando **"E"** (*enter*), ya que ofrece acceso directo a la memoria para realizar funciones de edición.

    - **Alt-R:** Se presenta alternativamente al comando **"R"**(*register*), permitiendo alterar el contenido de los registros a través de acceso directo a los mismos.

- Cuando se está ejecutando un programa, se ofrecen las siguientes funcionalidades:
  - **+:** Aumenta progresivamente la velocidad a la que se ejecuta un programa de usuario.
  - **-:** Disminuye progresivamente la velocidad a la que se ejecuta un programa de usuario.
  - **F5:** Muestra y oculta el papel de la impresora.
  - **F9:** Realiza un cambio de pantalla.
  - **F10:** Provoca una interrupción al PIC.
  - **ESC:** Da fin a la ejecución de un programa de usuario.
  - **0...7:** Conmutan el microinterruptor correspondiente

10. REFINAMIENTO DE LAS INSTRUCCIONES.

La CPU que no es sino un autómata, divide la ejecución de toda instrucción en una secuencia de operaciones básicas, que encadenadas en el tiempo, al ritmo de una señal de reloj, dan cumplimiento a las órdenes reflejadas en la instrucción.

Desde un punto de vista funcional éstas se pueden agrupar en fases que denominaremos ciclos de instrucción. Así, en un primer nivel de refinamiento, la ejecución de una instrucción, se puede ver como una secuencia de ciclos de instrucción.

A continuación, se detallan la secuencia de ciclos de instrucción para cada una de las instrucciones que forman parte del repertorio de la CPU SX88.

10.1. INSTRUCCIONES DE TRANSFERENCIA.

· MOV memoria, registro

· Direccionamiento directo:

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (2 ciclos de fetch).

· 2º Ciclo de instrucción: Búsqueda de información sobre los operandos (2 ciclos de lectura en memoria).

· 3er Ciclo de instrucción: Almacenamiento del operando en memoria (1 ó 2 ciclos de escritura en memoria, dependiendo de si es de tipo "byte" o "word", respectivamente).

- **Direccionamiento indirecto:**

  - 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (2 ciclos de fetch).

  - 2º Ciclo de instrucción: Almacenamiento del operando en memoria (1 ó 2 ciclos de escritura en memoria, dependiendo de si es de tipo "byte" o "word", respectivamente).

· MOV registro, memoria

- **Direccionamiento directo:**

  - 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (2 ciclos de fetch).

  - 2º Ciclo de instrucción: Búsqueda de información sobre los operandos (2 ciclos de lectura en memoria).

  - 3er Ciclo de instrucción: Búsqueda del operando en memoria (1 ó 2 ciclos de lectura en memoria, dependiendo de si es de tipo "byte" o "word", respectivamente).

- **Direccionamiento indirecto:**

  - 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (2 ciclos de fetch).

  - 2º Ciclo de instrucción: Búsqueda del operando en memoria (1 ó 2 ciclos de lectura en memoria, dependiendo de si es de tipo "byte" o "word", respectivamente).

· MOV memoria, dato inmediato

- **Direccionamiento directo:**

  - 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (2 ciclos de fetch).

  - 2º Ciclo de instrucción: Búsqueda de información sobre los operandos (2 ciclos de lectura en memoria).

· 3er Ciclo de instrucción: Búsqueda del operando en memoria (1 ó 2 ciclos de lectura en memoria, dependiendo de si es de tipo "byte" o "word", respectivamente).

· 4º Ciclo de instrucción: Almacenamiento del operando en memoria (1 ó 2 ciclos de escritura en memoria, dependiendo de si es de tipo "byte" o "word", respectivamente).

· **Direccionamiento indirecto:**

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (2 ciclos de fetch).

· 2º Ciclo de instrucción: Búsqueda del operando en memoria (1 ó 2 ciclos de lectura en memoria, dependiendo de si es de tipo "byte" o "word", respectivamente).

· 3er Ciclo de instrucción: Almacenamiento del operando en memoria (1 ó 2 ciclos de escritura en memoria, dependiendo de si es de tipo "byte" o "word", respectivamente).

· **MOV registro, dato inmediato**

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (1 ciclo de fetch).

· 2º Ciclo de instrucción: Búsqueda del operando en memoria (1 ó 2 ciclos de lectura en memoria, dependiendo de si es de tipo "byte" o "word", respectivamente).

· **MOV registro, REGISTRO**

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (2 ciclos de fetch).

· 2º Ciclo de instrucción: Transferencia entre registros.

10.2. INSTRUCCIONES ARITMÉTICAS, Y LÓGICAS DE DOS OPERANDOS.

En cada una de las instrucciones, OP hace referencia a la operación aritmética (ADD, ADC, SUB, SBB), ó lógica (AND, OR, XOR) que realiza la instrucción.

· OP memoria, registro

· Direccionamiento directo:

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (2 ciclos de fetch).

· 2º Ciclo de instrucción: Búsqueda de información sobre los operandos (2 ciclos de lectura en memoria).

· 3er Ciclo de instrucción: Búsqueda del operando en memoria (1 ó 2 ciclos de lectura en memoria, dependiendo de si es de tipo "byte" o "word", respectivamente).

· 4º Ciclo de instrucción: Búsqueda del operando en registro.

· 5º Ciclo de instrucción: Ejecución de la operación en la ALU.

· 6º Ciclo de instrucción: Almacenamiento del resultado en memoria (1 ó 2 ciclos de escritura en memoria, dependiendo de si es de tipo "byte" o "word", respectivamente).

· Direccionamiento indirecto:

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (2 ciclos de fetch).

· 2º Ciclo de instrucción: Búsqueda del operando en memoria (1 ó 2 ciclos de lectura en memoria, dependiendo de si es de tipo "byte" o "word", respectivamente).

· 3er Ciclo de instrucción: Búsqueda del operando en registro.

· 4º Ciclo de instrucción: Ejecución de la operación en la ALU.

· 5º Ciclo de instrucción: Almacenamiento del resultado en memoria (1 ciclo de escritura en memoria, con datos tipo "byte", ó 2 con datos tipo "word").

· OP registro, memoria

· Direccionamiento directo:

- · 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (2 ciclos de fetch).
- · 2º Ciclo de instrucción: Búsqueda de información sobre los operandos (2 ciclos de lectura en memoria).
- · 3er Ciclo de instrucción: Búsqueda del operando en memoria (1 ó 2 ciclos de lectura en memoria, dependiendo de si es de tipo "byte" o "word", respectivamente).
- · 4º Ciclo de instrucción: Búsqueda del operando en registro.
- · 5º Ciclo de instrucción: Ejecución de la operación en la ALU.
- · 6º Ciclo de instrucción: Almacenamiento del resultado en registro.

· Direccionamiento indirecto:

- · 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (2 ciclos de fetch).
- · 2º Ciclo de instrucción: Búsqueda del operando en memoria (1 ó 2 ciclos de lectura en memoria, dependiendo de si es de tipo "byte" o "word", respectivamente).
- · 3er Ciclo de instrucción: Búsqueda del operando en registro.
- · 4º Ciclo de instrucción: Ejecución de la operación en la ALU.
- · 5º Ciclo de instrucción: Almacenamiento del resultado en registro.

· OP memoria, dato inmediato

· Direccionamiento directo:

- · 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (2 ciclos de fetch).
- · 2º Ciclo de instrucción: Búsqueda de información sobre los operandos (2 ciclos de lectura en memoria).

· 3er Ciclo de instrucción: Búsqueda del operando inmediato en memoria (1 ó 2 ciclos de lectura en memoria, dependiendo de si es de tipo "byte" o "word", respectivamente).

· 4º Ciclo de instrucción: Búsqueda del operando en memoria.

· 5º Ciclo de instrucción: Ejecución de la operación en la ALU.

· 6º Ciclo de instrucción: Almacenamiento del resultado en memoria (1 ó 2 ciclos de escritura en memoria, dependiendo de si es de tipo "byte" ó "word", respectivamente).

· **Direccionamiento indirecto:**

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (2 ciclos de fetch).

· 2º Ciclo de instrucción: Búsqueda del operando inmediato en memoria (1 ó 2 ciclos de lectura en memoria, dependiendo de si es de tipo "byte" o "word", respectivamente).

· 3er Ciclo de instrucción: Búsqueda del operando en memoria.

· 4º Ciclo de instrucción: Ejecución de la operación en la ALU.

· 5º Ciclo de instrucción: Almacenamiento del resultado en memoria(1 ó 2 ciclos de escritura en memoria, dependiendo de si es de tipo "byte" ó "word", respectivamente).

· OP registro, dato inmediato

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (2 ciclos de fetch).

· 2º Ciclo de instrucción: Búsqueda del operando inmediato en memoria (1 ó 2 ciclos de lectura en memoria, dependiendo de si es de tipo "byte", ó, "word", respectivamente).

· 3er Ciclo de instrucción: Búsqueda del operando en registro.

· 4º Ciclo de instrucción: Ejecución de la operación en la ALU.

· 5º Ciclo de instrucción: Almacenamiento del resultado en registro.

· OP registro, REGISTRO

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (2 ciclos de fetch).

· 2º Ciclo de instrucción: Búsqueda del operando 1 en registro.

· 3er Ciclo de instrucción: Búsqueda del operando 2 en registro.

· 4º Ciclo de instrucción: Ejecución de la operación en la ALU.

· 5º Ciclo de instrucción: Almacenamiento del resultado en registro.

10.3. INSTRUCCIONES DE COMPARACIÓN

La ejecución de este tipo de instrucciones, consta de los mismos ciclos de instrucción que las del grupo anterior, a excepción del ciclo de almacenamiento del resultado, debido a que el único efecto que produce la ejecución de la operación de comparación, es la alteración de los flags, perdiéndose el resultado en el interior de la ALU.

10.4. INSTRUCCIONES DE INCREMENTO, DECREMENTO, Y LÓGICAS DE UN OPERANDO.

En cada una de las instrucciones, OP es la operación que realiza la instrucción (INC, DEC, NOT, ó NEG).

· OP memoria

· Direccionamiento directo:

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (2 ciclos de fetch).

· 2º Ciclo de instrucción: Búsqueda de información sobre el operando (2 ciclos de lectura en memoria).

· 3er Ciclo de instrucción: Búsqueda del operando en memoria (1 ó 2 ciclos de lectura en memoria, dependiendo de si es de tipo "byte" o "word", respectivamente).

· 4º Ciclo de instrucción: Ejecución de la operación en la ALU.

· 5º Ciclo de instrucción: Almacenamiento del resultado en memoria (1 ó 2 ciclos de escritura en memoria, dependiendo de si es de tipo "byte" ó "word", respectivamente).

· **Direccionamiento indirecto:**

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (2 ciclos de fetch).

· 2º Ciclo de instrucción: Búsqueda del operando en memoria (1 ó 2 ciclos de lectura en memoria, dependiendo de si es de tipo "byte" o "word", respectivamente).

· 3er Ciclo de instrucción: Ejecución de la operación en la ALU.

· 5º Ciclo de instrucción: Almacenamiento del resultado en memoria (1 ó 2 ciclos de escritura en memoria, dependiendo de si es de tipo "byte" ó "word", respectivamente).

· OP registro

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (2 ciclos de fetch).

· 2º Ciclo de instrucción: Búsqueda del operando en registro.

· 3er Ciclo de instrucción: Ejecución de la operación en la ALU.

· 4º Ciclo de instrucción: Almacenamiento del resultado en registro.

10.5. INSTRUCCIONES DE MANEJO DE PILA

· PUSH registro

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (1 ciclo de fetch).

· 2º Ciclo de instrucción: Almacenamiento del operando en memoria (1 ó 2 ciclos de escritura en memoria, según sea de tipo "byte" ó "word", respectivamente).

· POP registro

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (1 ciclo de fetch).

· 2º Ciclo de instrucción: Búsqueda del operando en memoria (1 ó 2 ciclos de lectura en memoria, dependiendo de si es de tipo "byte" ó "word", respectivamente).

· PUSHF

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (1 ciclo de fetch).

· 2º Ciclo de instrucción: Almacenamiento del operando en memoria (2 ciclos de escritura en memoria).

· POPF

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (1 ciclo de fetch).

· 2º Ciclo de instrucción: Búsqueda del operando en memoria (2 ciclos de lectura en memoria)

10.6. INSTRUCCIONES DE CAMBIO DE FLUJO DEL PROGRAMA.

En estas instrucciones, Jx referencia cualquier salto incondicional (JZ, JNZ, JS, JNS,...)

· JMP dirección

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (1 ciclo de fetch).

· 2º Ciclo de instrucción: Búsqueda de información sobre el operando (2 ciclos de lectura en memoria).

· 3er Ciclo de instrucción: Actualización del registro IP.

· Jx desplazamiento

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (1 ciclo de fetch).

· 2º Ciclo de instrucción: Búsqueda de información sobre el operando (1 ciclo de lectura en memoria).

· 3er Ciclo de instrucción: Actualización del registro IP (sólo si se cumple la condición que denota la instrucción).

10.7. INSTRUCCIONES ASOCIADAS A SUBRUTINAS.

· CALL dirección

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (1 ciclo de fetch).

· 2º Ciclo de instrucción: Búsqueda de información sobre el operando (2 ciclos de lectura en memoria).

· 3er Ciclo de instrucción: Almacenamiento del registro IP en memoria (2 ciclos de escritura en memoria).

· 4º Ciclo de instrucción: Actualización del registro IP.

· RET

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (1 ciclo de fetch).

· 2º Ciclo de instrucción: Actualización del registro IP (2 ciclos de lectura en memoria).

10.8. INSTRUCCIONES DE GESTIÓN DE LAS INTERRUPCIONES.

· INT número

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (1 ciclo de fetch).

· 2º Ciclo de instrucción: Búsqueda de información sobre el operando (1 ciclo de lectura en memoria).

· 3er Ciclo de instrucción: Almacenamiento del registro de flags en memoria (2 ciclos de escritura en memoria).

· 4º Ciclo de instrucción: Almacenamiento del registro IP en memoria (2 ciclos de escritura en memoria).

· 5º Ciclo de instrucción: Actualización del registro IP (2 ciclos de lectura en memoria).

· IRET

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (1 ciclo de fetch).

· 2º Ciclo de instrucción: Actualización del registro IP (2 ciclos de lectura en memoria).

· 3er Ciclo de instrucción: Actualización del registro de flags (2 ciclos de lectura en memoria).

· STI

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (1 ciclo de fetch).

· 2º Ciclo de instrucción: Actualización de flags.

· CLI

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (1 ciclo de fetch).

· 2º Ciclo de instrucción: Actualización de flags.

10.9. INSTRUCCIONES DE ENTRADA/SALIDA

· IN AL, dirección

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (1 ciclo de fetch).

· 2º Ciclo de instrucción: Búsqueda de información sobre los operandos (1 ciclo de lectura en memoria).

· 3er Ciclo de instrucción: Lectura desde el puerto (1 ciclo de lectura en I/O).

· IN AX, dirección

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (1 ciclo de fetch).

· 2º Ciclo de instrucción: Búsqueda de información sobre los operandos (1 ciclo de lectura en memoria).

· 3er Ciclo de instrucción: Lectura desde el puerto (2 ciclos de lectura en I/O)..

· OUT dirección, AL

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (1 ciclo de fetch).

· 2º Ciclo de instrucción: Búsqueda de información sobre los operandos (1 ciclo de lectura en memoria).

· 3er Ciclo de instrucción: Escritura en el puerto (1 ciclo de escritura en I/O).

· OUT dirección, AX

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (1 ciclo de fetch).

· 2º Ciclo de instrucción: Búsqueda de información sobre los operandos (1 ciclo de lectura en memoria).

· 3er Ciclo de instrucción: Escritura en el puerto (2 ciclos de escritura en I/O)..

· IN AL, DX

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (1 ciclo de fetch).

· 2º Ciclo de instrucción: Lectura desde el puerto (1 ciclo de lectura en I/O)..

· IN AX, DX

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (1 ciclo de fetch).

· 2º Ciclo de instrucción: Lectura desde el puerto (2 ciclos de lectura en I/O).

· OUT DX, AL

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (1 ciclo de fetch).

· 2º Ciclo de instrucción: Escritura en el puerto (1 ciclo de escritura en I/O).

· OUT DX, AX

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (1 ciclo de fetch).

· 2º Ciclo de instrucción: Escritura en el puerto.(2 ciclos de escritura en I/O).

10.10. INSTRUCCIONES DE CONTROL

· NOP

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (1 ciclo de fetch).

· HLT

· 1er Ciclo de instrucción: Búsqueda del código de operación y decodificación (1 ciclo de fetch).