MICROMANUALES

Amstrad CPC 464/664/6128. Manual de referencia avanzado

Rafael Sarmiento de Sotomayor

ANAYA MULTIMEDIA

Amstrad CPC 464/664/6128

Manual de referencia avanzado

Amstrad CPC 464/664/6128

Manual de referencia avanzado

Rafael Sarmiento
de Sotomayor

ANAYA MULTIMEDIA

MICROMANUALES

Diseño de colección: Narcís Fernández

Villafranca, 22. 28028 Madrid
Depósito legal: M. 21.541-1987
ISBN: 84-7614-133-5
Printed in Spain
Imprime: Anzos, S.A. - Fuenlabrada (Madrid)

Indice

1

Disposición de la memoria en el Amstrad

El microprocesador Z80A, de Zilog, única CPU de los Amstrad CPC, es capaz de direccionar 64 Kbytes de memoria y 64K puertos de E/S (entrada/salida). A pesar de esta limitación, los diseñadores de Amstrad han sabido desarrollar un sistema capaz de manejar hasta 4096 Kbytes de memoria.

Originalmente, el CPC464 direcciona 64K de RAM y 32K de ROM; el CPC664 direcciona 64K de RAM y 48K de ROM; finalmente, el CPC6128 puede direccionar 128K de RAM y 48K de ROM, tal como se verá más tarde.

La memoria común de estos sistemas es como sigue:

- 16K de ROM inferior, conteniendo el sistema operativo.
- 64K de RAM de acceso directo desde el BASIC.

Los 16K superiores forman la memoria de pantalla. Direcciones C000 a FFCF.

- 16K de ROM superior, conteniendo el BASIC del sistema.
- Posibilidad de direccionar 64K de E/S.

Los CPC664 y 6128 incluyen, además, 16K de ROM superior, que contienen el sistema operativo de la unidad de disco. Por último, el CPC6128 tiene 64K adicionales de RAM, que se conmutan con la básica en forma de bancos de memoria de 16K. En el capítulo dedicado a los accesos a ROM se explica la forma de conmutar bloques de 16 Kbytes.

El mapa de memoria impuesto por el sistema operativo presenta la configuración de la figura 1.1.

Memoria de pantalla

La memoria actúa como soporte de los datos que precisa el controlador del tubo de rayos catódicos (CRT). El CRT lee cada microsegundo dos bytes del contenido de la memoria de pantalla. El origen de pantalla (es decir, el punto de la esquina superior izquierda) no tiene por qué coincidir con el contenido del bit 7 de la posición C000. El CRT suma automáticamente la base y el *offset* de pantalla, para obtener este origen. Tanto la base como el *offset* de pantalla son programables a través de las rutinas del bloque de saltos. El origen coincide con la dirección C000 sólo después de un cambio de modo de pantalla (MODE).

Tras un cambio de modo, el mapa de la memoria de pantalla presenta la configuración de la figura 1.2.

Vemos que cada línea de pantalla ocupa 80 bytes con direcciones consecutivas. Si se continúa incrementando la dirección, saltamos ocho filas más abajo, que es justo donde comienza la nueva línea de caracteres. Resumiendo, la memoria de pantalla no es más que una sola matriz, formada por ocho filas y 80*25 columnas.

Dependiendo del modo de pantalla, los bits de cada byte de la memoria de pantalla se comportarán de

<table>
<tr><th>dirección
(HEX)</th><th>RAM</th><th>ROM</th></tr>
<tr><td>0000
003A</td><td>Area de instrucciones RST</td><td rowspan="3">ROM INFERIOR

Sistema
operativo</td></tr>
<tr><td>003B

016F</td><td>Area para interrupciones
de usuario</td></tr>
<tr><td>0170
3FFF
4000

A8FF</td><td>Comienzo
de los programas

en BASIC.
Zona de usuario.</td></tr>
<tr><td>A900

ABFF</td><td>Area de símbolos
y variables
definibles por el usuario
y por ROM secundarias</td><td rowspan="4"></td></tr>
<tr><td>AC00

B0FF</td><td>Bloque de control,
variables del sistema,
datos de ROM primaria</td></tr>
<tr><td>B110

BFFF</td><td>Pila, bloque de saltos,
firmware y datos</td></tr>
<tr><td>C000
FFFF</td><td>Memoria de pantalla</td><td>ROM SUPERIOR
BASIC + DISCO</td></tr>
</table>

Figura 1.1

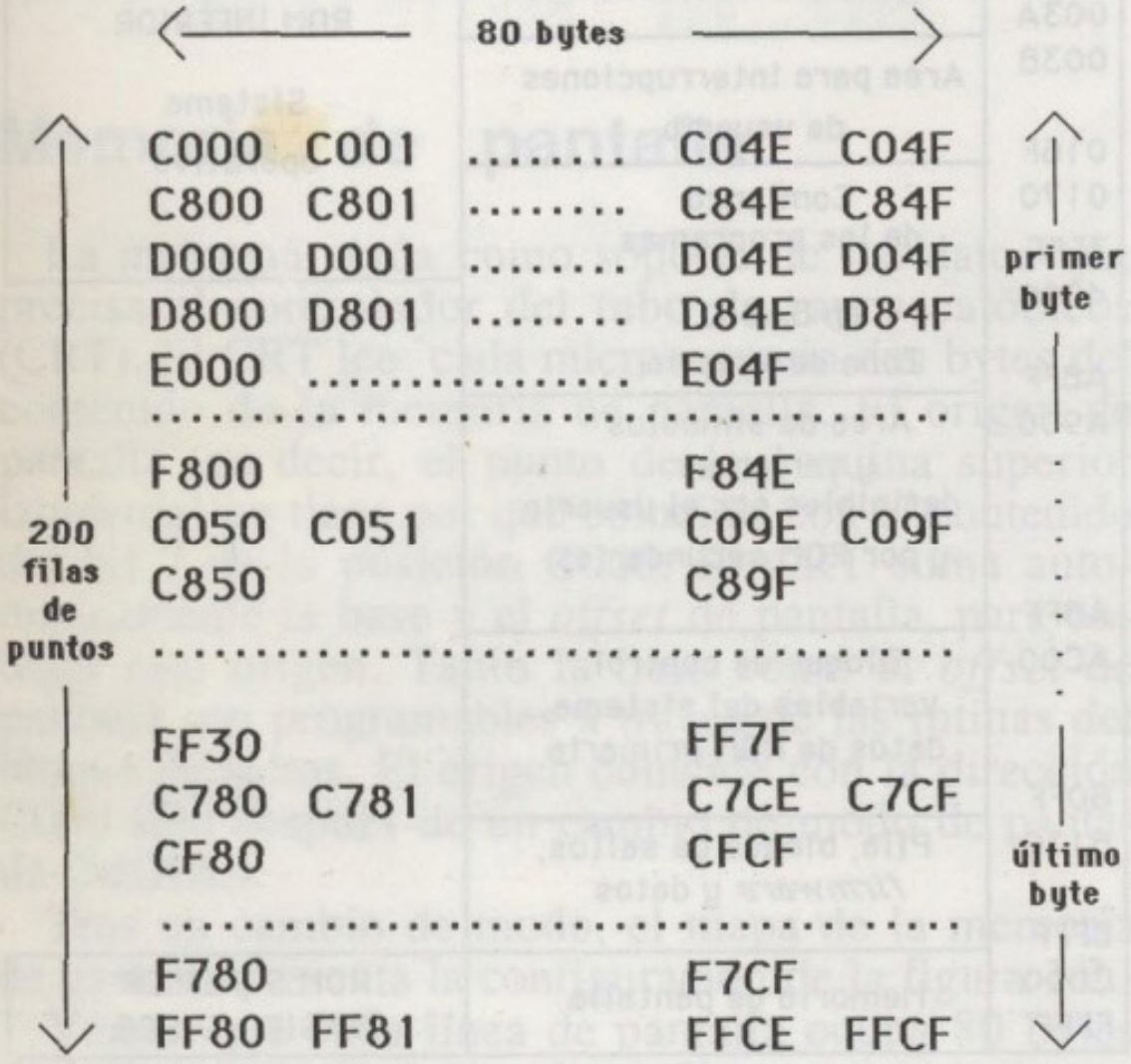
80 bytes
C000 C001 ..... C04E C04F
C800 C801 ..... C84E C84F
D000 D001 ..... D04E D04F
D800 D801 ..... D84E D84F
E000 ..... E04F
F800 F84E
C050 C051 C09E C09F
C850 C89F
FF30 FF7F
C780 C781 C7CE C7CF
CF80 CFCF
F780 F7CF
FF80 FF81 FFCE FFCF
200 filas de puntos
primer byte
último byte

Figura 1.2

forma diferente. En modo 2, cada *pixel* está asociado a un solo bit; en modo 1, a dos bits, y en modo 0, a cuatro bits. Estos 1, 2 ó 4 bits contienen un número entre (0 — 1), (0 — 3) y (0 — 15), respectivamente, que especifica el color asignado a ese *pixel*. Así, para cada byte, los bits asociados a un *pixel* serán:

	MODO 2 *bits*	*MODO 1* *bits*	*MODO 0* *bits*
pixel más a la izquierda	7	3-7	1-5-3-7
	6		
	5	2-6	
	4		
	3	1-5	
	2		
	1	0-4	
pixel más a la derecha	0		0-4-2-6
Total:	8	4	2

Direcciones de E/S

El mapa de entrada/salida sólo es accesible a través de las órdenes del BASIC, INP y OUT, o por medio de las instrucciones IN (C),C y OUT(C),C del Z80. Esta limitación se impone en beneficio de la gran cantidad de memoria direccionable. Además, el usuario sólo tiene un pequeño margen de direcciones para su propia utilización. Las direcciones para el usuario son:

F8E0-F8FE; F9E0-F9FF; FAE0-FAFF; FEB0-FBFF

El resto de direcciones posibles del sistema son:

71FXX	Matriz de la puerta de vídeo (salida).
BXXX	Controlador del tubo de rayos catódicos.

BCXX	Salida al registro elegido.
BDXX	Salida de datos.
BEXX	Entrada del registro de estado.
BFXX	Entrada de datos.
DFXX	Salida de datos de la ROM seleccionada.
EFXX	Canal de impresora (salida).
F4XX	Puerto A del PPI (E/S).
F5XX	Puerto B del PPI (E/S).
F6XX	Puerto C del PPI (E/S).
F7XX	Registro de control del PPI (salida).
F8FF	Reinicialización de los canales de expansión.
FB7X	Sistema de disco.
FBBX	Función reservada.
FBDX	Canal de comunicación.

2

Características del BASIC en el Amstrad

La memoria para BASIC disponible al inicializar el aparato es de 43533 bytes libres en modelo CPC464, 42619 en el CPC664 y de 42249 bytes en el CPC6128.

Los nombres de las variables, ya sean numéricas o de cadena, pueden tener hasta una longitud de 40 caracteres. La longitud máxima de una línea de programa o de una orden directa es de 255 caracteres.

Los programas pueden tener sus líneas numeradas entre 1 y 65535. Los programas en BASIC se almacenan en memoria a partir de la dirección 368(&170 en HEX). El margen de valores para las variables reales está entre —32768 y 32767. Todas las variables reales se consideran de simple precisión, pudiendo tomar valores entre +/—2.93874 E-39 y +/—1.70141 E+38.

Disposición del BASIC en la memoria

La estructura de una línea BASIC en la memoria responde a la configuración que se indica:

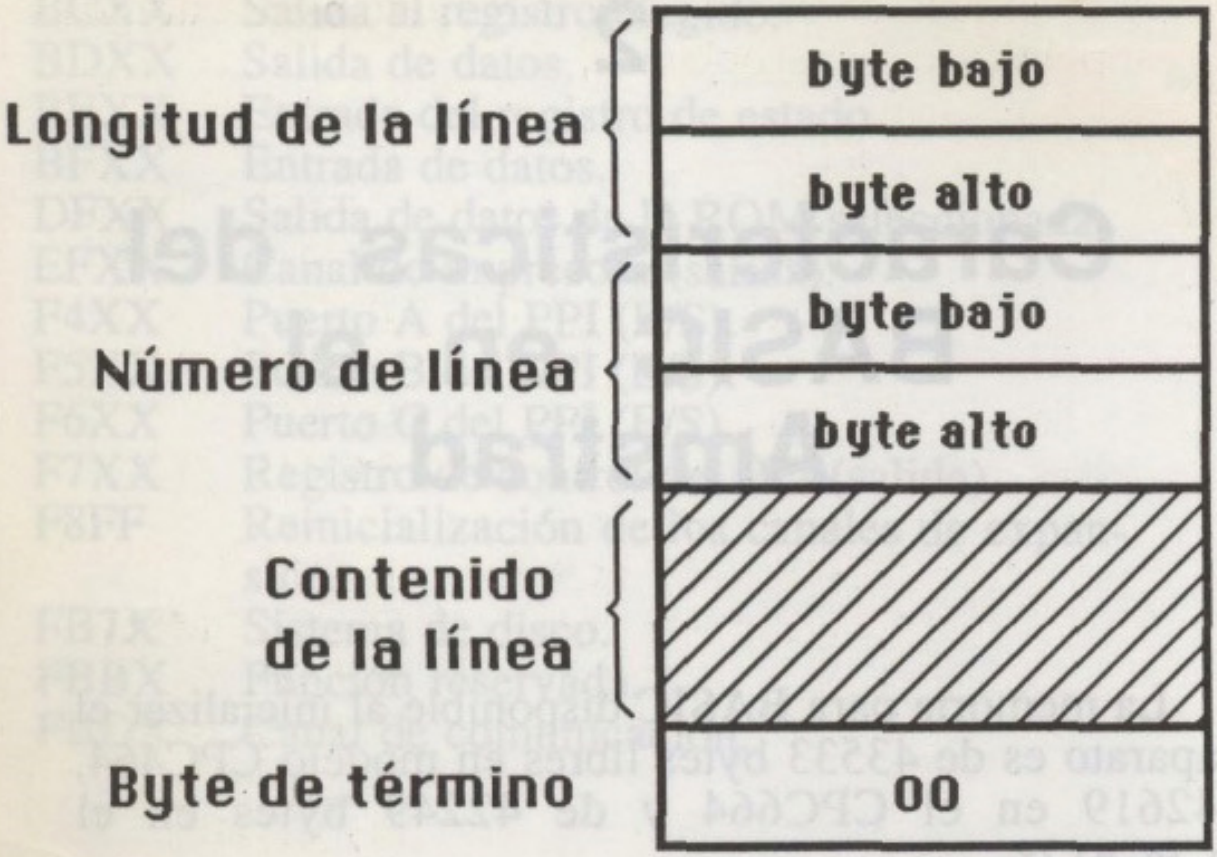

Figura 2.1

La longitud de la línea incluye los bytes de longitud de línea, el número de línea y el byte de término "0". Este byte terminal sirve para distinguir la separación entre unas líneas y otras. El texto está formado por todas las instrucciones, funciones, variables, cadenas y datos de la línea del programa.

Se escribirán literalmente en código ASCII:

- Los nombres de las variables, excepto el último carácter, al que se añade 128, incluso si tiene identificador de tipo.
- El texto de las sentencias REM.
- Las cadenas de caracteres.
- Las constantes.
- El código de paréntesis, comillas, etc. En cambio, las palabras clave del BASIC se insertan en la memoria en forma de *tokens* o códigos clave asociados a cada palabra.

A continuación se presenta la lista de estos códigos y su palabra clave asociada.

80	AFTER	B9	OUT
81	AUTO	BA	PAPER
82	BORDER	BB	PEN
83	CALL	BC	PLOT
84	CAT	BD	PLOTR
85	CHAIN	BE	POKE
86	CLEAR	BF	PRINT
87	CLG	C0	'
88	CLOSEIN	C1	RAD
89	CLOSEOUT	C2	RANDOMIZE
8A	CLS	C3	READ
8B	CONT	C4	RELEASE
8C	DATA	C5	REM
8D	DEF	C6	RENUM
8E	DEFINT	C7	RESTORE
8F	DEFREAL	C8	RESUME
90	DEFSTR	C9	RETURN
91	DEG	CA	RUN
92	DELETE	CB	SAVE
93	DIM	CC	SOUND
94	DRAW	CD	SPEED
95	DRAWR	CE	STOP
96	EDIT	CF	SYMBOL
97	ELSE	D0	TAG
98	END	D1	TAGOFF
99	ENT	D2	TROFF
9A	ENV	D3	TRON
9B	ERASE	D4	WAIT
9C	ERROR	D5	WEND
9D	EVERY	D6	WHILE
9E	FOR	D7	WIDTH
9F	GOSUB	D8	WINDOW
A0	GOTO	D9	WRITE
A1	IF	DA	ZONE
A2	INK	DB	DI

A3	INPUT	DC	EI
A4	KEY	EA	TAB
A5	LET	EB	THEN
A6	LINE	EC	TO
A7	LIST	ED	USING
A8	LOAD	EE	>
A9	LOCATE	EF	=
AA	MEMORY	F0	>=
AB	MERGE	F1	<
AC	MID$	F2	<>
AD	MODE	F3	<=
AE	MOVE	F4	+
AF	MOVER	F5	—
B0	NEXT	F6	*
B1	NEW	F7	/
B2	ON	F8	
B3	ON BREAK	F9	\
B4	ON ERROR GOTO	FA	AND
B5	ON SQ	FB	MOD
B6	OPENIN	FC	OR
B7	OPENOUT	FD	XOR
B8	ORIGIN	FE	NOT
FF+00	ABS	FF+1B	UNT
FF+01	ASC	FF+1C	UPPER$
FF+02	ATN	FF+1D	VAL
FF+03	CHR$	FF+40	EOF
FF+04	CINT	FF+41	ERR
FF+05	COS	FF+42	HIMEM
FF+06	CREAL	FF+43	INKEY$
FF+07	EXP	FF+44	PI
FF+08	FIX	FF+45	RND
FF+09	FRE	FF+46	TIME
FF+0A	NKEY	FF+47	XPOS
FF+0B	INP	FF+48	YPOS
FF+0C	INT	FF+71	BIN$
FF+0D	JOY	FF+72	DEC$
FF+0E	LEN	FF+73	HEX$

FF+0F	LOG	FF+74	INSTR
FF+10	LOG10	FF+75	LEFT$
FF+11	LOWER$	FF+76	MAX
FF+12	PEEK	FF+77	MIN
FF+13	REMAIN	FF+78	POS
FF+14	SGN	FF+79	RIGHT$
FF+15	SIN	FF+7A	ROUND
FF+16	SPACE$	FF+7B	STRING$
FF+17	SQ	FF+7C	TEST
FF+18	SQR	FF+7D	TESTR
FF+19	STR$	FF+7E	
FF+1A	TAN	FF+7F	VPOS

Disposición en la memoria de una línea BASIC

Considérese, por ejemplo, la siguiente línea:

10 VAR=4:PRINT VAR;"es un número"

Almacenada quedará:

26 00 0A 00 0D 07 00 56 41 D2 EF 19 0A 01 BF
20 0D 07 00 56 41 D2 3B 22 65 73 20 75 6E 20
6E 75 6D 65 72 6F 22 00

El significado de cada byte es:

26 00	38 en decimal.Es la longitud de la línea.
0A 00	Número de línea del programa (10).
0D	Indica que es variable numérica.
07	Longitud de la variable +4.
00	Separador.
56 41 D2	Código ASCII de VAR sumando 128 al último.

EF	Código del signo =.
19	Tamaño de la variable.
0A 01	Valor de la variable y separador.
BF 20	Código para PRINT y espacio.
0D 07 00	Véase arriba.
56 41 D2	Código de VAR.
3B 22	Código ASCII de los símbolos (;), (").
65 73 20	es
75 6E 20	un
6E 75 6D	núm
65 72 6F	ero
22 00	Símbolo de comillas y código terminal.

3

Instrucciones del BASIC

AFTER
AFTER t [,c] GOSUB n

Llama a la subrutina situada en *n*, después de que haya transcurrido un tiempo *t* medido en unidades de 0,02 segundos. El parámetro *c* es opcional e indica cuál de los cuatro temporizadores se empleará con esta instrucción. Se tomará el temporizador 0 por defecto.

AUTO
AUTO [n] [,i]

Genera automáticamente números de línea, comenzando desde la número *n* e incrementando según *i*. Por defecto, se tomará el valor 10 para ambos parámetros.

BORDER
BORDER a [,b]

El color del borde de la pantalla pasa a ser el indi-

cado por el parámetro *a*, y si se incluye el parámetro opcional, el borde de la pantalla cambiará alternativamente de uno a otro color.

CALL
CALL dir [,parámetros]

Llamada desde BASIC a una rutina en código máquina, situada en memoria en la dirección *dir*. Opcionalmente, se puede añadir una lista de parámetros cuya función y significado viene explicado en el capítulo 6.

CAT
CAT

Lee del disco o del *cassette* el nombre de los ficheros y programas almacenados. En el caso de disco, muestra la cantidad de memoria ocupada por cada fichero. En el caso de *cassette*, indicará el número de bloques y el tipo de fichero.

CHAIN
CHAIN "f" [,n]

CHAIN MERGE
CHAIN MERGE "f" [,n] [,DELETE m]

La primera instrucción introduce el programa "f" en la memoria, borrando el anterior, y opcionalmente, comenzará a ejecutarlo a partir de la línea *n*. La segunda instrucción enlaza el programa "f" con el que había en memoria, borrando si se quiere (DELETE) una gama de números de línea del programa en

memoria. En ambas intrucciones, el contenido de las variables se conserva.

CLEAR
CLEAR

Borra el contenido de todas las variables.

CLEAR INPUT
CLEAR INPUT*

Vacía el *buffer* de teclado, eliminando todos los caracteres presentes.

CLG
CLG

Borra la pantalla de gráficos.

CLOSEIN
CLOSEIN

CLOSEOUT
CLOSEOUT

Cierran, respectivamente, un fichero de entrada y de salida.

CLS
CLS [#N]

Limpia la ventana *N* y la pone con el color del papel asignado a ella. Por defecto, se toma el cauce 0.

CONT
CONT

Reanuda la ejecución de un programa, después de haberse producido un *break*, STOP o END. Esta opción puede ser de gran utilidad.

CURSOR
CURSOR flag-s,flag-u

Activa o desactiva el cursor de texto según el estado de los flags del sistema o de usuario. Si el *flag* está a uno, el cursor estará activado; por el contrario, si el *flag* del sistema está a cero, el cursor será inhibido.

DATA
DATA a, b, c

Permite definir una lista de valores de forma secuencial. Puede haber, como máximo, tantos parámetros como quepan en la línea del programa. Los valores sólo podrán leerse con la instrucción READ.

DEF FN
DEF FNz [(x,y..)] =expresión

Permite definir la función "z", que depende de las variables *x*, *y*, etc., como una expresión matemática que puede contener o no a las variables mencionadas.

DEFtipo

DEFINT A-F
DEFSTR I,J,K,L
DEFREAL RT,PT

Define la gama de variables que comienzan con la letra o letras indicadas, como variables enteras, de cadena o reales, respectivamente.

DEG
DEG

Impone los cálculos en grados para las funciones trigonométricas. Al conectar el aparato, los cálculos por defecto se hacen en radianes; si se desea utilizar otro sistema, es necesario especificarlo.

DELETE

DELETE n1,n2...
DELETE n1-n3

Elimina del programa las líneas n1, n2, etc., en el primer caso. En el segundo caso, borrará las líneas que haya entre n1 y n3, ambas inclusive.

DI
DI

Inhibe interrupciones. Todas las órdenes que generen interrupciones, excepto un *break,* dejarán de trabajar. La situación normal se alcanzará al ejecutarse una instrucción EI.

DIM
DIM num(n1,n2...),cad(m1...)

Dimensiona una o varias matrices de forma mono o multidimensional y con dimensiones n1, n2, etc. Por defecto, una variable es automáticamente dimensionada a 10.

DRAW
DRAW x,y [,c]

Dibuja una línea desde la posición actual del cursor de gráficos hasta el punto de coordenadas x, y con el color "c".

DRAWR
DRAWR x,y [,c]

Dibuja una línea desde la posición actual del cursor de gráficos hasta el punto de coordenadas relativas x, y con el color "c".

EDIT
EDIT n

Edita la línea *n* del programa.

EI
EI

Permite las interrupciones; cancela el efecto de DI.

END
END

Finaliza la ejecución de un programa que estaba funcionando.

ENT
ENT ne [,se]

Define la envolvente de tono que se quiere utilizar (efecto de vibrato). El parámetro *ne* indica el número de envolvente que se desea (de 0 a 15). Puede haber hasta 5 grupos de parámetros *se*, o secciones de envolvente, de la forma siguiente: número de pasos, valor de frecuencia de cada paso y tiempo para cada paso.

ENV
ENV ne [,se]

Define la envolvente de volumen para los sonidos. El parámetro *ne* indica el número de envolvente (de 0 a 15). Puede haber hasta 5 grupos de parámetros *se*, o secciones de envolvente, de la forma que se indica a continuación: número de pasos, nivel de volumen de cada paso y tiempo para cada paso.

ERASE
ERASE lista de variables

Libera el espacio de memoria que se reservó anteriormente por órdenes DIM para las variables especificadas.

ERROR
ERROR n

Comienza la acción propia para el error número *n*.

EVERY
EVERY t,c GOSUB nl

Cada *t* centésimas de segundo se ejecutará la subrutina de la línea número *nl*. El parámetro *c* indica qué temporizador se utilizará.

FILL *
FILL tinta

Rellena con color de *tinta* el área donde está situado el cursor de gráficos.

FRAME *
FRAME

Sincroniza la acción sobre la pantalla con el barrido de cuadro para evitar las fluctuaciones de los caracteres.

FOR
FOR var=i to f [STEP p]

Comienzo de un bucle. Esta instrucción hace que las líneas de programa contenidas entre esta intrucción y el primer NEXT que aparezca se repitan tantas veces como tarde la variable **var** en alcanzar el valor **f** partiendo del valor **i**, e incrementándose en **p** a cada paso.

Los parámetros **i, f** y **p** pueden ser positivos y negativos, enteros o reales.

GRAPHICS *

GRAPHICS PEN tinta
GRAPHICS PAPER tinta

Fijan, respectivamente, la tinta de la pluma de gráficos y la del papel de gráficos.

GOSUB
GOSUB nl

Llamada a la subrutina que comienza en la línea **nl**.

GOTO
GOTO nl

Salto a la línea de número **nl**.

IF
IF cond THEN in1 [ELSE in2]

Se evalúa la condición dada por **cond**. Si resulta ser cierta, se ejecutará la instrucción **in1**; en caso contrario, se ejecutará la siguiente línea del programa u opcionalmente la instrucción **in2**.

INK
INK tin,c1 [,c2]

Según el modo de pantalla actual, se dispone de un cierto número de tintas.

Esta instrucción asigna el color **c1** de la paleta de colores a la tinta **tin.** Opcionalmente, puede asignarse un segundo color **c2**, que se alternará con el primero a la velocidad especificada con SPEED INK .

INP
INP(dir-puerto)

Devuelve el contenido de puerto que está en la dirección especificada.

INPUT
INPUT [#c][;]["cadena";] lista

Lee datos desde el cauce **c** especificado (0 por defecto). Los datos leídos se asignarán consecutivamente a las variables de la lista, debiendo haber igual número de datos y de variables. La cadena literal **cadena** aparecerá siempre que los datos se introduzcan desde el teclado.

KEY
KEY nt,cadena de caracteres

Permite definir la tecla de función número **nt** (de 128 a 140), asignándole la cadena de caracteres de la instrucción.

KEY DEF
KEY DEF, numt,rep,numcódigo [,may [,ctrl]]

Cambia el código asignado a la tecla **numt**, indicando si admite repetición con **rep** distinto de cero

o si no la admite con **rep**=0. El código ASCII para la tecla viene dado por **numcódigo**, por **may** cuando se pulsa junto a MAYS (SHIFT) y por **ctrl** cuando se pulsa junto a CONTROL.

LET
LET var=expresión

Asigna el valor de la expresión de la derecha a la variable **var**. Es una instrucción opcional.

LINE INPUT
LINE INPUT [#c][;]["caden";] var

Lee una cadena de caracteres completa a través del cauce **c** especificado (0 por defecto) y la asigna a la variable **var**. Se diferencia de INPUT en que pueden introducirse tantas comas y espacios como se desee, incluyéndose todos ellos en la misma variable.

LIST
LIST [ini-fin] [,#c]

Lista el programa presente en la memoria a través del cauce **c** indicado (0 por defecto). Tanto **ini** como **fin** son parámetros opcionales e indican, respectivamente, la primera y la última línea de la sección que se desea listar del programa.

LOAD
LOAD "nombre" [,direcc]

Carga en la memoria el programa llamado **nombre**. Si se trata de un fichero ASCII o de un progra-

ma en binario, será necesario indicar la dirección **dir** a partir de la cual se almacenará en la memoria. Utilizando *cassette*, no es necesario indicar el nombre, ya que así se carga el primer programa que encuentre.

LOCATE
LOCATE [#v,] X,Y

Sitúa el cursor de texto en la columna X, fila Y, dentro de la ventana **v** especificada (la 0 por defecto). Las coordenadas empiezan por 1.

MEMORY
MEMORY dirección

Permite redefinir la dirección tope de memoria a la que tiene acceso el BASIC.

MERGE
MERGE "nombre"

Enlaza el programa o fichero **nombre** con el que reside actualmente en memoria. Si dos números de línea coinciden, permanecerá la del programa cargado. La diferencia con CHAIN MERGE es que ahora no se conservan las variables ni funciones definidas anteriormente.

MODE
MODE n

Modifica el actual modo de pantalla al de número **n**. Sólo son posibles los modos 0, 1 y 2.

MOVE
MOVE X,Y

Sitúa el cursor de gráficos en el punto de coordenadas X (abcisas), Y (ordenadas).

MOVER
MOVER X,Y

Sitúa el cursor de gráficos en el punto de coordenadas X (abcisas), Y (ordenadas) relativas a su posición actual.

NEW
NEW

Borra la memoria, afectando sólo al programa y variables definidas.

NEXT
NEXT [T,H,..]

Marca el final del bucle comenzado por el último FOR. Cuando aparece más de una variable opcional, la instrucción se interpretará como varias NEXT consecutivas.

ON...GOTO
ON n GOTO lista de números

ON...GOSUB
ON n GOSUB N1,N2,N3,...

Desvía la ejecución del programa a la línea *n*-

ésima, de las indicadas en la lista de números de línea. En el segundo caso, la línea marcará el comienzo de una subrutina.

ON BREAK CONT *
ON BREAK CONT

Desactiva la tecla ESC que genera los *break*.

ON BREAK GOSUB
ON BREAK GOSUB nlínea

Llama a la subrutina que comienza en **nlínea**, cuando se produce una interrupción del programa.

ON BREAK STOP
ON BREAK STOP

Cancela el efecto de la instrucción ON BREAK GOSUB.

ON ERROR GOTO
ON ERROR GOTO nlínea

Cuando aparece un error, bifurca la ejecución del programa hacia la línea indicada.

ON SQ GOSUB
ON SQ (c) GOSUB nlínea

Ejecutará la subrutina situada en **nlínea**, cuando en la cola de sonido correspondiente al canal **c** aparezca un hueco que no esté utilizado.

El parámetro será 1 para el canal A, 2 para el canal B y 3 para el canal C.

OPENIN
OPENIN "Nfichero"

Abre un fichero de entrada de la unidad de almacenamiento (disco o *cassette).* El fichero debe tener el mismo nombre que el indicado por la instrucción. Los datos se extraerán secuencialmente. En el caso de utilizar *cassette,* el fichero se irá leyendo de la cinta en bloques de dos Kbytes.

OPENOUT
OPENOUT "Nfichero"

Abre un fichero de salida para la unidad de almacenamiento. El fichero se guardará con el nombre **Nfichero** y los datos se almacenarán secuencialmente en bloques de dos Kbytes, o cuando se cierre el fichero.

ORIGIN
ORIGIN X,Y [,D,I,A,B]

Determina las coordenadas X e Y del origen para el cursor de gráficos. Opcionalmente, se pueden añadir cuatro parámetros que definan una nueva ventana para los gráficos.

OUT
OUT p,n

Envía el valor entero **n** al puerto **p.** El primero

puede tener valores entre 0 y 255, mientras que el puerto puede direccionarse entre 0 y 65535.

PAPER
PAPER [#v,] ntinta

Fija el color (número de tinta) para el fondo de la ventana **v**.

PEN
PEN [#v,] ntinta

Fija el color con el que serán escritos los siguientes caracteres en la ventana de texto **v**.

PLOT
PLOT x,y [,ntinta]

Dibuja en la pantalla de gráficos un punto en las coordenadas x,y. Opcionalmente, se puede incluir el número de la tinta con la que dibujará.

PLOTR
PLOTR x,y [,ntinta]

Dibuja en la pantalla de gráficos un punto en las coordenadas x,y, relativas a la posición actual del cursor. Opcionalmente, se puede incluir el número de la tinta con la que dibujará.

POKE
POKE dir,dato

Introduce en la dirección **dir** de la memoria el **dato** indicado. Este puede tomar valores entre 0 y 255.

PRINT
PRINT [#c,] datos

Escribe a través del cauce **c** los datos o cadenas literales entrecomilladas que aparecen a continuación. PRINT USING permite especificar diferentes formatos de impresión. El cauce 8 corresponde a la impresora.

RAD
RAD

Fija en radianes el modo de trabajo con las funciones trigonométricas.

RANDOMIZE
RANDOMIZE [n]

Define una nueva secuencia de números pseudoaleatorios. La secuencia comenzará por **n,** siendo éste un entero entre 0 y 65535. Por defecto, n = 0.

READ
READ lista de variables

Lee secuencialmente los datos contenidos en la línea DATA del programa y los asigna por orden a las variables de la lista.

RELEASE
RELEASE canal

Finaliza el estado de espera del canal de sonido indicado.

REM
REM comentario

Todos los comentarios incluidos después de REM, serán ignorados por el programa. Esta acción es válida solamente para esa línea del programa.

RENUM
RENUM [nn] [,ante] [,inc]

Renumera las líneas del programa según los parámetros indicados. A partir de la línea **ante,** las líneas se numerarán empezando por el nuevo número **nn,** incrementándose según **inc**. Omitiendo todos los parámetros, el programa se numerará desde el principio, empezando desde 10 e incrementándose de 10 en 10.

RESTORE
RESTORE [nlínea]

Indica cuál es la siguiente línea con DATA que será leída por READ. Si no se especifica ningún número de línea, la lectura comenzará en la primera línea del programa con DATA.

RESUME
RESUME [nlínea]

Permite continuar la ejecución de un programa en un determinado número de línea, después de detectado y corregido un error por medio de ON ERROR GOTO.

RETURN
RETURN

Regresa al programa principal tras finalizar la subrutina llamada por GOSUB.

RUN

RUN [nlínea]
RUN "programa"

Ejecuta el programa residente en la memoria. Opcionalmente, se puede ejecutar a partir de la línea número **nlínea**. La segunda versión cargará y ejecutará desde la primera línea el programa mencionado. Con *cassette,* RUN cargará y ejecutará el primer programa que encuentre en la cinta.

SAVE
SAVE "nombre" [,t][,dir,long][,com]

Guarda en la unidad de almacenamiento el programa o fichero **nombre**. Se puede especificar el tipo de fichero de que se trate, siendo **tip** = A para ficheros ASCII; **tip** = P para programas a proteger, y **tip** = B, para binarios, en los cuales hay que indicar la dirección donde se encuentran en memoria, su longitud en bytes y, opcionalmente, la dirección de comienzo.

SOUND
SOUND c,p,d [,v,evol,eton,r]

Emite un sonido, especificado por los siguientes parámetros: **c** es el número de canal; **p** es el período

del tono; **d** es la duración; **v** es el volumen de arranque; **evol** es el número de envolvente de volumen; **eton** es el número de envolvente de tono, y **r** es el indicador de ruido. Véase la explicación completa en el capítulo de SONIDOS.

SPEED INK
SPEED INK t1,t2

Modifica la velocidad de parpadeo entre dos tintas asignadas a un mismo papel, pluma o borde. El tiempo de duración de la primera tinta viene indicado por **t1**, y el de la segunda por **t2**.

SPEED KEY
SPEED KEY ret,p-rep

Determina el tiempo **ret**, que una tecla ha de esperar antes de repetir su código, y el período de repetición **p-rep**; ambos, en cincuentavos de segundo.

SPEED WRITE
SPEED WRITE n

Cambia la velocidad con que se grabarán programas en el *cassette*. Con n = 0, que es el valor al conectar, se graban a 1000 baudios; con n = 1, la velocidad es de 2000 baudios. En lectura, el ordenador se adapta por sí solo a la velocidad de grabación de la cinta.

STOP
STOP

Detiene la ejecución de un programa, permitiendo su reanudación mediante la instrucción CONT.

SYMBOL
SYMBOL n,lista de 8

Redefine la matriz (forma) del carácter número **n**. Cada uno de los ocho parámetros de la lista corresponde a la definición en binario de las ocho líneas que componen el carácter.

SYMBOL AFTER
SYMBOL AFTER n

Permite redefinir caracteres a partir del número **n** (hasta 255). Al conectar el aparato, n = 240.

TAG
TAG [#c]

Permite escribir caracteres en la posición actual del cursor de gráficos, dentro del cauce **c**.

TAGOFF
TAGOFF [#c]

Cancela el efecto de TAG para el cauce especificado (0 por defecto).

TRON
TRON

Instaura el modo *trace*, con el cual se escribirán consecutivamente los números de línea que se vayan ejecutando a lo largo del programa. Muy útil para el depurado de los mismos.

TROFF
TROFF

Cancela el efecto de TRON.

WAIT
WAIT puerto,masc,b

Interrumpe la ejecución del programa para leer a través de un puerto de entrada el byte presente en él. Con ese byte y con el byte de máscara **masc** realiza un AND lógico y, finalmente, un XOR con el byte b. La ejecución sólo se reanudará cuando el resultado de estas operaciones sea distinto de cero.

WEND
WEND

Fin del bucle iniciado por la orden WHILE.

WHILE
WHILE expresión

El programa ejecutará las instrucciones comprendidas entre WHILE y WEND mientras sea cierta la expresión lógica indicada.

WIDTH
WIDTH n

Fija el número de caracteres por línea que se quieren utilizar con la impresora.

WINDOW
WINDOW [#c,] I,D,A,B

Define la ventana de texto que representa el cauce **c** (0 para omisiones). I y D determinan la columna izquierda y derecha; A y B las líneas superior e inferior.

WINDOW SWAP
WINDOW SWAP v1,v2

Intercambia la definición de las ventanas **v1** y **v2**.

WRITE
WRITE [#c,] lista

Escribe la lista a través del cauce **c**, sin modificar los signos de puntuación que en ella aparezcan.

ZONE
ZONE n

Cambia la anchura entre columnas cuando se utiliza la orden PRINT con coma. Por defecto, esta anchura es de 13 espacios.

4

Funciones del BASIC

ABS
ABS(expresión)

Devuelve el valor absoluto de la expresión numérica o número entre paréntesis.

ASC
ASC(caracteres)

Devuelve el código ASCII del primer carácter de la cadena entre paréntesis.

ATN
ATN(expresión)

Devuelve el valor en radianes o grados (según el modo actual) del arcotangente de la expresión numérica que se especifique entre paréntesis. Si se desea, se puede cambiar el modo actual antes de esta orden.

BIN$
BIN$(N decimal [,k])

Convierte el entero decimal N, en un número binario formado por tantos dígitos como indique k.

CHR$
CHR$(N)

Devuelve el carácter ASCII del código N, siendo N un entero entre 0 y 255.

CINT
CINT(expresión)

Devuelve un número entero, resultado de redondear el valor de la expresión numérica.

*** COPYCHR$**
COPYCHR$(#v)

Variable que contiene el carácter situado en la posición actual del cursor dentro de la ventana v.

COS
COS(x)

Devuelve el valor del coseno del ángulo indicado. Este se tomará en radianes o grados, según el modo definido (DEG o GRA).

CREAL
CREAL(expresión)

Convierte el valor de la expresión numérica en un número real.

*** DEC$**
DEC$(expresión,forma)

Formatea un número, variable o expresión para su impresión. Análogo a PRINT USING.

*** DERR**
DERR

Escribe el número del último error producido.

EOF
EOF

Variable asociada a los ficheros de la unidad de almacenamiento. Si se ha alcanzado el final del fichero, EOF = —1; en otro caso, EOF = 0.

ERR
ERR

Variable que contiene el número del último error producido durante la ejecución del programa en curso.

ERL
ERL

Variable que contiene el número de línea donde se produjo el último error.

EXP
EXP(n)

Devuelve la potencia **n** del número **e**.

FIX
FIX(n)

Devuelve la parte entera del número **n**, limitándose a tomar los dígitos a la izquierda del punto decimal.

FRE
FRE(z) o FRE("x")

Devuelve el número de bytes que permanecen libres en la memoria. Cuando el argumento es un carácter literal, limpia la memoria de datos inútiles antes de devolver el número.

HEX$
HEX$(n)

Convierte en hexadecimal un número entero.

HIMEM
HIMEM

Variable que contiene la dirección de memoria más alta disponible para el BASIC.

INKEY
INKEY(t)

Examina el teclado y determina si la tecla **t** está pulsada. En caso afirmativo, INKEY=0; en caso negativo, su valor es **—1**. Si la tecla está pulsada a la vez que MAYS (SHIFT), el valor es 32; y 128 si se pulsa junto a CONTROL.

INKEY$
INKEY$

Contiene el carácter del teclado pulsado.

INSTR
INSTR([n,] a$,b$)

Determina a partir de qué carácter de a$ está contenida la cadena b$, si es que lo está. El parámetro opcional **n** indica que la búsqueda comenzará a partir del carácter *n-ésimo* de a$.

INT
INT(expresión)

Para números positivos, el resultado es la parte entera de la expresión; es decir, igual que con FIX. Para números negativos no enteros, el resultado es la parte entera redondeada.

JOY
JOY(n)

Lee el byte asignado al *joystick* n (0 ó 1). Véase apéndice para la interpretación del byte.

LEFT$
LEFT$(cadena,n)

Devuelve los **n** caracteres más a la izquierda de la cadena literal.

LEN
LEN(cadena)

Devuelve el número de caracteres de la cadena literal.

LOG
LOG(x)

Calcula el logaritmo neperiano de x.

LOG10
LOG10(x)

Calcula el logaritmo en base 10 de x.

LOWER$
LOWER$(cadena)

Transforma en minúsculas todas las letras mayúsculas de la cadena.

*** MASK**
MASK N,M

Permite dibujar líneas con DRAW de trazo no continuo, sino formadas por máscaras de 8 bits segú

la equivalencia binaria de N, comenzando la línea con el bit especificado en M. M, entre 0 y 7; N, entre 0 y 255.

MAX
MAX(lista numérica)

Devuelve el valor más alto encontrado en la lista de expresiones numéricas.

MID$
MID$(cadena,n [,p])

Devuelve los primeros **p** caracteres que se encuentren a partir del carácter que ocupa la posición **n**.

*** MID$**
MID$(cad1,n,h)=cad2

Inserta la cadena **cad2** en la **cad1**, comenzando a partir del carácter de la posición **n** y sustituyendo en **cad1**, tantos caracteres de **cad2** como indica **h**.

MIN
MIN(lista numérica)

Devuelve el valor más bajo encontrado en la lista de expresiones numéricas.

PEEK
PEEK(n)

Devuelve el valor contenido en el byte de memoria de dirección **n**.

PI
PI

Constante que contiene el valor del número PI.

POS
POS(#c)

Variable que contiene la posición horizontal del cursor de texto (o de impresora), dentro del cauce **c**.

REMAIN
REMAIN(t)

Desactiva el temporizador **t** especificado y devuelve el tiempo que restaba en dicho cronómetro. El parámetro **t** puede valer 0, 1, 2 ó 3.

RIGHT$
RIGHT$(cadena)

Devuelve los **n** caracteres más a la derecha de la cadena literal.

RND
RND(n)

Devuelve el número pseudoaleatorio siguiente de la secuencia determinada por la instrucción RANDOMIZE. Si el número **n** es negativo, entonces se repetirá exactamente la misma secuencia cada **n** números.

ROUND
ROUND(x [,n])

Sin el parámetro opcional **n** devuelve la parte entera de **x**. Con el parámetro opcional devuelve el número **x** con tantos decimales como indique **n**, redondeados. Con **n** negativo redondea la parte entera hasta el dígito que ocupa la potencia *n-ésima* de 10.

SGN
SGN(expresión)

Determina el signo de la expresión, devolviendo -1 si es negativa; 0 si es cero, y 1, si es positiva.

SIN
SIN(ángulo)

Devuelve el valor del seno del ángulo indicado. Este se tomará en radianes o grados, según el modo definido (DEG o GRA).

SPACE$
SPACE$(n)

Crea una cadena literal formada por **n** espacios consecutivos; **n** entre 0 y 255.

SPC
SPC(n)

Utilizado con PRINT genera **n** espacios.

SQ
SQ(cs)

Devuelve el número de espacios libres en la cola de sonido del canal **cs**.

SQR
SQR(x)

Devuelve la raíz cuadrada de **x**.

STR$
STR$([&] N)

Convierte la expresión numérica N a una cadena literal. Si N es un número hexadecimal, primero se convertirá en decimal.

STRING$
STRING$(n,car)

Crea una cadena literal formada por **n** caracteres **car**. El parámetro **car** puede ser un número de código ASCII o un carácter entre comillas.

TAB
TAB(n)

Utilizado con PRINT sitúa el cursor de texto en la columna **n** dentro de la línea en que se encuentre.

TAN
TAN(ángulo)

Devuelve el valor de la tangente del ángulo indicado. Este se tomará en radianes o grados, según el modo definido (DEG o GRA).

TEST
TEST(x,y)

Devuelve el valor de la tinta del punto de pantalla de coordenadas absolutas x,y.

TESTR
TESTR(x,y)

Devuelve el valor de la tinta del punto de pantalla de coordenadas x,y, relativas a la posición actual del cursor.

TIME
TIME

Variable que contiene el tiempo transcurrido desde la conexión del aparato. Esta variable se utiliza para la generación de números pseudoaleatorios.

UNT
UNT(n)

Realiza la función inversa al complemento a dos; es decir, los números enteros sin signo entre 0 y 65535 son convertidos en enteros entre —32767 y 32768. Siendo UNT(65535)=—32767.

UPPER$
UPPER$(cadena)

Transforma en mayúsculas, todas las letras minúsculas de la cadena.

VAL
VAL(cadena)

Devuelve el valor del número que encuentre en los primeros lugares de la cadena. Si no hay ninguno, devolverá un cero.

VPOS
VPOS(#c)

Variable que contiene la posición vertical del cursor de texto (o de impresora) dentro del cauce **c**.

XPOS
XPOS

Variable que contiene la posición horizontal del cursor de gráficos.

YPOS
YPOS

Variable que contiene la posición vertical del cursor de gráficos.

5

ROM de BASIC

Principales direcciones de la ROM superior del CPC 6128

La ROM superior contiene todas las rutinas de procesamiento de claves en BASIC.

C006	Inicialización y salida de BASIC 1.1 (mensaje)
C033	BASIC 1.1 (mensaje)
C046	Función EDIT
C058	Entrada principal (aparece READY)
C0D7	READY (mensaje)
C0EA	AUTO
C128	NEW
C12F	CLEAR
C239	PAPER
C224	PEN
C239	PAPER
C248	BORDER
C251	INK
C275	MODE
C280	CLS
C298	COPYCHR$
C2A1	VPOS

C2A5	POS
C2FF	LOCATE
C30E	WINDOW
C343	TAG
C34A	TAGOFF
C360	CURSOR
C42A	WIDTH
C44F	EOF
C4DE	ORIGIN
C506	CLG
C512	FILL
C52F	MOVE
C534	MOVER
C539	DRAW
C53E	DRAWR
C543	PLOT
C548	PLOTR
C571	TEST
C576	TESTR
C59A	GRAPHICS
C5C0	MASK
C5D4	FOR
C6A2	NEXT
C767	IF
C786	GOTO
C78C	GOSUB
C7B0	RETURN
C7E7	WHILE
C81A	WEND
C882	ON
C976	ON BREAK
C997	DI
C99D	EI
C9F5	ON SQ
CA22	AFTER
CA2A	EVERY
CA50	REMAIN
CB51	ERROR

CBF1	UNDEFINED LINE (mensaje)
CC01	Manda el mensaje "BREAK IN"
CC1C	BREAK (mensaje)
CC22	IN (mensaje)
CC26	STOP
CC31	END
CC93	CONT
CCCA	ON ERROR
CCD5	RESUME
CD14	Tabla de mensajes de error (parte de la palabra)
CFED	Tabla de puntos de entrada para operaciones lógicas y aritméticas
D01D	—
D028	NOT
D036	+
D117	Tabla de puntos de entrada para las funciones EOF, ERR, HIMEM, INKEY$, PI, RND, TIME, XPOS y YPOS.
D12B	DERR
D130	ERR
D139	TIME
D142	ERL
D148	HIMEM
D14E	@
D161	XPOS
D168	YPOS
D1E5	Tabla de puntos de entrada para funciones
D23F	MIN
D243	MAX
DE6A	ROUND
D968	CAT
D2A8	OPENOUT
D2B4	OPENIN
D2ED	CLOSEIN
D2F5	CLOSEOUT

D313	SOUND
D370	RELEASE
D37B	SQ
D39E	ENV
D3D4	ENT
D456	INKEY
D470	JOY
D486	KEY DEF
D4DB	SPEED
D51D	PI
D529	DEG
D52D	RAD
D531	SQR
D536	Rutina para elevar a potencias
D560	EXP
D565	LOG10
D56A	LOG
D57F	SIN
D574	COS
D579	TAN
D57E	ATN
D584	Mensaje *RANDOM NUMBER SEED?*
D599	RANDOMIZE
D5C1	RND
D650	DEFSTR
D654	DEFINT
D658	DEFREAL
D68E	LET
D6B6	DIM
D9F0	ERASE
DB13	LINE
DB43	INPUT
DB7A	*? redo from start* (mensaje)
DCC8	RESTORE
DCDA	READ
DEC1	TRON

DEC5	TROFF
DEE0	Tabla de puntos de entrada para las claves en BASIC
DFA3	Fin de la tabla
E0C3	Tabla de claves que deben ir precedidas de un número de línea (GOTO, RESTORE, AUTO, EDIT, etc.)
E1CD	LIST
E3A8	Rutina para posicionar la tabla de caracteres durante la búsqueda de una clave
E3EB	Busca una clave en la tabla
E418	Tabla de direcciones para cada una de las 26 letras del alfabeto
E44C	Tabla de claves con sus respectivos códigos
E735	Fin de la tabla
E7EE	DELETE
E89E	RENUM
E9A3	DATA
E9A7	REM
E9AD	ELSE
EA78	RUN
EAB5	LOAD
EAFD	CHAIN
EB54	MERGE
ECDC	SAVE
F208	PEEK
F20F	POKE
F219	INP
F223	OUT
F229	WAIT
F25C	CALL
F29D	ZONE
F2A9	PRINT
F383	PRINT USING
F508	WRITE
F56B	MEMORY

F784	SYMBOL
F8EC	LOWER$
F8F1	Rutina de conversión en minúsculas
F8FA	UPPER$
F964	BIN$
F969	HEX$
F98F	DEC$
F9BC	STR$
F9D3	LEFT$
F9D8	RIGHT$
FA07	MID$
FA69	LEN
FA6E	ASC
FA74	CHR$
FA7E	INKEY$
FA8D	STRING$
FAAD	SPACE$
FABE	VAL
FAE5	INSTR
FC53	FRE
FD0C	+
FD21	—
FD35	* (multiplicación)
FD52	/
FD67	División entera
FD79	MODULO (resto de una división)
FD87	Función AND (Y lógico)
FD92	Función OR (O lógico)
FD9C	Función XOR (O exclusivo)
FDB0	ABS
FE0E	FIX
FE13	INT
FEB6	CINT
FEEB	UNT
FF14	CREAL
FF1B	Acumulador a cero
FF2A	SGN
FF32	Coloca un entero en el acumulador

FF3E	Convierte un entero en real
FF45	Carga el tipo de variable en C
FF4B	Carga el tipo de variable en A
FF83	Copia el acumulador en el área apuntada por DE
FF92	Busca mayúsculas
FF9C	Busca números
FFAB	Convierte en mayúsculas
FFCA	Compara A con el contenido de HL
FFD8	Compara HL con DE
FFDE	Compara HL con BC
FFE4	DE = HL—DE
FFF2	LDIR
FFF8	LDDR
FFFB	JP(HL)
FFFC	Regresa a la dirección apuntada por BC
FFFE	Regresa a la dirección apuntada por DE

Principales direcciones de la ROM superior del CPC 464

La ROM superior contiene todas las rutinas de procesamiento de claves en BASIC.

C006	Inicialización y salida de BASIC 1.0 (mensaje).
C03F	BASIC 1.0 (mensaje)
C053	Función EDIT
C090	Entrada principal (aparece *Ready*)
C0CC	READY (mensaje)
C0DF	AUTO
C12B	NEW
C132	CLEAR
C20A	PAPER
C212	PEN

C221	BORDER
C22A	INK
C24F	MODE
C25A	CLS
C262	VPOS
C276	POS
C2D2	LOCATE
C2E1	WINDOW
C319	TAG
C320	TAGOFF
C337	Escribe el mensaje apuntado por HL
C3E3	WIDTH
C417	EOF
C48C	ORIGIN
C4B5	CLG
C4C6	DRAW
C4CB	DRAWR
C4D0	PLOT
C4D5	PLOTR
C4E9	TEST
C4EE	TESTR
C505	MOVE
C50A	MOVER
C529	FOR
C5FB	NEXT
C6C7	IF
C6E8	GOTO
C6ED	GOSUB
C70F	RETURN
C747	WHILE
C776	WEND
C7C3	ON
C8CB	ON BREAK
C8E1	DI
C8E7	EI
C940	ON SQ
C971	AFTER
C979	EVERY

C99F	REMAIN
CA8F	ERROR
CB23	*Undefined line* (mensaje)
CB33	Manda el mensaje *Break in*
CB4F	*Break* (mensaje)
CB55	IN (mensaje)
CB5A	STOP
CB65	END
CBC0	CONT
CBF8	ON ERROR
CC03	RESUME
CC5B	Tabla de mensajes de error (parte de la palabra)
CE66	Final de la tabla
CF81	Tabla de puntos de entrada para operaciones lógicas y aritméticas
D0CA	Tabla de puntos de entrada para las funciones EOF, ERR, HIMEM, INKEY$, PI, RND, TIME, XPOS y YPOS
D0DC	ERR
D0F4	HIMEM
D107	XPOS
D10E	YPOS
D190	Tabla de puntos de entrada para funciones
D219	ROUND
D1EA	MIN
D1EE	MAX
D256	OPENOUT
D25F	OPENIN
D298	CLOSEIN
D2A1	CLOSEOUT
D2C0	SOUND
D31E	RELEASE
D329	SQ
D34E	ENV
D385	ENT

D409	INKEY
D423	JOY
D439	KEY DEF
D494	SPEED
D4DB	PI
D4E7	DEG
D4EB	RAD
D4EF	SQR
D4F4	Rutina para elevar a potencias
D520	EXP
D525	LOG10
D52A	LOG
D52F	SIN
D534	COS
D539	TAN
D53E	ATN
D543	Mensaje *RANDOM NUMBER SEED?*
D559	RANDOMIZE
D584	RND
D614	DEFSTR
D618	DEFINT
D61C	DEFREAL
D654	LET
D67D	DIM
D9C0	ERASE
DAF8	LINE
DB28	INPUT
DB77	*? redo from start* (mensaje)
DCD9	RESTORE
DCEB	READ
DDED	TRON
DDEF	TROFF
DE01	Tabla de puntos de entrada para las claves en BASIC
DEBA	Fin de la tabla
DFDC	Tabla de claves que deben ir precedidas de un número de línea (GOTO, RESTORE, AUTO, EDIT, etc.)

E0F7	LIST
E2DD	Rutina para posicionar la tabla de caracteres durante la búsqueda de una clave.
E327	Busca una clave en la tabla
E354	Tabla de direcciones para cada una de las 26 letras del alfabeto
E388	Tabla de claves con sus respectivos códigos
E64A	Fin de la tabla
E728	DELETE
E7DF	RENUM
E8EF	DATA
E8F3	REM
E9BD	RUN
E9F6	LOAD
EA3C	CHAIN
EAA6	MERGE
EC09	SAVE
F158	PEEK
F15F	POKE
F16D	INP
F177	OUT
F17D	WAIT
F1BA	CALL
F1F7	ZONE
F1FD	PRINT
F2C4	PRINT USING
F47B	WRITE
F4EF	MEMORY
F69D	SYMBOL
F834	LOWER$
F839	Rutina de conversión en minúsculas
F842	UPPER$
F8BA	BIN$
F8C4	HEX$
F8EA	DEC$
F91E	STR$

F930	LEFT$
F943	RIGHT$
F993	MID$
FA0A	LEN
FA10	ASC
FA16	CHR$
FA24	INKEY$
FA36	STRING$
FA57	SPACE$
FA77	VAL
FAA1	INSTR
FC2D	FRE
FCCC	+
FCE1	—
FCF5	* (multiplicación)
FD12	/
FD37	División entera
FD49	MODULO (resto de una división)
FD58	Función AND (Y lógico)
FD63	Función OR (O lógico)
FD6D	Función XOR (O exclusivo)
FD85	ABS
FDE8	FIX
FDED	INT
FE8D	CINT
FEC2	UNT
FEEC	CREAL
FEF3	Acumulador a cero
FF02	SGN
FF0A	Coloca un entero en el acumulador
FF16	Convierte un entero en real
FF1D	Carga el tipo de variable en C
FF23	Carga el tipo de variable en A
FF27	Comprueba si el acumulador contiene el puntero de una cadena
FF62	Copia el acumulador en el área apuntada por DE
FF71	Busca mayúsculas

FF7B	Busca números
FF8A	Convierte en mayúsculas
FFAA	Compara A con el contenido de HL
FFB8	Compara HL con DE
FFBE	Compara HL con BC
FFC4	DE=HL—DE
FFCF	HL=HL—DE
FFDA	BC=HL—DE
FFE7	HL=HL—BC
FFF2	LDIR
FFF5	LDDR
FFF8	JP(HL)
FFF9	Regresa a la dirección apuntada por BC
FFFB	Regresa a la dirección apuntada por DE

Principales direcciones de la ROM superior del CPC 664:

La ROM superior contiene todas las rutinas de procesamiento de claves en BASIC.

C006	Inicialización y salida de BASIC 1.1 (mensaje).
C033	BASIC 1.1 (mensaje)
C046	Función EDIT
C058	Entrada principal (aparece *Ready)*
C0D7	*Ready* (mensaje)
C0EA	AUTO
C128	NEW
C12F	CLEAR
C23C	PAPER
C227	PEN
C24B	BORDER
C254	INK
C278	MODE

C283	CLS
C29B	COPYCHR$
C2A4	VPOS
C2A8	POS
C302	LOCATE
C311	WINDOW
C346	TAG
C34D	TAGOFF
C363	CURSOR
C42D	WIDTH
C452	EOF
C4E1	ORIGIN
C509	CLG
C515	FILL
C532	MOVE
C537	MOVER
C53C	DRAW
C541	DRAWR
C546	PLOT
C54B	PLOTR
C574	TEST
C579	TESTR
C59D	GRAPHICS
C5C3	MASK
C5D7	FOR
C6A5	NEXT
C7A6	IF
C789	GOTO
C78F	GOSUB
C7B3	RETURN
C7EA	WHILE
C81D	WEND
C885	ON
C979	ON BREAK
C99A	DI
C9A0	EI
C9F8	ON SQ
CA25	AFTER

CA2D	EVERY
CA53	REMAIN
CB54	ERROR
CB74	*Undefined line* (mensaje)
CC04	Manda el mensaje *Break in*
CC1F	*Break* (mensaje)
CC25	IN (mensaje)
CC29	STOP
CC34	END
CC96	CONT
CCCD	ON ERROR
CCD8	RESUME
CD17	Tabla de mensajes de error (parte de la palabra)
CFF0	Tabla de puntos de entrada para operaciones lógicas y aritméticas
D11A	Tabla de puntos de entrada para las funciones EOF, ERR, HIMEM, INKEY$, PI, RND, TIME, XPOS y YPOS
D12E	DERR
D133	ERR
D14B	HIMEM
D164	XPOS
D16B	YPOS
D1E8	Tabla de puntos de entrada para funciones
D242	MIN
D246	MAX
D26D	ROUND
D2AB	OPENOUT
D2B7	OPENIN
D2F0	CLOSEIN
D2F8	CLOSEOUT
D316	SOUND
D373	RELEASE
D37E	SQ
D3A1	ENV

D3D7	ENT
D459	INKEY
D473	JOY
D489	KEY DEF
D4DE	SPEE D
D520	PI
D52C	DEG
D530	RAD
D534	SQR
D539	Rutina para elevar a potencias
D563	EXP
D568	LOG10
D56D	LOG
D572	SIN
D577	COS
D57C	TAN
D581	ATN
D587	Mensaje *RANDOM NUMBER SEED?*
D59C	RANDOMIZE
D5C4	RND
D653	DEFSTR
D657	DEFINT
D65B	DEFREAL
D691	LET
D6B9	DIM
D9F4	ERASE
DB18	LINE
DB48	INPUT
DB7F	*? redo from start* (mensaje)
DCCD	RESTORE
DCDF	READ
DEC6	TRON
DECA	TROFF
DEE5	Tabla de puntos de entrada para las claves en BASIC
DFA8	Fin de la tabla

E0C8	Tabla de claves que deben ir precedidas de un número de línea (GOTO, RESTORE, AUTO, EDIT, etc)
E1D2	LIST
E3AD	Rutina para posicionar la tabla de caracteres durante la búsqueda de una clave
E3F0	Busca una clave en la tabla
E41D	Tabla de direcciones para cada una de las 26 letras del alfabeto
E451	Tabla de claves con sus respectivos códigos
E73A	Fin de la tabla
ES7F3	DELETE
E8A3	RENUM
E9A8	DATA
E9AC	REM
EA7D	RUN
EABA	LOAD
EB02	CHAIN
EB59	MERGE
ECE1	SAVE
F20D	PEEK
F214	POKE
F21E	INP
F228	OUT
F232	WAIT
F261	CALL
F2A2	ZONE
F2A9	PRINT
F383	PRINT USING
F50D	WRITE
F570	MEMORY
F784	SYMBOL
F8EC	LOWER$
F8F1	Rutina de conversión en minúsculas
F8FA	UPPER$
F964	BIN$

F969	HEX$
F98F	DEC$
F9BC	STR$
F9D3	LEFT$
F9D8	RIGHT$
FA07	MID$
FA69	LEN
FA6E	ASC
FA74	CHR$
FA7E	INKEY$
FA8D	STRING$
FAAD	SPACE$
FABE	VAL
FAE5	INSTR
FC53	FRE
FD0C	+
FD21	—
FD35	* (multiplicación)
FD52	/
FD67	División entera
FD79	MODULO (resto de una división)
FD87	Función AND (Y lógico)
FD92	Función OR (O lógico)
FD9C	Función XOR (O exclusivo)
FDB0	ABS
FE0E	FIX
FE13	INT
FEB6	CINT
FEEB	UNT
FF14	CREAL
FF1B	Acumulador a cero
FF2A	SGN
FF32	Coloca un entero en el acumulador
FF3E	Convierte un entero en real
FF45	Carga el tipo de variable en C
FF4B	Carga el tipo de variable en A
FF83	Copia el acumulador en el área apuntada por DE

FF92	Busca mayúsculas
FF9C	Busca números
FFAB	Convierte en mayúsculas
FFCA	Compara A con el contenido de HL
FFD8	Compara HL con DE
FFDE	Compara HL con BC
FFE4	DE=HL—DE
FFF2	LDIR
FFF8	LDDR
FFFB	JP(HL)
FFFC	Regresa a la dirección apuntada por BC
FFFE	Regresa a la dirección apuntada por DE

6

Código máquina desde BASIC

La inserción del código máquina Z-80, variables o bloques de datos desde BASIC, se efectúa con la función POKE, de la forma:

```
POKE dirección,dato
```

donde "dirección" puede ser cualquier posición de la memoria; "dato" es un número comprendido entre 0 y 255 (&0 - &FF). Los valores mayores de 255 ocupan 2 bytes de memoria, que se almacenarán con el formato de Z-80; es decir, primero el byte bajo.

Por ejemplo, almacenar el valor 10000 en la posición &A000 sería:

```
POKE &A000,10000—(10000/256)*256 (byte bajo)
POKE &A001,10000/256 (byte alto)
```

El acceso a los programas en código máquina lo realizamos con la instrucción CALL. El formato de esta instrucción es:

```
CALL dirección,(lista opcional de parámetros)
```

donde "dirección" indica el punto de comienzo del

programa. Los parámetros de la lista opcional son datos o cadenas literales que se pueden transferir en dos sentidos. Esto es, desde BASIC a código máquina, y viceversa. Veamos las condiciones para cada uno de los sentidos.

Transferencia de datos de BASIC a código máquina

Los datos pueden ser de tres tipos: números enteros, variables enteras y variables literales. Las variables siempre deben estar previamente inicializadas.

Al ejecutar la instrucción CALL, el registro A queda cargado con el número de parámetros transferidos, y el registro IX apunta a una dirección donde están almacenados los parámetros transferidos.

Si los parámetros son constantes numéricas, el registro IX apuntará directamente al byte bajo de la última constante transferida. Así:

```
CALL dirección,100,0,32767
```

Cada constante ocupará dos bytes del bloque de parámetros, aun siendo menor que 256. Los números negativos se transferirán en complemento a dos. Los valores admisibles deben pertenecer al intervalo -32768 a 32767.

En el caso de variables numéricas, éstas han de ser enteras y cumplir los mismos requisitos que las constantes numéricas. El registro IX apuntará a un bloque de parámetros formado por los valores de las variables transferidas. Por ejemplo:

```
A%=6666:ZD%=0: CALL dirección,A%,ZD%
```

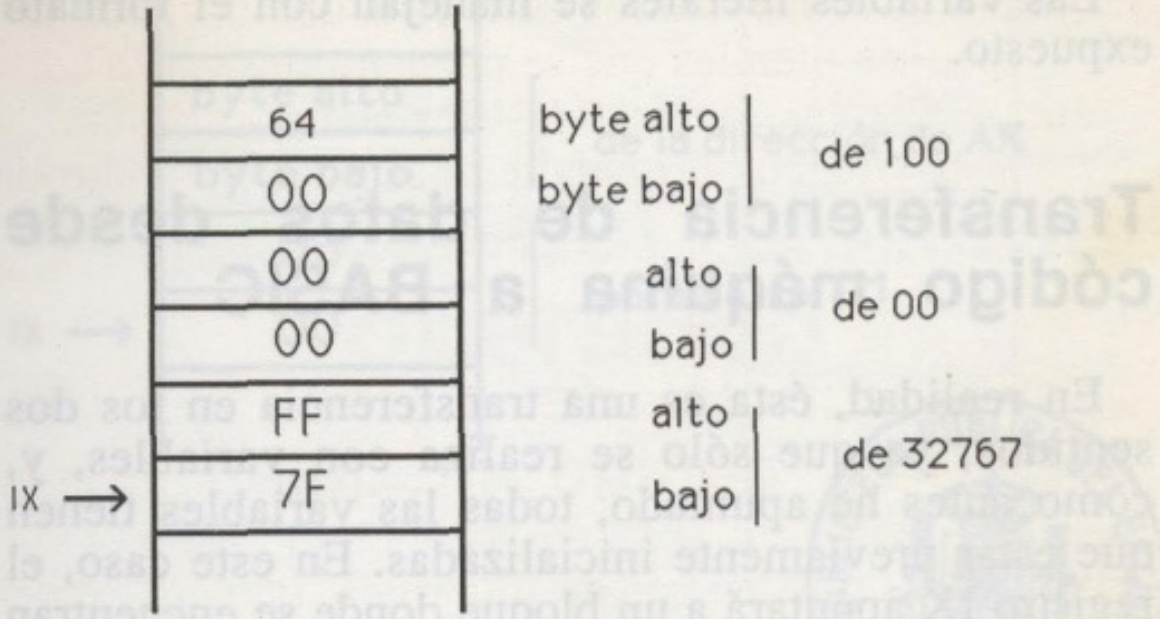

Figura 6.1

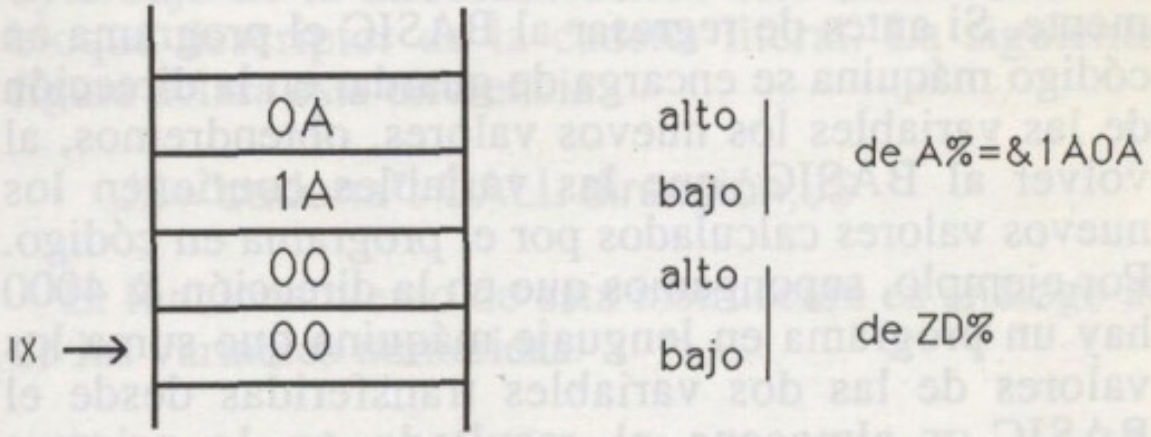

Figura 6.2

Las variables literales se manejan con el formato expuesto.

Transferencia de datos desde código máquina a BASIC

En realidad, ésta es una transferencia en los dos sentidos, ya que sólo se realiza con variables, y, como antes he apuntado, todas las variables tienen que estar previamente inicializadas. En este caso, el registro IX apuntará a un bloque donde se encuentran las direcciones de las variables. Esto es válido al menos para variables numéricas. Veremos más adelante la diferencia que existe con las variables literales.

La ventaja de este tipo de transferencia está en que podemos recuperar valores o cadenas de caracteres que han sido calculados o formados por el programa en código máquina. Para distinguir a este tipo de parámetros, se les precede con el símbolo "@". Al ejecutar la instrucción CALL, el registro IX apuntará a un bloque de parámetros como los descritos anteriormente. Si antes de regresar al BASIC el programa en código máquina se encarga de guardar en la dirección de las variables los nuevos valores, obtendremos, al volver al BASIC, que las variables contienen los nuevos valores calculados por el programa en código. Por ejemplo, supongamos que en la dirección & 4000 hay un programa en lenguaje máquina que suma los valores de las dos variables transferidas desde el BASIC y almacena el resultado en la primera variable. En este supuesto, la instrucción

```
M=5: A%=10: CALL &4000,@A%,M
```

hará que al regreso la variable A% contenga el valor

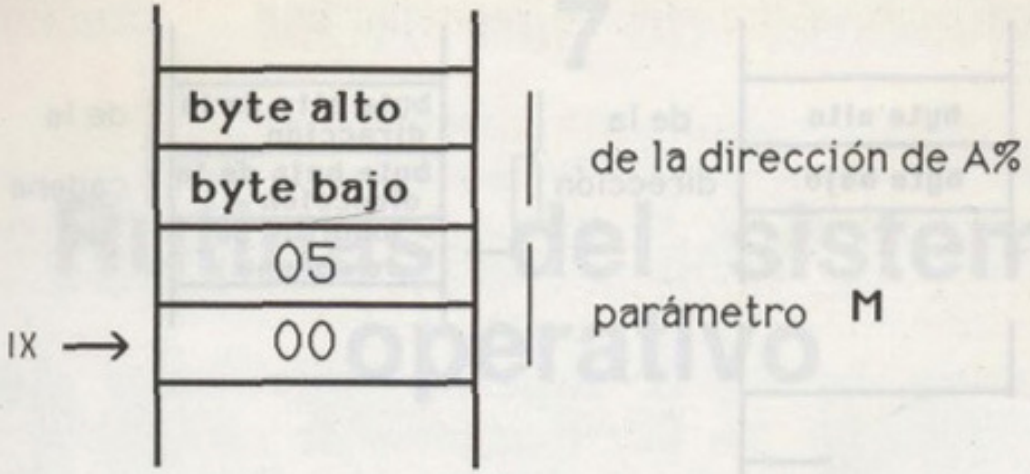

Figura 6.3

A%=15. Con este tipo de parámetros los bloques quedarán dispuestos en la memoria como indica la figura 6.3. Las variables literales se transfieren de forma diferente. En este caso, el registro IX apunta al byte bajo de la dirección donde está almacenado un bloque descriptor de la cadena literal. La siguiente figura aclara esta diferencia:

```
J$="cadena": CALL dirección,J$
```

El funcionamiento de esta instrucción es análogo al de las variables numéricas.

Finalmente, hay que destacar que es posible combinar todos los tipos de parámetros descritos en una misma instrucción CALL. En cuanto al número de ellos, el límite viene impuesto por la longitud de las líneas en BASIC. El registro IX siempre apuntará al byte bajo del último de los parámetros de la lista.

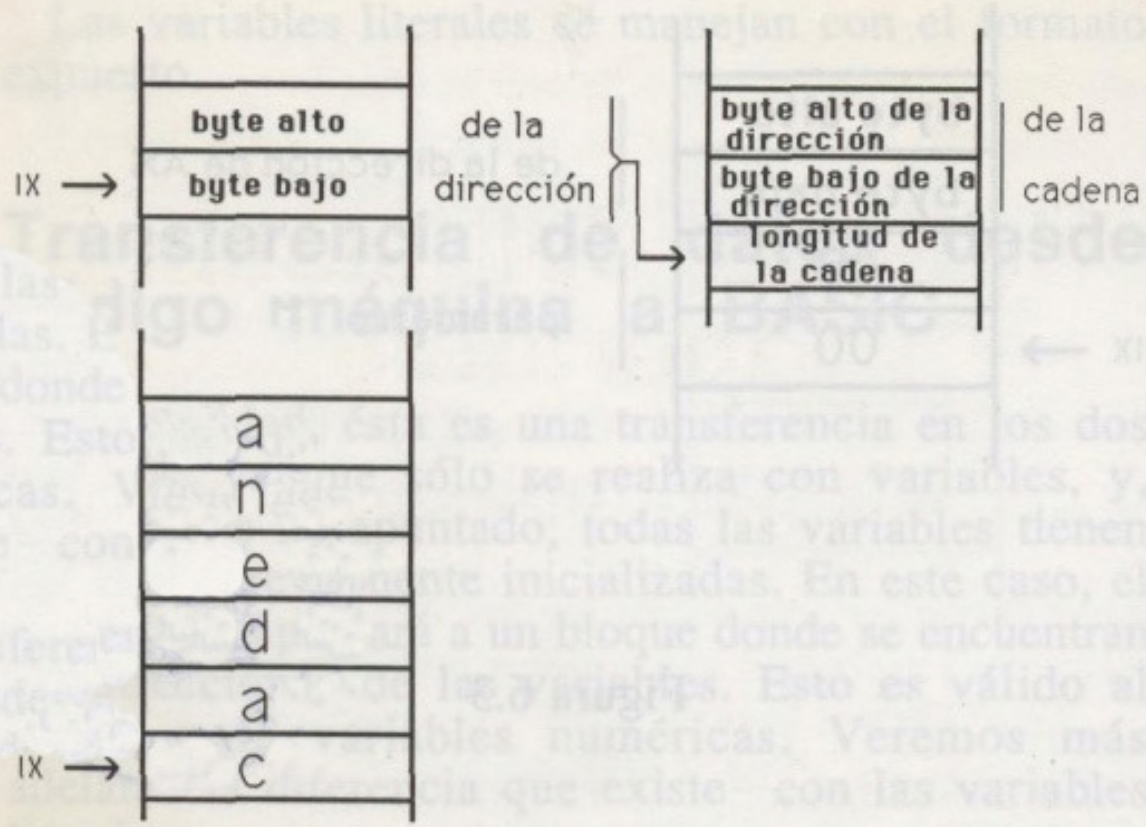
byte alto
IX →
byte bajo
de la
dirección
byte alto de la dirección
byte bajo de la dirección
longitud de la cadena
de la
cadena
a
n
e
d
a
IX →
c

Figura 6.4

7

Rutinas del sistema operativo

Una de las más poderosas ayudas que ofrece el ordenador es la posibilidad de acceder a las muchas rutinas implementadas en el sistema operativo. En la mayoría de los casos será necesario acudir a estas llamadas al sistema operativo desde el código máquina; esto es, desde una pequeña rutina en lenguaje máquina que inicialice adecuadamente el contenido de los registros del procesador Z-80.

Al final del capítulo aparece el listado de un programa en BASIC con el que se puede comprobar el funcionamiento de cada una de las rutinas explicadas.

El formato de la explicación de cada rutina es el siguiente:

Primero aparece la dirección en RAM de la entrada desde donde el ordenador efectuará el salto a la rutina requerida y contenida en la ROM; después se indica la dirección en ROM donde se encuentra dicha rutina en cada uno de los tres sistemas CPC 464, 664 y 6128, respectivamente. En negrita aparece la acción que realiza esta rutina, y finalmente se indican los requisitos de entrada y condiciones de salida de cada uno de los registros del procesador. Todas las direcciones que se indican están expresadas en hexadecimal.

Activación de las ROM

Común	CPC464	CPC664	CPC6128
B900	BA5E	BA5F	BA5F

Activa la ROM superior que haya sido seleccionada. No hay requisitos de entrada. A la salida, el registro A contiene el byte de estado de la ROM anterior; los demás registros inalterados.

B903	BA68	BA66	BA66

Desactiva la ROM seleccionada previamente. No hay requisitos de entrada. A la salida, el registro A contiene el estado anterior de la ROM; los demás registros inalterados.

B906	BA4A	BA51	BA51

Activa la ROM inferior. No hay requisitos de entrada. A la salida, el registro A contiene el estado anterior de la ROM; los demás registros quedan inalterados.

B909	BA54	BA58	BA58

Desactiva la ROM inferior. No hay requisitos de entrada. A la salida, el registro A contiene el estado anterior de la ROM; los demás registros inalterados.

B90C BA72 BA70 BA70

Restaura la ROM en su estado anterior. A la entrada, A debe contener el estado previo de ROM, que pasará a ser el actual. A la salida, aparece alterado el registro A.

B90F BA7E BA79 BA79

Selecciona y activa una ROM superior. A la entrada, C contiene el número de la ROM requerida; a la salida, C contendrá el número de la ROM anterior y B el byte de estado de la misma. A es alterado.

B912 BAA2 BA7D BA7D

Devuelve el número de la ROM actual. Sin requisitos de entrada. A la salida, A contiene el número de la ROM actualmente seleccionada.

B915 BAA2 BA7E BA7E

Devuelve el tipo y versión de una ROM. A la entrada, C contiene el número de la ROM a comprobar. A la salida, H contiene la versión, L el número de marca y A el tipo de ROM. B es alterado.

B918 BA8C BA87 BA87

Selecciona la ROM anterior. A la entrada, C contiene el número de la ROM anterior y B su byte de estado. A la salida, C contendrá el número de la ROM actual. B es alterado.

B91B	BAA6	BAA1	BAA1

Realiza la operación LDIR con las ROM inhibidas. A la entrada y salida, todos los registros deben estar igual que para dicha instrucción. Durante la ejecución, las ROM se desactivarán y después regresarán a su estado previo.

B91E	BAAC	BAA7	BAA7

Realiza la operación LDDR con las ROM inhibidas. A la entrada y salida, todos los registros deben estar igual que para dicha intrucción.

B921	B921	B921	B921

Comprueba si hay algún suceso con mayor prioridad de la definida. Sin requisitos de entrada. A la salida, el acarreo estará a uno si lo hay, o a cero en caso contrario.

Funciones sobre el teclado

BB00	19E0	1B5C	1B5C

Inicialización del sistema de teclado. Sin requisitos de entrada. Alterados todos los registros.

BB03	1A1E	1B98	1B98

Reinicializa las indirecciones y *buffers* del teclado. Sin requisitos de entrada. Alterados todos los registros.

BB06 1A42 1BBF 1BBF

Espera el siguiente código del teclado. Sin requisitos de entrada. A la salida, acarreo a uno y A contiene el carácter.

BB09 1A3C 1BC5 1BC5

Intenta tomar un carácter del *buffer* del teclado. Sin requisitos de entrada. A la salida, si había algún carácter disponible, acarreo a uno y A contiene el carácter. En caso contrario, acarreo a cero y A aparecerá alterado.

BB0C 1A77 1BFA 1BFA

Deposita un carácter en el *buffer* del teclado. A la entrada, A contiene el carácter. Ningún registro se altera.

BB0F 1ABD 1C46 1C46

Fija la cadena de expansión asociada con una clave de expansión. A la entrada, B contiene la clave de expansión; C la longitud de la cadena, y HL la dirección de la cadena. A la salida, el acarreo estará a cero si la cadena era demasiado larga o si la clave era inválida. En el caso correcto, el acarreo estará a cero. Todos los registros se alteran.

BB12 1B2E 1CB3 1CB3

Lee un carácter de una cadena de expansión. Los caracteres se numeran desde cero. A la entrada,

A contiene la clave de expansión y L el número de carácter. A la salida, acarreo a uno si se encontró el carácter que estará en A; en caso contrario, acarreo a cero. Todos los registros son alterados.

BB15 1A7B 1C04 1C04

Define un *buffer* para las cadenas de expansión. A la entrada, DE contiene la dirección del *buffer* y HL su longitud. A la salida, acarreo a uno si el *buffer* es correcto. Todos los registros son alterados.

BB18 1B56 1CDB 1CDB

Espera hasta que haya algún código en el *buffer* del teclado. A la salida, acarreo a uno y A contiene el carácter o clave de expansión.

BB1B 1B5C 1CE1 1CE1

Intenta tomar un código del *buffer* del teclado. Si hay alguno disponible a la salida, acarreo a uno y A contiene el código. En el caso contrario, acarreo a cero.

BB1E 1CBD 1E45 1E45

Comprueba si una tecla está pulsada. A la entrada, A contiene el número de la tecla. A la salida, C contiene el estado de MAYS (SHIFT) y de CONTROL (CTRL); si la tecla está pulsada, el *flag* Z estará a cero, y si no, a uno. Los registros A y HL son alterados.

BB21	1BB3	1D38	1D38

Comprueba el estado de FIJA MAYS (CAPS LOCK) y de CONTROL+FIJA MAYS (CTRL+ CAPS LOCK). A la salida, L contiene el estado de CONTROL+FIJA MAYS y H el estado de FIJA MAYS. El contenido &FF indica activación, mientras que &00 indica desactivación. A es alterado.

BB24	1C5C	1DE5	1DE5

Comprueba el estado del *joystick*. A la salida, los registros A y H contienen el estado del *joystick* 0, y L el estado del 1 (véanse apéndices).

BB27	1052	1ED8	1ED8

Define el código generado por una tecla aislada. A la entrada, A contiene el número de tecla y B el nuevo código o clave. HL y A son alterados.

Las claves con uso especial son:

&80-&9F	Claves de expansión.
&E0-&FC	Movimiento del cursor, edición y rupturas.
&FD	FIJA MAYS (CAPS LOCK), modo *toggle* o alternado.
&FE	CONTROL+FIJA MAYS, modo alternado.
&FF	Se ignora.

BB2A	1D3E	1EC4	1EC4

Obtiene el código generado por una tecla aislada. A la entrada, A contiene el número de tecla

y, a la salida, contendrá la interpretación o código. HL es alterado.

BB2D 1D57 1EDD 1EDD

Define el código generado por una tecla pulsada junto a MAYS (SHIFT). A la entrada, A contiene el número de tecla y B el nuevo código o clave. HL y A son alterados.

BB30 1D43 1EC9 1EC9

Obtiene el código generado por una tecla pulsada junto a MAYS (SHIFT). A la entrada, A contiene el número de tecla y, a la salida, contendrá la interpretación o código. HL es alterado.

BB33 1D5C 1EE2 1EE2

Define el código generado por una tecla pulsada junto a CONTROL (CTRL). A la entrada, A contiene el número de tecla y B el nuevo significado. Tanto el registro HL como el A son alterados.

BB36 1D48 1ECE 1ECE

Obtiene el código generado por una tecla pulsada junto a CONTROL (CTRL). A la entrada, A contiene el número de tecla y, a la salida, contendrá el significado. Sólo el registro HL es alterado.

BB39 1CAB 1E34 1E34

Determina si a una tecla se le permite la repetición. A la entrada, A contiene el número de tecla; B contiene &FF, si se permite la repetición, o &00 si no se permite. A, HL y BC son alterados.

BB3C 1CA6 1E2F 1E2F

Comprueba si se permite la repetición a una tecla. A la entrada, A contiene el número de la tecla. A la salida, el *flag* Z estará a uno si no se permite la repetición, y a cero si se permite. A y HL son alterados.

BB3F 1C6D 1DF6 1DF6

Asigna el retardo de comienzo y la velocidad de repetición. A la entrada, H contiene el retardo de comienzo y L el período de repetición. Tanto uno como otro se miden en unidades de 0.04 segundos. A es alterado.

BB42 1C69 1DF2 1DF2

Obtiene el retardo de comienzo y la velocidad de repetición. A la salida, H contiene el retardo de comienzo y L el período de repetición. Se miden en unidades de 0.04 segundos. A es alterado.

BB45 1C71 1DFA 1DFA

Permite que se generen sucesos de rupturas. A la entrada, DE contiene la dirección de la rutina

asociada al suceso de ruptura y C el número de ROM de esa rutina. Todos los registros son alterados.

BB48	1C82	1E0B	1E08

Impide que se generen sucesos de rupturas. No hay requisitos. A y HL son alterados.

BB4B	1C90	1E19	1E19

Genera un suceso de ruptura (si está permitido). No hay requisitos. A y HL son alterados.

Funciones sobre la unidad de texto

BB4E	1078	1070	1074

Inicialización de la unidad de vídeo para texto. No hay requisitos. Todos los registros son alterados.

BB51	1088	1080	1084

Reinicializa las indirecciones y la tabla de códigos de control. No hay requisitos. Todos los registros son alterados.

BB54 1415 1455 1459

Permite mostrar caracteres en la pantalla. No hay requisitos. A es alterado.

BB57 144B 144E 1452

No permite mostrar caracteres en la pantalla. No hay requisitos. A es alterado.

BB5A 1400 13FA 13FE

Envía un carácter o código de control a la unidad de vídeo para texto. A la entrada, A contiene el código a enviar.

BB5D 1334 1331 1335

Escribe un carácter. A la entrada, A contiene el código del carácter. Todos los registros son alterados.

BB60 13AB 13A8 13AC

Lee un carácter de la pantalla. H contiene la columna física y L la fila física. Si identificó algún carácter, el acarreo estará a uno y A contendrá su código. En caso contrario, el acarreo estará a cero. Los demás registros son preservados.

BB63 137A 13A4 13A8

Habilita o deshabilita la opción de caracteres gráficos. A la entrada, A = 0 para activar la opción

gráficos y A distinto de cero para desactivarla. A la salida, A es alterado.

BB66	120C	1204	1208

Determina el tamaño de la ventana de texto. A la entrada, H contiene la columna extrema izquierda de la ventana, L la fila superior de la ventana y E la fila inferior. Todos los registros son alterados.

BB69	1256	124E	1252

Obtiene el tamaño de la ventana. No hay requisitos de entrada. A la salida, H contiene la columna extrema izquierda de la ventana, D la columna extrema derecha, L la fila superior y E la fila inferior. El acarreo regresará a cero o a uno según la ventana se ocupe toda o no. A es alterado.

BB6C	1540	154B	154F

Borra la ventana y la rellena con la tinta del papel. No hay requisitos. Todos los registros son alterados.

BB6F	115E	1156	115A

Mueve el cursor a la columna requerida. A la entrada, A contiene el número de columna lógica requerida. A y HL son alterados.

BB72 1174 1161 1165

Mueve el cursor a la fila requerida. A la entrada, A contiene la fila lógica requerida. A y HL son alterados.

BB75 1174 116C 1170

Fija la posición del cursor. A la entrada, H contiene la columna lógica y L la fila lógica. A y HL son alterados.

BB78 1180 1178 1176

Devuelve la posición del cursor y el número de desplazamientos de la ventana. No hay requisitos. A la salida, H contiene la columna lógica del cursor; L la fila lógica, y A la cuenta actual de desplazamientos. DE y BC son preservados.

BB7B 1289 1282 1286

Habilita el cursor para el cauce actual. No hay requisitos. A es alterado.

BB7E 129A 1293 1297

Deshabilita el cursor para el cauce actual. No hay requisitos. A es alterado.

BB81 1279 1272 1276

Muestra el cursor en la pantalla. No hay requisitos. Todos los registros son preservados.

BB84 1281 127A 127E

Impide mostrar el cursor en la pantalla. No hay requisitos. Todos los registros son preservados.

BB87 11CE 11C6 11CA

Comprueba si la posición del cursor queda dentro de los límites de la ventana. A la entrada, H contiene la columna lógica de la posición a comprobar, y L la fila lógica. A la salida, H contiene la columna lógica donde se imprimiría el carácter, y L la fila lógica. Si al escribir el carácter la ventana no se desplaza, el acarreo estará a uno; si se desplaza hacia arriba, estará a cero y B contendrá &FF; si se desplaza hacia abajo, el acarreo estará a cero y B contendrá &00. A es alterado.

BB8A 1268 1261 1265

Pone el cursor en la pantalla. No hay requisitos de entrada. A es alterado.

BB8D 1268 1261 1265

Quita el cursor de la pantalla. No hay requisitos de entrada. A es alterado.

BB90 12A9 12A2 12A6

Define la tinta de escritura. A la entrada, A debe contener el número de la tinta. A y HL son alterados.

BB93 12BD 12B6 12BA

Comprueba la tinta de escritura. No hay requisitos. A la salida, A contiene el número de la tinta.

BB96 12AE 12A7 12AB

Define la tinta del papel. A la entrada, A contiene el número de la tinta. A y HL son alterados.

BB99 12C3 12BC 12C0

Comprueba la tinta del papel. No hay requisitos. A la salida, A contiene el número de la tinta.

BB9C 12C9 12C2 12C6

Intercambia las tintas de la pluma y del papel. No hay requisitos. A y HL son alterados.

BB9F 137A 1377 137B

Fija el modo opaco o transparente. A la entrada, A = 0 para el modo opaco y A<>0 para el transparente. A y HL son alterados.

BBA2 1387 1384 1388

Comprueba el modo de escritura. No hay requisitos. A la salida, A = 0 para el modo opaco y A<>0 para el modo transparente. DE y HL son alterados.

BBA5 12D3 12D0 12D4

Obtiene la dirección del modelo de un carácter. A la entrada, A contiene el código del carácter que ha de encontrarse. A la salida, el acarreo estará a uno si la matriz está en la tabla definida por el usuario, o a cero si pertenece a las definiciones de la ROM baja. H contiene la dirección de la matriz. A es alterado.

BBA8 12F1 12EE 12F2

Define una matriz de carácter. A la entrada, A contiene el código asignado al carácter, y HL la matriz que se utilizará para definir el modelo. A la salida, el acarreo estará a uno si el carácter es definible por el usuario o a cero si no lo es. Todos los registros son alterados.

BBAB 12FD 12FA 12FE

Define una tabla de matrices de caracteres. A la entrada, DE contiene el primer carácter de la tabla y HL la dirección de comienzo de la nueva tabla. Condiciones de salida: si no había otra tabla de usuario definida el acarreo estará a cero; si antes había alguna, el acarreo estará a uno y, en este caso, A contendrá el primer carácter de la tabla antigua, y HL la dirección de la tabla antigua. BC y DE son alterados.

BBAE 132A 1327 132B

Comprueba la dirección de la tabla de matrices definidas por el usuario y el primer carácter de la tabla. No hay requisitos. A la salida,

si no hay ninguna tabla definida por el usuario, el acarreo estará a cero; si hay alguna, el acarreo estará a uno; A contendrá el código del primer carácter de la tabla, y HL la dirección de comienzo de la tabla. Los demás registros son preservados.

BBB1 14CB 14D0 14D4

Obtiene la dirección de la tabla de códigos de control. A la salida, HL contiene la dirección de la tabla.

BBB4 10E8 10E0 10E4

Selecciona un cauce. A la entrada, A contiene el número del cauce a seleccionar. HL es alterado.

BBB7 1107 10FF 1103

Intercambia dos cauces. A la entrada, C y B contienen los cauces a cambiar. Todos los registros son alterados.

Funciones sobre la unidad de gráficos

BBBA 15B0 15A4 15A8

Inicializa la unidad de vídeo para gráficos. No hay requisitos. Todos los registros son alterados.

BBBD	15DF	15D3	15D7

Reinicializa la unidad de vídeo para gráficos. No hay requisitos. Todos los registros son alterados.

BBC0	15F4	15FA	15FE

Mueve el cursor a una posición en coordenadas absolutas. A la entrada, DE contiene la coordenada X y HL la coordenada Y. Todos los registros son alterados.

BBC3	15F1	15F7	15FB

Mueve el cursor a una posición en coordenadas relativas. A la entrada, DE contiene la coordenada X y HL la coordenada Y. Todos los registros son alterados.

BBC6	15FC	1602	1606

Examina la posición del cursor de gráficos. No hay requisitos. A la salida, DE contiene la coordenada X absoluta y HL la coordenada Y. A es alterado.

BBC9	1604	160A	160E

Determina el origen de coordenadas del cursor de gráficos. A la entrada, DE contiene la coordenada X, y HL la coordenada Y. Todos los registros son alterados.

BBCC 1612 1618 161C

Examina la posición del origen de coordenadas. No hay requisitos. A la salida, DE contiene la coordenada X estándar del origen y HL la coordenada Y.

BBCF 1734 16A1 16A5

Define los márgenes derecho e izquierdo de la ventana de gráficos. A la entrada, DE y HL contienen las coordenadas X estándar de los márgenes. Todos los registros son alterados.

BBD2 1779 16E6 16EA

Define los bordes superior e inferior de la ventana de gráficos. A la entrada, DE y HL contienen las coordenadas Y estándar de los bordes. Todos los registros son alterados.

BBD5 17A6 1713 1717

Obtiene los márgenes derecho e izquierdo de la ventana de gráficos. No hay requisitos. A la salida, DE contiene la coordenada X estándar del borde izquierdo y HL la coordenada X del borde derecho. A es alterado.

BBD8 17BC 1729 172D

Obtiene los límites superior e inferior de la ventana de gráficos. No hay requisitos. A la salida, DE contiene la coordenada Y estándar del límite

superior y HL la coordenada Y del límite inferior. A es alterado.

BBDB	17C5	1732	1736

Borra la ventana de gráficos. No hay requisitos. Todos los registros son alterados.

BBDE	17F6	1763	1767

Define una tinta para la pluma de gráficos. A la entrada, A contiene el número de tinta. A es alterado.

BBE1	1804	1771	1775

Obtiene el color actual de la pluma de gráficos. No hay requisitos de entrada. A la salida, A contiene el número de tinta. Los demás registros son alterados.

BBE4	17FD	176A	176E

Define una tinta para el papel de gráficos. A la entrada, A contiene el número de tinta. A es alterado.

BBE7	1807	1776	177A

Obtiene el color del papel de gráficos. No hay requisitos. A la salida, A contiene el número de tinta. Los demás registros son preservados.

BBEA 1813 177F 1783

Activa un *pixel* en una posición en coordenadas absolutas. A la entrada, DE contiene la coordenada X y HL la coordenada Y. Todos los registros son alterados.

BBED 1810 177C 1780

Activa un *pixel* en una posición en coordenadas relativas. A la entrada, DE contiene la coordenada X y HL la coordenada Y. Todos los registros son alterados.

BBF0 1827 1793 1797

Obtiene la tinta de un *pixel* en coordenadas absolutas. A la entrada, DE contiene la coordenada X del punto y HL la coordenada Y. A la salida, A contiene el número de tinta. Los registros BC, DE y HL son alterados.

BBF3 1824 1790 1794

Obtiene la tinta de un *pixel* en coordenadas relativas. A la entrada, DE contiene la coordenada X del punto y HL la coordenada Y. A la salida, A contiene el número de tinta. Los registros BC, DE y HL son alterados.

BBF6 1839 17A5 17A9

Dibuja una línea hasta la posición indicada en coordenadas absolutas. A la entrada, DE

contiene la coordenada X del punto final y HL la coordenada Y. Todos los registros son alterados.

BBF9 1836 17A2 17A6

Dibuja una línea hasta la posición indicada en coordenadas relativas. A la entrada, DE contiene la coordenada X del punto final y HL la coordenada Y. Todos los registros son alterados.

BBFC 1945 193C 1940

Escribe un carácter en la posición actual del cursor de gráficos. A la entrada, A contiene el carácter a escribir. Todos los registros son alterados.

Funciones sobre el sistema de pantalla

BBFF 0AA0 0ABB 0ABF

Inicializa el sistema de pantalla. No hay requisitos de entrada. Todos los requisitos son alterados.

BC02 0AB1 0ACC 0AD0

Reinicializa las indirecciones, las tablas de color, velocidad de parpadeo y el modo de texto. No hay requisitos de entrada. Todos los registros son alterados.

BC05 0B3C 0B33 0B37

Impone el desplazamiento del origen de la pantalla. A la entrada, HL contiene el desplazamiento requerido. A y HL son alterados.

BC08 0B45 0B38 0B3C

Define la dirección base de la pantalla. A su entrada, A contiene el byte alto de dicha dirección. A y HL son alterados.

BC0B 0B50 0B52 0B56

Examina la dirección base de pantalla y el desplazamiento actual. No hay requisitos de entrada. A la salida, A contiene el byte alto de la base y HL el desplazamiento. Los demás registros son preservados.

BC0E 0ACA 0AE5 0AE9

Define un nuevo modo de pantalla. A la entrada, A contiene el número de modo. Todos los registros son alterados.

BC11 0AEC 0B0B 0B0C

Comprueba el modo actual de pantalla. Al regreso, el modo actual queda indicado por el estado de los *flags* de cero y de acarreo.

Modo 0: C = 1 Z = 0
Modo 1: C = 0 Z = 1
Modo 2: C = 0 Z = 0

BC14 0AF2 0B13 0B17

Borra la pantalla. No hay requisitos de entrada. Todos los registros son alterados.

BC17 0B57 0B59 0B5D

Examina el tamaño en caracteres de pantalla. No hay requisitos de entrada. A la salida, B contiene la última columna física de la pantalla y C la última fila física de pantalla. A es alterado.

BC1A 0B64 0B66 0B6A

Calcula la dirección de pantalla de la esquina superior izquierda de un carácter. A la entrada, H contiene la columna física del carácter; L, la fila del carácter. A la salida, HL contiene la dirección de la esquina superior izquierda del carácter; B contiene el ancho en bytes del carácter. A es alterado.

BC1D 0B95 0BAB 0BAF

Calcula la dirección de pantalla de un *pixel.* A la entrada, DE contiene la coordenada X del *pixel;* HL contiene la coordenada Y. A la salida, HL contiene la dirección de pantalla del *pixel;* C la máscara del *pixel* y B el número de *pixels* que caben en un byte menos uno. A y DE son alterados.

BC20 0BF9 0C01 0C05

Calcula la dirección del byte situado a la derecha de una dirección dada. A la entrada, HL

contiene una dirección de pantalla. A la salida, HL contendrá la dirección requerida. A es alterado.

BC23 0C05 0C0D 0C11

Calcula la dirección del byte situado a la izquierda de una dirección dada. A la entrada, HL contiene una dirección de pantalla. A la salida, HL contendrá la dirección requerida. A es alterado.

BC26 0C13 0C1B 0C1F

Calcula la dirección del byte situado una línea por debajo de una dirección dada. A la entrada, HL contiene la dirección de pantalla. A la salida, HL contiene la dirección de pantalla requerida. A es alterado.

BC29 0C2D 0C35 0C39

Calcula la dirección del byte situado una línea por encima de una dirección dada. A la entrada, HL contiene la dirección de pantalla. A la salida, HL contiene la dirección de pantalla requerida. A es alterado.

BC2C 0C86 0C8A 0C8E

Codifica una tinta que rellenará todos los *pixels* de un byte. A la entrada, A contiene un número de tinta. A la salida, A contiene la tinta codificada. Los demás registros son preservados.

BC2F 0CAC 0CA3 0CA7

Convierte una tinta decodificada en el número de tinta correspondiente. A la entrada, A contiene una tinta decodificada. A la salida, A contiene el número de tinta. Los demás registros son preservados.

BC32 0CEC 0CEE 0CF2

Fija los colores de un número de tinta. A la entrada, B contiene el primer color, C el segundo y A el número de tinta a chequear. A la salida, B contiene el primer color y C el segundo. Todos los registros son alterados.

BC35 0D14 0D16 0D1A

Examina los colores relacionados con una tinta dada. A la entrada, A contiene el número de tinta a comprobar. A la salida, B contiene el primer color y C el segundo. A, DE y HL son alterados.

BC38 0CF1 0CF3 0CF7

Fija los colores del borde. A la entrada, B contiene el primer color y C el segundo. A la salida, B contiene el primer color y C el segundo. Todos los registros son alterados.

BC3B 0D19 0D1B 0D1F

Examina los colores relacionados con el borde. No hay requisitos de entrada. A la salida, B

contiene el primer color y C, el segundo. A, DE y HL son alterados.

BC3E 0CE4 0CE6 0CEA

Impone los períodos de parpadeo. A la entrada, H contiene el período del primer color y L, el período del segundo color. Los registros A y HL son alterados.

BC41 0CE8 0CEA 0CEE

Examina los períodos de parpadeo. No hay requisitos. A la salida, H contiene el período del primer color y L, el período del segundo color. A es alterado.

BC44 0DB3 0DB5 0DB9

Rellena un área rectangular de la pantalla con una tinta dada. A la entrada, A contiene la tinta codifica; H, la columna física más a la izquierda; D, la columna más a la derecha; L, la fila superior, y E contiene la fila inferior. Todos los registros son alterados.

BC47 0DB7 0DB9 0DBD

Rellena un área rectangular de la pantalla con una tinta dada. A la entrada, C contiene la tinta codificada; HL, la dirección de pantalla de la esquina superior izquierda del rectángulo; D, el ancho en bytes, y E la altura del rectángulo en líneas de pantalla. Todos los registros son alterados.

BC4A 0DDF 0DE1 0DE5

Intercambia las tintas de papel y pluma de un carácter dado. A la entrada, B y C contienen las tintas codificadas; H contiene la columna física del carácter, y L la fila física. Todos los registros son alterados.

BC4D 0DFA 0DFC 0E00

Desplaza la pantalla verticalmente ocho líneas. A la entrada, B = 0 si el desplazamiento es hacia abajo y B<>0 si es hacia arriba; A contiene la nueva línea que aparece. Todos los registros son alterados.

BC50 0E3E 0E40 0E44

Desplaza verticalmente un área de la pantalla ocho líneas. A la entrada, B = 0 indica un desplazamiento hacia abajo; si es hacia arriba entonces B<>0; A contiene la tinta codificada de la línea que aparece; H, la columna física de la izquierda del área; D, la columna derecha; L, la fila superior, y E la fila inferior. Todos los registros sufrirán alteración.

BC53 0EF3 0EF5 0EF9

Define la matriz que define un carácter para adaptarla al modo actual de pantalla. A la entrada, HL contiene la dirección donde está definida la matriz y DE la dirección de un área de memoria que se empleará para realizar la expansión. Todos los registros son alterados.

BC56 0F49 0F26 0F2A

Convierte un carácter al tamaño estándar. A la entrada, A contiene la tinta codificada del carácter; H la columna física del carácter; L, la fila que ocupa el carácter, y DE la dirección del área donde se formalizará la matriz estándar. Todos los registros son alterados.

BC59 0C49 0C51 0C55

Fija el modo de gráficos para la unidad de vídeo. A la entrada, A contiene el modo requerido. Todos los registros son alterados.

BC5C 0C6B 0C70 0C74

Escribe en la pantalla un *pixel*, ignorando el modo actual. A la entrada, B contiene la tinta codificada del *pixel;* C, la máscara para el *pixel*, y HL la dirección de pantalla. A es alterado.

BC5F 0FC4 0F8F 0F93

Dibuja una línea horizontal. A la entrada, A contiene la tinta codificada para la línea; DE, la coordenada X de comienzo; BC, la coordenada X final, y HL la coordenada Y de la línea. Todos los registros son alterados.

BC62 102F 0F97 0F9B

Dibuja una línea vertical. A la entrada, A contiene la tinta codificada; DE, la coordenada X de

la línea; HL, la coordenada Y de comienzo de la línea, y BC la coordenada Y final. Todos los registros son alterados.

Funciones sobre monitor de disco o de *cassette*

BC65 2370 24BC 24BC

Inicializa el monitor de *cassette*. No hay requisitos de entrada. Todos los registros son alterados.

BC68 237F 24CE 24CE

Define la velocidad de escritura. A la entrada, HL contiene la duración en microsegundos de medio período de un bit a nivel bajo; A debe contener la precompensación. A y HL son alterados.

BC6B 238E 24E1 24E1

Activa o desactiva los mensajes de ayuda. A la entrada, A = 0 para activar o A<>0 para desactivar. A es alterado.

BC6E 2A4B 2BBB 2BBB

Enciende el motor del *cassette*. No hay condiciones de entrada. A la salida, el acarreo estará a uno si el motor empezó a funcionar; estará a cero si

se pulsó ESC. A contiene el estado interior del motor. Todos los demás registros quedarán sin ser modificados.

BC71 2A4F 2BBF 2BBF

Desconecta el motor del *cassette*. No hay requisitos de entrada. A la salida, A contiene el estado anterior del motor. El acarreo estará a uno si se realizó la desconexión y a cero si se pulsó ESC. Los demás registros son preservados.

BC74 2A51 2BC1 2BC1

Restaura el estado anterior del motor. A la entrada, A debe contener el estado previo. A la salida, el acarreo estará a uno si no se pulsó ESC.

BC77 2392 288B 24E5

Abre un fichero de entrada. A la entrada, B contiene la longitud del nombre del fichero, HL contiene la dirección del nombre, y DE contiene la dirección de un área de 2K que será el *buffer* para la carga. A la salida, HL contiene la dirección del *buffer* donde está la cabecera del fichero, DE contiene la dirección de los datos según la cabecera, BC contiene la longitud del fichero según la cabecera y A contiene el tipo de fichero. Estas condiciones de salida sólo son ciertas si el acarreo está a uno y el *flag* Z a cero. Si se pulsó ESC el acarreo y Z quedarán a uno. En el resto de los casos el acarreo y Z quedarán a cero.

BC7A 23FC 288B 2550

Cierra un fichero de entrada. No hay requisitos de entrada. A la salida, el acarreo estará a uno si se realizó la acción. Todos los registros quedarán alterados.

BC7D 2401 288B 257F

Abandona un fichero de entrada. No hay requisitos. Todos los registros son alterados.

BC80 2435 288B 25A0

Lee un carácter del fichero de entrada. No hay requisitos. A la salida, A contiene el código del carácter, lo cual queda indicado si el acarreo está a uno y el *flag* de cero está a cero. Si se encontró el final del fichero, Z = 0 y C = 0. Si se pulsó ESC, C = 0 y Z = 1. Todos los demás registros son preservados.

BC83 24AB 288B 2618

Lee un fichero completo. A la entrada, HL contiene la dirección donde se almacenará el fichero. A la salida, C = 1, Z = 0 si se leyó el fichero correctamente y HL contiene la dirección de entrada según la cabecera. Si el fichero no estaba abierto, C = 0 y Z = 0. Si se pulsó ESC, C = 0 y Z = 1. Todos los registros alterados.

BC86 249A 288B 2607

Devuelve al fichero de entrada el último carácter leído. No hay requisitos de entrada. Todos los registros son preservados.

BC89 2496 288B 2603

Comprueba si se ha alcanzado el final del fichero. No hay requisitos de entrada. A la salida, C = 1 y Z = 0 si aún no se ha llegado al final del fichero; C = 0 y Z = 0 si se alcanzado; C = 0 y Z = 1 si se pulsó ESC. A es alterado.

BC8C 23AB 288B 24FE

Abre un fichero de salida. A la entrada, B contiene la longitud del nombre del fichero; HL, la dirección del nombre, y DE la dirección de un área de 2K que actúa como *buffer*. A la salida, C = 0 si el fichero ya estaba abierto. Si el fichero se abre correctamente, C = 1 y HL contiene la dirección donde está la cabecera. A, BC, DE y HL son alterados.

BC8F 2415 288B 256D

Cierra un fichero de salida y lo graba. No hay requisitos. A la salida, C = 1 y Z = 0 si se realizó correctamente la operación; C = 0 y Z = 0 si el fichero no estaba abierto; C = 0 y Z = 1 si se pulsó ESC. Todos los registros son alterados.

BC92 242E 288B 2599

Abandona un fichero de salida. No hay requisitos de entrada. Todos los registros son alterados.

BC95 245B 288B 25C6

Escribe un carácter en el fichero de salida. A la entrada, A contiene el carácter a escribir. A la salida, C = 1 y Z = 0 si se realizó bien la operación; C = 0 y Z = 0 si el fichero no estaba abierto; C = 0 y Z = 1 si se pulsó ESC. Tan sólo el registro A es alterado.

BC98 24EA 288B 2653

Escribe un fichero completo. A la entrada, HL contiene la dirección donde están los datos a escribir; DE, la longitud de los datos; BC, la dirección de entrada, y A el tipo de fichero, ambos para el encabezamiento. A la salida, si el fichero se escribió, C = 1 y Z = 0; si el fichero no estaba abierto, C = 0 y Z = 0; si se pulsó ESC, entonces C = 0 y Z = 1. Todos los registros son alterados.

BC9B 2528 288B 2692

Lista el contenido de una cinta. A la entrada, DE contiene la dirección de un *buffer* de 2 Kbytes para la información. A la salida, el acarreo estará a uno si se realizó bien la operación. Además, C = 0 y Z = 0 si había algún fichero abierto; C = 0 y Z = 1 si se produjo error. Todos los registros quedarán alterados.

BC9E 283F 29AF 29AF

Escribe una cadena de caracteres en la cinta. A la entrada, HL contiene la dirección de los datos a escribir; DE, la longitud de los datos, y A el carácter de sincronismo a escribir al final de la antegrabación. A la salida, el acarreo estará a uno si todo fue bien, pero si el acarreo está a cero, entonces hubo algún error y A contiene el código del error. BC, DE y HL son alterados.

BCA1 2836 29A6 2946

Lee un registro o cadena de la cinta. A la entrada, HL contiene la dirección donde se guardarán los datos; DE, la longitud de los datos a leer, y A el carácter de sincronismo esperado al final de la antegrabación. A la salida, acarreo a uno si todo fue bien, pero acarreo a cero si hubo algún error; entonces, A contiene un código de error. En este caso, los registros BC, DE y HL quedarán modificados.

BCA4 2851 29C1 29C1

Compara una grabación con el contenido de la memoria. A la entrada, HL contiene la dirección de los datos a comparar; DE, la longitud de los datos, y A el carácter de sincronismo esperado. A la salida, si la comparación es correcta, acarreo a uno; si hubo algún error, C = 0 y A contiene un código del error detectado.

Funciones sobre el monitor de sonido

BCA7 1E68 1FE9 1FE9

Inicializa el monitor de sonido. No hay requisitos. Todos los registros son alterados.

BCAA 1F9F 2114 2114

Añade un sonido a la cola de sonidos. A la entrada, HL contiene la dirección de un bloque que define un sonido. A la salida, el acarreo a uno si el sonido se añadió a la cola y a cero si la cola estaba llena. A, BC y DE son alterados.

BCAD 206C 21CE 21CE

Comprueba si hay espacio libre en la cola. A la entrada, A contiene el bit propio del canal a examinar. A la salida, A contiene el estado del canal. Los registros BC, DE y HL son alterados por esta función.

BCB0 2089 21EB 21EB

Actualiza un suceso que se ejecutará cuando la cola de sonido se vacíe. A la entrada, A contiene el bit propio del canal y HL contiene la dirección de la rutina del suceso asociado. Todos los registros son alterados.

BCB3 204A 21AC 21AC

Permite la producción de sonido en los canales especificados. A la entrada, A contiene los bits propios de los canales que se desean. Todos los registros son alterados.

BCB6 1ECB 2050 2050

Detiene la ejecución de sonidos. No hay requisitos. A la salida, el acarreo estará a uno si había algún sonido ejecutándose; si no, acarreo a cero. A, BC y HL son alterados.

BCB9 1EE6 206B 206B

Permite continuar los sonidos que la rutina anterior había retenido. No hay requisitos. A, BC y DE son alterados.

BCBC 2338 2495 2495

Define una envolvente de volumen programable. A la entrada, A contiene el número de envolvente que se va a definir; HL incluye la dirección de un bloque que contiene los datos de la envolvente. A la salida, el acarreo estará a uno si se definió correctamente y HL contendrá la dirección del bloque, más 16. Si el número de la envolvente es inválido, acarreo a cero. A, B y HL son preservados.

BCBF 233D 249A 249A

Define una envolvente de tono. A la entrada, A contiene el número de envolvente y HL la dirección

del bloque que contiene los datos de la envolvente. Las condiciones de salida son las mismas que para BCBC. DE es alterado.

BCC2 2349 24A6 24A6

Obtiene la dirección de un bloque definitorio de envolvente de volumen. A la entrada, A contiene el número de la envolvente. A la salida, el acarreo estará a uno si se encontró el bloque. Entonces, HL contendrá la dirección del bloque y BC la longitud de la envolvente. Si el número de la envolvente no es correcto el acarreo estará a cero. A es alterado.

BCC5 234E 24AB 24AB

Obtiene la dirección de un bloque definitorio de envolvente de tono. A la entrada, A contiene el número de la envolvente. A la salida, el acarreo estará a uno si se encontró el bloque. Entonces, HL contendrá la dirección del bloque y BC la longitud de la envolvente. Si el número de la envolvente no es correcto, el acarreo estará a cero. A es alterado.

Funciones sobre el núcleo de sucesos e interrupciones

BCC8 005C 005C 005C

Inicializa el núcleo, sucesos y temporizaciones. A la salida, B contiene el número

de la ROM primaria actual; C contiene el número de selección de ROM para un programa primario en RAM, y DE contiene la dirección de entrada de la ROM primaria. A y HL son alterados.

BCCB	0329	0326	0326

Identifica e inicializa las ROM secundarias. A la entrada, DE y HL contienen respectivamente las direcciones del primero y último bytes utilizables de la memoria.

BCCE	0332	0330	0330

Inicializa una ROM secundaria específica. A la entrada, C contiene el número de ROM a inicializar; DE contiene la dirección más baja del byte utilizable, y HL la más alta dirección utilizable. A la salida, DE y HL contienen los nuevos valores mínimo y máximo utilizables de la memoria. A y B son alterados.

BCD1	02A1	02A0	02A0

Introduce una extensión RSX para el *firmware*. A la entrada, BC contiene la dirección de la tabla de órdenes de la RSX; HL contiene la dirección de un área de 4 bytes de la RAM para uso del núcleo. DE es alterado.

BCD4	02B2	02B1	02B1

Busca una extensión RSX, una ROM primaria o una secundaria que procese una orden dada. A la entrada, HL contiene la dirección donde se

encuentra almacenado el nombre de la orden. A la salida, acarreo a uno si se encontró la orden RSX o de ROM; C contendrá el número de selección de ROM, y HL la dirección de la rutina. Si no se encontró la orden, acarreo a cero y HL alterado. Siempre se alteran A, B y DE.

BCD7	0163	0163	0163

Inicializa y añade un bloque en la lista de síncronos asociados al barrido de pantalla. A la entrada, HL contiene la dirección del bloque; B, la clase de suceso; C, el número de ROM de la rutina del suceso, y DE la dirección de la rutina asociada al suceso. A, DE y HL son alterados.

BCDA	016A	016A	016A

Añade un bloque a la lista de este tipo de sucesos. A la entrada, HL contiene la dirección del bloque. A, DE y HL son alterados.

BCDD	0170	0170	0170

Elimina un bloque de la lista de este tipo de sucesos. A la entrada, HL contiene la dirección del bloque. A, DE y HL son alterados.

BCE0	0176	0176	0176
BCE3	017D	017D	017D
BCE6	0183	0183	0183

Estas tres rutinas ejercen la misma función que las tres inmediatamente anteriores, pero refiriéndose a los

sucesos de temporización rápida. Mismos requisitos de entrada que los anteriores. A, DE y HL son alterados.

BCE9 01B3 01B3 01B3

Pone un bloque en la lista de sucesos síncronos lentos. A la entrada, HL contiene la dirección del bloque; DE, el valor inicial del byte "cuenta", y BC el valor de recarga. Todos los registros son alterados.

BCEC 015C 01C5 01C5

Elimina un bloque de la lista de sucesos lentos. A la entrada, HL contiene la dirección del bloque. A la salida, si el bloque se encontró en la lista, el acarreo estará a uno, y DE contendrá la cuenta remanente; si no se encontró el acarreo, estará a cero. A y HL son alterados.

BCEF 01D2 01D2 01D2

Inicializa un bloque de sucesos. A la entrada, HL contiene la dirección del bloque; B, la clase de suceso; C, la dirección de selección (número) de ROM para ese suceso, y DE la dirección de la rutina asociada. A la salida, HL contiene la dirección del bloque de suceso más 7. Los demás registros son preservados.

BCF2 01E2 01E2 01E2

Atiende un bloque de suceso. A la entrada, HL

contiene la dirección del bloque de suceso. No hay condiciones de salida. Todos los registros son alterados.

BCF5 0228 0227 0227

Borra la secuencia de sucesos síncronos. No hay requisitos de entrada ni condiciones de salida. A y HL alterados.

BCF8 0285 0284 0284

Borra un suceso síncrono de la secuencia de sucesos. A la entrada, HL debe contener la dirección del bloque de suceso. Todos los registros son alterados.

BCFB 0256 0255 0255

Toma el siguiente suceso de la secuencia. No hay requisitos de entrada. A la salida, si hay algún suceso que procesar, el acarreo estará a uno; HL contiene la dirección del bloque del suceso, y A la prioridad del último suceso procesado. Si no hay sucesos que procesar, el acarreo estará a cero. DE es alterado.

BCFE 021A 0219 0219

Llama a la rutina asociada a un suceso y la ejecuta. No hay requisitos. Todos los registros son alterados.

BD01 0277 0276 027C

Termina la ejecución de un suceso. A la entrada, A contiene la prioridad del suceso anterior, y HL la dirección del bloque del suceso. Todos los registros son alterados.

BD04 0295 0294 0294

Inhibe los sucesos síncronos normales, conservando los clasificados como urgentes. No hay requisitos de entrada. HL es alterado.

BD07 029B 029A 029A

Habilita los sucesos síncronos normales. No hay requisitos de entrada. Sólo el registro HL es alterado.

BD0A 028E 028D 028D

Impide la ejecución de un suceso. A la entrada, HL contiene la dirección del bloque del suceso. A es alterado.

BC0D 0099 0099 0099

La rutina regresa con la cuenta del tiempo transcurrido. A la salida, DEHL contienen los cuatro bytes del contador: D el byte más alto y L el menos significativo. Los demás registros son preservados.

BD10	00A3	00A3	00A3

Fija un valor inicial para el contador de tiempo. A la entrada, DEHL contendrán los cuatro bytes para el contador: D el byte más significativo y L el menos significativo. A es alterado.

Funciones que relacionan *hardware* y *software*

BD13	05DC	05D7	05ED

Carga y ejecuta un programa. A la entrada, HL debe contener la dirección de una rutina que se encargará de cargar el programa. Si la carga se efectuó bien, acarreo a uno; HL contendrá la dirección del punto de entrada al programa. Si HL = 0000 entonces se pasará el control al intérprete de BASIC en la dirección C.000. Si el programa cargado termina de ejecutarse, se lleva a cabo una completa reinicialización del sistema. CALL 0. No hay condiciones de salida.

BD16	060B	0606	061C

Ejecuta un programa principal. A la entrada, HL contiene la dirección de comienzo y C el número de ROM a emplear. Al comienzo de esta rutina y de la anterior se produce una inicialización completa del sistema y después se ejecuta el programa. No hay condiciones de salida.

Funciones sobre la unidad de pantalla

BD19 07BA 07A4 07B4

Retrasa toda acción en la pantalla hasta que se produzca un nuevo barrido del haz de rayos catódicos. No hay requisitos de entrada. Todos los registros son preservados.

BD1C 0776 0766 0776

Fija el modo de pantalla. A la entrada, A contiene el modo requerido. A la salida, A es alterado.

BD1F 077C 07B0 07C0

Fija el *offset* de pantalla. A la entrada, A contiene el byte superior de la base requerida y HL contiene el *offset* o desplazamiento. A es alterado.

BD22 0786 0776 0786

Borra las tintas. A la entrada, DE contiene la dirección de un vector de tintas. A es alterado.

BD25 0799 077C 078C

Asigna un color a cada una de las tintas y al borde. A la entrada, DE contiene la dirección de un vector de tintas. A es alterado.

BD28 07E6 07D0 07E0

Reinicializa la indirección BDF1. No hay requisitos de entrada. A la salida, todos los registros son alterados.

BD2B 07F2 080B 081B

Intenta mandar un carácter al puerto Centronics. A la entrada, A contiene el carácter a enviar a la impresora. Si se envió el carácter el acarreo estará a uno; si no, estará a cero. A es alterado.

BD2E 081B 0848 0858

Comprueba si el puerto Centronics está ocupado. No hay requisitos de entrada. A la salida, si el puerto está ocupado, el acarreo estará a uno; si no lo está, C = 0. Los demás registros son preservados.

BD31 0817 0834 0844

Envía un carácter a la impresora. A la entrada, A contiene el carácter a enviar. El registro A es alterado.

BD34 0826 0853 0863

Programa un registro del generador programable de sonido. A la entrada, A contiene el número de registro y C el dato a introducir en el registro. A y BC son alterados.

BD37	0888	08BB	08BD

Restaura las direcciones e indirecciones del bloque de saltos. No hay requisitos de entrada. A la salida, todos los registros son alterados.

Indirecciones del *firmware*

BDCD	1263	125F	125F

Coloca el cursor en la pantalla. No hay requisitos de entrada. A es alterado.

BDD0	1263	125F	125F

Retira el cursor de la pantalla. No hay requisitos de entrada. A es alterado.

BDD3	134A	134B	134B

Escribe un carácter en la pantalla. A la entrada, A contiene el carácter a escribir; H, la columna física, y L la fila física donde se escribirá el carácter. Todos los registros son alterados.

BDD6	13C0	13BE	13BE

Lee un carácter de la pantalla. A la entrada, H contiene la columna física desde donde leer, y L la fila física. A la salida, si encontró algún carácter identificable, el acarreo estará a uno y A contendrá el código del carácter leído. Si no se pudo reconocer

ningún carácter, el acarreo aparecerá a cero. BC, DE y HL son alterados.

BDD9 140C 140A 140A

Manda un carácter o un código de control. A la entrada, A contiene el código del carácter. Todos los registros son alterados.

BDDC 1816 1786 1786

Dibuja un punto. A la entrada, DE contiene la coordenada X y HL la coordenada Y. Todos los registros son alterados.

BDDF 182A 179A 179A

Obtiene la tinta de un *pixel* en coordenadas absolutas. A la entrada, DE contiene la coordenada X del punto, y HL la coordenada Y. A la salida, A contiene la tinta decodificada. BC, DE y HL son alterados.

BDE2 183C 17B4 17B4

Dibuja una línea. A la entrada, DE contiene la coordenada X del punto final, y HL la coordenada Y. Todos los registros son alterados.

BDE5 0C82 0C8A 0C8A

Lee un punto de la pantalla y decodifica su tinta. A la entrada, HL contiene la dirección de

pantalla del punto y C la máscara del punto. A la salida, A contiene la tinta decodificada del punto. Los demás registros son preservados.

BDE8 0C68 0C71 0C71

Activa uno o varios puntos de la pantalla en el modo de gráficos actual. A la entrada, HL contiene la dirección de pantalla del *pixel;* C la máscara del *pixel*, y B la tinta codificada en la que escribe el *pixel.* A es alterado.

BDEB 0AF7 0B17 0B17

Pone toda la pantalla con la tinta 0. No hay requisitos de entrada. Todos los registros son alterados.

BDEE 1C90 1DB8 1DB8

Comprueba si la tecla ESC está pulsada. A la entrada, interrupciones inhibidas; C contiene el estado de las teclas MAY y CONTROL. A y HL son alterados.

BDF1 07F8 0835 0835

Escribe un carácter por impresora o espera cierto tiempo. A la entrada, A contiene el carácter a enviar a la impresora. A la salida, el acarreo estará a uno si se envió el carácter. A y BC son alterados.

Funciones sobre la unidad de disco

BC77 2392 288B 24E5

Abre un fichero de entrada. A la entrada, B contiene la longitud del nombre del fichero; HL contiene la dirección del nombre, y DE contiene la dirección de un área de 2K que será el *buffer* para la carga. A la salida, HL contiene la dirección del *buffer* donde está la cabecera del fichero; DE contiene la dirección de los datos según la cabecera; BC contiene la longitud del fichero según la cabecera y A contiene el tipo de fichero. Estas condiciones de salida sólo son ciertas si el acarreo está a uno y el *flag* Z a cero. Si se pulsó ESCape, el acarreo y Z quedarán a uno. En el resto de los casos el acarreo y Z quedarán a cero.

BC7A 23FC 288B 2550

Cierra un fichero de entrada. No hay condiciones de entrada. A la salida, el acarreo estará a uno si se realizó la acción.

BC7D 2401 288B 257F

Abandona un fichero de entrada. No hay requisitos.

BC80 2435 288B 25A0

Lee un carácter del fichero de entrada. No hay

requisitos. A la salida, A contiene el código del carácter, lo cual queda indicado si el acarreo está a uno y el *flag* de cero está a cero. Si se encontró el final del fichero, Z = 0 y C = 0. Si se pulsó ESC, C = 0 y Z = 1.

BC83 24AB 288B 2618

Lee un fichero completo. A la entrada, HL contiene la dirección donde se almacenará el fichero. A la salida, si el fichero se leyó correctamente, C = 1, Z = 0 y HL contiene la dirección de entrada según la cabecera. Si el fichero no estaba abierto, LC = 0 y Z = 0. Si se pulsó ESC, C = 0 y Z = 1.

BC86 249A 288B 2607

Devuelve al fichero de entrada el último carácter leído. No hay requisitos.

BC89 2496 288B 2603

Comprueba si ha alcanzado el final del fichero. No hay requisitos. A la salida, C = 1 y Z = 0 si aún no se ha llegado al final del fichero; C = 0 y Z = 0 si se ha alcanzado; C = 0 y Z = 1 si se pulsó ESC.

BC8C 23AB 288B 24FE

Abre un fichero de salida. A la entrada, B contiene la longitud del nombre del fichero, HL la dirección del nombre y DE la dirección de un área de

2K que actúa como *buffer*. A la salida, si el fichero ya estaba abierto, C = 0; si el fichero se abre correctamente, C = 1 y HL contiene la dirección donde está la cabecera del fichero especificado en la función.

BC8F 2415 288B 256D

Cierra un fichero de salida y lo graba en el disco. No hay requisitos. A la salida, C = 1 y Z = 0 si se realizó bien la operación; C = 0 y Z = 0 si el fichero no estaba abierto; C = 0 y Z = 1 si se pulsó ESC.

BC92 24E2 288B 2599

Abandona un fichero de salida. No hay requisitos.

BC95 245B 288B 25C6

Escribe un carácter en el fichero de salida. A la entrada, A contiene el carácter a escribir. A la salida, C = 1 y Z = 0 si se realizó bien la operación; C = 0 y Z = 0 si el fichero no estaba abierto; C = 9 y Z = 1 si se pulsó ESC.

BC98 24EA 288B 2653

Escribe un fichero completo. A la entrada, HL contiene la dirección donde están los datos a escribir; DE la longitud de los datos; BC la dirección de entrada, y A el tipo de fichero, ambos para el encabezamiento. A la salida, si el fichero se escribió,

C = 1 y Z = 0; si el fichero no estaba abierto, C = 0 y Z = 0; si se pulsó ESC, entonces C = 0 y Z = 1.

BC9B 2528 288B 2692

Lista el contenido de un disco. A la entrada, DE contiene la dirección de un *buffer* de 2Kbytes para la información. A la salida, el acarreo estará a uno si se realizó bien la operación. Además, C = 0 y Z = 0 si había algún fichero abierto; C = 0 y Z = 1 si se produjo algún error.

Sistema operativo de la unidad de disco

BE80 CA72

Activa o desactiva los mensajes de error para disco. A la entrada, A contiene &00 para activar los mensajes y &FF para desactivarlos. A la salida, A contiene el estado anterior.

BE83 C60D

Inicializa los diferentes parámetros relativos a la unidad de disco. A la entrada, HL contiene la dirección de un bloque de nueve parámetros. La interpretación del bloque es:

Bytes 0-1 Tiempo de arranque del motor en 1/50 segundos.
Bytes 2-3 Tiempo de parada del motor en 1/50 segundos.

Byte 4 Tiempo de escritura en unidades de 10 microsegundos.
Byte 5 Tiempo de asentamiento de la cabeza lectora en milisegundos.
Byte 6 Tiempo entre paso y paso en milisegundos.
Byte 7 Retardo de escritura de la cabeza lectora.
Byte 8 Retardo de carga de la cabeza lectora.

BE86 C581

Selecciona uno de los formatos dispuestos para Amstrad. A la entrada, A contiene el número del primer sector del formato elegido. Siendo 65 para formato del sistema; &C1, formato de sólo datos; 01, formato IBM. E = 0 para unidad de disco A y E = 1 para la unidad B.

BE89 C666

Lee un sector físico del disco. A la entrada, HL contiene la dirección de un *buffer* en memoria para el sector; E el número de unidad de disco (0 para la A y para la B); D el número de pista, y C el número de sector. A la salida, acarreo a uno si se leyó bien y HL contiene la dirección del *buffer* donde está almacenado el sector.

BE8C C64E

Escribe un sector del disco. A la entrada, HL contiene la dirección de memoria donde están almacenados los 512 bytes de datos; E contiene el número de unidad de disco; D el número de pista, y C

el número de sector. A la salida, acarreo a uno si se escribió bien y HL contiene la dirección del *buffer* de datos. Acarreo a cero si hubo error; A contiene el byte de error, y HL la dirección del *buffer* de errores.

BE8F C652

Formatea una pista según el formato seleccionado. A la entrada, E contiene el número de unidad de disco; D el número de pista, y HL la dirección de un *buffer* de información de cabeceras. Este *buffer* se interpreta así:

Entrada al sector para el primer sector

Byte 0: número de pista
Byte 1: número de cabecera
Byte 2: número de sector
Byte 3: log2 (número de bytes del sector) —7

Entrada al sector para el segundo sector

Byte 0: número de pista
Byte 1: número de cabecera
Byte 2: número de sector
Byte 3: log2 (número de bytes del sector) —7

Entrada al sector para el último sector

Byte 0: número de pista
Byte 1: número de cabecera
Byte 2: número de sector
Byte 3: log2 (número de bytes del sector) —7

A la salida, acarreo a uno si se formateó correctamente.

BE92 C763

Mueve la cabeza lectora a la pista especificada. A la entrada, E contiene el número de unidad de disco y D el número de pista. A la salida, acarreo a uno si todo fue bien; acarreo a cero si hubo algún fallo.

BE95 C630

Devuelve el estado de la unidad de disco especificada. A la entrada, A contiene el número de unidad de disco (0 para la A, 1 para la B). A la salida, acarreo a uno si no hubo error; A contiene el byte de estado. La interpretación de este byte es:

bit 0	Unidad seleccionada.
bits 1-2	Siempre a cero.
bit 3	Indefinido.
bit 4	Línea de la pista 0 a 1.
bit 5	Línea READY a uno (preparada).
bit 6	Línea de protección contra escritura a uno.
bit 7	Indefinido.

BE98 C603

Determina el número de veces que se vuelve a intentar una operación en caso de error. A la entrada, A contiene el nuevo valor; a la salida, A contiene el valor anterior.

BE9B C168

Llama a una rutina del *firmware*. A la entrada,

todos los registros y *flags* deben estar tal y como requiera la rutina llamada; IX contiene la dirección de la rutina. A la salida, todos los registros se devolverán tal y como la rutina establezca.

BE9E C0DB

Indica si se deben guardar los registros alternativos y el registro IY. A la entrada, A contiene &00 si se han de guardar en la pila y &FF para no guardarlos. A la salida, A contiene el valor anterior.

BEA1 C389

Inicializa los canales del interfaz serie (SIO). A la entrada, HL contiene la dirección de un bloque de parámetros. Este se interpretará así:

Byte 0	Registro 4 del canal A de la SIO
Byte 1	Registro 5 del canal A
Byte 2	Registro 3 del canal A
Byte 3	Registro 4 del canal B
Byte 4	Registro 5 del canal B
Byte 5	Registro 3 del canal B
Byte 6-7	Temporizador 0 del CI8253
Byte 8-9	Temporizador 1 del CI8253
Byte 10-11	Temporizador 2 del CI8253

BEA4 C301

Inicializa el *buffer* que contiene las órdenes de comienzo. Los caracteres se introducirán desde el teclado. A la entrada, A = &00 para vaciar el *buffer*

al pulsar una tecla; A = &CC para añadir al *buffer* la tecla pulsada; HL contiene la dirección del *buffer*.

BEA7 C3DB

Examina si hay algún carácter disponible en el dispositivo 0 de E/S. A la salida, A contiene &FF si hay carácter, o &00 si no lo hay. El dispositivo 0, por omisión, es el canal A de la SIO.

BEAA C3F7

Toma un carácter del dispositivo de E/S. Si no hay ninguno disponible, esperará hasta que aparezca uno. A la salida, A contiene el carácter de entrada.

BEAD C435

Examina si hay algún carácter preparado para salir desde el dispositivo 0. A la salida, A contiene &FF si está preparado; A = &00 si el dispositivo está ocupado.

BEB0 C445

Envía un carácter al dispositivo 0 de E/S. Si el dispositivo está ocupado, esperará hasta que dé la señal de listo. A la entrada, C contiene el carácter a enviar.

BEB3	C363
BEB6	C3FF
BEB9	C43A
BEBC	C44B

Estas cuatro rutinas son idénticas, respectivamente, a las cuatro anteriores, pero refiriéndose al dispositivo 1 de E/S. Se toma por defecto el canal B de la SIO.

8
Acceso a las diferentes ROM

Cierto es que las órdenes PEEK y POKE del BASIC, nos permiten conocer o modificar el contenido de la memoria; sin embargo, con ellas sólo trabajamos en la RAM. El acceso a las memorias de tipo ROM no es posible con ninguna orden directa del BASIC. Para volcar en pantalla o unidad de almacenamiento el contenido de una ROM, e incluso para ejecutar una rutina de dicha memoria, son necesarios dos pasos previos: primero seleccionarla y después activarla.

Estas dos operaciones son posibles, gracias a las subrutinas del *firmware* localizadas en la zona &B900 a &B918. Hay que señalar que desde BASIC, las ROM permanecen continuamente desactivadas.

Acceso a la ROM inferior

No es necesario seleccionarla; únicamente activarla o desactivarla. Veamos una rutina BASIC que explica el funcionamiento de un volcado:

```
10 FOR A=30000 TO 30019:READ D$
20 POKE A,VAL("&"+D$):NEXT
```

```
30 DATA CD,06,B9,DD,6E,00,DD,66,01,F5,7E
40 DATA DD,6E,02,DD,66,03,77,F1,C9
50 H%=0:CLS:INPUT "Dirección inicial ",DI
60 INPUT "Dirección final ",DF:PRINT
70 FOR T=DI TO DF:CALL 30000,@H%,T
80 PRINT HEX$(H%,2);" ";:NEXT
```

Primero se almacena el código máquina que accederá a las direcciones requeridas. A continuación, se acude a él para cada una de las direcciones del volcado y se escribe en la pantalla en código hexadecimal.

Acceso a las ROM superiores

En total, el sistema admite hasta 255 posibles ROM (o RAM), teniendo cada una de ellas un número propio de selección. El número de la ROM que contiene el BASIC es el 00. La ROM para el manejo de unidad de disco tiene el número 07. Todas aquellas que quieran añadirse, deberán tener un número de selección diferente. A continuación se lista un programa en BASIC para el volcado de cualquiera de las ROM superiores.

```
10 FOR A=30100 TO 30128:READ D$
20 POKE A,VAL("&"+D$):NEXT
30 DATA DE,4E,00,CD,0F,B9,DD,6E,02,DD,
   66,03
40 DATA C5,7E,DD,6E,04,DD,66,05,77,C1,
   CD,18
50 DATA B9,CD,00,B9,C9
60 H%=0:CLS:INPUT "Dirección inicial ",DI
70 INPUT "Dirección final ",DF
80 INPUT "Número de ROM ",C
```

```
90 FOR T=DI TO DF:CALL 30100,@H%,T,C
100 PRINT HEX$(H%,2);" ";:NEXT
```

Es interesante, por ejemplo, listar en código ASCII el contenido de la ROM de BASIC entre las direcciones correspondientes a los mensajes de error.

La capacidad máxima de una ROM añadida al sistema es de 16 Kbytes. Normalmente, todas las ROM que se introduzcan serán primarias *(foreground* ROM); sin embargo, pueden incluirse hasta siete memorias de tipo ROM secundarias *(background* ROM). Cada conjunto homogéneo de ROM debe estar numerado consecutivamente, siendo los números más bajos los de selección de la primaria, que puede estar compuesta, a su vez, por cuatro ROM. El margen de direccionamiento de las ROM superiores va desde C000 hasta FFFF. El formato del contenido es libre, excepto para los seis primeros bytes; estos deben contener:

C000: Tipo de ROM
0 para ROM primaria
1 para ROM secundaria
2 extensión de ROM primaria
&80 ROM de BASIC del sistema
C001: Número/marca de la ROM
C002: Número de versión
C003: Nivel de modificación
C004-5: Dirección de la tabla de instrucciones externas

9

Conexiones de usuario

* Salida de vídeo

pin 1: rojo
pin 2: verde
pin 3: azul
pin 4: sincronismo
pin 5: masa
pin 6: brillo

Figura 9.1

* Puerto de usuario o de *joystick*

Como *joystick*:	Como puerto de usuario de entrada
pin 1: arriba	pin 1: bit 0
pin 2: abajo	pin 2: bit 1
pin 3: izquierda	pin 3: bit 2
pin 4: derecha	pin 4: bit 3
pin 5: no se usa	pin 5: bit 6
pin 6: fuego 2	pin 6: bit 4
pin 7: fuego 1	pin 7: bit 5
pin 8: masa	pin 8: masa
pin 9: masa para el 2	pin 9: masa

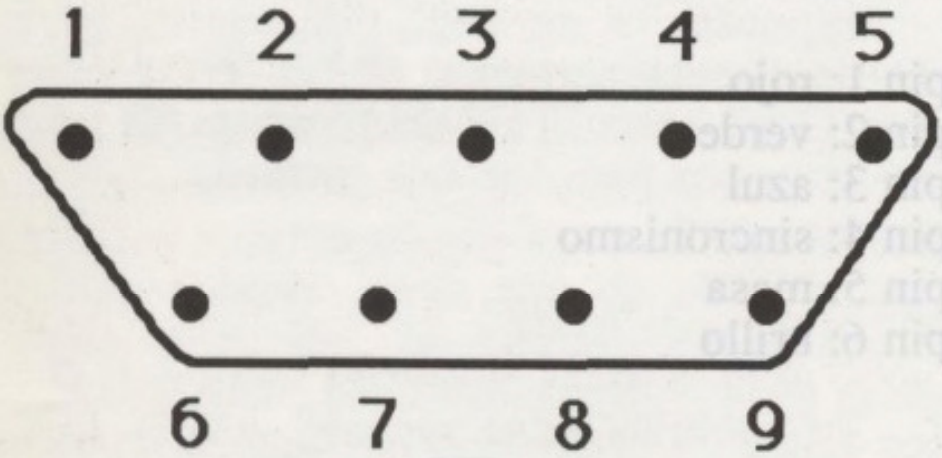

Figura 9.2

Esta conexión es sólo de entrada, así que, funcionando como puerto de usuario, nos servirá únicamente para introducir datos; justo lo contrario que el puerto de impresora.

Como puede observarse en la lista de significado de los pines, sólo son activos los 7 bits menos significativos. Además, es importante señalar que estos pines son activos a nivel bajo, es decir, realizando un cortocircuito entre el pin que nos interesa y tierra.

A pesar de ello, cuando consultemos el estado de

este puerto con JOY o con la rutina situada en BB24, obtendremos un 1 para cada uno de los pines en activo. Es necesario tener esto en cuenta para no equivocarse.

Puerto de salida de impresora

Sólo se utilizan 22 de los 34 pines del conector. Todo dato enviado a este puerto, ya sea con PRINT #8 o con cualesquiera de las rutinas del *firmware,* sólo aparecerá en sus pines cuando la señal de entrada BUSY esté a cero. La señal de salida STROBE se pone a nivel bajo cada vez que se intenta mandar un dato a través de este puerto, esté o no el dato presente en el conector. Sirve junto a la señal BUSY, para la sincronización con la impresora, pero el usuario puede aprovecharlas según sus necesidades. Los pines activos son los siguientes:

pin 1 STROBE	pin 16 GND
pin 2 D0	pin 19 GND
pin 3 D1	pin 20 GND
pin 4 D2	pin 21 GND
pin 5 D3	pin 22 GND
pin 6 D4	pin 23 GND
pin 7 D5	pin 24 GND
pin 8 D6	pin 25 GND
pin 9 D7	pin 26 GND
pin 11 BUSY	pin 28 GND
pin 14 GND(masa)	pin 33 GND

Puerto de expansión

pin 1	Sonido	S	pin 26	D0	E/S
pin 2	GND (masa)		pin 27	Vcc (+5 volts.)	
pin 3	A15	E/S	pin 28	MREQ/	S
pin 4	A14	E/S	pin 29	M1/	S
pin 5	A13	E/S	pin 30	RFSH/	S
pin 6	A12	E/S	pin 31	IORQ/	S
pin 7	A11	E/S	pin 32	RD/	S

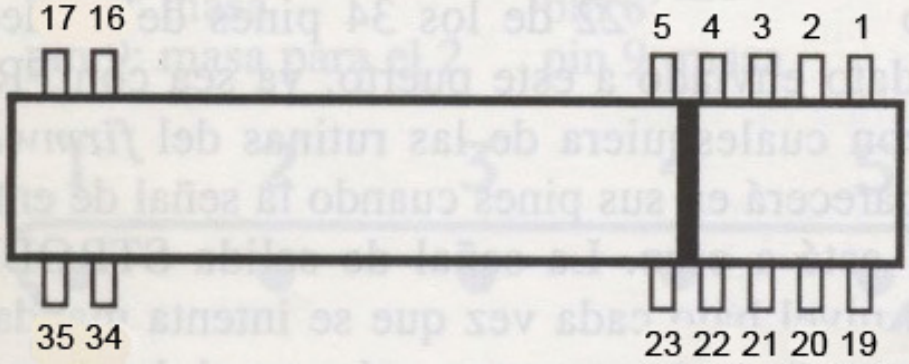

Figura 9.3

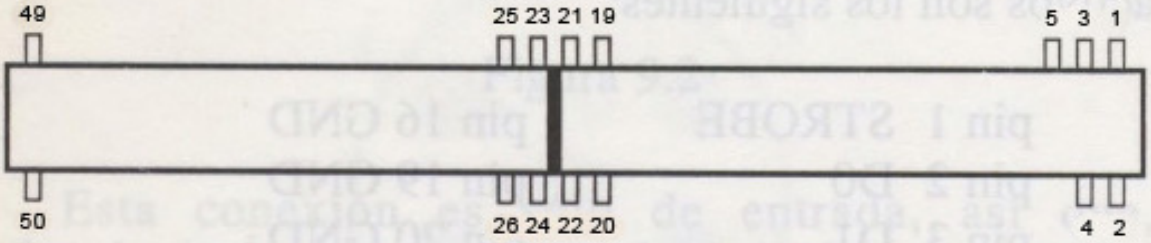

Figura 9.4

pin 8	A10	E/S	pin 33	WR/	S
pin 9	A9	E/S	pin 34	HALT/	S
pin 10	A8	E/S	pin 35	INT/	E
pin 11	A7	E/S	pin 36	NMI/	E
pin 12	A6	E/S	pin 37	BUSRD/	E

pin 13	A5	E/S	pin 38	BUSAK/	S
pin 14	A4	E/S	pin 39	READY	E
pin 15	A3	E/S	pin 40	BUS RESET/	E
pin 16	A2	E/S	pin 41	RESET/	S
pin 17	A1	E/S	pin 42	ROMEM/	S
pin 18	A0	E/S	pin 43	ROMDIS	E
pin 19	D7	E/S	pin 44	RAMRD/	S
pin 20	D6	E/S	pin 45	RAMDIS	E
pin 21	D5	E/S	pin 46	CURSOR	S
pin 22	D4	E/S	pin 47	Lápiz óptico	E
pin 23	D3	E/S	pin 48	EXP/	E/S
pin 24	D2	E/S	pin 49	GND	
pin 25	D1	E/S	pin 50	CLK (reloj)	S

E: Entrada
S: Salida

10

Bloques de control de cada subsistema

Espacio de trabajo del monitor de sonido

CPC6128 y 664		CPC464
B1EC	*Flags* para nuevas entradas	B550
B1ED	*Flags* para retención de canales activos	B551
B1EE	*Flags* para los canales activos	B552
B1EF	Contador del control de interrupciones	B553
B1F0	Contador de acciones pendientes	B554
B1F1-1F7	Bloque de sucesos	B555-5B
B1F8-236	*Buffer* del canal A	B55C-9A
B237-275	*Buffer* del canal B	B59B-D9
B276-2B4	*Buffer* del canal C	B5DA-618
B2B5	Byte de activación del GPS	B619
B2B6-3A5	Envolventes de volumen	B61A-709
B3A6-495	Envolventes de tono	B70A-F9

Formato del *buffer* de un canal relativo a la dirección del *buffer*

&00	Número de canal
&01	Bit del canal

&02	Bit de acordes del canal
&03	*Flags* de estado del canal
&04	Llamadas para el establecimiento del tono si el bit 0=1
&05	Contador 1 de interrupciones
&06	Contador 2 de interrupciones
&07	Llamadas para el establecimiento del volumen si el bit 0=1
&08-&09	Duración negada (complementada)
&0A-&0B	Dirección de la envolvente de volumen
&0C	Número de secciones de la envolvente de volumen
&0D-&0E	Dirección de la sección de la envolvente de volumen

Espacio de trabajo del monitor de teclado

B496-E5	Tabla 1: tecla MAY sin pulsar	B34C-9B
B4E6-535	Tabla 2: tecla MAY pulsada	B39C-EB
B536-85	Tabla 3: tecla CONTROL pulsada	B3EC-4B
B586-605	Tabla de control de repeticiones	B34C-49B
B590-627	*Buffer* del teclado	B446-DD
B628	Puntero del *buffer*	B4DE
B629	Señal de la cadena actual	B4DF
B62A	Devuelve carácter	B4E0
B62B-C	Comienzo del *buffer* de cadenas	B4E1-2
B62D-E	Final del *buffer* de cadenas	B4E3-4
B62F-30	Comienzo del espacio libre en el *buffer* de cadenas	B4E5-6
B631-2	Estado FIJA/MAY del teclado	B4E7-8
B633	Velocidad de repetición	B4E9
B634	Pausa o retardo de repetición	B4EA
B635-3E	Mapa 3	B4EB-F4
B63F-48	Mapa 1	B4F5-FE
B649-52	Mapa 2	B4FF-508
B653	Cuenta de repetición	B509
B654	Número del mapa de bytes	B50A
B655	Mapa de máscaras de bits	B50B
B656	*Flag* para permitir rupturas	B50C
B657-5D	Bloque de ruptura de sucesos	B50D-13
B65E-85	*Buffer* del teclado	B514-3B
B686	Espacio libre del *buffer* + 1	B53C

B687	Puntero de entrada	B53D
B688	Entradas del *buffer* + 1	B53E
B689	Puntero de salida	B53F
B68A	Entradas del *buffer*	B540
B68B-C	Puntero de la tabla 1 de tecla/código	B541-2
B68D-E	Puntero de la tabla 2 de tecla/código	B543-4
B68F-90	Puntero de la tabla 3 de tecla/código	B545-6
B691-2	Puntero de la tabla de repeticiones	B547-8

Bloque referente a la pantalla

B7C3	Modo	B1C8
B7C4-5	*Offset*	B1C9-A
B7C6	Base (byte alto)	B1CB
B7C7-6	Instrucción de saltos	B1CC-E
B7CA-D1	Máscaras de puntos	B1CF-D6
B7D2	Segundo tiempo de parpadeo	B1D7
B7D3	Primer tiempo de parpadeo	B1D8
B7D4-E4	Tabla 2 de colores	B1D9-E9
B7E5-F5	Tabla 1 de colores	B1EA-FA
B7F6	*Flag* de selección de tabla	B1FB
B7F7	Contador de parpadeos	B1FC
B7F8	Tiempo de cada color	B1FD
B7F9-801	Bloque de sucesos	B1FE-206
B802	Bits que forman el punto, negados	B207

Bloque referente a gráficos

B693-4	Coordenada X del origen	B328-9
B695-6	Coordenada Y del origen	B32A-B
B697-8	Posición X	B32C-D
B699-A	Posición Y	B32E-F
B69B-C	Izquierda de la ventana	B330-1
B69D-E	Derecha de la ventana	B332-3
B69F-A0	Borde superior de la ventana	B334-5
B6A1-2	Borde inferior de la ventana	B336-7
B6A3	Pluma codificada	B338
B6A4	Papel codificado	B339
B6A5-C	Copia de Matriz	B33A-41
B6AD-E	Coordenada X elegida	B342-3
B6AF-B0	Coordenada Y elegida	B344-5

B6B2	No hay	
B6B3	No hay	
B6B4	Modo de gráficos actual	No hay

VDU de texto

B6B5	Cauce actual	B20C
B6B4-C3	Datos del cauce 0	B20D-1B
B6C4-D1	Datos del cauce 1	B21C-2A
B6D2-F	Datos del cauce 2	B22B-39
B6E0-D	Datos del cauce 3	B23A-48
B6EE-FB	Datos del cauce 4	B249-57
B6FC-709	Datos del cauce 5	B258-66
B70A-17	Datos del cauce 6	B267-75
B718-25	Datos del cauce 7	B276-84
B726	Fila actual	B285
B727	Columna actual	B286
B728	Tipo de desplazamiento actual	B287
B729	Borde superior actual	B288
B72A	Borde izquierdo actual	B289
B72B	Borde inferior actual	B28A
B72C	Borde derecho actual	B28B
B72D	Cuenta de desplazamiento actual	B28C
B72E	*Flag* del cursor actual	B28D
B72E	*Flag* de activación del cursor y de la pantalla actual	B28E
B72F	Pluma actual	B28F
B730	Papel actual	B290
B731-2	Conexión para modo de impresión	B291-2
B733	*Flag* de escritura de gráficos	B293
B734	Código de la primera matriz de RAM	B294
B735	*Flag* de la matriz	B295
B736-7	Dirección de la matriz de la RAM	B296-7
B738-57	Modelo elegido	B298-B7
B758	Contador de parámetros	B2B8
B759	Código de control	B2B9
B75A-62	Parámetros de control	B2BA-C2
B763-C2	Tabla de saltos de control	B2C3-322

Area de datos del núcleo

B82D-E	Base de la lista de inatendidos	B100-1
B82F-30	Fin de la lista de inatendidos	B102-3
B831	Byte de *flag*	B104
B832-3	Contiene el SP (puntero de pila)	B105-6
B834-B3	Pila especial	B107-86
B8B4-8	Contador de tiempo	B187-D
B8B9-A	Lista de sucesos asociados barridos	B18C-D
B8BB-C	Lista de temporización rápida	B18E-F
B8BD-E	Lista de temporización lenta	B190-1
B8BF	Contador de temporización lenta	B192
B8C0-1	Base de lista de síncronos	B193-4
B8C2	Prioridad actual	B195
B8C3-D2	Copia de las palabras-órdenes	B196-A5
B8D3-4	Base de la cadena de órdenes	B1A6-7
B8D6	ROM actual	B1A8
B8D7-8	Dirección lejana característica	B1A9-B
B8D9-E5	Punteros del área de datos	B1AC-B8

Tabla de caracteres de control

0.	14EB	Regreso inmediato. Sin acción	14E2
1.	1335	TXT escribe carácter	1334
2.	139B	TXT inhibe cursor usuario	139A
3.	1286	TXT activa cursor usuario	1289
4.	0AE9	PNT fija modo	0ACA
5.	1940	GRA escribe carácter	1945
6.	1459	TXT activa VDU	1451
7.	14E1	COLA SONIDO con HL=14D8 (pitido)	14D8
8.	1519	Cursor izquierda	150A
9.	151E	Cursor derecha	150F
10.	1523	Cursor abajo	1514
11.	1528	Cursor arriba	1519
12.	154F	TXT limpia ventana	1540
13.	153F	Columna=Borde izquierdo ventana	1530
14.	12AB	TXT color papel	12AE
15.	12A6	TXT color pluma	12A9
16.	155E	Borrar	154F

17.	1599	Limpia ventana a la izquierda del cursor	158E
18.	158F	Limpia ventana a derecha del cursor	1584
19.	1578	Limpia ventana encima del cursor	156D
20.	1565	Limpia ventana debajo del cursor	1556
21.	1452	TXT desactiva VDU	144B
22.	14EC	TXT fija modo	14E3
23.	0C55	PNT modo gráficos	0C49
24.	12C6	TXT invierte colores	12C9
25.	150D	TXT impone matriz	1504
26.	1501	TXT activa ventana	14F8
27.	14FB	Regreso inmediato sin acción	14E2
28.	14F1	PNT impone tinta	14E8
29.	14FA	PNT impone borde	14F1
30.	1539	Cursor a esquina superior izquierda	152A
31.	1543	posiciona cursor	1538

11

Ficheros y programas

Un fichero puede ser secuencial o de acceso aleatorio. Este último tipo sólo es posible con unidad de disco y, además, sólo sirve para almacenamiento de datos. Otra forma de clasificar los ficheros es según el contenido del mismo. Así, tenemos ficheros binarios, ASCII, BASIC, de sólo datos, etc. Su manejo puede hacerse a través de un sistema operativo desde el BASIC, o incluso desde código máquina. Veamos las diferentes posibilidades.

Ficheros desde BASIC

Las órdenes SAVE y LOAD son recomendables para el almacenamiento de programas en general y, en ciertos casos, para datos. Ya sea para disco o *cassette,* el formato de estas órdenes es:

```
SAVE "nombre",tipo,dir-ini,long,ejec
LOAD "nombre",dir-carga
```

Si se trata de un programa en BASIC únicamente es necesario el nombre; si se quiere, se puede añadir "P", como tipo en SAVE, para indicar que el programa debe ser protegido.

Los ficheros ASCII y códigos binarios en general, necesitan todos los parámetros menos "ejec", siendo: tipo = B para binarios y tipo = A para ficheros ASCII; dir-ini es la dirección inicial a partir de la que están los datos en memoria; long es la longitud en bytes del fichero; dir-carga es la dirección donde se cargará el fichero desde la unidad de almacenamiento. Opcionalmente se puede incluir el parámetro "ejec", que indica la dirección de comienzo de ejecución del programa, lo cual indica que sólo es válido para programas en binario. Si se utiliza este parámetro, la única forma de hacer funcionar correctamente el programa es cargándolo mediante la orden

```
RUN "nombre",dir-carga
```

Las instrucciones OPEN y CLOSE, para ficheros de entrada (IN) o de salida (OUT), trabajan exclusivamente con datos, ya sean números o cadenas literales. Son estrictamente secuenciales. Esto quiere decir que un dato introducido en el fichero en una posición n, sólo se recuperará después de leer los n-1 datos anteriores. Los datos pueden tener la longitud que se desee, pero siempre han de ser números o cadenas de caracteres ASCII.

Las órdenes OPENIN y OPENOUT irán siempre acompañadas del nombre del fichero entre comillas. No puede haber nunca más de un fichero abierto. Si es un fichero de salida se grabará cada vez que los datos ocupen un bloque de 2 Kbytes. Si el fichero es de entrada, primero se carga en memoria el primer bloque de 2 Kbytes y sólo cuando el programa termine de leer todos sus datos, se cargará el siguiente bloque. Las órdenes CLOSEIN y CLOSEOUT, respectivamente, harán que cese la carga de bloques de ese fichero y que se grabe el bloque de datos, aunque se encuentre incompleto.

La forma de guardar o recuperar los datos de un fichero, es escribir o leer a través del cauce número nueve. Así,

```
FOR I=1 TO 10:PRINT #9, A(I):NEXT
```

escribirá secuencialmente en el fichero el contenido de las variables A(1), A(2),..., A(10). Por otra parte, mediante la línea

```
FOR K=1 TO 5:INPUT #9,M$(I):NEXT
```

podremos leer del fichero las cadenas literales M$(1), M$(2),..., M$(5). Es muy importante tener en cuenta a la hora de escribir en el fichero que las cadenas literales o de caracteres ASCII deben estar aisladas unas de otras. Por ejemplo, no debe escribir una línea de programa de la forma:

```
PRINT #9,M$;C$;A$
```

ya que en el momento de leer el fichero, encontraríamos una sola cadena literal formada por M$, C$ y A$. Por tanto, debe evitarse el uso del punto y coma al escribir cadenas ASCII. Es preferible programar tantos PRINT como cadenas a guardar.

Ficheros desde código máquina

Desde el lenguaje máquina prácticamente no tiene sentido hablar de ficheros de datos, pero sí de ficheros de bytes. A pesar de esto, sigue siendo posible definir un fichero como ASCII como programa BASIC, aunque no lo sea, o como realmente queramos definirlo. Por medio de las

rutinas del *firmware*, podemos cargar en la memoria el encabezamiento de un fichero y conocer, leyendo en los sitios adecuados, de qué tipo de fichero se trata, su longitud y todos los demás datos de interés. Asimismo, podemos guardar parte del contenido de la memoria como fichero con sólo indicar el nombre y las direcciones donde se encuentre.

La información completa sobre cada una de las rutinas encargadas del manejo de la unidad de almacenamiento se encuentra en el capítulo 7. Utilizando *cassette*, consultar el intervalo de direcciones BC65 a BCA4; para la unidad de disco, sólo son válidas las rutinas comprendidas en el intervalo BC77 a BC9B. En cualesquiera de los dos casos, es de gran utilidad conocer la disposición del bloque de control de ficheros. Es posible cambiar en cualquier momento el contenido de este bloque, incluso desde el BASIC. De esta forma podremos cargar programas o ficheros sin las rígidas normas impuestas desde el BASIC. A continuación se hace un listado de las direcciones mas importantes que contienen la información sobre el estado de los ficheros.

Espacio de trabajo para cassette/disco

CPC6128 y 664		CPC464
B118	*Flag* para mensajes (0-sí 1-no)	B800
B119	Contador de columnas de pantalla (para los mensajes)	B801

Bloque de ficheros de entrada

B11A	Estado del fichero	B802
B11B/C	Dirección del *buffer*	B803/4
B11D/E	Puntero del *buffer*	B805/6

B11F/2E	Nombre del fichero	B807/16
B12F	Número de bloques	B817
B130	*Flag* para EOF (final del fichero)	B818
B131	Tipo de fichero	B819
B132/3	Bytes en el *buffer*	B81A/B
B134/5	Dirección de escritura de datos	B81C/D
B136	*Flag* de primer bloque	B81E
B137/8	Longitud de datos	B81F/20
B139/A	Dirección de ejecución	B821/2
B13B/5D	Sin definir	B823/46

Bloque de ficheros de salida

B15F	Estado del fichero	B847
B160/1	Dirección del *buffer*	B848/9
B162/3	Puntero del *buffer*	B84A/B
B164/73	Nombre del fichero	B84C/5B
B174	Numero de bloques	B85C
B175	Flag para EOF (final del fichero)	B85D
B176	Tipo de fichero	B85E
B177/8	Bytes en el *buffer*	B85F/60
B179/A	Dirección de lectura de datos	B861/2
B17B	*Flag* de primer bloque	B863
B17C/D	Longitud de datos	B864/5
B17E/F	Dirección de comienzo de ejecución	B866/7
B180/A3	Sin definir	B868/8B

Copia del encabezamiento

B1A4/B3	Nombre del fichero	B8CC/9B
B1B4	Número de bloque	B89C
B1B5	*Flag* de último bloque	B89D
B1B6	Tipo de fichero	B89E
B1B7/8	Longitud de datos	B89F/A0
B1B9/A	Dirección de los datos	B8A1/2
B1BB	*Flag* de primer bloque	B8A3
B1BC/D	Longitud total de datos	B8A4/5
B1BE/F	Dirección de comienzo de ejecución	B8A6/7
B1C0/E3	Sin definir	B8A8/CB

General

B1E4	*Flag* para mensajes de ayuda	B8CC
B1E5	Carácter de sincronismo	B8CD
B1E6/7	Temporización	B8CD/F
B1E8	Temporización	B8D0
B1E9	Precompensación	B8D1
B1EA	Velocidad	B8D2
B1EB/C	Temporización	B8D3/4

Encabezamiento de fichero en disco

A754/5	Dirección del *buffer*
A756	Número de usuario
A757/5E	Nombre de fichero
A75F/61	Identificador de tipo
A762/67	A cero
A768	Tipo de fichero
A769/6A	Dirección del *buffer*
A76B/C	Dirección de comienzo
A76D	*Flag* de primer bloque
A76E/6F	Longitud del fichero

El byte "tipo de fichero" se interpreta de la siguiente manera:

bit 0	1:	Protegido; 0: sin proteger
bits 3-1	000:	BASIC
	001:	Binario
	010:	Volcado de pantalla
	011:	ASCII (de CP/M)
bits 7-4	0000:	No es ASCII
	0001:	ASCII

Las demás combinaciones no están definidas y pueden emplearse a discreción por el usuario.

12

Códigos ASCII

Carácter	Códigos ASCII		
	Octal	Hexadecimal	Decimal
NUL (CTRL @)	0	0	0
SOH (CTRL A)	1	1	1
STX (CTRL B)	2	2	2
ETX (CTRL C)	3	3	3
EOT (CTRL D)	4	4	4
ENQ (CTRL E)	5	5	5
ACK (CTRL F)	6	6	6
BEL (CTRL G)	7	7	7
BS (CTRL H)	8	8	10
HT (CTRL I)	9	9	11
LF (CTRL J)	A	10	12
VT (CTRL K)	B	11	13
FF (CTRL L)	C	12	14
CR (CTRL M)	D	13	15
SO (CTRL N)	E	14	16
SI (CTRL O)	F	15	17
DLE (CTRL P)	10	16	20
DC1 (CTRL Q)	11	17	21
DC2 (CTRL R)	12	18	22
DC3 (CTRL S)	13	19	23
DC4 (CTRL T)	14	20	24
NAK (CTRL U)	15	21	25

SYN (CTRL V)	16	22	26
ETB (CTRL W)	17	23	27
CAN (CTRL X)	18	24	30
EM (CTRL Y)	19	25	31
SUB (CTRL Z)	1A	26	32
ESC	1B	27	33
FS	1C	28	34
GS	1D	29	35
RS	1E	30	36
US	1F	31	37
(espacio)	20	32	40
!	21	33	41
"	22	34	42
#	23	35	43
$	24	36	44
%	25	37	45
&	26	38	46
'	27	39	47
(	28	40	50
)	29	41	51
*	2A	42	52
+	2B	43	53
,	2C	44	54
-	2D	45	55
.	2E	46	56
/	2F	47	57
0	30	48	60
1	31	49	61
2	32	50	62
3	33	51	63
4	34	52	64
5	35	53	65
6	36	54	66
7	37	55	67
8	38	56	70
9	39	57	71
:	3A	58	72
;	3B	59	73

<	3C	60	74
=	3D	61	75
>	3E	62	76
?	3F	63	77
@	40	64	100
A	41	65	101
B	42	66	102
C	43	67	103
D	44	68	104
E	45	69	105
F	46	70	106
G	47	71	107
H	48	72	110
I	49	73	111
J	4A	74	112
K	4B	75	113
L	4C	76	114
M	4D	77	115
N	4E	78	116
O	4F	79	117
P	50	80	120
Q	51	81	121
R	52	82	122
S	53	83	123
T	54	84	124
U	55	85	125
V	56	86	126
W	57	87	127
X	58	88	130
Y	59	89	131
Z	5A	90	132
[	5B	91	133
\	5C	92	134
]	5D	93	135
^	5E	94	136
_	5F	95	137
`	60	96	140
a	61	97	141

b	62	98	142
c	63	99	143
d	64	100	144
e	65	101	145
f	66	102	146
g	67	103	147
h	68	104	150
i	69	105	151
j	6A	106	152
k	6B	107	153
l	6C	108	154
m	6D	109	155
n	6E	110	156
o	6F	111	157
p	70	112	160
q	71	113	161
r	72	114	162
s	73	115	163
t	74	116	164
u	75	117	165
v	76	118	166
w	77	119	167
x	78	120	170
y	79	121	171
z	7A	122	172
{	7B	123	173
|	7C	124	174
}	7D	125	175
	7E	126	176
DEL (BORR)	7F	127	177

13

Códigos de control

Código	*Efecto*
0	Sin efecto.
1	Expone el símbolo asociado al valor de un parámetro comprendido entre 0 y 255. Permite mostrar los símbolos asociados a los caracteres de la gama 0 a 31.
2	Vuelve invisible el cursor de texto.
3	Vuelve visible el cursor de texto.
4	Un parámetro que será el modo de pantalla.
5	Parámetro entre 0 y 225, correspondiente al código ASCII del carácter a exponer.
6	Activa la pantalla de texto.
7	Produce un pitido.
8	Retrocede el cursor una posición.
9	Avanza el cursor una posición.
10	Mueve el cursor una línea hacia abajo.
11	Mueve el cursor una línea hacia arriba.
12	Limpia la ventana de texto.
13	Desplaza el cursor al borde izquierdo de la línea en que se encuentre.
14	Un parámetro (número de tinta del papel).
15	Un parámetro (número de tinta de la pluma).

16 Borra el carácter cubierto por el cursor de texto.

17 Limpia la ventana desde la posición actual del cursor hasta el borde izquierdo.

18 Limpia la ventana desde la posición actual del cursor hasta el borde derecho.

19 Limpia la ventana desde el principio hasta la posición actual del cursor.

20 Limpia la ventana desde la posición actual del cursor hasta el final.

21 Desactiva la pantalla de texto.

22 Un parámetro; 1 activa la opción de transparencia y 0 la desactiva.

23 Un parámetro que fija el modo de gráficos:

1 modo XOR
2 modo AND
3 modo OR
0 modo absoluto o forzado.

24 Intercambia la tinta del papel y pluma.

25 Define un carácter con 9 parámetros. El primero es el código ASCII del símbolo a definir. El resto son las definiciones de cada una de las líneas del carácter.

26 Define una ventana. Tiene cuatro parámetros. Los dos primeros, especifican los bordes izquierdo y derecho de la ventana; los otros dos son las filas superior e inferior de la ventana.

27 Sin efecto.

28 Tres parámetros. Elige como tinta una pareja de colores.

29 Dos parámetros que especifican los dos colores.

30 Desplaza el cursor a la esquina superior izquierda de la ventana.

31 Dos parámetros que sitúan el cursor en la posición X,Y.

14

Tabla de colores

Nº tinta	*Color*	*Nº tinta*	*Color*
0	Negro	14	Azul pastel
1	Azul	15	Naranja
2	Azul brillante	16	Rosa
3	Rojo	17	Magenta pastel
4	Magenta	18	Verde brillante
5	Malva	19	Verde marino
6	Rojo brillante	20	Cyan brillante
7	Púrpura	21	Verde lima
8	Magenta brillante	22	Verde pastel
9	Verde	23	Cyan pastel
10	Cyan	24	Amarillo brillante
11	Azul cielo	25	Amarillo pastel
12	Amarillo	26	Blanco brillante
13	Blanco		

15

Sonidos en Amstrad

El Generador Programable de Sonido del sistema (GPS), es el circuito integrado AY-3-8912 de General Instruments; y es el encargado de producir toda la gama de sonidos y ruidos disponibles en el sistema.

Veamos las dos formas posibles de sacar el máximo partido a este dispositivo.

Sonidos desde el BASIC

El formato típico de la orden SOUND es el siguiente:

SOUND canal,per,dur,vol,evol,eton,ruido

siendo opcionales los cuatro últimos parámetros. Veamos las posibilidades de cada uno de estos parámetros.

• Canal es uno o varios de los tres canales disponibles. Su valor estará comprendido entre 1 y 255. La interpretación es:

1	canal A
2	canal B
4	canal C
8	sincronización con el canal A
16	sincronización con el canal B
32	sincronización con el canal C
64	detención del sonido
128	cancela los sonidos

Por ejemplo, la combinación: 49=1+16+32 significa que la nota se envía al canal A, pero simultáneamente pueden sonar las notas de los canales B y C.

• Per, es el periodo de la nota. Puede tomar valores entre 0 y 4095. La frecuencia emitida se calcula mediante la expresión

frecuencia=125000 / per

• Dur, indica la duración del sonido, medido en centésimas de segundo. Puede ser negativo o estar en la gama 0-32767. El valor cero especifica que la duración viene controlada por la envolvente de volumen que veremos después. Un valor negativo indicará, en valor absoluto, el número de veces que se ha de repetir la envolvente.

• Vol, es el volumen de la nota; pudiendo tomar los valores de 0 a 15 con envolvente, y de 0 a 7 sin ella. Por defecto, se tomarán los valores 12 y 7, respectivamente.

• Evol, indica el número de una envolvente de volumen, previamente definida. Valores entre 0 y 15.

• Eton, indica el número de una envolvente de tono, previamente definida. Valores entre 0 y 15.

• Ruido, es el parámetro que regula la frecuencia del ruido que se suma a la nota emitida. Es el mismo para los tres canales. Por defecto se toma el valor 0.

La función ENV sirve para señalar la forma de la envolvente de volumen, esto es, la secuencia de volumen de la nota a lo largo de su emisión. Su formato es :

ENV,num,n1,a1,t1,n2,a2,t2,n3,a3,t3,n4,a4,t4,n5,a5,t5

siendo "num", el número de envolvente (de 0 a 15). La envolvente viene definida por una serie de secciones, cinco como máximo, definida cada una de ellas por tres parámetros. Estos son:

- n, que indica el número de pasos que tiene la sección. Valores entre 0 y 127.
- a, que indica cuanto aumenta o disminuye el volumen en cada paso. El volume n asignado a la nota se divide entre 127. Este parámetro especifica cuantas fracciones se toman a cada paso. Valores entre -128 y 127.
- t, que indica el número de centésimas de segundo de duración de cada paso. Valores entre 0 y 127.

La función ENT sirve para señalar la forma de la envolvente de tono, es decir, la secuencia de variaciones de la frecuencia de la nota a lo largo de su duración. Su formato es:

ENV,num,n1,a1,t1,n2,a2,t2,n3,a3,t3,n4,a4,t4,n5,a5,t5

siendo "num", el número de envolvente (de 0 a 15). La envolvente viene definida por una serie de secciones, cinco como máximo, definida cada una de ellas por tres parámetros. Estos son:

- n, que indica el número de pasos que tiene la sección. Valores entre 0 y 239.
- a, que indica cuánto se desvía la frecuencia en cada paso. El período asignado a la nota se divide entre 127. Este parámetro especifica cuántas fracciones se toman a cada paso. Valores entre -128 y 127.
- t, que indica el número de centésimas de segundo de duración de cada paso. Valores entre 0 y 255.

Sonidos desde código máquina

La programación directa del Generador Programable de Sonido implica tener ciertos conocimientos previos sobre sus registros. Sin embargo, su funcionamiento se puede resumir en pocas líneas.

El GPS tiene 16 registros, programables en el Amstrad mediante una rutina del *firmware* situada en la dirección &BD34 de la memoria. Para acceder a ella, el registro C del Z-80 debe contener un valor entre 0 y 255, que es el que se introducirá en el registro elegido. El registro A contendrá el número de ese registro.

Los registros del GPS se numeran del 0 al 15. Debido a que los registros 14 y 15 no se emplean para la generación de sonido, es muy recomendable olvidarse de ellos por el momento. Veamos la función del resto:

Registros 0 a 5: Son los que controlan la frecuencia de los sonidos emitidos. El 0 y el 1 se encargan del canal A; el 2 y 3, del canal B y el 4 y 5, del canal C. Los registros de número más bajo de cada canal son de 8 bits, y contienen la parte menos significativa del contenido total. Variando su contenido, provocaremos

variaciones pequeñas del tono emitido. El otro registro asignado a cada canal tiene cuatro bits y es el control grueso del tono; pequeñas variaciones en él provocan grandes variaciones del tono emitido. Cuanto mayor sea el contenido de estos registros, más grave será la nota generada.

Registro 6: Se encarga de controlar el ruido que acompaña a las notas producidas. Tiene 5 bits, es decir, admite valores entre 0 y 31. Según aumente el valor contenido en este registro, más grave será el ruido emitido. Este registro es único para los tres canales, ya que no es posible programar un ruido diferente para cada canal. Su volumen será el mismo que el del tono del canal correspondiente, por lo que no es necesario especificarlo.

Registro 7: Formado por 8 bits, controla la activación o desactivación de los canales y si éstos van a generar tonos puros, ruido, o ambas cosas a la vez. La siguiente figura muestra la función de cada uno de los bits.

Los bits 6 y 7 están encargados de la activación del teclado y, si no quieres que éste se quede inactivo, es recomendable dejarlos continuamente a cero. Un bit puesto a cero significa activación; puesto a uno, significa desactivación.

Registros 8 a 10: Todos tienen 5 bits y controlan el volumen de los tonos generados por cada canal. El registro 8 es asignado al canal A; el 9 al canal B y el 10, al canal C. Si el valor que tiene uno de ellos es mayor de 15, entonces la amplitud del sonido de su canal está gobernada por una de las envolventes *hardware* del sistema. Un valor entre 0 y 15, indica

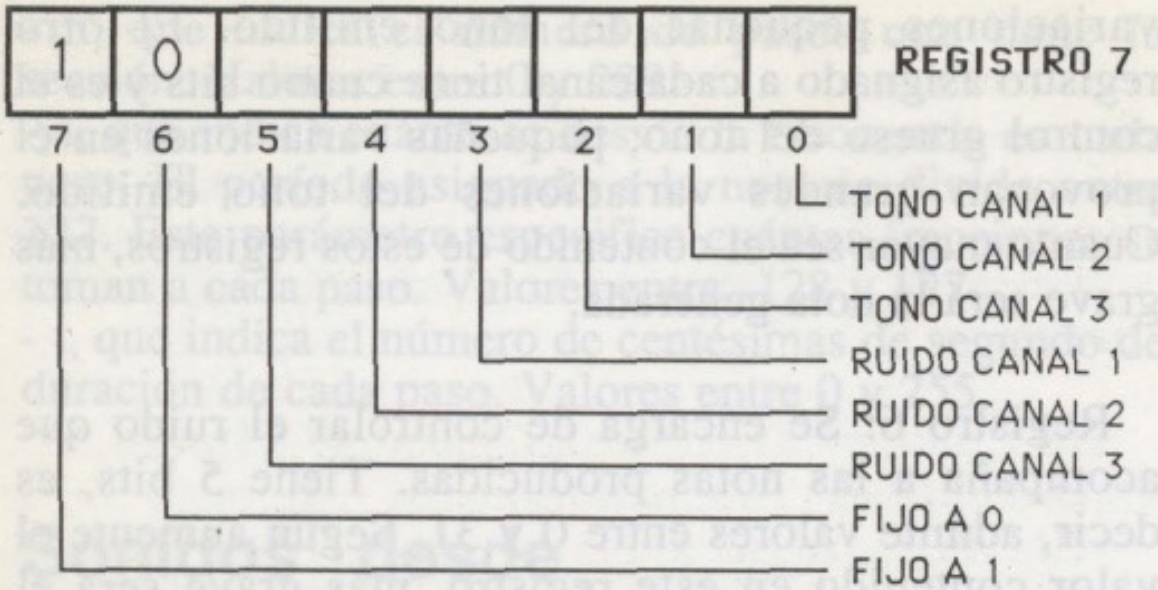

Figura 15.1

que el volumen es proporcional a ese mismo valor; siendo 15 el máximo y 0 el silencio.

Registros 11 y 12: Controlan el período de la envolvente de volumen, P-E (véase la figura 15.2).

Ambos registros son de 8 bits, siendo R12 el byte más significativo. La duración de la envolvente será mayor según aumenta el valor de estos registros, es decir, a mayor período, más duración.

Registro 13: Su contenido selecciona la forma de la envolvente de volumen. Como se indicó hablando de los registros 8 a 10, el volumen del tono emitido estará gobernado por una envolvente, únicamente si el bit 4 está puesto a uno.

Los números y formas de las envolventes, son indicados en la figura 15.2.

Se observa que no existe ningún registro encargado de controlar envolventes de tono. En efecto, la única forma de conseguir este efecto es mediante *software;* tal es el modo en que trabaja la función ENT del BASIC.

La siguiente rutina en BASIC facilita la programación del GPS:

```
10 CLS:FOR T=35000 TO 35009:READ A$
20 POKE T,VAL("&"+A$):NEXT
30 DATA DD,7E,02,DD,4E,00,CD,34,BD,C9
40 INPUT "Registro ",R:INPUT "Contenido ",C
50 CALL 35000,R,C:GOTO 40
```

Valores del período de las notas

OCTAVA —3

Nota	Período
DO	3822
DO#	3608
RE	3405
RE#	3214
MI	3034
FA	2863
FA#	2703
SOL	2551
SOL#	2408
LA	2273
LA#	2145
SI	2025

OCTAVA —2

Nota	Período
DO	1911
DO#	1804
RE	1703
RE#	1607
MI	1517
FA	1432
FA#	1351
SOL	1276
SOL#	1204
LA	1136
LA#	1073
SI	1012

OCTAVA —1

Nota	Período
DO	956
DO#	902

OCTAVA 0

Nota	Período
DO	478
DO#	451

RE	851	RE	426
RE#	804	RE#	402
MI	758	MI	379
FA	716	FA	308
FA#	676	FA#	338
SOL	638	SOL	319
SOL#	602	SOL#	301
LA	568	LA	284
LA#	536	LA#	268
SI	506	SI	253

OCTAVA 1

Nota	Período
DO	239
DO#	225
RE	213
RE#	201
MI	190
FA	179
FA#	169
SOL	159
SOL#	150
LA	142
LA#	134
SI	127

OCTAVA 2

Nota	Período
DO	119
DO#	113
RE	106
RE#	100
MI	95
FA	89
FA#	84
SOL	80
SOL#	75
LA	71
LA#	67
SI	63

OCTAVA 3

Nota	Período
DO	60
DO#	56
RE	53
RE#	50
MI	47
FA	45
FA#	42
SOL	40
SOL#	38

OCTAVA 4

Nota	Período
DO	30
DO#	28
RE	27
RE#	25
MI	24
FA	22
FA#	21
SOL	20
SOL#	19

LA	36	LA	18
LA#	34	LA#	17
SI	32	SI	16

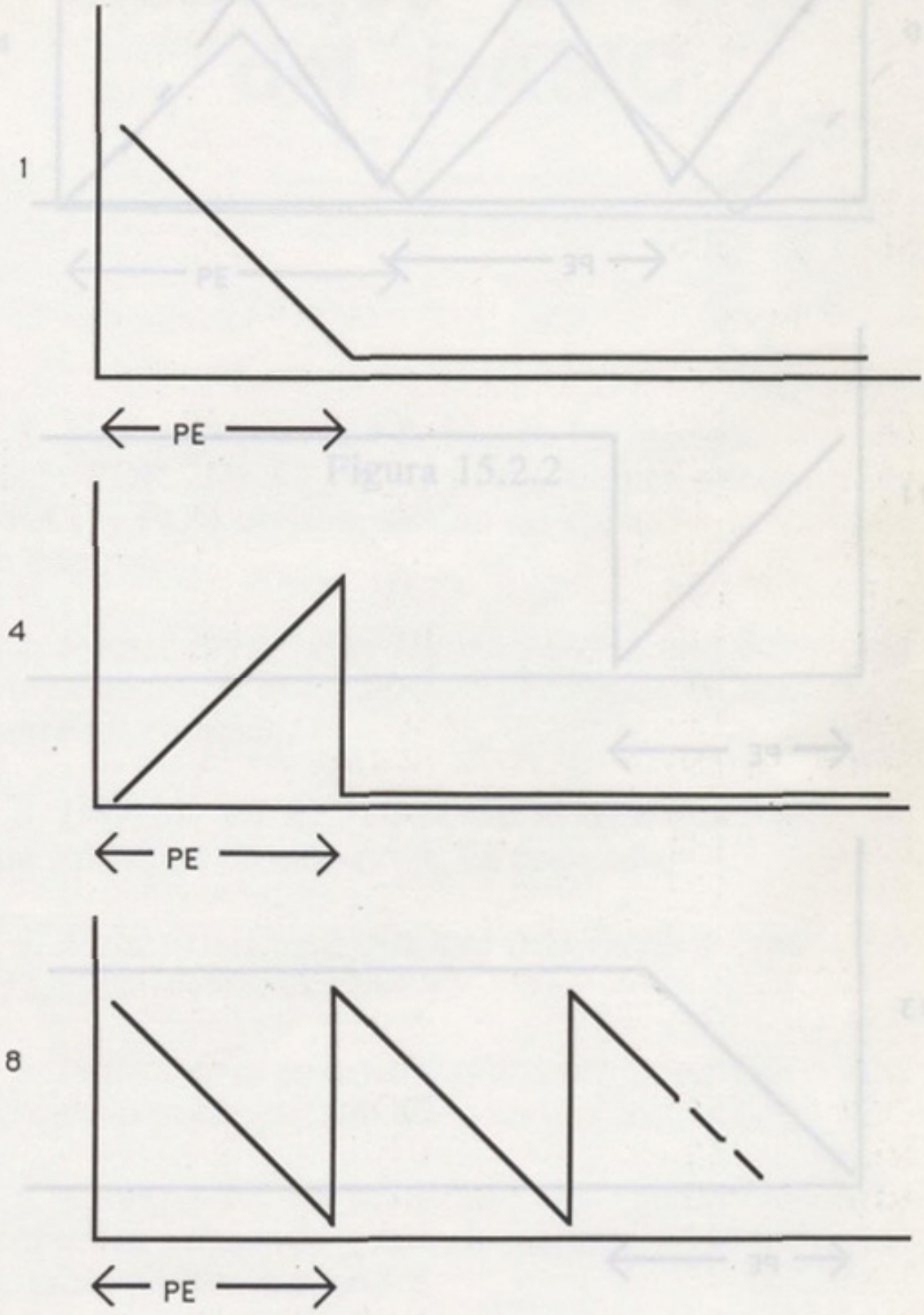

Figura 15.2

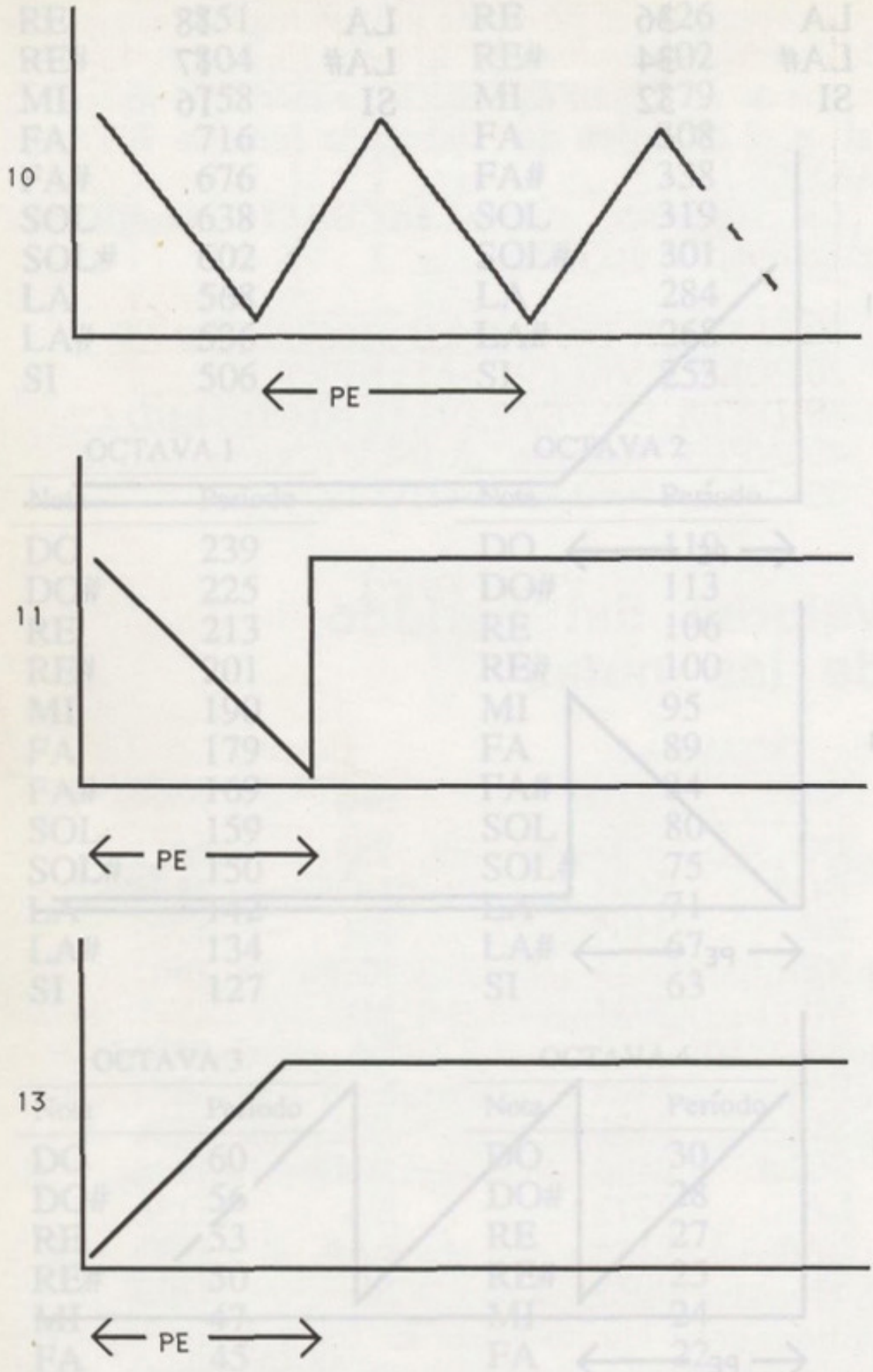

Figura 15.2.1

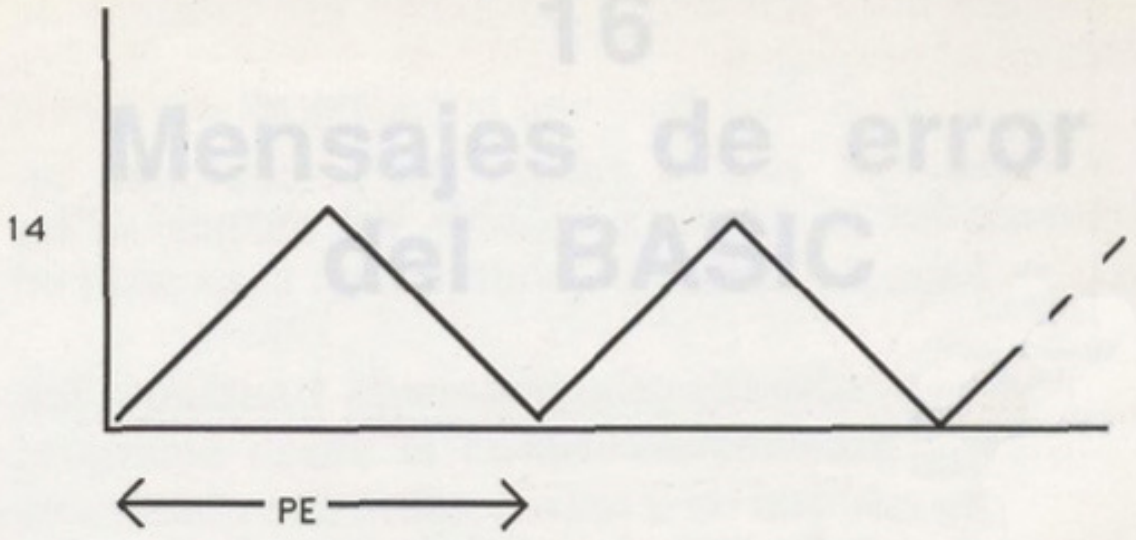

Figura 15.2.2

16
Mensajes de error del BASIC

1. *Unexpected NEXT*. No se ha iniciado ningún bucle FOR...NEXT, o las instrucciones encontradas NEXT y FOR no concuerdan en cuanto a la variable de control.

2. *Syntax Error*. Hay alguna palabra mal deletreada o inexistente, o el número de paréntesis de apertura y cierre no coincide.

3. *Unexpected RETURN*. Se le pide que vuelva de una subrutina cuando no se ha desviado.

4. *Data exhausted*. No hay más datos o constantes en las instrucciones DATA.

5. *Improper argument*. Argumento impropio de una función o parámetro inválido en una instrucción.

6. *Overflow*. El resultado de una operación aritmética rebasa la gama de valores permitida a la variable (entera o real).

7. *Memory full*. La capacidad de memoria se ha cubierto.

8. *Line does not exist.* No existe ese número de línea en el programa.

9. *Subscript out of range.* Al seleccionar un elemento de una matriz, uno de los sufijos se ha elegido fuera de gama.

10. *Array already dimensioned.* La matriz en cuestión ya está dimensionada.

11. *Division by cero.* Los valores propuestos dan una división entre cero.

12. *Invalid direct command.* Esa palabra clave sólo se admite como instrucción, no como orden directa.

13. *Type mismatch.* Se da como valor de una variable literal una constante numérica o viceversa.

14. *String space full.* Se ha cubierto el espacio interno reservado para cadenas de caracteres.

15. *String too long.* La cadena sobrepasa los 255 caracteres.

16. *String expression too complex.* Las expresiones literales son demasiado complejas, excediendo de un límite razonable los resultados.

17. *Cannot CONTinue.* Detectado un error y corregida la instrucción no se puede continuar el programa en el punto en que se interrumpió.

18. *Unknown user function.* No conoce esa instrucción definible por el usuario; debe ser avisado previamente para su utilización.

19. *RESUME missing.* Falta esta instrucción para que se reanude la ejecución en algún punto del programa, después que éste haya tratado un error.

20. *Unexpected RESUME.* Si no ha habido error no tiene nada que reanudar.

21. *Direct command found.* Tras cargar un programa desde la unidad de almacenamiento, ha encontrado una orden directa y no una instrucción.

22. *Operand missing.* En la expresión falta algún operando.

23. *Line too long.* Línea demasiado larga.

24. *EOF met.* Tomando información de la unidad de almacenamiento, se ha llegado al final del fichero.

25. *File type error.* Error en el tipo de fichero (cuidado con las órdenes OPENIN, LOAD y RUN).

26. *NEXT missing.* Estando dentro de un bucle no encuentra la instrucción que le manda hacer otra ronda.

27. *File already open.* El fichero ya está abierto.

28. *Unknown command.* Orden directa desconocida.

29. *WEND missing.* No encuentra la instrucción WEND que corresponde a un WHILE anterior en el programa.

30. *Unexpected WEND*. Encontró una instrucción WEND sin estar dentro de un bucle WHILE.

31. *Unknown error*. Este mensaje aparece ante todos los errores cuyo valor ERR sea superior o igual a 31.

Apéndice

Rutina de recuperación de ficheros

Los programas y ficheros almacenados en disco se guardan consecutivamente, aunque esto no sea transparente para el usuario. El directorio de un disco contiene todos los datos referentes a cada fichero (nombre, número de usuario, longitud del fichero, pistas y sectores donde se encuentra, etc.).

Con el formato usual de CP/M, el disco queda conformado así:

- 40 pistas por cara, numeradas de 0 a 39
- 9 sectores de 512 bytes por pista, numerados de 65 a 74
- 2 pistas reservadas: 0 y 1

Las pistas 0 y 1 contienen los datos necesarios para el CP/M. El directorio se encuentra en la pista 2, sectores 65 y 66. El número máximo de ficheros para un directorio es de 64. El resto del disco permanece libre para programas y ficheros.

Cuando se ejecuta la orden ERA del sistema de disco, los datos no son alterados hasta que se vuelva a guardar un nuevo fichero, pero sí el directorio. Incluso el nombre del fichero permanece intacto; lo

único que se varía es el número que precede al nombre, poniéndolo fuera de un margen legal. Es posible, por tanto, modificar el directorio de forma que el fichero borrado pueda volver a formar parte de él. Se presenta aquí una pequeña rutina en BASIC, que se encarga de buscar el nombre del fichero perdido y devolverlo al directorio, recuperándose así todo el contenido del fichero. Tal vez sea una rutina muy lenta, pero es segura. Es necesario ejecutarla desde AMSDOS, para lo cual habrá que llamar desde el CP/M al sistema AMSDOS, y una vez allí escribir el programa o cargarlo en memoria si ya estaba grabado.

```
10  CLS:PRINT "Escribe el nombre del fichero
    perdido:";
20  LINE INPUT A$:A$=UPPER$(A$)
30  P=INSTR(1,A$,"."):N$=LEFT$(A$,P—1):
    S$=RIGHT$(A$,LEN(A$)-P)
40  N$=N$+STRING$(8—LEN(N$),32):
    S$=S$+STRING$(3-LEN(S$),32)
50  A$=N$+S$:SEC=65
60  FOR T=15000 TO 15031:READ WS$:POKE
    T,VAL("&"+WS$):NEXT
70  DATA 0E,07,CD,0F,B9,C5,CD,00,B9,1E,
    00,21,20,4E,DD,56,02,DD,4E,00,CD,89,
    BE,21,08,52,77,C1,CD,18,B9,C9
80  CALL 15000,2,SEC
90  GOSUB 130
100 SEC=SEC+1:CALL 15000,2,SEC
110 GOSUB 130
120 PRINT "NOMBRE ERRONEO O DESAPA
    RECIDO":PRINT:END
130 FOR T=20000 TO 20500
140 G=0:FOR H=T TO T+10
150 IF PEEK(H) = ASC(MID$(A$,H—T+1,1)
    THEN G=G+1
160 NEXT
```

```
170  IF G=11 THEN POKE T—1,0:POKE
     15021,140:GOTO 200
180  NEXT
190  RETURN
200  CALL 15000,2,SEC:END
```