ZEN 80

Z80 Assembly Language Programming System
for the MSX MICRO-COMPUTER

ISBN: 07457-0002-0
Editado por: Kuma Computes Ltd.
12 Horseshoe Park
Pangbourne
Berks RGB 7JW
Telex 849462
Tel 07357 4335

Remaquetado por: Aitor Lekerika
April, 15th, 2.007
Forma de contacto: msx-assembler@hotmail.com

ADVICE

Este sistema de programación se entrega con el convencimiento de que funciona tal y como se describe, pero la compañía **KUMA COMPUTERS LTD** no se hace responsable de ninguna perdida de datos y/o daños que pudieran sufrir en su ordenador a cualquier persona como resultado de fallos ó defectos en los productos entregados, y por lo tanto la compañía declina cualquier responsabilidad que de ello se derive.

INTRODUCCION

Gracias por adquirir esta copia de ZEN. Si tienes cualquier pregunta sobre ZEN, por favor, siéntete libre de escribir a AVALON SOFTWARE, que responderá a todas las preguntas que reciba.

Los lenguajes de alto nivel tienen limitaciones. Cuanto se necesitan mayores velocidades y flexibilidad, uno busca la solución en el lenguaje de programación Ensamblador. ZEN te proporciona las herramientas para generar ó analizar los programas escritos para Ensamblador Z80.

Al contrario del BASIC, el cual está permanéntemente disponible en ROM, necesitas cargar ZEN desde cassette. Está almacenado en la cinta como un fichero binario diseñado para ejecutarse en la dirección &HA000.

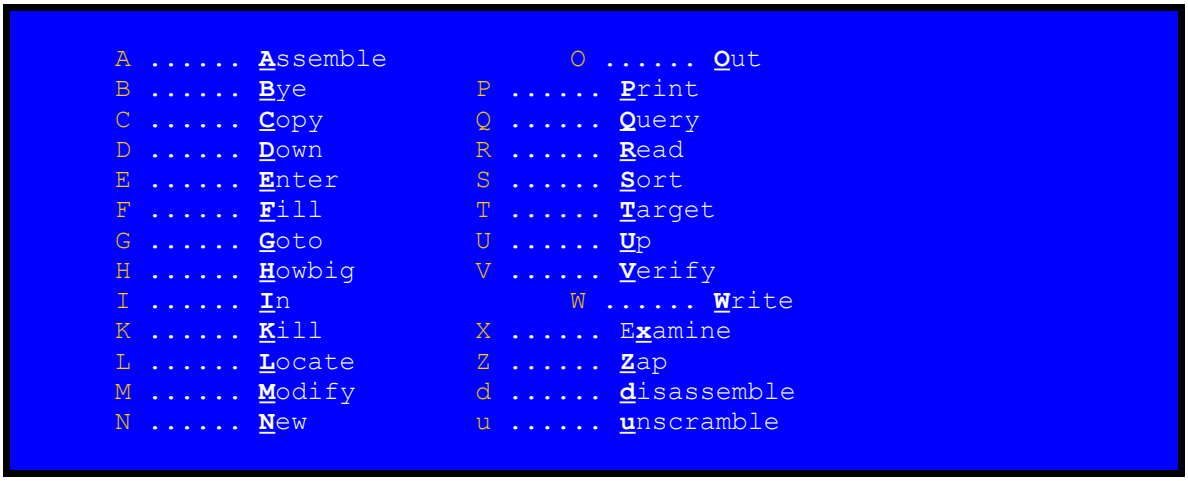
BASIC reserva una zona de memoria para su funcionamiento. Pero esta zona de memoria podría entrar en conflicto con ZEN. Para asegurarte que esto no ocurre y que ambos programas puedan funcionar sin problemas, sería conveniente efectuar el siguiente paso en BASIC para cargarlo de la forma apropiada:

```
CLEAR 200, &H9FFF
Ok
BLOAD"CAS:ZEN", R
```

Después de esto, ZEN será cargado en la memoria y se ejecutará automáticamente cuando acabe la carga.

NIVEL DE COMANDOS

Cuando en pantalla se muestre el prompt del sistema ZEN>, estarás en lo que se llama el Nivel de Comandos, y podrás ejecutar cualquiera de los siguientes comandos:



Para seleccionra un comando determinado, basta que pulses la primera tecla de su nombre, seguido de un parámetro -siempre que sea posible- para después pulsar la tecla RETURN.

La tecla BS (Back Space) borra el caracter anterior, mientras que DEL activará ó desactivará el motor en cassette (MOTOR ON / MOTOR OFF).

El uso de cada uno de los Comandos principales se explican con mas detenimiento en los siguientes sub-apartados.

Si introducimos una orden, y esta no es entendida por ZEN -por ejemplo, hemos pulsado una tecla incorrecta- ZEN mostrará el mensaje de error HUH? -algo así como "¿Qué dices?"-.

Si pulsamos solamente RETURN en el Nivel de Comandos, la pantalla se limpiará y pasará a pantalla de modo de texto ancho. Este modo, es el usado por ZEN.

NEW (NUEVO)

Este comando te permite modificar la línea de texto actual.

La línea se muestra con el cursor situado en la posición mas a la derecha. Cambia la línea y pulsa RETURN para reestablecer la nueva línea de texto al fichero.

OUT (SALIDA)

Con este comando podrás sacar un valor al puerto I/O que especifiques en los parámetros. Si introduces sólamente el comando -sin parámetros- se te solicitarán los datos a introducir.

PRINT (IMPRIMIR)

Con este comando podrás pasar un número determinado de líneas del fichero de texto por la impresora.

El número de líneas se especifica en el parámetro del Comando. Por ejemplo, P9 imprimirá 9 líneas. El comando por defecto es 1.

La impresión comienza en la línea actual, y la ultima línea que se muestre, pasará a ser la línea actual.

QUERY (VOLCADO)

Con este comando mostramos 64 bytes de la memoria en formato Hexadecimal y ASCII.

Con los parámetros del comando especificamos la dirección de comienzo, por ejemplo, QA000H, mostrará los primeros 64 bytes del programa ZEN.

Si no se especifica ningún parámetro, la dirección a partir de la cual se empieza a mostrar, será la última en la que se encontraba.

READ (CARGAR)

Con este comando leemos un fichero desde el cassette y lo pasamos a la memoria.

Con los parámetros del comando, especificamos el tipo de fichero a leer:

R (LEER TEXTO)

Leerá un fichero de texto y lo añadirá al final del texto existente en memoria.

Se solicitará un nombre de fichero. Puede tener de 0 a 6 caracteres de longitud. En caso de no dar ningún nombre -0 caracteres- ZEN procederá a leer el primer fichero de texto que encuentre en el cassette.

En caso de que el fichero de texto, supere el tope de memoria asignado por ZEN, la lectura terminará y se mostrará el error MEMORY.

RB (LEER BINARIO)

Leerá un fichero binario y lo pasará a la memoria.

Se solicitará una dirección de memoria en la que cargar el fichero. Si se da la dirección, ZEN comenzará la lectura del fichero y lo colocará en la dirección de memoria que le hemos dado. Por defecto, pulsando solamente RETURN, el fichero se cargará en la dirección que apunte la cabecera del fichero. En caso de ser necesario, la dirección de ejecución del fichero se coloca en el Contador del Programa del Usuario, para una posterior ejecución.

Se puede pulsar CTRL-STOP para abortar cualquier operacion I/O en progreso, pero no tiene ningun efecto sobre ZEN.

GOTO (IR A...)

Con este comando, cargamos los registros del Z80 con la "Imagen de Usuario" y se transfiere el control a la dirección especificada por el parámetro del comando.

Por ejemplo, **GO** -sin parámetros- realiza y reinicio en frío del sistema.

Si no se especifica ningún comando, el control se transfiere a la dirección a la que apunta el Contador de Programa de Usuario. Después se solicita una dirección de "punto de ruptura". Si se introduce una dirección válidad, el "punto de ruptura" se activa.

Un "punto de ruptura" es una manera de parar un programa en ejecución. En esencia, lo que se hace es colocar una instrucción CALL, para que se devuelva el control del sistema a un gancho ZEN. Este gancho, guarda todos los registros del Z80 en el Area de Imagen de Usuario para reestablecer el código bajo el punto de ruptura antes de retornar al programa.

Se puede continuar la ejecución usando el Nivel de Comando B, de forma que el Contador de Programa se guarda como parte del proceso de gancho. Los programas de apllicaciones deberían terminar con un simple salto de retorno a la **ENTRADA** ZEN ó puntos de **REENTRADA**. La única diferencia entre esos dos puntos, es que la **REENTRADA** guarda los registros del Z80, mientras que la **ENTRADA**, no lo hace.

ZEN intentará realizar una inicialización tan pequeña como sea posible para perder la menor cantidad de datos.

HOWBIG (TAMAÑO)

Con este comando se puede saber -en formato Hexadecimal- la dirección de comienzo y final de un fichero de texto y el máximo de memoria.

ZEN permitirá que el fichero de texto aumente hasta el límite máximo, pero no mas allá. Se puede cambiar este límite si es necesario (ver la constante LIMIT).

IN (ENTRADA)

Este comando mostrará -en Hexadecimal y Binario- los datos leídos desde el Puerto I/O especificado en el parámetro del comando.

Por ejemplo, 98H leerá el estado del puerto de la impresora.

KILL (BORRADO)

Este comando borra el fichero de texto de la memoria, tal y como lo hace el comando NEW del BASIC.

Cuando se borra un fichero de texto, ZEN iguala los punteros EOF[1] y SOF[2] del fichero, pero el fichero de texto permanece en la memoria, de forma que -si por accidente- hemos borrado el fichero, podemos recuperarlo. Para ello, basta con hacer lo siguiente:

- Buscar el último caracter del texto -este debería ser el código de Retorno de Carro- (&H90).
- Increméntarlo en 1 y usar el comando **MODIFY** para reestablecer el puntero de EOF (ver EOFP).

LOCATE (BUSCAR)

Este comando busca una cadena de texto determinada en el fichero de texto.

La cadena de caracteres forman parte de los parámetros del comando, por ejemplo **LBIT 7, A**. Con esto se procede a buscar la primera cadena de texto que tenga la instrucción **BIT 7, A** a partir de la línea siguiente a la línea actual. Si se encuentra el texto, la línea que lo contiene pasa a ser la línea actual. En caso de no encontrarse el texto, la última línea de texto pasará a ser la actual.

No hay restricciones en el contenido de la cadena del parámetro.

MODIFY (MODIFICAR)

Este comando permite examinar y despues modificar el contenido de la memoria.

La dirección de memoria se especifica en el parámetro del comando. Por ejemplo, MD000H situará el comienzo en &HD000.

Si no se especifica la dirección en el parámetro el comando continuará desde la última posición. Se muestra el de la dirección en Hexadecimal y ZEN solicita los parámetros.

Si se introduce un parámetro, se toma la dirección especifica. Si se omite, ZEN solamente pasará a la siguiene dirección

Para volver al Nivel de Comandos basta con teclear **STOP**.

ASSEMBLE (ENSAMBLAR)

Con este comando, se leen una serie de instrucciones en lenguaje Ensamblador para convertirlo en el correspondiente Código Máquina del Z80.

El fichero de texto en la memoria es una serie de instrucciones -llamado Código Objeto- que se puede editar con los comandos ZEN, el cual se pasa después a Ensamblador -llamado también Código Máquina-. El fichero resultante del Código Máquina -normalmente llamado Fichero Binario- es controlado por el operador **LOAD** (ver PSEUDO-OPERADORES).

ZEN solicitará cual será el formato de salida del lista, indicando en pantalla con el mensaje OPTION>. Hay 4 opciones posibles:

V

Salida por pantalla

E

Salida por dispositivo externo.

P

Salida del listado por impresora.

RETURN

Sin listado de salida.

El fichero de texto comienza a leerse desde la primera línea, y acaba al llegar a un operador **END**[3].

BYE **(SALIR)**

Este comando devuelve el control del sistema al BASIC. Los programas en BASIC de la memoria no se verán afectados.

Una vez en el BASIC se puee regresar a ZEN sin necesidad de recargar el programa -lo que tambien se puede hacer, pero se podrían perder datos-, de la siguiente forma:

```
DEFUSR=&HA000
Ok
A=USR(0)
```

El paso de control entre BASIC y ZEN y viceversa, no afectará a los programas situados en memoria (programas BASIC, fichero de texto, etc...).

COPY (COPIAR)

Este comando permite copiar bloques de memoria.

Se solicitan 3 parámetros que son:

START>

Dirección de comienzo del bloque a copiar.

STOP>

Dirección final del bloque a copiar.

DESTINATION>

Dirección destino del bloque a copiar.

Estas direcciones pueden darse en formato Decimal, Hexadecimal u Octal.

Los numeros Decimales no llevan ningun indicador, tan solo son números.

Los números Hexadecimales lleva el indicador "H" al final del numero.

Los números Octales llevan el indicador "O" al final del numero.

Por ejemplo, si se quiere copiar un bloque de memoria desde &H0200 hasta &H02FF a la dirección &HC000, hay que teclear 200H, 2FFH y C000H. para cada uno de los parámetros que nos solicita.

DOWN (BAJAR)

Este comando mueve el editor una línea por debajo de la actual tantas veces como numeros de línea se especifiquen en el parámetro del comando.

Por ejemplo, D37 movera 37 líneas abajo.

El editor de ZEN es muy parecido al del BASIC, salvo que no usa numeros de línea, en vez de eso usa varios comandos para moverse a lo largo del fichero de texto hasta que alcances la posición requerida.

Pulsando ENTER se inserta una línea, y si usando el comando **ZAP** se borra una línea.

En caso de alcanzar el final del fichero, se muestra en pantalla el mensaje EOF.

ENTER (INSERTAR)

Este comando introduce líneas de texto en el fichero de texto. ZEN mostará el numero de línea actual, y una vez escrito el texto deseado -una instrucción ó un comentario- presionamos RETURN y quedará introducido en el fichero de texto.

Cuando se haya editado lo requerido, basta con teclear **STOP** al comienzo de una línea y ZEN mostrará el Nivel de Comandos. La ultima línea introducida es la actual, y las que se hayan introducido anteriormente son movidas abajo hacia EOF.

Aunque se muestran los numeros de línea de texto, estos no son almacenados en el fichero de texto.

FILL (RELLENAR)

Este comando rellena un bloque de memoria con un valor.

Hay tres datos a especificar y son:

START>

Dirección de comienzo

STOP>

Dirección final

DATA>

Valor con el que se rellena la memoria.

Este comando es especialmente util para borrar la memoria si utilizamos el valor 0.

SORT (ORDENAR)

Este comando ordena y muestra la tabla de símbolos producida durante el útlimo ensamblado.

ZEN solicita una de las opciones de salida, que son las mismas que para el comando **ASSEMBLER**.

Se puede elegir el proceso de ordenado segun la letra de comienzo de los símbolos introduciendo la primera letra como parámetro del comando. Por ejemplo, SB listará unicamente los símbolos que comiencen con la letra "B".

Los símbolos son ordenados por la primera letra, no por el resto del nombre.

TARGET (DESTINO)

Este comando se desplazará a una línea de texto y la pondrá como línea actual.

El parámetro del comando especifica el numero de línea, por ejemplo. T1435 pasara el editor a la línea 1.435.

El parámetro por defecto del comando, T, moverá el editor al comienzo del fichero.

UP ARRIBA)

Este comando mueve el fichero de texto tantas posiciones como numeros de líneas especificados en el parámetro de comandos.

El parámetro por defecto del comando es 1.

VERIFY (VERIFICAR)

Este comando verifica que un fichero ha sido correctamente grabado en cassette. El parámetro del comando especifica el tipo de fichero a comprobar.

Este comando debería ser usado inmediatamente despues de efectuar la grabación, ya que los ficheros MSX no tienen prodecimientos de detección de grabación de errores internamente. Cuando se graba un fichero, ZEN guarda un codigo "Cheksum"[4], el cual se compara con el de lectura. Los parámetros disponibles son:

V

Verifica un fichero de texto.

VB

Verifica un fichero Binario.

WRITE (ESCRITURA)

Este comando graba a cassette una zona de memoria. Los parámetros disponibles son:

W

Guarda el texto ASCII de la memoria en el cassete.

Se solicita el nombre del fichero que será guardado con el comando READ. Los ficheros de texto ZEN son ficheros de texto estandard del MSX -como los generados por el comando SAVE del BASIC), con un CR y LF entre líneas, y un CTRL-Z al final del fichero.

WB

Guarda el contenido de una zona de memoria como fichero Binario en el cassette.

Se solicitan tres parámetros:

START>

Dirección de comienzo.

STOP>

Dirección final.

EXEC>

Dirección de ejecución -en caso de omitirla pulsando RETURN, se entenderá que es la misma que START-.

EXAMINE (EXAMINAR)

Este comando muestra los registros del Z80 que han sido guardados en el Area de Imagen del Usuario.

La línea superior muestra los principales registros y la línea inferior muestra los registros alternativos.

ZAP (BORRAR LÍNEAS)

Este comando elimina tantos numeros de línea del fichero de texto como indica el número dado en el parámetro del comando.

Por ejemplo, Z108 eliminará 108 líneas de texto, comenzando por la línea actual.

El parámetro por defecto es 1.

DISASSEMBLE (DESENSAMBLAR)

Este comando realiza un desensamblado simbolico de un area de la memoria y genera un fichero de texto ó lo lista como salida.

Se solicitan los parámetros:

START>

Dirección desde la cual comenzar el desensamblado.

STOP>

Dirección final del desensamblado.

RUNS AT>

Dirección de ejecución del programa

Hay ocasiones en las que se necesita ubicar un programa en una posición de memoria poco habitual. El desensamblador puede reubicar cualquier dirección con sus Labels[5] hacia la salida.

Si se omite el parámetro START>, ZEN asume que este se situa en una posición de memoria habitual.

Se solicitaran repetidamente los dos primeros parámetros cuando se encuentren areas de datos en la región a desensamblar, las cuales, no deben ser decodificadas como instrucciones, sino como Bytes de datos.

Para terminar este proceso, basta con poner 0 cuando se solicite una dirección. Hay un máximo de 64 areas de datos separadas. Si se excede este número ZEN mostará el mensaje de error FULL.

Una vez acabado, se solicita el parámetro OPTION>. Se puede especificar tres vias:

V

Salida a través de la pantalla.

P

Salida por impresora.

E

Salida a través de un dispositivo externo.

En caso de que el desensamblado exceda el límite de memoria se mostrará el mensaje de error MEMORY, y el desensamblado finaliza.

El otro error posible que se puede dar durante el desensamblado es cuando la tabla de símbolos está llena, en cuyo caso se muestra el mensaje de error FULL.

El desensamblador y el ensamblador usan la misma tabla de símbolos, por lo que al usar uno de estos comandos, esta tabla se pierde en favor de la nueva que se genera, aunque las Tablas de símbolo solo suelen ser de interes en el caso de que el usuario necesite usar la opcion **SORT**.

Cualquier OpCode inválido encontrado durante el desensamblado, será tratado como dato.

Las Labels de formato Lnnnn (donde nnnn es una dirección de memoria) se generará en la posicion mas apropiada, si es posible.

UNSCRAMBLE (MINI)

Este comando es una versión simplificada del Desensamblador.

Basicamente desensambla 8 instrucciones del Z80 comenzando por la dirección de memoria especificada en el parametro del comando.

Por ejemplo, u0, desensamblará 8 dirección a partir del comienzo del BASIC ROM.

La opción por defecto continuara en la ultima posición donde se estaba.

Los OpCodes no válidos que se encuentren durante este comando, serán tomados como Bytes de datos.

ZEN tratará de presentar los operandos de forma diferente:

> Los números menores que 10, serán mostrados como números decimales.
>
> Los números entre &H41-&H5A y &H61-&H7A son mostrados como caracteres literales en ASCII. Los restantes, son mostrados como numeros Hexadecimales, empezando por un 0 si fuera necesario.

LISTADO DE SALIDA

Los comandos **ASSEMBLE**, **SORT** y **DISASSEMBLE** pueden generar grandes cantidades de datos a la salida en formato de listado, tanto a video, impresora ó a otros dispositivos externos. Con esos comandos, se generarán una página cada vez, efectuando una pequeña pausa entre cada página. Presionando cualquier tecla se parará la salida al final de la página, y para reanudar el listado bastará con pulsar cualquier tecla -excepto "Q"-. Si pulsamos esta tecla, se abortará el listado y volverá al Nivel de Comandos.

Se da por sabido que tanto la impresora como los dispositivos externos son de 80 caracteres de ancho por 66 líneas, ejemplo típico de una impresora. Si se dispone de otro tipo, será necesario modificar ZEN. El usuario puede cambiar la longitud de la página modificando el procedimiento PAGE (ver listado ZEN). Se pueden modificar el grupo de constantes COMWIDTH/SYMWIDTH. El primer byte de cada una de esas constantes define el ancho de un campo del dispositivo de video. Tambien se puede cambiar el numero de símbolos por línea durante un **SORT**, ya que alli se dispone de un código especifico para ese proposito.

Se entiende que los dispositivos externos e impresoras responden a los caracteres de Control Forefeed (&H0C), Retorno de Carro (&H0D) y LineFeed (&H0A).

ZEN utiliza un ForeFeed seguido de 62 líneas de texto para cada página, y cada línea termina con un CR y un LF..

El dispositivo externo e impresora está conectado a la salida del MSX, el conector RS-232, mientras que el controlador de la salida de la impresora va al puerto Centronics. Los controladores del tipo de impresoras EPSON FX-89 son de ese tipo. En caso de disponer de otro tipo de dispositivos, hay espacio suficiente en los controladores para insertar parches, para filtrar los LF, por ejemplo.

Se asume que el dispositivo de Video tiene 37 caracteres de ancho, pero es algo que se puede cambiar, así como para las impresoras y dispositivos externos, mientras soporten un modo de 80 caracteres por línea.

ZEN no muestra los numeros de línea al generar los listados por pantalla ya que no entran por el ancho.

Los símbolos, operandos y comentarios de una instrucción del Z80 pueden ser de una longitud indefinida. Si es necesario, ZEN truncará esos campos para que se ajusten al formato requerido.

LA TABLA DE SÍMBOLOS

Hay un area de la memoria, llamada Tabla de Símbolos, usada por ZEN para almacenar símbolos durante el Ensamblado/Desensamblado. Se situa entre ZEN y el fichero de texto. Si deseas incrementar este tamaño, es necesario cambiar el comienzo del puntero del fichero al valor Hexadecimal requerido, de la manera siguiente:

(1) Borrar el fichero de texto.
(2) Usar **MODIFY** para cambiar el SOFP
(3) Borrar otra vez el fichero de texto para copiar SOFP a EOFP y CURRENT.
(4) Realizar un Ensamblado para eliminar la Tabla de Símbolos.
(5) Usar **WB** para escribir una nueva versión al cassette.

SINTAXIS DE ENSAMBLADO

El Ensamblador de ZEN funciona de acuerdo a la Sintaxis definida por el **MANUAL DE PROGRAMACION DE LENGUAJE ENSAMBLADOR DE ZILOG Z80**.

Hay una excepción. ZEN, cuando se trata de la instrucción EX AF, AF', la cambia por EX AF, AF.

La siguiente sección contiene -ordenados alfabeticamente la- la forma en que se introducen las instrucciones Z80. Hay 4 campos lógicos que se pueden dividir en:

LABEL (ETIQUETA)

Una etiqueta es una manera de marcar una instruccion, de tal forma que pueda ser invocada en un momento dado por otra parte del programa. Los numero de lina sirven de igual forma que en BASIC, así que se puede usar GOTO 24, por ejemplo. En Ensamblador se permite llamar a una línea con un nombre simbolico.

Cuando se declara una Label debe ir precedida de dos puntos ":" para que el Ensamblador sepa que es una etiqueta. Deben empezar por una letra pero despues pueden contener letras y/o numeros. ZEN permite que la longitud de las etiquetas sea cualquiera que acepte los caracteres válidos.

No se pueden usar ni los registros ni los nombres de Pseudo-Operadores como nombre, así mismo como tampoco se pueden usar símbolos que hayan sido reservados como identificadores. Cualquie intento en este sentido, tendra un error por respuesta.

OPERADOR

Hay 67 operadores diferentes en el Z80. ZEN añade 7 mas, llamados Pseudo-Operadores:

END

Al encontrarse con este Pseudo-Operador, el Ensamblado del programa termina. **¡DEBE PONERSE SIEMPRE!**.

DS / DEFS

Define una variable simple de 1 Byte

DW / DEFW

Define una palabra (Word) de 2 Bytes. Estos deben en orden inverso, es decir, primero el Byte de menor peso, y luego el de mayor peso.

DB / DEFB

Define Bytes, separados por comas ",". Puede ser un valor entre 0-255 o una cadena literal. Estas pueden ser de cualquier longitud, pero no pueden formar parte de una expresión.

EQU

Asigna el valor del operando a un identificador de símbolo. Cualquier identificador simbolico en la expresion del operando debe estar declarada al comienzo del programa. Esta regla de "sin referencia hacia adelante" está diseñada para prevenir referencias circulares.

ORG

Este Pseudo-Operador se usa para definir en que posición de memoria va a empezar a Ensamblarse el Código Objeto del fichero. La regla "sin referencia hacia adelante" se aplica a este comando.

LOAD

Comienza cargando el código a la memoria empezando por la dirección del operando. Si se usa ORG este proceso se desactiva. y se requiere el reestablecimiento del proceso de carga.

OPERANDO

El número de operandos en una línea depende del operador. Los hay *Niladicos*, *Sonadicos* y *Diadicos*[6]. Estos utilizan 0, 1 ó 2 operandos respectivamente. Hay tres clases de operandos:

Con registros:
A, B, C, D, E, H, L, I, R, HL, DE, AF, IX, IY, SP

Códigos Condicionales:
NZ, Z, NC, C, PO, PE,P, M

Expresiones numéricas:
Una expresión numérica esta compuestra de uno ó mas de cualquiera de los siguientes elementos delimitados por el operador matemático:

- Un número Decimal, Hexadecimal u Octal. Los numeros Decimales no usan sufijos, pero los Hexadecimales usan el "H", y los Octales la "O". En algunos casos, los Hexadecimales pueden llevar un "0" por delante (por ejemplo, 0AH).
- Un caracter literal encerrado en comillas dobles o simples.
- El caracter "$". Esta variable se usa en la asociacion de valores en la evaluación de expresiones.

Los Operadores matemáticos pueden ser:

+	Suma
-	Resta
*	Multiplicación
/	División
&	"Y" Lógica
·	"O" Lógica

Las expresiones son evaluadas **EXCLUSIVAMENTE DE IZQUIERDA A DERECHA** sin precedencia en el orden. La aritmética es de 16 Bits sin signo y enteros y las desbordamientos son ignorados.

COMENTARIOS

Los comentarios son ignorados por el Ensamblador. Son muy útiles para aclarar partes del programa. Van al final de las líneas precedidas de un punto y com ";", y terminan al final de la línea.

GESTOR DE ERRORES DEL ENSAMBLADOR

En el caso de que el Ensamblador encuentre un error, sucederá lo siguiente:

(1) Finaliza el Ensamblado.
(2) Se muestra un error.
(3) Se muestran las líneas en las que se ha producido el error.
(4) Se produce un bucle de reentrada.

Ahora se puede corregir el error y reensamblar. Es imposible efectuar un error de sintaxis que produzca daños a ZEN ó a la memoria.

Los posibles mensajes de error son los siguientes:

UNDEFINED

Se ha intentado usar un símbolo no declarado.

SYMBOL

Se ha declarado un símbolo de longitud 0 ó se ha olvidado el símbolo necesario con un Pseudo-Operador EQU.

RESERVED

Se ha intentado usar una palabra reservada para un símbolo.

FULL

La Tabla de Símbolos esta llena.

DOUBLE SYMBOL

Se ha declarado el mismo símbolo mas de una vez.

EOF

El Ensamblador no encuentra el Pseudo-Operador END.

HUH?

No se entiende el comando introducido, ó no es correcto.

OPERAND

Se ha hecho algo mal con un operando. Muchos de los errores vendrán bajo este tipo. Suelen producirse tambien cuando se intenta hacer un salto relativo que excede de lo permitido, ó en las instrucciones de indexado.

FIN DEL DOCUMENTO

Posiblemente, la firma AVALON SOFTWARE haya pasado hace tiempo a mejor vida. Pero dejó tras de si uno de los Ensambladores-Desensambladores más utilizados. Aún así, el MSX está bastante olvidado -en lo que se refiere a las firmas comerciales-. Como muchos programas, este está descatalogado, y hoy día -salvo que alguien me indique lo contrario- pertenece al mundo del Abandonaware, lo que significa, que no se están vulnerando derechos de Copyright por hacer este manual.

Sabiendo todo esto, a uno le dan ganas de hacer muchas cosas. Por ejemplo, traducir manuales para ponerlos a disposición de tod@s. ¿A quien no le ha apetecido realizar este programa ó aquel? Pero siempre ha existido un límite: No hay información disponible.

La información hoy día es fácil de conseguir en Internet. No tan fácil encontrarla en Español. Y aún con todo, no tampoco demasiado fácil encontrarla en Inglés (muchos documentos los he tenido que "tirar" por estar en Holandés, Alemán, Japonés, etc...). Bien, ahora voy abriendo camino y proporcionando herramientas con las que trabajar. Así que si dispones de algun documento interesante, y te apetece ponerte a eso de traducir, pues hazlo, y entre todos quizás consigamos una Biblioteca MSX decente para nosotros. Este manual es un ejemplo y hay mas...

A lo largo de este año, iré colgando en la Red este tipo de manuales para que todo el mundo pueda disfrutar de ellos y quien sabe... ¡a lo mejor aparece por ahí otro *Hideo Kojima*. Por lo que a mi se refiere, es por el simple placer de hacerlo y saber que a alguien le puede resultar de ayuda.

Por último, mi e-mail de contacto está al principio del documento. Si tienes algo interesante que contar, ¡anímate y escribe!

Un saludo.

Aitor Lekerika Alcedo

[1] **EOF**
Abreviatura de End Of File (Final de Fichero). Es un indicador de final fichero, que suele tener el código ASCII &H90 (retorno de carro), para los ficheros de texto de ZEN.

[2] **SOF**
Abreviatura de Start Of File (Comienzo de fichero). Es un indicador de comienzo de fichero.

[3] **END**
Hay que tener en cuenta que la función END puede ser mal interpretada. No se trata del final de una Rutina ó Subrutina, sino el final mismo del programa. Por lo tanto, solo puede incluirse una vez en cada programa. En caso contrario, podemos encontrarnos con el desagradable problema de haber ensamblado tan solo una porcion del programa total.

[4] **CHEKSUM**
Abreviatura de **Chek Summary** (Sumario de comprobación). Consiste en un algoritmo matemático que se efectua durante la grabación de un fichero. Según la información que contenga y el formato que tenga, el valor que se genera es uno u otro. Teóricamente, al leer el fichero, se comprueba que el Checksum sea el mismo. En caso de no serlo, se produce el archiconocido "Cheksum Error", es decir, que la lectura -ó la grabación- es defectuosa. No todos los formatos de fichero tienen esta opción.

[5] **LABELS**
Traducido como etiquetas. Son puntos de referencia de zonas de memoria que contienen Rutinas ó Subrutinas. En vez de especificar una posición de memoria, se le asigna un nombre -LABEL- para luego ser llamada sin necesidad de recordad esa posición de memoria.

Me ha parecido oportuno dejar este nombre en Inglés original, pues es el que se utiliza normalmente a la hora de comentar programas en CM.
[6] Soy consciente de que estas tres palabras suenan como muy "mañicas" -muy Aragonesas, para que se entienda mejor- pero es la mejor traducción que se me ha ocurrido. Si conoces las reales, ya sabes lo que hacer...