LA PRIMERA REVISTA ESPAÑOLA DE ORDENADORES PERSONALES

EL ORDENADOR PERSONAL

la revista informática para todos

N.° 39 Especial 1985 350 Pts.

APPLE,ATOM-ACORN,CASIO FX-702-P,COMMODORE-64 VIC-20,DRAGON,HP-41,NEW BRAIN, ORIC,OSBORNE,PC-1500,SHARP MZ-80-B,SHARP PC-1211,SPECTRUM,TI-59,ZX-81

LA UNICA Y DEFINITIVA SOLUCION EN COLOR COMPATIBLE CON SU SISTEMA

Modelos	Pixels
14" Standard	452 x 585
14" Media	653 x 585
14" Alta	895 x 585
20" Standard	505 x 585
20" Alta	860 x 625

1431/AP DS

ESTAMOS EN INFORMAT
PALACIO FERIAL - NIVEL 2
STAND - 202

1431/MS-DS

1451/DQ

¿QUIEN NECESITA UN MONITOR EN COLOR?

Por supuesto toda persona que tenga un ordenado..

Hasta ahora Vd. probablemente usaba su televisor doméstico con su ordenador y habrá notado bastantes interferencias, especialmente cuando visualiza textos. Los televisores no están básicamente diseñados para visualizar datos, ya que están construidos con circuitos de codificación y modulación para aceptar únicamente las ondas de televisión a través del aire.

La diferencia entre su televisor y un monitor CUB, es que éste último está especialmente diseñado para la visualización de textos y gráficos, ésto se evidencia inmediatamente en la imagen estable y clara que reduce notablemente el esfuerzo de la vista.

El monitor CUB está preparado para desarrollar las capacidades sofisticadas de visualización de los ordenadores de hoy y del mañana.

¿POR QUE ELEGIR UN MONITOR CUB?

Sólo la gama CUB de Microvitec, es suficientemente completa para cubrir la compatibilidad de casi todos los micro ordenadores del mercado.

Estos magníficos monitores británicos, son los únicos elegidos por el Gobierno inglés para usarlos en las escuelas primaria y secundaria de todo el país.

Nuestra gama de monitores de resoluciones standar, media y alta, más los modelos PAL/RGB, son compatibles totalmente con IBM PC/PCjr, APPLE II/IIe/III, SINCLAIR SPECTRUM/QL, COMMODORE 64/VIC 20, DRAGON 32/64, ORIC, BBC, ACORN ATOM, ATARI, ACT APRICOT, SHARP, ITT, TANDY, ADVANCE, CROMMENCO 501, LYNX, TEXAS INSTRUMENTS T 99/4A y muchos más.

Piense, cuando tome su decisión final, que sólo los CUB de Microvitec le pueden proporcionar la mayor calidad, rendimiento y fiabilidad al mejor precio.

TODOS LOS MONITORES CUB INCLUYEN:

* Garantía total por un año.
* Chasis aislado para máxima seguridad.
* Interruptor de potencia para un mejor rendimiento.
* Mínimo error de convergencia esencial para visualización de textos gráficos.
* Diseñados para introducir los standars reconocidos de seguridad (i.e. BS415).
* Chasis preparado para bajo consumo de potencia.
* Componentes de alta calidad para asegurar la máxima fiabilidad.
* Mando de conexión de potencia, plug y RGB.
* Diseño práctico, atractivo y moderno.
* Aprobación por la B.E.A.B. de nuestros más populares modelos.
* La mejor relación calidad-precio.
* La garantía de una gran firma como Microvitec que acaba de ganar el PREMIO REAL AL DESARROLLO TECNOLOGICO 1984 EN INGLATERRA.

MICROVITEC CUB

MONITORES COLOR

IMPORTADO Y DISTRIBUIDO EN EXCLUSIVA PARA ESPAÑA:

multilogic

COMERCIALIZADORA DE ARTICULOS DE INFORMATICA MULTILOGIC, S. A.
P.° de la Habana, 145.
28036-MADRID Tel. 458 74 75
Telex: 42710 FONOTXE

sumario

N.° 39 Especial - Año 1985

ESPECIAL TRUCOS

Director:
Javier San Román
Director Adjunto:
Santiago Mondet Peyrou

REDACCION:
Coordinador de Redacción:
S.M. Peyrou
Director Técnico:
J. Antonio Deza
Jefe de Redacción:
José Luis Sanabria
Secretaría de Redacción:
Julia Peña
Maquetación:
José Ramón Andréu
Composición:
M.ª José Raboso
Montaje:
Vicente Hernández
Fotografía:
Barahona

Colaboradores: S. Almeida - José Luis Bañesa Sanz - Iñaki Cabrera - Antonio Castaño Sánchez - Víctor Manuel Delgado - José Antonio Deza Navarro - Víctor Manuel Díaz - Pedro Díaz Cuadra - Jaime Díez Medrano - Fabio Gil Miguel - Juan Carlos González - Santiago González Ascensión - Félix Gutiérrez Fernández - Gerardo Izquierdo Cadalso - Miguel Angel Lerma Usero - Ramón López Cabrera - José Antonio Mañas Valle - Justo Maurín - Sebastián M. Yañez - Juan Carlos Ordoñez Vela - Manuel Otero Raña - Alberto Requena Rodríguez - José Manuel Rodríguez Prolongo - Francisco Romero - Isidoro Ruiz Sánchez - Gilberto Sánchez García - Pedro San Esteban Díaz - Víctor Manuel Sevilla - José María Vidal - Isabel Yañez Thos.

PUBLICIDAD VENTAS Y ADMINISTRACION:

Director de Publicidad:
Santiago Mondet
Asistido por: Julia Peña

Suscripciones:
Lucía Pérez

REDACCION - PUBLICIDAD ADMINISTRACION:

Para España y Extranjero:
Calle Ferraz, 11, 3°
Tel.: (91) 247 30 00 - 241 34 00
28008 MADRID

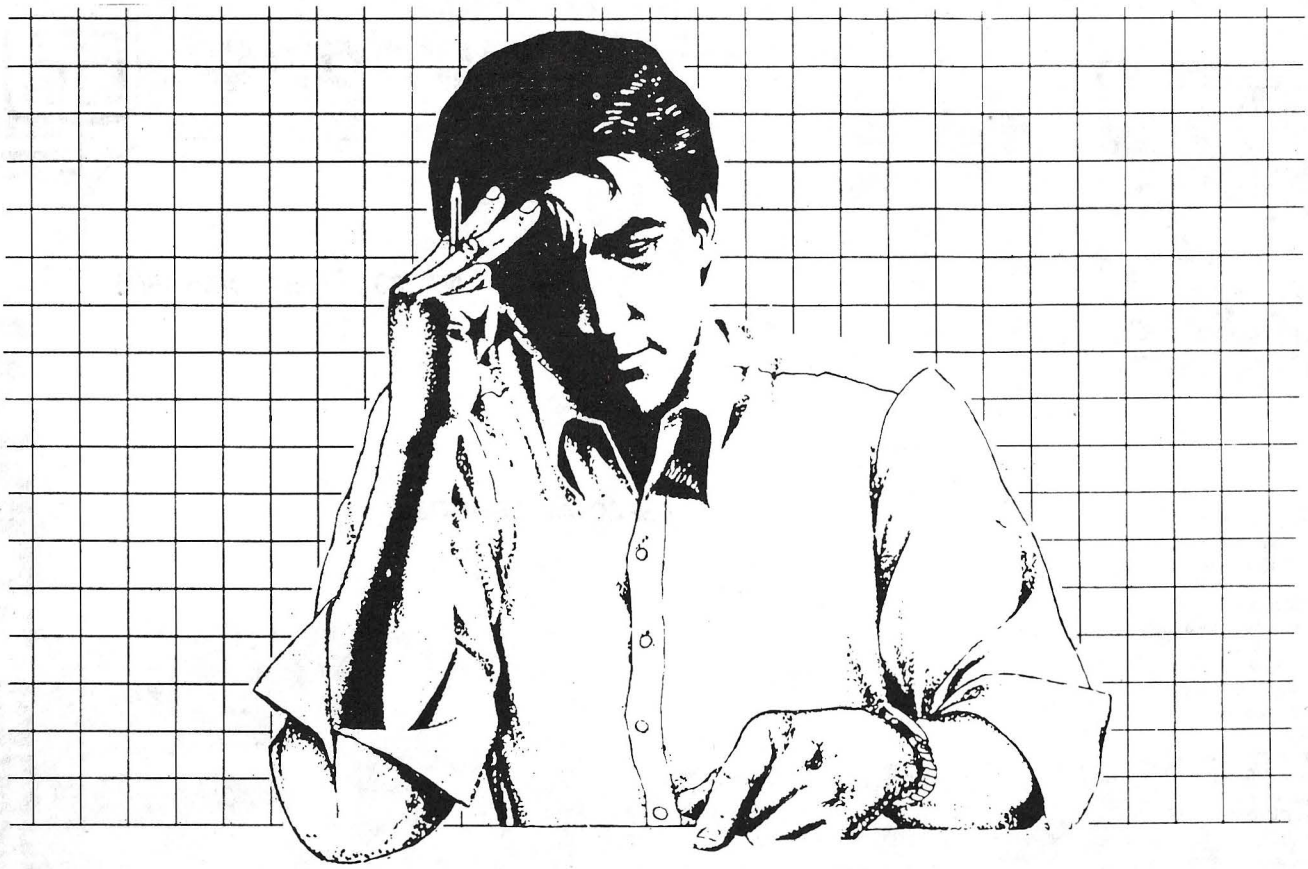
Imprenta:
Pentacrom, S.L.
Hachero, 4 - Madrid

Distribuye:
SGEL
Avda. Valdeparra, s/n.
ALCOBENDAS (Madrid)

Controlado

EL ORDENADOR PERSONAL
es una publicación de:
EL ORDENADOR INDIVIDUAL, S.A.
Director de publicación:
JAVIER SAN ROMAN
Depósito Legal: M-4256-1982.

EL ORDENADOR PERSONAL



ORDENADORES PARA EL MAÑANA

DISPONIBLES HOY. METHAMORPHIC ES CAPAZ HOY DE ESTAR LISTO PARA FUNCIONAR Y AMPLIARSE. **LE OFRECE YA:** MONITORES (VERDE Y COLOR), CONTROLADORES PARA SUS UNIDADES DE DISCO Y WINCHESTERS, Y ESTOS, A PRECIOS ASEQUIBLES, SISTEMAS OPERATIVOS, LENGUAJES PARA CONVERSAR USUARIO CON EL ORDENADOR, INTERFACES PARA EXPANDIRSE CON TODO LO PENSADO Y POR PENSAR (RELOJ, MODEMS, JOYSTICKS, PADDLES, ANALOGICO-DIGITAL, AMPLIACIONES DE MEMORIA, PADS GRAFICOS, WILDCARDS), VISUALIZACION A VOLUNTAD DEL USUARIO DE LA INFORMACION EN PANTALLA (40, 80, 124, ETC. COLUMNAS), TRANSMITIR LA INFORMACION CORRESPONDIENTE A PAPEL POR MEDIO DE IMPRESORAS Y PLOTTERS A TRAVES DE CONEXIONES INTELIGENTES, CON POSIBILIDAD DE BUFFERS Y CONECTAR VARIOS ORDENADORES A UNA SOLA IMPRESORA. UNIDO A UNA CANTIDAD DE PROGRAMAS DISPONIBLES INMEDIATAMENTE, COMO SON MAS DE 10.000 PROGRAMAS, PERO REALES NO AQUELLOS QUE AUN SE HAN DE HACER. SABER ADEMAS QUE HAY EN EL MUNDO MAS DE 500.000 PERSONAS CON UN ORDENADOR COMPATIBLE A METHAMORPHIC. Y ADEMAS QUE GUARDE LA MEJOR RELACION CALIDAD PRECIO DEL MERCADO... COMO METHAMORPHIC.
HOY DECIMOS **"LOS ORDENADORES DEL MAÑANA SERAN COMO METHAMORPHIC HOY"**.

METHAMORPHIC

Ordenadores más Personales!

DISTRIBUIDOR GENERAL EXCLUSIVO
PARA ESPAÑA DE METHAMORPHIC

FIRST S.A.

C. ARIBAU, 62
08011 BARCELONA
323 03 90
TELEX 53947 FIRS

Editorial

Estamos comprometidos en la ardua tarea de aproximar poco a poco al mundo de la informática a todas esas personas que un día se preguntaron ¿Qué es un ordenador?.

Al principio la gran mayoría cae en la tentación del juego. Despilfarrar adrelanina por los cuatro costados es muy propio de los tiempos en los que nòs movemos. Tarde o temprano uno se pregunta que es lo que permite que subamos las pulsaciones del corazón delante de la tele, sin conectar al programa nacional.

Una mañana primaveral nos sorprendemos delante del espejo con un libro de BASIC en la mano. ¡Ya no hay retorno!

En nuestra lucha con el "chip" no hay reglas. Hay que atacar o apagar y es por eso que ya no sirven los manuales, se quedaron atras junto con los "guacaguacas". Es el momento del truco, la astucia, la magia electrónica, es el momento de abrir estas páginas y retozar entre PEEKs, POKEs, DATAs y otros CALLs.

Recopilar los mejores trucos no es fácil, satisfacer a todos, tampoco, no dudamos que vuestras críticas ayudaran a superarnos.

Y si después de todo, os queda tiempo libre:

¡ ¡Felices Vacaciones!!

GARANTIA UN AÑO

KATSON

92.500

La mayor variedad en tarjetas y accesorios para tu APPLE*

SEGUIMOS BUSCANDO DISTRIBUIDORES

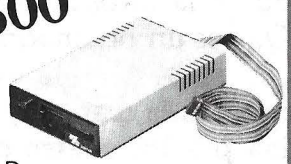
* (APPLE) es marca registrada de Apple Computer Inc

KATSON II

DISK DRIVE MEDIA ALTURA

NUEVO 49.500

TRACCION DIRECTA GRAN FIABILIDAD



16 K RAM CARD 12.900ptas.

CP/M CARD 13.500ptas.

DISK DRIVER CARD 11.000ptas

LANGUAGE CARD 13.500ptas.

PAL CARD 15.500ptas.

PARALELL PRINTER CARD 12.375ptas.

80 COLUMNAS CARD 16.000ptas.

estos son nuestros precios sin competencia

ORDENADORES PERSONALES

KA-001	KATSON II	92.500
KA-002	KATSON II con teclado numérico	98.500
KA-003	KATSON II con teclado numérico 64 K RAM y doble CPU (6502 + Z80)	118.000

TARJETAS Y ACCESORIOS

CD-001	8088 CARD	117.300
CD-002	A/D - D/A CARD	96.850
CD-003	A/D CARD	63.200
CD-004	IEEE-488 INTERFACE CARD	55.000
CD-005	6809 CARD	60.700
CD-006	SERIAL INTERFACE RS-232 C	14.900
CD-007	SUPER SERIAL CARD	36.750
CD-008	COMMUNICATION CARD	14.250
CD-009	128 K RAM CARD	44.000
CD-010	CP/M CARD	13.500
CD-011	WILD CARD	18.500
CD-012	GRAPPLER + BUFFER CARD	39.500
CD-013	TIME II CARD	19.125
CD-014	PARALLEL PRINTER CARD	12.375
CD-015	EPROM WRITER	16.500
CD-016	80-COLUMN CARD	16.000
CD-017	CONTROLADOR	11.000
CD-018	LANGUAGE CARD	13.500
CD-019	16 K RAM CARD	12.900
CD-020	PAL CARD	15.500
CD-021	6522 PARALLEL CARD	16.200
CD-022	MUSIC CARD	18.750
CD-023	SPEECH CARD	20.000
CD-024	80 COLUMN SOFT SWITCH CARD	22.500
CD-025	RF Modulador	3.500
CD-026	COOLING FAN	10.000
CD-027	JOYSTICK para APPLE	5.700
CD-028	SWITCHES 40/80 COLUMNAS	2.500
CD-029	TABLERO GRAFICO PLOT II	17.500

DISK DRIVER

DD-001	Disk driver - Unidad de disco flexible simple cara simple densidad 143 K. Mecánica Shugart.	47.500
DD-002	Disk Driver - Unidad de disco flexible simple cara simple densidad 160K - Tracción directa - Media Altura	62.500

MONITORS

MN-001	Monitor fósforo verde antirreflexivo Philips TP-200 12 Pulgadas alta resolución.	29.000
MN-002	Monitor fósforo verde antirreflexivo Philips PCT-1202 12 Pulgadas muy alta resolución	34.500

KATSON es una exclusiva de:

ANGLEX
Anglo-Española de Trading, S. A.

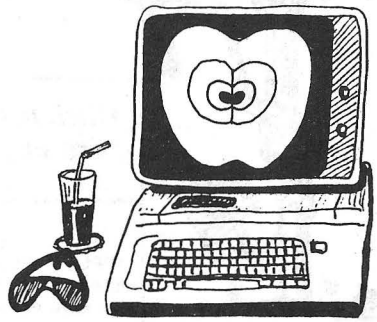
Ayala, 13
MADRID-28001
Tels. 276 22 74
276 22 75
Telex: 42.597 ANLE

PARA MAS INFORMACION MANDARNOS ESTE CUPON

Nombre .....
Dirección .....
Ciudad .....
Provincia .....

KATSON

el Apple pelado



En los remotos tiempos de Adan y Eva las manzanas se comían, ¡que tiempos!. Hoy se cacharrea, se ordena, se juega, se gana dioptrías día tras día, truco tras truco. ¡No te quemes mucho y que te aproveche!

Ha llegado el nuevo catalogo

Este programa en Basic Applesoft presenta el CATALOG del diskette en una única página, lo que proporciona una mejor visión de sus ficheros. Por otra parte, aporta numerosas informaciones útiles, como:

- Número de ficheros en el diskette.
- Tipo de ficheros (A, I, T, R o S).
- El cierre o no (lock = *) del fichero.

El programa lee el catálogo y, al mismo tiempo, contabiliza el número de sectores ocupados, así como el total de los ficheros (pista 11, sector 15 a 1). Todos estos parámetros se presentan arriba y a la izquierda de la pantalla en forma de contador decimal.

La velocidad de desarrollo del contador puede modificarse poniendo, por ejemplo, una instrucción SPEED = 200 tras la línea 85. Habrá que dar de nuevo el valor 255 a SPEED tras la línea 90. Puede hacer pruebas para fijar el valor que le convenga.

Veamos ahora los puntos peculiares del programa.

La línea 65 escribe un pequeño programa en binario, POKEado en la dirección 002 (página cero, empleada a menudo por los programadores para ubicar pequeñas rutinas, estando libres las direcciones 2 a 31). Se hubiera podido elegir otra dirección; por ejemplo, 768 ($300 hexadecimal) que corresponde a la posición de los vectores del monitor. El equivalente a los seis POKE hubiera consistido en entrar en el monitor mediante CALL-151 y teclear los siguientes códigos:

2:20 E3 03
5:4C D9 03

Podrá darse cuenta listando las direcciones 2 a 7: haga RUN, entre en el monitor mediante CALL-151 y teclee 2L Esta operación vuelve a desensamblar las direcciones a partir de la 2 (hágalo antes y después de que funcione el programa para comprobar la diferencia). Este pequeño subprograma binario de 12 octetos llama a otro subprograma de la dirección $03E3(JSR) y después de una bifurcación incondicional (JMP) a la dirección $03D9. Estos pequeños programas pertenecen al sistema de explotación del disco (Sed).

$03E3: búsqueda de la tabla IOB (Input Output Block), de los vectores de RWTS (Real-Write/Track Sector).

$03D9: lectura-escritura de un sector (RWTS). En nuestro caso, empleo en lectura.

Líneas 70 a 75: inicialización de los vectores del bloque IOB mediante POKE,s. Sirven para poner el volumen a cero, para buscar la dirección tampón en la memoria activa (MEV) y para leer la pista/sector deseada.

Línea 85: lectura de la pista 17 ($11) y de los sectores 15 a 1, en un bucle FOR NEXT y después llamada al subprograma mediante CALL 2 (POKEada en la línea 65).

Línea 95: dimensionado de la tabla de variables (T$).

Línea 100: se fija el número de caracteres de cada fichero, en función del número de columnas necesarias. El DIRECTORY puede contener como máximo 105 ficheros, y la visualización en pantalla se hará en ese caso en cinco columnas. Se define el número máximo de ficheros por columna (veintiuno en esta ocasión).

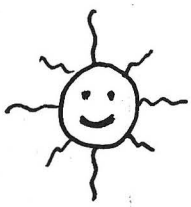
La línea 120 fija el valor de la variable COL (columna).

Las líneas 125 a 155 trazan la fisonomía del catálogo y escriben los rótulos.

La línea 160 coloca el cursor en la línea 23 columna 0, ¡y se acabó!

Marcel Cottini

```
10  REM  ********************************
15  REM  ***                          ***
20  REM  ***          CATALOG         ***
25  REM  ***                          ***
30  REM  ***  AUTOR.- M. COTTINI      ***
35  REM  ***                          ***
40  REM  ***  (C) EL O. PERSONAL      ***
45  REM  ***            Y             ***
50  REM  ***         EL AUTOR         ***
55  REM  ********************************
60  NORMAL : TEXT : HOME
65  POKE 2,32: POKE 3,227: POKE 4,3: POKE 5,76: POKE 6,217: POKE 7,3
 70 PIST = 17:RW = 1: POKE 47083,0: POKE 47084,PIST: POKE 47091,0: POKE 47092,RW
75 LOC = 8192: POKE 47088,LOC -  INT (LOC / 256) * 256: POKE 47089, INT (LOC / 2
56)
80  TEXT : HOME : PRINT "ESPERE, POR FAVOR.... "
85  FOR SE = 15 TO 1 STEP  - 1: POKE 47085,SE: CALL 2: FOR X = LOC + 11 TO LOC +
 221 STEP 35:P =  PEEK (X): IF P > 0 AND P < 255 THEN NF = NF + 1:USAD = USAD +
 PEEK (X + 33) + PEE
90  VTAB 3: INVERSE : PRINT " CONTADOR:";: NORMAL : FLASH : PRINT "PISTA 11": NO
RMAL : PRINT : PRINT "SECTOR  :";SE" ": PRINT "OSUPADOS  :";USAD: PRINT "FICHER
OS :";NF: IF P THEN  NEXT : NEXT
95  DIM T$(105):T$(0) = "T":T$(1) = "I":T$(2) = "A":T$(4) = "B":T$(8) = "S":T$(1
6) = "R":T$(32) = "A":T$(64) = "B": REM  CODIGOS DE LOS FICHEROS (I=ENTERO, T=TE
XTO, A=APPLESOFT, B=BINARIO S&R SON FICHEROS ESPECIALES.
100 L = 30: IF NF > 21 AND N < 43 THEN L = 17:L = 17: REM  L=LONGITUD DEL FICHER
O
105  IF NF > 42 AND NF < 64 THEN L = 10
110  IF NF > 63 AND NF < 85 THEN L = 7
115  IF NF > 84 THEN L = 5
120 COL = L + 3: IF L = 30 THEN COL = 40: REM  COL=POSICION DE LA COLUMNA
125  HOME : INVERSE : PRINT  SPC( 40);: FOR V = 2 TO 22: FOR H = 1 TO 40 STEP COL
: VTAB V: HTAB H: PRINT  SPC( 1);: HTAB 40: PRINT  SPC( 1);: NEXT : NEXT
130  VTAB 23: PRINT  SPC( 40);: VTAB 1: HTAB 3: PRINT "CATALOG :";NF;" FICHEROS,
 496 - USAD;" SECT.LIBRES": VTAB 23: PRINT "  ******* ORDENADOR PERSONAL  *****
** ";
135  FOR SEC = 15 TO 1 STEP  - 1: POKE 47085,SE: CALL 2: FOR X = LOC + 11 TO LOC
 + 221 STEP 35: IF  PEEK (X) = 0 THEN 160
140  IF  PEEK (X) = 255 THEN 155
145 N = N + 1: VTAB 1 + N -  INT ((N - 1) / 21) * 21: HTAB 1 + COL *  INT ((N -
1) / 21):P =  PEEK (X + 2):P = P - 128 * (P > 127)
150  INVERSE : PRINT T$(P);: NORMAL : PRINT  CHR$ (32 + 10 * ( PEEK (X + 2) > 12
7));: FOR Y = X + 3 TO X + L + 2: PRINT  CHR$ ( PEEK (Y));: NEXT Y
155  NEXT X,SE
160  VTAB 23: END
```



Miniensamblador para el Apple II plus

Según dice el manual del APPLE II PLUS o EUROPLUS, este modelo no permite el uso del mini-ensamblador normal.

Y sin embargo... es posible si posee Vd. el System Master del DOS (sistema operativo de disco) aun cuando usted no tenga ni DOS ni INTEGER.

En el primer caso se comienza escribiendo:

```
BLOAD INTBASIC
CALL-151
3537:35
355B:35
35BF:36
35DD:36
35E7:36
3633:35
3668:35
```

Se sustituye después el System Master por un diskette cualquiera en el que se pueda escribir, y se pulsa:

BSAVE MINI-ENSAMBLADOR, A$3500, L$170

Existe sin embargo un medio para hacer hablar a su mini-ensamblador. En efecto, este último comienza en la dirección $3500 y se acaba en $365 F. Por otro lado en la dirección $3666 hay un salto JMP $3592 que es la dirección de entrada del mini-ensamblador.

Para utilizar el mini-ensamblador debe hacerse:

```
BLOAD MINI-ENSAMBLADOR CALL-151
CALL-151
3666G
```

3500	E9	81	41	D0	14	A4	3F	A6
3508	3E	D0	01	88	CA	8A	18	E5
3510	3A	85	3E	10	01	C8	98	E5
3518	3B	D0	6B	A4	2F	B9	3D	00
3520	91	3A	88	10	F8	20	1A	FC
3528	20	1A	FC	20	D0	F8	20	53
3530	F9	84	3B	85	3A	4C	95	35
3538	20	BE	FF	A4	34	20	A7	FF
3540	84	34	A0	17	88	30	4B	D9
3548	CC	FF	D0	F8	C0	15	D0	E8
3550	A5	31	A0	00	C6	34	20	00
3558	FE	4C	95	35	A5	3D	20	8E
3560	F8	AA	BD	00	FA	C5	42	D0
3568	13	BD	C0	F9	C5	43	D0	0C
3570	A5	44	A4	2E	C0	9D	F0	88
3578	C5	2E	F0	9F	C6	3D	D0	DC
3580	E6	44	C6	35	F0	D6	A4	34
3588	98	AA	20	4A	F9	A9	DE	20
3590	ED	FD	20	3A	FF	A9	A1	85
3598	33	20	67	FD	20	C7	FF	AD
35A0	00	02	C9	A0	F0	13	C8	C9
35A8	A4	F0	92	88	20	A7	FF	C9
35B0	93	D0	D5	8A	F0	D2	20	78
35B8	FE	A9	03	85	3D	20	34	36
35C0	0A	E9	BE	C9	C2	90	C1	0A
35C8	0A	A2	04	0A	26	42	26	43
35D0	CA	10	F8	C6	3D	F0	F4	10
35D8	E4	A2	05	20	34	36	84	34
35E0	DD	B4	F9	D0	13	20	34	36
35E8	DD	BA	F9	F0	0D	BD	BA	F9
35F0	F0	07	C9	A4	F0	03	A4	34
35F8	18	88	26	44	E0	03	D0	0D
3600	20	A7	FF	A5	3F	F0	01	E8
3608	86	35	A2	03	88	86	3D	CA
3610	10	C9	A5	44	0A	0A	05	35
3618	C9	20	B0	06	A6	35	F0	02
3620	09	80	85	44	84	34	B9	00
3628	02	C9	BB	F0	04	C9	8D	D0
3630	80	4C	5C	35	B9	00	02	C8
3638	C9	A0	F0	F8	60	20	7D	F4
3640	A5	F8	10	13	C9	8E	D0	F5
3648	24	F9	10	0A	A5	FB	F0	06
3650	E6	FA	D0	02	E6	F9	60	A9
3658	00	85	F9	85	FA	60	FF	FF
3660	FF	FF	FF	FF	FF	FF	4C	92
3668	35							

Se verá aparecer entonces en la pantalla el "prompt" (I) propio del mini-ensamblador y se ahorrará simultaneamente la necesidad de disponer de una carta de lenguaje. Y como incluso las mejores cosas tienen su fin, se vuelve al monitor escribiendo: $FF69 G.

En el segundo caso (no dispone ni de DOS ni de INTEGER), debe copiar el programa del mini-ensamblador, y luego aplicar el método del primer caso haciendo CALL-151 y luego 3666G.

El programa mini-ensamblador podrá ser desplazado en la memoria a condición de tener el cuidado de cambiar las direcciones de los diferentes JMP y JSR. Estas direcciones están encuadradas en el listado que les damos del miniensamblador.

¡Feliz utilización!

J. F. Mabilat

Alta resolución en Apple II

El tema de la alta resolución en los ordenadores recientemente anunciados en el mercado es de difícil apreciación y se presta a fácil manipulación.

Siempre que nos dan una resolución nos definen el número de puntos horizontales y verticales.

Con el barrido de los televisores y monitores actualmente en el mercado, el número de líneas direccionables, o definición vertical, está limitado a un tope práctico de 246 líneas. Para sobrepasar dicha definición hay que ir a un tipo de monitor de alta resolución, cuyo precio se va a las nubes.

En los equipos de desarrollo americano, o pensados para su venta allí, se parte del barrido de 525 líneas, y los equipos generalmente tienen la alta resolución vertical limitada a 192 líneas.

En los nuevos equipos anunciados, que pretenden ser la segunda generación de Ordenadores Personales, aún se mantiene la definición vertical reducida, pero se ha aumentado la definición horizontal a 400 o 600 puntos.

Realmento esto no nos aporta nada nuevo ya que los gráficos no saldrán mejorados. La posición del punto dentro de un entorno servirá, si utilizamos un monitor de color, para definir el color de este entorno.

En el programa que incluimos a continuación, tenemos un método de, valiéndonos de la opción de color, obtener del APPLE II una definición horizontal de 0 a 558 sobre un monitor de blanco y negro.

El experimentar con este programa nos servirá para aprender a valorar aún más el APPLE II y no dejarnos engañar por propagandas que insisten sobre parámetros cuyo valor es muy relativo.

```
*
JLIST
5  HGR
10  CLEAR
12  INPUT "COORDENADAS INICIALES
    (X1,Y1)=";X1,Y1
20  INPUT "COORDENADAS FINALES
    (X2,Y2)=";X2,Y2
25  IF X1 > X2 THEN S = X1:X1 = X
    2:X2 = S
30  GOSUB 1000: REM  EQUIVALENTE
    A HPLOT X1,Y1 TO X2,Y2
40  GOTO 10
1000 C = Y1 - ((Y1 - Y2) / (X1 -
    X2)) * X1
1010 A = (Y1 - Y2) / (X1 - X2)
1020  FOR X = X1 TO X2
1030 Y = A * X + C: REM  ECUACIO
    N DE LA RECTA
1040  GOSUB 2000: REM  EQUIVALENT
    E A HPLOT X,Y
1050  NEXT X
2000 P = 0:PP = 0
2010 M = X / 2
2020  IF (M -  INT (M)) = 0 THEN
    P = 1
2030 M =  INT (M)
2040  IF (M / 2 -  INT (M / 2)) =
    0 THEN PP = 1
2050  IF P = 1 AND PP = 1 THEN  HCOLOR= 2
2060  IF P = 0 AND PP = 1 THEN  HCOLOR= 6
2070  IF P = 1 AND PP = 0 THEN  HCOLOR= 1
2080  IF P = 0 AND PP = 0 THEN  HCOLOR= 5
2090  HPLOT X,Y
2100  RETURN
```

Explorando por los diskettes habréis notado la existencia de un programa que se llama FID. Se trata de un programa multiuso que entre otras cosas permite hacer copias de programas de un disco a otro. Otra utilidad menos conocida es la de ver el espacio disponible en un diskette cualquiera. Esta rutina resulta muy útil pues casi nunca le sacamos todo el juego posible a los 140 Kb que podemos aprovechar y cuando no ocurre esto, ocurre lo contrario (¿qué lógica no?), es decir, que vamos a escribir un fichero y resulta que no hay sitio, con lo que habrá muchas probabilidades de perder todos los datos.

Claro que el acceder al programa FID resulta un poco lento y aburrido y nadie lo utiliza —por lo menos mi barita mágica y yo no lo hacemos—. En su lugar utilizemos el pequeño programa que nos describe hoy Stephan Burlot. Espero que os resulte interesante.

Sectores utilizados

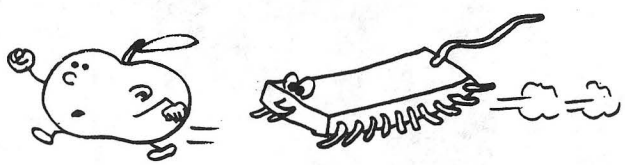
El programa permite conocer cuantos sectores están utilizados en un diskette de 16 sectores (DOS 3.3). Para ello se utiliza la rutina RWTS (ver el manual del DOS 3.3 para mayor información pg. 95) que va a leer la pista $11 del diskette y los sectores $F a $1 en orden descendente (SC a $1 en el DOS 3.2). En 3087 ($COF) se pone el número del sector que se va a leer y luego se lee siete veces el bloque que se acaba de transferir. En efecto este contiene los datos de siete "files" diferentes.

Línea 70: Si PEEK (A-33) = 255 ($FF), la "file" no existe (si bien su nombre sigue existiendo en el catálogo) y no es preciso contar el número de sectores que utiliza. El número de sectores utilizados por el programa está en las direcciones $21 y $22 contando a partir del comienzo de cada file (de las siete que hay). El número de sectores es ($226)*256 + ($21). (Para más detalle puede leerse la página 129 del manual del DOS "The diskette directory").

Después, el sistema de explotación de discos utiliza 3 pistas de 16 sectores cada una (3 del 13 para el DOS 3.2) más una de 16 (resp. 13) sectores para el catálogo, lo que suma en total 24 sectores (52 en DOS 3.2) a sumar al total de sectores utilizados.

Para el programa IOB, (ver manual página 95), se lee el "slot" 6 "drive" 2 y se transfieren los datos a la memoria tampón situada en $2000.
Los data se refieren a las direcciones de los lugares en los que se encuentra el número de sectores utilizados por cada una de las 7 "files"

```
]LIST

 10   PRINT  CHR$ (4);"BLOAD IOB"
 20   FOR K = 15 TO 1 STEP  - 1
 30   POKE 3087,K: REM  3087=$COF
 40   CALL 3072: REM  3072=$C00
 50   FOR T = 1 TO 7
 60   READ A
 70   IF  PEEK (A - 33) = 255 THEN
       110
 80   REM  SI PEEK(A-33)=255 LA "FI
      LE" HA SIDO BORRADA
 90   IF ( PEEK (A) +  PEEK (A + 1)
      ) = 0 THEN 140
100 X = X + ( PEEK (A) +  PEEK (A
      + 1) * 256)
110   NEXT T
120   RESTORE
130   NEXT K
140   PRINT X + 48 + 16;"SECTORES
      UTILIZADOS"
150   PRINT 560 - (X + 48 + 16);"S
      ECTORES LIBRES"
160   DATA  8236,8271,8306,8341,83
      76,8411,8446

]CALL-151

*C00.C24

0C00- A9 0C A0 0A 20 D9 03 60
0C08- 49 D0 01 60 02 00 11 0F
0C10- 20 0C 00 20 00 00 01 00
0C18- FE 60 02 45 20 22 3B 46
0C20- 00 01 EF D8 D0
```



Este programa está inspirado del programa FID del diskette Master Dos 3.3. Sólo que el lenguaje de máquina es muy lento de cargar en memoria y he preferido crear mi propio programa.

Stephan Burlot

Tabla

```
C:01 <= Indicador de IOB (INPUT/OUTPUT BLOCK)
00 <= Slot x 16
02 <= Numero de drive
0C <= volumen esperado (0C es el valor por defecto)
11 <= Pista
01 <= Sector
20 <  0C20 es la direccion de la
0C <=  "DEVICE CHARACTERISTICS TABLE"
00 <
20 <= 2000 es la direccion del buffer
00 <= no utilizado
00 <= no utilizado
01 <= Comando de lectura
00 <= Codigo de error
FF <= Volumen encontrado (254)
60 <= Slot x 16 encontrado
02 <= Drive encontrado
```

El quid del Pascal

Los poseedores de Apple 2e que sólo dispongan de un lector de disquetes (¡pobres!) y quieran trabajar en Pascal, pueden quedar muy sorprendidos si siguen el procedimiento descrito en *Apple Pascal: Language Reference Manual* con el título *«the two-steps startup»* (arranque en dos etapas). Este indica que, tras haber enchufado el Apple con el disquete Apple 3 se debe cambiar este por el Apple 0, cuando aparezca el mensaje: «Insert boot disk with System. Pascal on it, then press RESET» y después teclear RESET como amablemente nos pide. Este proceso funciona perfectamente con Apple 2 y 2+, pero fracasa estrepitosamente con los nuevos Apple 2e, ya que el RESET borra la parte correspondiente de la tarjeta de memoria y el sistema presenta el sibilino mensaje: «NO FILE SYSTEM APPLE».

Existen dos soluciones: esperar a la versión 1.2 del Pascal (que ya se puede conseguir en EE.UU.) o proceder del siguiente modo:

- Crear un disquete suplementario, Apple 0 boot, que comprenda los ficheros System Pascal, System Apple y System Miscinfo.

- Insertar el disquete Apple 3 y enchufarlo (o manzana-abierta/RESET, si el Apple ya está encendido).
- Reemplazar Apple 3 por Apple 0 boot y pulsar RESET. El Pascal ya está cargado y se sustituye el disquete Apple 0 boot por Apple 0 trabajo.

Hay una tercera solución: vamos a modificar el Sistema Pascal del disquete Apple 3 con el fin de evitar el efecto devastador del RESET. Tras haber tecleado y compilado MODIF. TEXT, se coloca el código resultante de la compilación MODIF. CODE, en el disquete Apple 3 conteniendo solamente Sistema Apple. La ejecución de MODIF. modificará elSistema Apple en *el disquete*.

El procedimiento de carga con un único lector será el que describe el manual.

- Enchufar (o manzana-abierta/RESET) con Apple 3 modificado;
- El mensaje anterior se cambia por «Insert boot disk with System. Pascal on it, then press (RET)»;
- Entonces se coloca el disquete Apple 0 normal y sencillamente se pulsa RETURN.

TARJETA DE INFORMACION PUBLICITARIA

INFORMACION SOBRE PUBLICIDAD EN
EL ORDENADOR PERSONAL

Sr. Director:

Estando interesado en conocer las Tarifas de Publicidad en esa Revista, le ruego me envíe un ejemplar sin compromiso por mi parte.

Nombre de la Empresa .....
A la atención de Don. .....
Calle ..... Tfno. .....
Población ..... Código Postal ..... Provincia .....
Fecha .....

Firma

TARJETA DE PETICION DE LIBRERIA

SERVICIO DE LIBRERIA

Les Ruego me remitan, contra reembolso, los siguientes libros de su fondo editorial.

Bassic, nº de ejemplares .....
Autor: Sanchez-Izquierdo — Precio: 1.100 Pts.

Programación Fichero Bassic:

Tomo I ..... Precio: 650 Pts.
Tomo II ..... Precio: 950 Pts.

TOTAL ..... 1.500 Pts.

Nombre .....
Domicilio ..... Firma
Ciudad .....

PETICION DE NUMEROS ATRASADOS

BOLETIN DE PEDIDO
O.P. EL ORDENADOR PERSONAL

☐ Deseo los siguientes números atrasados:

1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☐ 8 ☐ 9 ☐ 10 ☐ 11 ☐ (Al precio de 200 pts. ejemplar).

12 ☐ 13 ☐ 14 ☐ 15 ☐ 16 ☐ 17 ☐ 18 ☐ 19 ☐ GUIA (450 pts.) 20 ☐ 21 ☐ 22 ☐ 23 ☐ 24 ☐

25 ☐ 26 ☐ 27 ☐ 28 ☐ 50 Programas BASIC (450 pts.) 29 ☐ 30 ☐ 31 ☐ GUIA (500 pts.)

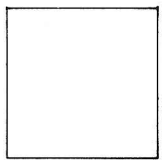
32 ☐ 33 ☐ 34 ☐ 35 ☐ 36 ☐ 37 ☐ 38 ☐

(Al precio de 250 pts. ejemplar).

☐ Deseo me envíen ..... tapas para encuadernar la revista (12 números) al precio de 500 pts. una.
☐ Tomo n.º 1 encuadernado (1 - 11), precio: 3.000 pts.

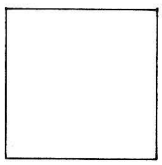
El importe total de ..... Pts. lo mando por giro postal número .....
o por su importe en sellos de correos nuevos. (Tachar las menciones útiles).
Nombre ..... Apellidos .....
Calle ..... N.º ..... puerta ..... piso .....
Ciudad ..... Código Postal .....
Provincia .....

Firma:



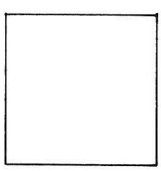
EL ORDENADOR INDIVIDUAL S.A.

Ferraz, 11 - 28008-Madrid (España)
Tels. 247 30 00 y 241 34 00



EL ORDENADOR INDIVIDUAL S.A.

Ferraz, 11 - 28008-Madrid (España)
Tels. 247 30 00 y 241 34 00



EL ORDENADOR INDIVIDUAL S.A.

Ferraz, 11 - 28008-Madrid (España)
Tels. 247 30 00 y 241 34 00

```
PROGRAM MODIF;

VAR
  BUFFER : PACKED ARRAY[0..31,0..511] OF 0..255 ;
  F:FILE ;
  I:INTEGER;


BEGIN
  RESET(F,'#4:SYSTEM.APPLE');
  I:=BLOCKREAD(F,BUFFER,32);
  CLOSE(F);
  BUFFER[17,105]:=40;
  BUFFER[17,106]:=82;
  BUFFER[17,107]:=69;
  BUFFER[17,108]:=84;
  BUFFER[17,109]:=41;
  BUFFER[17,451]:=173;
  BUFFER[17,452]:=0;
  BUFFER[17,453]:=192;
  BUFFER[17,454]:=16;
  BUFFER[17,455]:=251;
  BUFFER[17,456]:=141;
  BUFFER[17,457]:=16;
  BUFFER[17,458]:=192;
  BUFFER[17,459]:=173;
  BUFFER[17,460]:=131;
  BUFFER[17,461]:=192;
  BUFFER[17,462]:=76;
  BUFFER[17,463]:=158;
  BUFFER[17,464]:=214;
  RESET(F,'#4:SYSTEM.APPLE');
  I:=BLOCKWRITE(F,BUFFER,32);
  CLOSE(F);
END.
```

Este programa no tiene nada de misterioso y los números que se escriben directamente en el disquete representan los códigos de operación de las instrucciones máquinas que van a sustituir la espera del RESET por la espera de la pulsación de cualquier caracter.

```
LDA $ C000
BPL $ F3C3
STA § C083
JMP $ D69E
```

También se sustituyen los caracteres ASCII RESET por (RET) en el mensaje.

Thierry Leconte

Visto y no visto (o la instrucción 'NEW')

Cuando damos la instrucción 'NEW' el programa aparentemente desaparece. Sin embargo no es así. En efecto vamos a escribir un pequeño programa, por ejemplo el de la figura-1. Si miramos ahora a su forma codificada (recordar que para ello hay que decir CALL-151 y mirar a partir de $800) veremos que el programa se acaba con tres '00' consecutivos. Ahora volvemos al Basic con un Ctrl-C por ejemplo y ejecutamos 'NEW'. Como se puede comprobar con 'LIST' el programa ya no está (si todavía está más vale que empecéis a preocuparos de la salud de vuestro APPLE. . .).

Ahora volvemos otra vez al monitor y miramos a partir de la dirección $800. Como se ve por comparación con lo anterior, lo único que cambió fueron los octetos $801 y $802 que se transformaron en '00', indicando con ello que el programa se acaba justo al principio, en otras palabras, que no hay programa alguno. Por eso al decir 'LIST' el programa no escribe nada. Para comprobar que el 'NEW' solo tiene como efecto el poner en cero dichos octetos, vamos a restaurar los antiguos valores de estos:

```
]LIST

10  PRINT "HOLA SOY EL APPLE"
20  PRINT "AQUI ESTOY"
30  END

]CALL-151

*800

0800- 00
*
 1A 08 0A 00 BA 22 48
*       10  PRINT " H

0808- 4F 4C 41 20 53 4F 59 20
*       O  L  A     S  O  Y

0810- 45 4C 20 41 50 50 4C 45
*       E  L     A  P  P  L  E

0818- 22 00 2C 08 14 00 BA 22
*       "     ,           20  PRINT "

0820- 41 51 55 49 20 45 53 54
*       A  Q  U  I     E  S  T

0828- 4F 59 22 00 32 08 1E 00
*       O  Y  "     /           30

0830- 80 00 00 00 4C 9E 95 A9
*3D08 END

]LIST

10  PRINT "HOLA SOY EL APPLE"
20  PRINT "AQUI ESTOY"
30  END

]NEW

]LIST


]CALL-151

*801:1A 08

*3D08

]LIST

10  PRINT "HOLA SOY EL APPLE"
20  PRINT "AQUI ESTOY"
30  END
```

Fig. 1

$801: 1A Ø8

Ahora volvemos al Basic y decimos 'LIST' y ... Hop! El programa aparece de nuevo!.

Una cosa más. El programa puede borrarse y seguir ejecutándose sin más siempre y cuando no haya saltos del tipo 'GOTO' o 'GOSUB' pues en ese caso el interpretador se pone a buscar la dirección desde el octeto $801 y si llega a la señal de fin de programa ($00, $00) sin haberla encontrado, entonces da error por línea no definida. Como este es siempre el caso al poner en cero las direcciones $801, $802, entonces no podremos ejecutar este tipo de instrucciones a menos que previamente hayamos restaurado el antiguo valor de estos.

Para ver que esto es posible os podéis referir al programa de la figura 2.

```
]
]LIST

10 X = 8 * 256:A1 =  PEEK (X + 1)
    :A2 =  PEEK (X + 2)
20  REM  *AHORA VA A DESAPARECER
30  POKE X + 1,0: POKE X + 2,0
40  INPUT "PARAR?";A$
50  IF A$ = "S" THEN  STOP
55  REM *AHORA APARECE DE NUEVO
56  REM *PARA PODER HACER EL'GOTO
    '
60  POKE X + 1,A1: POKE X + 2,A2:
    GOTO 30

]RUN
PARAR?NO
PARAR?NO
PARAR?S

BREAK IN 50
]LIST


]POKE X+1,A1:POKEX+2,A2

]LIST

10 X = 8 * 256:A1 =  PEEK (X + 1)
    :A2 =  PEEK (X + 2)
20  REM  *AHORA VA A DESAPARECER
30  POKE X + 1,0: POKE X + 2,0
40  INPUT "PARAR?";A$
50  IF A$ = "S" THEN  STOP
55  REM *AHORA APARECE DE NUEVO
56  REM *PARA PODER HACER EL'GOTO
    '
60  POKE X + 1,A1: POKE X + 2,A2:
    GOTO 30
```

Fig. 2

Un programa larguísimo:

Aprovechando nuestros conocimientos anteriores vamos a ensayar otro truco. En el programa de la figura 1 cambiamos los octetos $801 y $802 con los valores $FF. Luego damos 'LIST'. ¿Qué pasa?.

Pues que el programa no se acaba nunca de listar y aparecen todo tipo de símbolos raros en la pantalla.

Para parar el listado, si todavía no lo habéis logrado, utilizar 'RESET'. . .

Jaime Díez Medrano.

Radiografía de un programa Applesoft

Cuando escribimos un programa, éste no se almacena tal y como lo introducimos, sino de una forma condensada mediante el uso de 'claves'. En esencia, a cada instrucción le corresponde un número hexadecimal de un octeto. Así por ejemplo la instrucción 'input' se codifica con el número $84, y el 'print' con el $BA.

Para poder 'explorar' en las entrañas de vuestros programas, es suficiente con decir las palabras mágicas: CALL151 y os encontrareis metidos en el misterioso mundo del lenguaje de máquina. El objetivo que perseguimos con esto es únicamente mirar a partir de la dirección $800 a ver que hay. (Para los 'nuevos' debo aclarar que esta es la dirección a partir de la cual se encuentra el programa codificado).

Lo que vereis se muestra en el ejemplo de la figura 1. Es mejor que exploreis mediante programas simples como el del ejemplo pues se evitan complicaciones. Básicamente la estructura de una línea de programa es como sigue:

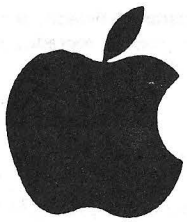
a1/ah/11/1h/c1/c2. . .cn/00/
cn+1. . . cm/00

Para una mayor comprensión de lo que esto significa, debo aclarar que las letras 'al' y 'ah' son la dirección donde empieza la siguiente línea codificada en hexadecimal ('1' y 'h' son la parte baja y alta respectivamente de esta dirección).

Las letras '11' y 1h' son el número de línea del programa también en hexadecimal, y 'c1' . . . 'cn' ó 'cn+1'. . . 'cm' son los códigos de las instrucciones. Por último, las instrucciones se acaban siempre con '00'.

Para que se pueda comprender mejor este sistema, he preferido escribir directamente la traducción de cada octeto en el ejemplo 1, pero para profundizar más creo que deberíais explorar por cuenta propia.

Otra cosa, los nombres de variables, y número de línea de los 'goto' o 'gosub' se codifican en formato ASCII. El programa empieza siempre con un '00' en la dirección $800 y acaba con tres '00' consecutivos.



Goto X y Gosub X

Una vez que ya hemos visto cómo se codifican los programas, vamos a desarrollar algunas aplicaciones simples.

Una de estas consiste en transformar una línea del programa de forma que cambie la jnstrucción. Lo mejor sería, a juzgar por lo que hemos visto, cambiar el número de la línea que sigue a un 'GOTO' o 'GOSUB'. Para ello sólo tenemos que resolver unas pocas cuestiones:

A) Reservar espacio para números de línea grandes. En efecto, si escribimos por ejemplo:

10 GOTO 30.

en el programa codificado, el número de octetos reservados para el número de línea será de dos pues 30 tiene sólo dos digitos. Por tanto, si queremos cambiarlo por 10 GOTO 1230 no podremos pues 1230 tiene cuatro digitos y por tanto no podemos 'insertarlo' en el espacio reservado a esta línea. Para resolver esto, tendremos que declarar por ejemplo: 1 GOTO 10000 con lo cual reservamos sitio para números de línea de hasta cinco dígitos.

```
PR#1
]POKE 1657,50

]LIST

0  GOTO 00020
1  GOTO 00000
20   INPUT "GOTO O GOSUB(1 O 2)";A
30   IF A <  > 1 AND A <  > 2 THEN 20
35   REM  *PONEMOS EN $810 EL CO-
36   REM  **DIGO DEL 'GOTO' O DEL
37   REM  **'GOSUB', SEGUN SEA EL
38   REM  **CASO.
39   REM
40   IF A = 1 THEN  POKE 2064,171
50   IF A = 2 THEN  POKE 2064,176
60   INPUT "NUMERO DE LINEA?";NL
61   REM
62   REM  **LA LINEA 70 SEPARA LOS
63   REM  **DIGITOS DEL NUMERO DE
64   REM  **LINEA, Y LOS CONVIERTE
65   REM  **EN ASCII.LUEGO LOS PO-
66   REM  **NE EN LAS DIRECCIONES
67   REM  **$811 A $815.
68   REM
70   FOR I = 1 TO 4:N1 =  INT (NL / 10): POKE
     2070 - I,NL - N1 * 10 + 48:NL = N1: NEXT

80   IF NL > 10 THEN  PRINT "ERROR": GOTO 60
     : REM  *EN ESTE CASO, EL NUMERO DE LINE
     A
81   REM  **ERA MAYOR DE LO PERMITIDO
90   POKE 2065,NL + 48: REM  **ULTIMO DIGITO
100  GOTO 1: REM *SALTAMOS A LA INSTRUCCION

110  REM  **'GOTO'O'GOSUB' MODIFICADA
200  PRINT "ESTOY EN LINEA 200": RETURN
210  PRINT "ESTOY EN LINEA 210": GOTO 20

]RUN
GOTO O GOSUB(1 O 2)1
NUMERO DE LINEA?210
ESTOY EN LINEA 210
GOTO O GOSUB(1 O 2)2
NUMERO DE LINEA?200
ESTOY EN LINEA 200
GOTO O GOSUB(1 O 2)11
GOTO O GOSUB(1 O 2)300
GOTO O GOSUB(1 O 2)1
NUMERO DE LINEA?300

?UNDEF'D STATEMENT ERROR IN 1
*
]LIST

10   INPUT A
20   PRINT A
30   FOR I = 1 TO 10
40   NEXT
50  Y = X + Y
60  Y = X - Y
70  Y = X * Y
80  Y = X / Y
90   GOTO 100
100   GOSUB 010
110   RETURN

]CALL-151

*800

0800- 00   (Siempre es cero)
*
 08 08 0A 00 84 41 00
*    (!)10 INPUT A   /      (!)
0808- 0F 08 14 00 BA 41 00 1B
*          20 PRINT  A /

0810- 08 1E 00 81 49 D0 31 C1
*          30  FOR I = 1 TO

0818- 31 30 00 21 08 28 00 82
*       1 0  /        40   NEXT

0820- 00 2B 08 32 00 59 D0 58
*       /        50    Y  = X

0828- C8 59 00 35 08 3C 00 59
*         + Y /        60    Y

0830- D0 58 C9 59 00 3F 08 46
*         = X - Y /        70

0838- 00 59 D0 58 CA 59 00 49
*           Y = X * Y  /

0840- 08 50 00 59 D0 58 CB 59


*
           80    Y = X / Y
0848-.00 52 08 5A 00 AB 31 30
*       /        90 GOTO 1  0
0850- 30 00 5B 08 64 00 B0 30
*       0 /        100 GOSUB 0
0858- 31 30 00 61 08 6E 00 B1
*       1  0 /        110  RETURN
0860- 00 00 00 02 29 CA 49 3A
       / / / FIN DE PROGRAMA.
```

B) El segundo problema está en saber la dirección donde se encuentra la instrucción 'GOTO' o 'GOSUB' a modificar. Para no complicarnos la vida adoptamos la siguiente solución. (ver programa de ejemplo):

```
0 GOTO 00020
1 GOTO 00000
```

De esta forma, al estar estas dos líneas justo al principio de programa, sabremos en qué dirección de la memoria se encuentran codificadas.

La línea cero tiene un salto a la dirección del principio del programa y podeis poner el número que querais. Fijaros bien que hay que poner un número de cinco dígitos, o sea que si es preciso pondréis ceros por delante.

La línea 1 contiene la instrucción 'GOTO' o 'GOSUB' a modificar por programa, en la que también reservamos espacio para números de línea de hasta cinco digitos.

De esta forma, y después de explorar en la memoria, veremos que la instrucción 'GOTO' o 'GOSUB' de la línea 1 empieza en la dirección $810 ó 2064, y los octetos con el número de línea son los que van desde $811 a $815 (2065 a 2069).

Con estos datos he escrito el programa de la figura -2, que os aconsejo estudieis detenidamente, poniendo especial cuidado en escribir las líneas o y 1 sin 'REM's' pues modificarías las direcciones en que se encuentra el 'GOTO' o 'GOSUB'. Por cierto que para cambiar el uno por el otro basta con poner el código correspondiente ($AB y $BO respectivamente). en la dirección $810 (2064).

Por lo demás, he procurado poner todos los comentarios que me han parecido útiles para una mayor comprensión del programa ejemplo.

Espero que os guste este truco y encontreis aplicaciones interesantes de éste. En un próximo artículo describiré un ejemplo de aplicación muy útil.

Espero también que me comuniquéis cualquier sugerencia que os pareza interesante a propósito de este truco.

Jaime Diez Medrano.

** Si os parece debiera extenderme más o menos en explicaciones decírmelo por favor.*

Como generar notas musicales

Puede emitirse un "bip" con el Apple II mediante: X = PEEK (–16336)

Se obtiene así un sonido que no es modulable.

También mediante:

PRINT CHRS (07)

Se obtiene un pitido correspondiente al carácter ASCII Bell (07 hex). Todo esto es normal.

Sin embargo, el Apple tiene posibilidad de cuatro octavas. ¿Por qué no utilizarlas? Se presenta un programa en lenguaje máquina, a introducir mediante POKE.

```
20  REM
190  REM RUTINA EN L
ENGUAJE MAQUINA
199  REM
200  DATA 173,48,192
,136,208,5,206,1,3,2
40,9,202,208,245,174
,0,3,76,2,3,96,0,0
210  FOR I = 770 TO
792
220  READ X: POKE I,
X
230  NEXT I
240 DF = 768:DL = 76
9
```

Para producir una nota se recurre a:

```
500   CALL 770
```

La altura (frecuencia) de la nota, debe introducirse mediante POKE en 768, con valores de 0 (nota muy aguda) a 255 (nota muy grave).

La duración de la nota debe introducirse mediante POKE en 769, con valores de 1 (nota muy corta) a 254 (nota muy larga). Se aconsejan los valores

50	♩
100	𝅗𝅥
150	𝅗𝅥.
200	𝅝

En la línea 240 se han definido DF como "dirección frecuencia" y DL como "dirección duración", para emplear en la POKE.

Para obtener todas las posibles notas (y más) introducir:

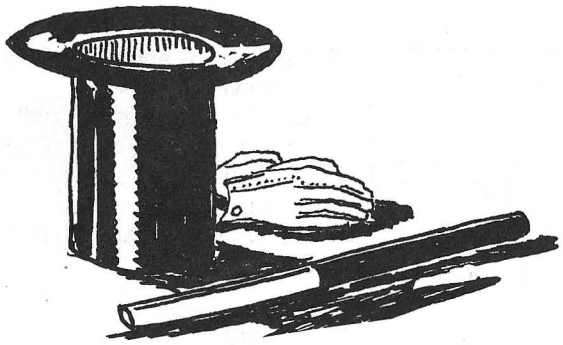
```
90  REM
100  INPUT "DURACION
 ? ";D
110  IF D < 0 OR D >
 254 THEN 100
120  FOR I = 255 TO
1 STEP  - 1
130  POKE DF,I: REM
ALTURA
140  POKE DL,L: REM
DURACION
150  CALL 770: REM S
UBRUTINA
160  NEXT
170  END
```

Nota: Puede comprobarse que las notas muy agudas (I=30 e inferiores) apenas se oyen. Si D es muy bajo (1 á 5) no llega a oirse la nota.

La tabla de correspondencia es la siguiente.

	OCTAVA			
	BAJA	MEDIA BAJA	MEDIA ALTA	RUTA
DO	255	128	64	32
DO#	242	121	60	30
RE	228	114	57	29
RE#	215	108	54	27
MI	203	102	51	25
FA	192	96	48	24
FA#	181	91	45	
SOL	171	85	43	
SOL#	161	81	40	
LA	152	76	38	
LA#	144	72	36	
SI	136	68	34	

Miguel Solano



Para pasar de lenguaje máquina a DATA

Este corto programa para Apple 2 le permitirá transformar un subprograma en lenguaje de máquina en una serie de datos en forma DATA en Basic.

A las preguntas «PRINCIPIO:» y «FIN», debe responder con las direcciones de principio y fin del subprograma en lenguaje de máquina que vaya a convertir (direcciones decimales).

A la pregunta: «NUMERO DE LINEA:», indicará el número de la primera línea del programa Basic en la que deberán implantarse los datos del programa en lenguaje máquina. Este número de línea debe ser estrictamente superior al número de la última línea del program Basic; es decir, que los datos sólo pueden implantarse al final del programa Basic, si no, el sistema se «planta».

El programa que publicamos crea las líneas de DATA. Hecho esto, sólo queda por hacer DEL 5-150 y crear un bucle FOR-NEXT.

Observación: el LOMEN de la línea 5 hay que adaptarlo según su configuración de memoria. Sencillamente, hay que prever un poco de sitio entre el programa y las primeras variables colocadas a partir de LOMEN, porque mi programa implanta los datos sin ocuparse de LOMEN. De todas maneras, haciendo DEL 5-150, se vuelve a modificar el valor de LOMEM.

Pascal Anquetin

Del lenguaje de máquina a DATA

```
]LIST

5  LOMEM: 24576
10  INPUT "COMIENZO : ";DE: INPUT
    "FIN : ";FI
20  INPUT "NUMERO DE LINEA : ";LI
    : IF LI > 63900 THEN 20
30 X1 =  INT (LI / 256):X2 = LI -
    X1 * 256
40 A =  PEEK (175) - 3:B =  PEEK
    (176):AB = A:AH = B:A = A +
    2
50  POKE B * 256 + A,X2: POKE B *
    256 + A + 1,X1
60 A = A + 2: POKE B * 256 + A,13
    1:A = A + 1: POKE B * 256 +
    A,32
70 A$ =  STR$ ( PEEK (DE)):LO = L
    O +  LEN (A$) + 1: IF LO > 2
    00 THEN LO = 0:EG = 1: GOTO
    100
80  FOR L = 1 TO  LEN (A$): POKE
    B * 256 + A + L, VAL ( MID$
    (A$,L,1)) + 48: NEXT L:A = A
     +  LEN (A$) + 1: POKE B * 2
    56 + A,44
90 DE = DE + 1: IF DE <  = FI THEN
    70
100  POKE B * 256 + A,0: POKE B *
    256 + A + 1,0: POKE B * 256 +
    A + 2,0
110 A = A + 1:B = B +  INT (A / 2
    56):A = A - ( INT (A / 256) *
    256)
120  POKE AH * 256 + AB,A: POKE A
    H * 256 + AB + 1,B
130 A = A + 3:B = B +  INT (A / 2
    56):A = A -  INT (A / 256) *
    256: POKE 175,A: POKE 176,B
140  IF EG = 1 THEN EG = 0:LI = L
    I + 10: GOTO 30
150  END
```

Como burlarse del Applesoft

El Applesoft rehusa tener en cuenta las cadenas de caracteres que contengan una coma, un punto y coma o dos puntos bien sea partiendo del teclado o del disquete. En este caso, emite el mensaje EXTRA IGNORED y sólo registra la porción de serie que precede al carácter prohibido.

El problema puede soslayarse mediante el empleo de sucesivos GET o de un subprograma ensamblador que sustituya la función INPUT.

También hay un modo muy sencillo: teclear " antes de la cadena propiamente dicha.

Para evitar este problema en el momento de la lectura de un fichero en disco, en la creación del fichero registrar " antes de cada cadena. Sencillamente, puede hacerse mediante PRINT CHR$(34)+A$.

Cuando el INPUT para la lectura, el Applesoft quedará confundido por el " y todo funcionará bien (ver el programa).

Señalaremos una limitación: la cadena no debe contener el símbolo " que indicaría al INPUT el final de la serie.

Roland Jost

```
]LIST

5 D$ =  CHR$ (4)
10  INPUT A$: REM  INTRODUCIR UNA
     CADENA DE CARACTERES CONTEN
    IENDO . O ; O : Y PRECEDIDA
    POR "
20  PRINT A$
30  PRINT D$"OPEN PRUEBA-INPUT": PRINT
    D$"WRITE PRUEBA-INPUT": PRINT
     CHR$ (34) + A$: PRINT D$"CL
    OSE"
35  REM   CHR$(34)= CODIGO ASCII
    DE ". SITUA UNA " AL COMIENZ
    O DE LA CADENA SALVADA EN DI
    SCO
40  PRINT "RELECTURA"
50  PRINT D$"OPEN PRUEBA-INPUT": PRINT
    D$"READ PRUEBA-INPUT": INPUT
    C$: PRINT D$"CLOSE"
60  PRINT C$
```

Para que «crezcan» sus disquetes

```
B       00 01 EF D8 00
*300L

0300-   A9 00       LDA   #$00
0302-   85 48       STA   $48
0304-   20 F7 AF    JSR   $AFF7
0307-   A9 00       LDA   #$00
0309-   85 48       STA   $48
030B-   A9 FF       LDA   #$FF
030D-   8D FB B3    STA   $B3FB
0310-   A9 E0       LDA   #$E0
0312-   8D FC B3    STA   $B3FC
0315-   20 FB AF    JSR   $AFFB
0318-   A9 00       LDA   #$00
031A-   85 48       STA   $48
031C-   60          RTS
```

¡Viva la transgresión octal!

Un consejo para aumentar la capacidad de sus disquetes de 2,6 Ko; es decir, para liberar los once sectores de la tercera pista, reservados para el DOS 3.3.

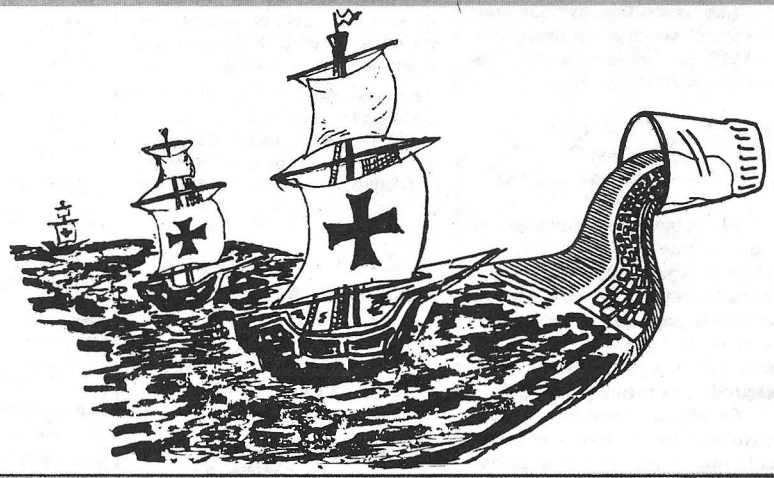
Ante todo, teclee el siguiente programa en el monitor. Después, salvaguardarlo mediante «BSAVE PROG, A$ 300, L & 1D.

Ponga el disquete en la unidad 1 y termine triunfalmente bien mediante BRUN PROG o bien con BLOD PROG, o CALL 768.

Richard Mau

ATOM-ACORN

Vamos AtoMar algo

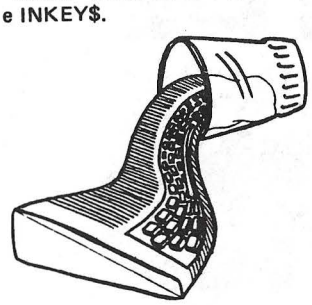


Si vas ATOMar algo, que sea fresquito, como estos trucos que hemos sacado del congelador para tí. No dudes en enviar los descubrimientos de este verano ¡que nos refresquemos todos!

GET$ e INKEY$

El ALGO sobre el ATOM va de las direcciones #B001 y #FE71.

Cuando por primera vez vamos arealizar con un ATOM un programa de ésos que, mientras que se están ejecutando, pueden tomar diferentes caminos en función de que se pulse(n) o no determinada(s) tecla(s), nos encontramos con que el equipo "carece" de los comandos GET$ e INKEY$.



Para simular algo así podríamos pensar en utilizar INPUT's, pero entonces el programa quedaría parado hasta la introducción de un valor y la posterior e inevitable pulsación de la tecla RETURN. Y esto, evidentemente, no es lo deseado. Veamos un par de soluciones:

Una primera sería hacer uso del contenido de la posición #B001.

En el ATOM disponemos de una PIA 8255.

La PIA tiene tres puertos de ocho bits: El A (#B000), el B (#B001), y el C (#B002).

En este caso nos fijaremos en el puerto B.

Preguntando al ATOM qué contenido tiene (P.?#B001) nos contestará que 255.

Si escribimos el siguiente programa:

```
10 DO
20 P.?#B001
30 U.0
```

Y lo ejecutamos, rápidamente la pantalla se llena de 255's.

Pero, ¿y si, cuándo nos parezca, pulsamos la tecla Q?.

— Entonces en vez de 255 sale 239.

Si pulsáramos SHIFT, obtendríamos 127. Y pulsando ambas a la vez, 111.

Así que ya estamos frente a una clara solución.

Para un programa donde queremos pilotar una base según se pulse SHIFT o CTRL se mueve para un lado o para otro:

```
>>L.
    1 REM -BASE: ESTRUCTURA Y MOVIMIENTO-
   10 CLEAR4
   20 E=50;G=2;N=5
   30 GOS.c
  110xIF?#B001=191;N=7;GOS.c;E=E-3;N=5;GOS.c
  120 IF?#B001=127;N=7;GOS.c;E=E+3;N=5;GOS.c
  130 G.x
 3000cCOLOUR3
 3100 MOVEE,G
 3110 PLOTN,E,(G+6)
 3120 MOVE(E+2),G
 3130 PLOTN,(E+2),(G+6)
 3140 MOVE(E+3),G
 3150 PLOTN,(E+3),(G+10)
 3160 MOVE(E+4),G
 3170 PLOTN,(E+4),(G+6)
 3180 MOVE(E+6),G
 3190 PLOTN,(E+6),(G+6)
 3200 MOVE(E+1),(G+1)
 3210 PLOTN,(E+5),(G+1)
 3220 MOVE(E+1),(G+2)
 3230 PLOTN,(E+5),(G+2)
 3240 R.
```

¿Cuál es el problema de adoptar este método?.

— Sencillamente, que el contenido de la posición #B001 únicamente cambia cuando se pulsan determinadas teclas —pero sólo una cuantas—.

Y esto nos obliga a usar siempre unas teclas concretas.

La segunda solución sería hacer uso de una pequeña línea en Ensamblador: [JSR #FE71; STY #80;RTS;].

Aparentemente es más complejo, pero en realidad es muy fácil también.

En el caso del movimiento de la base, si queremos utilizar esta solucion habría que cambiar las primeras líneas por:

```
>L.1,130
    1 REM -BASE: ESTRUCTURA Y MOVIMIENTO TIPO II-
    2 DIMP-1
    5 E=50;G=2;N=5
   10 P.$21;[;JSR #FE71;STY #80;RTS;];P.$6
   20 CLEAR4
   30 GOS.c
  100xLINK TOP;R=?#80
  110 IF?R=60;N=7;GOS.c;E=E-3;N=5;GOS.c
  120 IF?R=92;N=7;GOS.c;E=E+3;N=5;GOS.c
  130 G.x
```

Mediante JSR #FE71 hacemos uso de una subrutina muy especial:

Almacena el código de la tecla pulsada en el Registro Y (uno de los tres registros de 8 bits presentes en la 6502).

Con STY #80 lo que hacemos es pasar este código desde el Registro Y hasta una posición de memoria RAM (en este caso la #80).

De manera que después ya sólo tenemos que ver qué número hay en esta posición, y tomar diferentes opciones según ello.

Esta solución es bastante mejor que la primera. Mediante ella el ATOM tampoco necesita esperar que se introduzca algo por el teclado, y además (y esto es ideal) permite usar cualquier tecla o conjunto de ellas.

Para saber qué códigos corresponden a cada tecla podeis utilizar algo así:

```
10 DIMP-1
20[JSR #FE71;STY #80;RTS;]
30 LI.T.
40 P."CODIGO:"?#80'
50 G.30
```

Victor M. Delgado.

ATOM – ACORN

Las posibilidades que nos ofrece el sistema operativo y el BASIC de leer un caracter del teclado son las siguientes:

BASIC —— INPUT A$
S.O. —— rutina OSRDCH
—— rutina OSECHO

El problema de la primera es que hay que pulsar "RETURN", el de la segunda es que hay que soltar primero una tecla y luego apretarla para que la reconozca, y el de la tercera es que no es más que una combinación de la segunda y la rutina OSWRCH.

En algunos programas, especialmente los de juegos es particularmente interesante el poder actuar de forma repetitiva sobre una tecla (mando de juegos. . .). Aunque para conseguir esto bastaría con mantener pulsada la tecla REPT o instalar un pequeño interruptor en un costado para simular esto último, tiene el inconveniente que la rutina OSRDCH no "suelta prenda" hasta que no se pulsa una tecla, deteniendo toda posible acción, con pérdida de realismo.

Para simular esto presento este pequeño programa asembler dividido en dos partes.

La primera parte es opcional y sirve para almacenar el resultado leído en la posición # 80 para que el Basic vaya a buscarlo, si se programa en asembler no es necesario ya que la otra parte de la rutina deja el carácter leído en el registro A. Nótese que si no se pulsa ninguna tecla devuelve el valor # FF valor este dado en la línea 150.

El programa usa diversas subrutinas presentes en el sistema operativo, y son de uso interno del ordenador.

```
>L.
  10 DIM RR(2)
  20 P=#2800
  30[:RR2
  40 JSR RR0        Primera
  50 STA #80        parte
  60 RTS
  70 RR0
  80 PHP            Segunda
  90 CLD            parte
 100 STX #E4
 110 STY #E5
 120 JSR #FE71
 130 BCC RR1
 140 LDA @#FF
 150 LDX #E4
 160 LDY #E5
 170 PLP
 180 RTS
 200:RR1
 210 TYA
 220 LDX @#17
 230 JSR #FEC5
 240 LDA #FEE3,X
 250 STA #E2
 260 LDA @#FD
 270 STA #E3
 280 TYA
 290 JMP (#E2)
 300[
 310 END
```

La dirección de la línea 20 se puede cambiar a discrección, y representa la dirección donde estará situada la rutina. También se puede poner 20 DIM P(-1).

Si ensamblamos la rutina en # 2800, podremos luego cargar en memoria cualquier programa BASIC, haciendo un SAVE, y no alteraremos esta rutina.

Por utilizar las rutinas internas de decodificación, no es conveniente pulsar las teclas ↕ ↔ y LOCK por quedar congelado hasta que se apriete otra tecla. Ojo con ellas, y al abordaje.

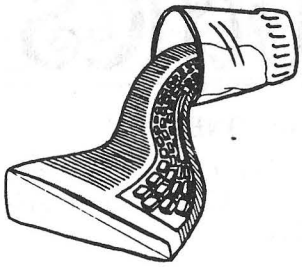
Gerardo Izquierdo Cadalso.

ATOM – ACORN
Leyendo teclas

Algunas veces es conveniente que un Programa pueda leer una pulsación, sin que el usuario tenga que apretar la tecla RETURN.

El siguiente Programa de ejemplo permite esto. Utiliza para ello una subrutina en código máquina que espera una pulsación y luego coloca el correspondiente código ASCII en la localización #80 antes de volver al Programa en BASIC. R continuación saca en Pantalla el código ASCII generado.

```
>L.
  1 REM LEYENDO TECLAS
  2 DIM P(-1)
  3 [:JSR #FFE6:STA #80:RTS:]
  4 PRINT"PULSE UNA TECLA"'
  5 LINK TOP
  6 @=0
  7 PRINT'"CODIGO= "?#80'
  8 GOTO 4
```



Veamos alguna característica del ATOM

Alguna vez, haciendo algo con el ATOM, hemos pulsado, por ejemplo CTRL-N, para elegir el modo paginado, para a continuación dar LIST y el resultado es «BLEEP ERROR 94». ¿Por qué este comportamiento tan insólito? Cuando activamos un carcter, este no sólo pasa por la rutina de salida a pantalla, sino que también se guarda en el bufer de entrada, siendo necesario el pulsar ESCAPE antes de introducir algún comando válido.

Esta propiedad la p odemos utilizar en algunos casos puesto que si queremos borrar la pantalla, para después escribir «TITULO» normalmente hacemos: PRINT $12, «TITULO».

Podemos hacer PRINT «TITUL donde «º» es el resultado de hacer CNTRL-L, y no saldrá representado en la pantalla, sino que borrará la pantalla, y esto cada vez que listemos el programa con un LIST, o podemos hacer sonar el altavoz incluyendo un CTRL-G en una cadena, con lo que cada vez que listemos ese trozo, sonará el altavoz.

Lo dicho hasta anora tiene poca utilidad práctica, pues dificulta la edición de un programa, (para alterar una línea que contenga un carácter especial de estos, deberá primero asegurarse donde está, y luego al llegar a este punto la tecla COPY no actúa) pero puede representar un ahorro de memoria de dos o tres octetos por caracter. Y facilitar la edición en impresora de Programas.

Cuando en un programa queremos introducir algún carácter semigráfico alguna vez hemos tenido problemas al tener que introducirlo como $xxx, pero si antes de editar la línea hacemos ejecutar el comando «FOR I=0 TO 255; =8000? I=I; NEXT I», dispondremos en la parte superior de la pantalla de todo el juegó de caracteres, y posicionándonos con el cursor, podremos incluir el carácter que deseamos en nuestro programa utilizando la tecla COPY.

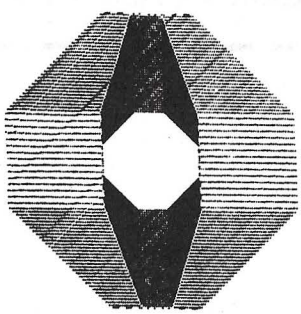
De sacar los REM a doble ancho haciendo:
98 b REM b ELECCION

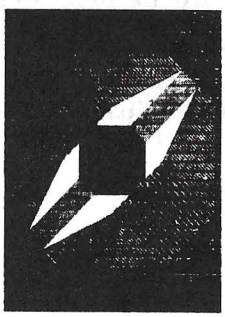
CTRL-N
CTRL-O

Inversión Video

Les propongo una rutina para el Atom (líneas 60 a 90) que os permitirá invertir un gráfico de alta resolución.

El programa siguiente os da un ejemplo de utilización: líneas 130 a 170.
¡A Vd. de jugar!





```
 60 DIM VV1;P=#2800
 70[:VV0;LDA@#80;STA#81;LDY@0
 74 STY #80;:VV1 LDA(#80),Y
 81 EOR@#FF;STA(#80),Y;INY
 84 BNE VV1;INC#81;LDA#81
 93 CMP@#99;BNE VV1;RTS;]
130 REM Ejemplo de utilizacion
140 X=30;Y=0;CLEAR4;F.I=1TO182
151 MOVE(128+X),(96+Y)
152 X=X+Y/16;Y=Y-X/16
160 PLOT2,(X-Y),(Y-X)
161 WAIT;PLOT2,(X+Y),(X+Y)
162 N.;LI.VV0;LI.#FFE3;G.150
```

Traductor de números

Este pequeño programa permite expresar números en castellano, por lo que puede resultar de gran utilidad aplicado a facturación o enseñanza.

Está escrito para traducir cantidades de cuatro cifras, aunque con ligeras modificaciones se podría adaptar para números de mayor o menor longitud. (La subrutina para escribir las decenas valdría perfectamente para las decenas de millar, etc.).

Tal y como está, los números deberán introducirse bajo la forma de cuatro caracteres (p.e. 0023 para 23), y, tras haberlo hecho, el ATOM responderá demostrando que sabe mucho de números.

En esta versión a continuación de la cantidad en letra escribirá siempre PTAS

Cualquier entrada incorrecta será, por supuesto, rechazada. (Se trata de un programa muy astuto).

V. Manuel Delgado

```
CANTIDAD?9999
NUEVE MIL NOVECIENTAS NOVENTA Y NUEVE PTAS

CANTIDAD?0623
SEISCIENTAS VEINTITRES PTAS

CANTIDAD?0003
TRES PTAS

CANTIDAD?6513
SEIS MIL QUINIENTAS TRECE PTAS

CANTIDAD?2001
DOS MIL UNA PTAS

CANTIDAD?7234
SIETE MIL DOSCIENTAS TREINTA Y CUATRO PTAS
```

```
?#FE=0
>P.$2
>L.
    1 P.$12;G.10
    2
    3
    4       TRADUCTOR DE NUMEROS
    5       VICTOR M. DELGADO
    6       (C) EL AUTOR Y EL ORDENADOR PERSONAL
    7
    8
    9
   10 T=#3A00;U=0
   15aP." PTAS"''"CANTIDAD....."
   16 P.$88888888;IN.$T;IFL.T<>4;P.$7;G.a
   17 U=0;F.I=0TO3;IFI?T(#300R I?T>#39;U=1
   18 N.;IFU=1;P.$7;G.a
   20 GOS.(100+((?T)-#30));IF?T>#30;P." MIL "
   30 IFT?1=#31;P."CIEN";IFT?2=#30;IFT?3=#30;G.a
   35 IFT?1=#31;P."TO ";G.b
   40 IFT?1=#35;P."QUINIENTAS ";G.b
   50 IFT?1=#37;P."SETECIENTAS ";G.b
   60 IFT?1=#39;P."NOVECIENTAS ";G.b
   70 GOS.(100+((T?1)-#30));IFT?1>#30;P."CIENTAS "
   80bIFT?2=#30;GOS.(100+((T?3)-#30));IFT?3=#31;P."UNA";G.a
   90 IFT?2=#31;GOS.(300+((T?3)-#30));G.a
   95 GOS.(400+((T?2)-#30)*10);G.a
  100 R.
  101 R.
  102 P."DOS";R.
  103 P."TRES";R.
  104 P."CUATRO";R.
  105 P."CINCO";R.
  106 P."SEIS";R.
  107 P."SIETE";R.
  108 P."OCHO";R.
  109 P."NUEVE";R.
  300 P."DIEZ";R.
  301 P."ONCE";R.
  302 P."DOCE";R.
  303 P."TRECE";R.
  304 P."CATORCE";R.
  305 P."QUINCE";R.
  306 P."DIECISEIS";R.
  307 P."DIECISIETE";R.
  308 P."DIECIOCHO";R.
  309 P."DIECINUEVE";R.
  400 R.
  420 P."VEINT";IFT?3=#30;P."E";R.
  421 P."I";GOS.(100+(T?3)-#30);IFT?3=#31;P."UNA"
  422 R.
  430 P."TREINTA";IFT?3=#30;R.
  431 GOS.r;R.
  440 P."CUARENTA";IFT?3=#30;R.
  441 GOS.r;R.
  450 P."CINCUENTA";IFT?3=#30;R.
  451 GOS.r;R.
  460 P."SESENTA";IFT?3=#30;R.
  461 GOS.r;R.
  470 P."SETENTA";IFT?3=#30;R.
  471 GOS.r;R.
  480 P."OCHENTA";IFT?3=#30;R.
  481 GOS.r;R.
  490 P."NOVENTA";IFT?3=#30;R.
  491 GOS.r;R.
  500rP." Y ";GOS.(100+(T?3)-#30);IFT?3=#31;P."UNA";R.
  501 R.
```

SOLUCIONES DELTRONICS

COMUNICACIONES

- ACOPLADORES ACUSTICOS de 300 Baudios, origen y respuesta + AUTO. Conectable a cualquier vía RS-232 serie.
- Modems 1200 Baudios
- Conversores de Interface serie-paralelo/paralelo-serie (y ambos a la vez) + Buffer de 59 Kb (80 Kb).

MEMORIA DE MASA para APPLE

WINCHESTER de 10 Mb de INSTALACION INTERNA.
Compatible PRODOS/CPM/PASCAL/DOS 3.3.
Incluye CONTROLADOR, FUENTE DE ALIMENTACION Y VENTILADOR para refrigerar todo el ordenador.

CONFIGURABLE por el USUARIO y particionable en distintos sistemas operativos.

2 X 655 KB

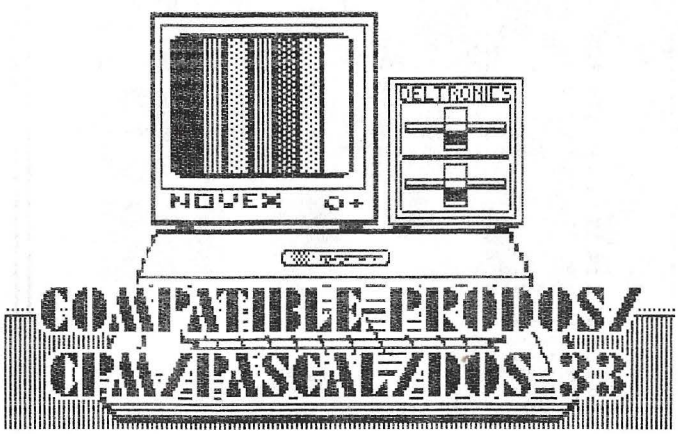
Y PUEDE LEER DISCOS DE 143 KB

Doble Floppy de 1,3 Mb (2 x 6,55 Kb).

Compatible PRODOS/CPM/PASCAL/DOS 3.3/ DIVERSI-DOS.

COMPATIBLE CON DISCOS NORMALES de 143 Kb.

INCLUYE CONTROLADOR Y UTILIDADES.

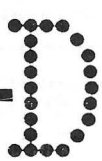


OTRAS SOLUCIONES

- TERMINALES CON TECLADO SEPARABLE, VIA AUXILIAR PARA IMPRESORA Y AMPLIAS POSIBILIDADES DE EDICION
- MONITORES EN COLOR MEDIA/ALTA RESOLUCION CON ENTRADAS PAL/RGB. 14 Pulgadas. COMPATIBLES IBM.
- IMPRESORAS VELOCIDAD 180 CPS Y CALIDAD TEXTO (35 CPS) INTERFACE SERIE + PARALELO IBM COMPATIBLES

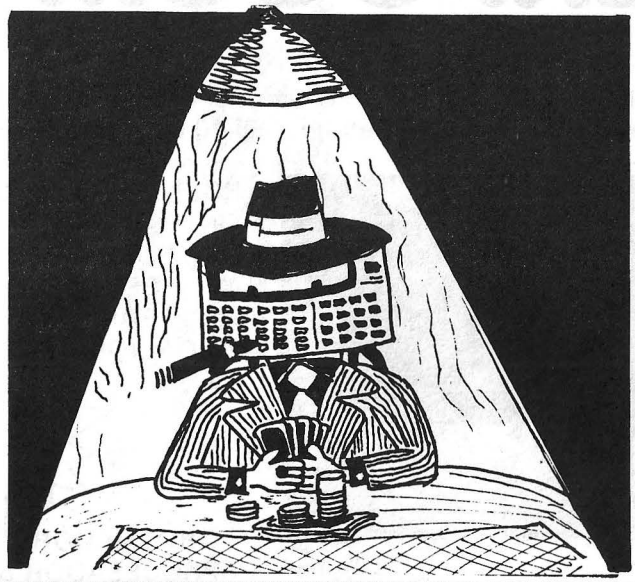
Enviar a DELTRONICS, S.A.
C/ Estébanez Calderón, 5 - 1° B
28020 MADRID

Sr. .....
EMPRESA .....
CARGO .....
DOMICILIO .....
C.P. ..... POBLACION .....
PROVINCIA ..... Tel. .....
INTERESADO EN .....
.....
.....
.....
.....



DELTRONICS S.A.
Estébanez Calderón, 5, 1.° B - 28020-Madrid
Tels. 450 76 09 - 616 22 75
Telex: 49739 Gerb e

CASIO FX 702 P



¡La mano un niño!. Si tu puedes "envidar más" ya sabes donde encontrarnos.

Acceso directo de las memorias

Veamos un truco difícil de utilizar, pero de inmensas posibilidades. Júzguelo por la siguiente manipulación del Casio 702 P. Haga VAC, después CLR ALL, introduzca en P8 una línea de STEPSTEP... STEPTO sin espacio. Espere dos segundos tras pulsar EXE y después OFF/ON. Hoy día eso es clásico, pero la continuación se realza.

Haga MODE 1, después DEFM 20 (EXE), siga con F1 P9 e introduzca la línea 1A (EXE). Después F1P8 e introduzca la línea 9999 AAAAA (EXE). Seguidamente se va a borrar (i) todo eso por CLR # 9, después F1 P8 y 9999 EXE. Si ha seguido bien todo, le quedan en ese momento 1925 pasos de programa disponibles. Finalmente teclee 5352 EXE: quedan 2013 pasos. Ahora todo está dispuesto para introducir el programa: Autoprogramación en PO, después VAC y teclear 5253 PRTA, 5254 PRTA, 5255 PRTA.

Finalmente haga MODE 0, después $ = T9$ y D$ = MID (1, 7). Está dispuesto.

```
1 $="ABCDEFGHIJKL
  MNOPQRSTUVWXYZ"
2 INP A:C$=MID(A,
  1):$=D$:T9$="HS
  "+MID(3,1)+C$+M
  ID(5,1)
5 PRT
```

¿Qué hace este programa? Haga RUN y responda 7EXE: ise presenta el contenido de la variable G! Si hubiera tecleado 1, hubiera obtenido A; 26 hubiera dado Z, etc. ¿Cómo funciona? Ante todo, el programa Autoprog ocupa exactamente 80 octetos. No es una casualidad, corresponde al tamaño de la memoria tampón que hemos creado. Seguidamente, si listamos de nuevo este programa, ¡sorpresa!, las líneas 5253 y 5254 ya no existen. En cambio se ha generado una línea nueva: 5247 PRT G. Ahora bien, si se sabe que el código de H es 47; el de S, 52 y que el tercer caracter de D$ (que proporciona un $ especial en la presentación) corresponde a PRT, todo se hace más claro. El cuarto caracter es A, es el parámetro de PRT, y el quinto es el retroceso de carro que termina la línea. Pero, ¿quién me explicará por qué en esas circunstancias raras, la instruccion MID (A, 1) no ocasiona un error? Pregunta suspense...

Denis Vincent

Abreviando los "Save"

Lo siguiente puede interesar a aquellos que utilicen el cassette con frecuencia. En efecto quiero señalar que son notablemente menores los tiempos para la carga de un programa y el contenido de las memorias, por una orden "SAVE ALL" o "LOAD ALL" que por "SAVE" o "LOAD" simples.

He aquí los tiempos medidos con un programa de 1.220 pasos:

SAVE "XXX" 2'28" (1220 pasos).
SAVE ALL "XXX" 1'11"(1220 pasos +26 memorias).
LOAD ALL "XXX" 1'28"(1220 pasos +6 memorias).

La razón aparece al escuchar la cinta, con un SAVE simple cada tren de señales es separado del siguiente por un "blanco" bastante largo, que no existe para "SAVE ALL".

Jean-Luc Florin.

Potencias de números negativos

¿Ha descubierto que su querida Casio realiza las potencias de los números negativos?. ¡Pero como todavía no había intentado $-1 \uparrow 2$!! El sistema es:

$(-1) \uparrow 2 = 1$
$(-1) \uparrow 3 = -1$
$(-1) \uparrow (-2) = -1$.

$0 \uparrow 0$ da ERR-3 lo que suele ser normal (compruébelo en otra máquina y $(-1) \uparrow 3{,}412$ da ERR-3 igualmente lo que es bastante normal (en calculadoras).

Serge Boisse.

Otro nuevo carácter

Este nuevo carácter tiene la ventaja de poder ser «generado» directamente en el teclado, sin utilizar la tortura por asfixia del Casio.

Ya conocíamos un carácter: el adoquín lleno (en la impresión). Señalemos de paso que este último se puede obtener en modo 0 y en modo 1, conectando al 702 la impresora (en OFF), y pulsando la tecla F1, y luego muy rápidamente la tecla B.

Pero vean Vds. esto:
— Conecte la impresora FP 10 (en OFF) al Casio;
— pulse F2;
— pulse F1 (sin soltar F2);
— suelte F2;
— pulse (rápido) B;
... y la pantalla visualiza F seguido de algunos blancos.

En la impresión, obtenemos el signo: «normal» porque la FP 10 no interpreta este código de la misma manera que el 702.

Frédéric Lacroix

¡No a la tortura!

Es inútil suprimir las pilas de la máquina para «poner patas arriba» la memoria: haga F1 P3 en modo PRT, con P4 y P5 llenos. Escriba CLR EXE y, durante la ejecución del CLR (¡hay que ir rápido!), pare la máquina.

Unas veces se encontrará con P3 borrado normalmente, y otras con todos los programas borrados, pero con un poco de práctica...

Si lista P3, P4 y P5 en MODE RUN, primero no ocurre nada, pero luego se encuentra con los caracteres especiales que tanto esperaba.

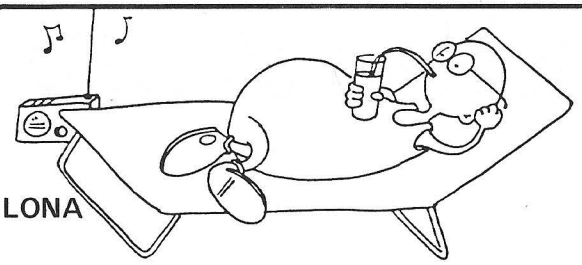
Para ponerlos en memoria, hay que listar en modo WRT, hacer aparecer las líneas interesantes para hacer S «...» EXE, después de haber hecho desaparecer lo que es conocido.

Un truco, si tiene tres líneas 10, escriba dos veces 10 EXE en modo WRT... ya no hay problemas.

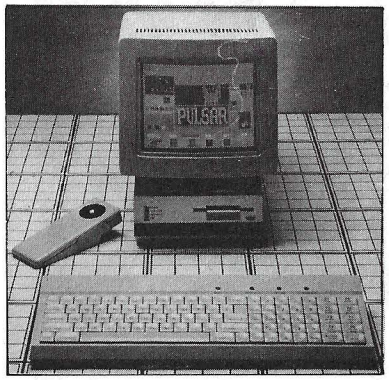
Observe que se produce un fenómeno similar, a veces, si se para la máquina durante la ejecución de un PASS. Le dejo la sorpresa de lo que ocurre entonces.

Christian Riche

EXPOCOM

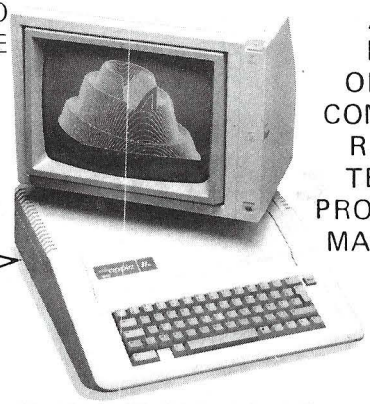


Villarroel, 68 Tienda - Teléfono: 254 88 13 - 08011 BARCELONA
Toledo, 83 Tienda - Teléfono: 265 40 69 - 28005 MADRID



ORDENADOR F1 CON DISCO
MONITOR FOSFORO VERDE
IMPRESORA DP-100
CABLE CENTRONICS
5 PROGRAMAS DE APLICACION

PRECIO MUY ESPECIAL CONSULTENOS

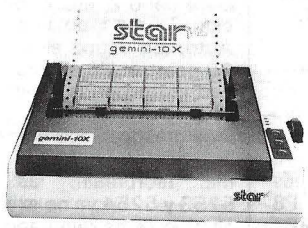


APPLE II e
EL UNICO ORDENADOR CON TODAS LAS RESPUESTAS TENEMOS LA PROGRAMATECA MAS COMPLETA

ORDENADOR
PROCESADOR 8087 A 4,77 MHZ. RAM 256K. AMPLIABLE a 768K. DISCO SONY 3 1/2" 720K. PANTALLA 640x256 PIKELS
92 TECLAS INCLUIDAS
PORT RS232
PORT CENTRONICS

195.839

IMPRESORA GEMINI-10X
MATRIZ DE PUNTOS
VELOCIDAD 120CPS.
BUFER 816 CARACTERES
MATRIZ ESTANDAR, ENFATIZADA
DOBLE PICA,
BLOQUE GRAFICO
ARRASTRE Y FRICCION
CENTRONICS



55.000

IMPRESORA ADMATE DP-100
VELOCIDAD 100CPS. MATRIZ DE IMPACTOS.
BIDIRECCIONAL OPTIMIZADA NORMAL
80 COLUMNAS NORMAL EXPANDIDO
COMPRIMIDO. COMPRIMIDO
EXPANDIDO CENTRONICS
O RS232.

PROMOCION DE VERANO

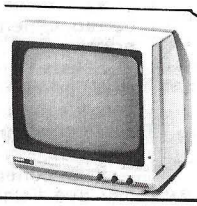
MACINTOSH
EL ORDENADOR MAS SENCILLO DE MANEJAR, CON 128K.RAM
UNIDAD DISCO DE 400K. CON MICROPROCESADOR 6800 DE 32BITS, PANTALLA INCORPORADA LO MISMO QUE EL RATON Y SALIDA IMPRESORA

560.789

ORDENADOR KATSON
DISCO 170K.
TARJETA CONTROLADORA
IMPRESORA DP-100
TARJETA CENTRONICS
MONITOR FOSFORO VERDE

220.000

MONITOR PHILIPS
ES EL MONITOR PREFERIDO POR SU ORDENADOR PARA LA ALTERESOLUCION



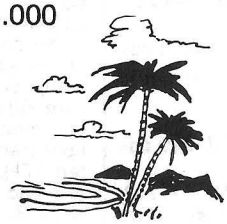
24.500

DISCOS 3" UNIDAD	1.300
DISCOS 5 1/4" SC/DD CAJA	3.995
CARTUCHO TINTA DP-100	1.000
CARTUCHO TINTA C. ITOH	1.000

IMPRESORA DP-100 COMMODORE
COMPATIBLE CON LOS ORDENADORES COMMODORE Y CON CABLE INCLUIDO

59.000

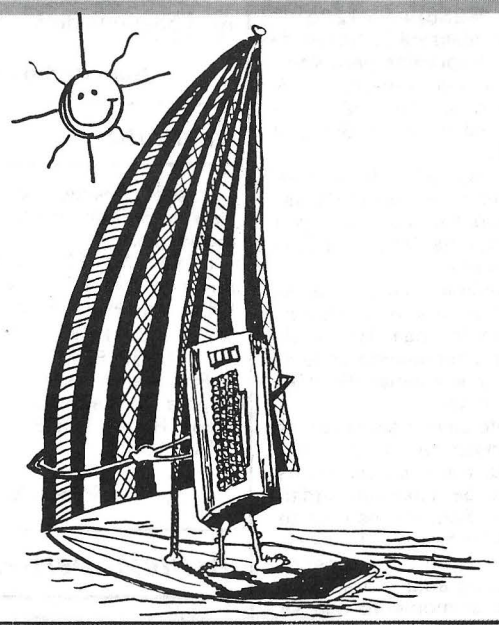
BUSCAMOS DISTRIBUIDORES



NEW BRAIN
EL PEQUEÑO GRAN ORDENADOR
MEMORIA RAM 32K. AMPLIABLE a I MGB. ROM 28K., 40 y 80 COLUMNAS
INTERFACE RS232 Y TAMBIEN PARA IMPRESORA, SALIDA MONITOR Y TV.

40.000

COMMODORE VIC-20



¡Dos mejor que uno!. Este duo seguro que dará guerra este verano. ¿Quién no recuerda tandems famosos?: musicales, cinematográficos, políticos, deportivos, electrónicos, etc... He aquí la selección preparada para tí con el fin de que no te aburras en la cálida hora de la siesta.

Modificar el encuadre de la pantalla

Para modificar el encuadre de la pantalla basta con actuar sobre las direcciones 36864 y 36865.

En la 36864 se determina la anchura del cuadro. Su valor inicial es 12; aumentándolo o disminuyéndolo, desplazará el cuadro a la izquierda o derecha respectivamente.

Ocurre igual con la dirección 36865, que opera sobre la altura y puede subir o bajar el cuadro modificando el valor de esta dirección que en principio es 38.

Todo ello se puede hacer sin alterar el correcto funcionamiento del sistema.

También se pueden modificar las dimensiones del cuadro transdormando los valores de las direcciones 36866 y 36867.

Actuando sobre la primera, aumentará o disminuirá el ancho del cuadro; igual ocurre con la segunda, pero en este caso existe modificación del sistema de impresión. ¡Cuidado con las falsas maniobras!

Esta es una idea a tener en cuenta para un eventual aumento de la pantalla del Vic 20, al que se reprocha (con razón) los pocos caracteres que contiene una línea.

Si tiene su Vic 20 desde hace algún tiempo y es un virtuoso de las teclas de control; sobre todo en el momento de corregir un programa; puede acelerar la velocidad del curos actuando sobre la dirección 36903 que contiene, al principio, el valor 55.

Cuanto mayor sea el valor contenido en esta dirección de memoria, más lentitud tendrá el cursor. Por ejemplo, si quiere un cursor MUY lento, haga: POKE 36903,254.

Atención a la tecla SPACE y sobre todo a la tecla INST-DEL, porque en lugar de borrar dos caracteres, corre el riesgo de encontrarse con tres líneas menos. Se vuelven a encontrar, más allá, otras direcciones con los mismos significados (36881, 36880, 36967, etc.).

Thierry Melin

Utilización de la función "del"

Para que una línea de programa no aparezca en listado, pero sea ejecutada, es necesario aplicar el metodo siguiente.

Teclear la línea Basic seguida de (:REM"), después pulsar RETURN. Volver al final de esta línea (después de las comillas), a continuación pulsad SHIFT e INST/DEL tantas veces como caracteres tenga la línea (desde el número de línea hasta las comillas) hacer lo mismo con INST/DEL (sin SHIFT).

Deberá aparecer el símbolo «T» invertido, que es el símbolo de borrado (tantas veces como caracteres hay en la línea).

Ejemplo:

Para borrar la línea 20

20 PRINT « INTENTO »: REM "

(22 veces la «T» invertida)

Este método puede ser aplicado a las notas (REM) para no hacer aparecer más que el contenido de la nota.

Ejemplo:

10 REM «(7 veces la «T» invertida) FIN DEL JUEGO.

Hará aparecer en el listado FIN DEL JUEGO.

Carol Limbard

Ultimas noticias sobre el VIC 20

Siguen algunos trucos que pueden ayudar a su VIC 20:

POKE 650, 128: repetición automática para todas las teclas.

POKE 650, 100: anula completamente la repetición automática.

POKE 650, 0: repetición automática normal (teclas de edición).

POKE 36864: desplazamiento de la pantalla en relación con la televisión; la posición de la imagen en el centro de la televisión se obtiene por POKE 36880,12.

POKE 36865: como la anterior, pero con un desplazamiento vertical; el centrado se obtiene por POKE 36881, 38.

POKE 36867→POKE 36883, no estrecha la pantalla.

SYS 64802: reinicializa completamente el VIC 20.

WAIT 653, 1: espera que se pulse la tecla SHIFT.

WAIT 653, 2: espera que se pulse la tecla G.

WAIT 653, 4: espera que se pulse la tecla CTRL.

Cuando se pulsa una tecla, PEEK (203) contiene el código correspondiente a la tecla pulsada.

Philippe Raffard

Como crear la instrucción Merge

La función MERGE no aparece en el Basic del Vic. Sin embargo, puede conseguirse fácilmente, con la condición, de tener espacio libre en memoria:

* Leer las direcciones 45 y 46 indicando la dirección del final de programa almacenado en RAM (no es obligatorio pero nos indicará la RAM vacía);

* Decir al Vic que el Basic comienza solamente en este lugar tecleando:

POKE 43, PEEK (45)-2 después: POKE 44, PEEK (46)

(PEEK (45)-2, con el fin de suprimir los 00 indicadores de fin de programa);

* Teclear: LOAD... (programa a cargar), después de colocar el disquete o K7;

* Una vez efectuada la carga, colocad de nuevo el punto de partida del Basic:

POKE 43,1 y POKE 44,E

Para el vic de base, E=16 : (16 * 256) + 1 + 4097...Si + 3K, E=4 : (4 * 256) + 1 = 1205.

El Vic realizará la fusión de los dos programas pero, ¡cuidado con los números de línea!, el orden no será restablecido (caso de los GOTO y GOSUB).

Si no hay saltos, funcionará incluso en desorden, el Vic espera encontrar los «00» para decidir que ha llegado al final de programa.

Se puede así tener (y salvaguardar) un programa comenzando en la línea 100 y siguientes, continuando en la línea 20 y siguientes (la 100 comprendida).

Excepto ciertas aplicaciones a determinar, es mejor, generalmente, tratar de poner los números de línea en orden creciente. Para ello es preciso que el programa a cargar tenga números de línea superiores al programa situado en RAM en el momento que se desea utilizar el comando MERGE.

Miguel Piperand

A próposito de semejanzas

El hecho de que CBM 64 y Vic sean muy parecidos en su organización interna, hace que sea muy frecuente un fenómeno fundamental.

En efecto, una gran proporción de trucos y astucias del CBM 64 se pueden emplear en el Vic 20. Veamos un ejemplo.

El título del CBM 64 contiene una interesante astucia de P. Parro, que permite modificar el habitual funcionamiento de los INPUT. Por supuesto, si trata de emplear el programita tal cual fracasará; pero sustituya el SYS 44025 de la línea 1030 por un SYS 52217 y la astucia del CBM 64 se convierte en una astucia del Vic...

Para sacar partido de esta propiedad, basta con saber que las direcciones del Vic pueden calcularse fácilmente partiendo de las del CBM 64 mediante una sencilla suma:

(Dirección del Vic) = = (Dirección del CBM 64) + + 8192

En la práctica este cálculo es válido para todas las direcciones del CBM 64 comprendidas entre $0000 y $BFED (en decimal 0 a 49133). Para las siguientes direcciones, hasta $FFFF, no puede emplearse este modo de cálculo.

¿Quién dijo que no hablamos bastante del Vic?

Juan Pedro Lalevée

A mano

Puede conseguir la repetición automática de todas las teclas con:

POKE 650, 128; un PRINT AT (x, y) que, a falta de Basic CBM, corresponde a: POKE 211, X: POKE 214, Y; SYS58732: PRINT «TEXTO».

Finalmente, una rutina de espera de caracteres de tecleo. Haga: POKE 198, 9: WAIT 198, 1.

Pascal Meurisse

Astucias periféricas

¿Hacer una «copia» de pantalla en la impresora? Fácil:

```
10 OPEN 1,3 : OPEN 2,4
20 PRINT CHR$(19);:FOR I
= 0 TO 999 : GET#1,X$ :
PRINT #2,X$ ;: NEXT I
30 CLOSE 1 : CLOSE 2
```

Perfecto para una impresora de 40 columnas. Si la suya imprime en 80 columnas o más, la corrección es sencilla:

```
10 OPEN 1,3 : OPEN 2,4
20 PRINT CHR$(19);:FOR I
=0 TO 24:FOR J=0 TO 39
30 GET#1,X$:PRINT #2,X$;
:NEXT J:PRINT #2:NEXT I
40 CLOSE 1 : CLOSE 2
```

Para el Vic 20 modifique los valores de I y J, que representan respectivamente el número de líneas y columnas de la pantalla.

¿Conocer la dirección de comienzo de un programa registrado en disquete? Sencillo:

10 OPEN 1, 8, 2, «nombre del programa»

```
10 OPEN 1,8,2, " nombre
del programa "
20 GET#1,A$,B$
30 PRINT ASC (A$+CHR$
(0))+256*ASC (B$+CHR$
(0))
40 CLOSE 1
```

Por supuesto, ¡esto también funciona con el Vic 20!

¿Salvar en casete una parte de la memoria cuando no se dispone de un monitor LM? Un poco más difícil:

Ejemplo: quiere almacenar los octetos comprendidos entre PRINCIPIO = LA + 256x HA y FIN + 1 = LB + 256 × HB (no olvide sumar 1 a la dirección final del programa a almacenar).

Haga:

SAVE «nombre del programa», 1, 1

Pulsad RETURN y STOP. Componga ahora:

```
POKE 780,253
POKE 253,LA:POKE 254,
HA
POKE 781,LB:POKE 782,
HB
SYS 62941
```

Se almacenará normalmente. La dirección 62941 ($F5DD) es el comienzo de la rutina SAVE del Basic. En el Vic 20, la dirección correspondiente en hexadecimal es $F675 (63093 decimal).

Hervé Lemarchand

Una palabra por otra

Este programa permite modificar la finalidad de las cuatro teclas de función. Con las teclas SHIFT, C =, ESCAPE y CTRL se pueden programar dieciséis palabras.

Estas dieciséis palabras se eligen entre las líneas 140 y 290 del programa. Puede cambiarlas, pero ninguna puede sobrepasar quince caracteres.

Este programa emplea el principio del vector de interrupción; es decir, que el programa cambia las direcciones 788 y 789 con el fin de que el ordenador ejecute la rutina en lenguaje máquina que se encuentra en la dirección 49152 cada 1/60 de segundo.

Frank Bonbled

```
70 I=49152
80 READ A:IF A=-1 THEN 100
90 POKEI,A:I=I+1:GOTO 80
100 :
110 REM--MENSAJE PARA PROGRAMAR--
120 REM  NO DEBERA SOBREPASAR 15 CARAC.
130 DIM ME$(15)
140 ME$(0)="PRINT"      :REM 'F1'
150 ME$(1)="INPUT"      :REM 'F3'
160 ME$(2)="POKE"       :REM 'F5'
170 ME$(3)="PEEK"       :REM 'F7'
180 ME$(4)="GOSUB"      :REM 'F1 + SHIFT'
190 ME$(5)="GOTO"       :REM 'F3 + SHIFT'
200 ME$(6)="DATA"       :REM 'F5 + SHIFT'
210 ME$(7)="READ"       :REM 'F7 + SHIFT'
220 ME$(8)="FOR"        :REM 'F1 + COMM.'
230 ME$(9)="TO"         :REM 'F3 + COMM.'
240 ME$(10)="STEP"      :REM 'F5 + COMM.'
250 ME$(11)="NEXT"      :REM 'F7 + COMM.'
260 ME$(12)="IF"        :REM 'F1 + CTRL'
270 ME$(13)="THEN"      :REM 'F3 + CTRL'
280 ME$(14)="SAVE"      :REM 'F5 + CTRL'
290 ME$(15)="VERIFY"    :REM 'F7 + CTRL'
300 :
310 FOR I=0 TO 15
320 IF LEN(ME$(I))>15 THEN PRINT"MENSAJE NO.";I;"DEMASIADO LARGO":GOTO 370
330 FOR A=1 TO LEN(ME$(I))
340 POKE 49407+I*16+A,ASC(MID$(ME$(I),A,1))
350 NEXT
360 POKE 49407+I*16+LEN(ME$(I))+1,0
370 NEXT
380 :
390 REM OCULTARLAS INTERRUPCIONES VERS 49152
400 FOR I=49244 TO 49256
410 READ A:POKEI,A
420 NEXT
430 SYS 49244
440 :
500 DATA 165,197,201,64,208,4,169,254,133,2,230,2,240,3,76,49,234,201,3,208,5
510 DATA 169,48,76,56,192,201,4,208,5,169,0,76,56,192,201,5,208,5,169,16,76,56
520 DATA192,201,6,208,5,169,32,76,56,192,76,49,234,174,141,2,224,0,240,20,224,1
530 DATA208,5,105,63,76,83,192,224,2,208,5,105,127,76,83,192,105,191,160,193
540 DATA32,30,171,76,49,234
550 DATA -1
560 DATA120,169,192,141,21,3,169,0,141,20,3,88,96

READY.
```

Por consiguiente, basta con chequear las posiciones de memoria 56320 ó 56321 para saber las acciones que desea efectuar el mando.

El programa reproducido permite desplazar un trazo por toda la pantalla por medio del mando del conector 1.

Franck Bonbled

Para utilizar los mandos de juego

El Commodore 64 tiene dos conectores con contactos que se pueden emplear con los mandos de juego o con un lápiz luminoso.

Cada uno de ellos admite un mando de juego. El lápiz luminoso sólo puede emplearse con el conector 1.

Un mando puede definirse como cinco interruptores: uno para el botón de tiro y otros para cada dirección.

Los cinco interruptores corresponden a los cinco primeros bits de las direcciones 56320 (conector 2) y 56321 (conector 1).

Ejemplo: tiro (interruptor 4)

Arriba (0)

Izquierda (2) — + — Derecha (3)

Abajo (1)

Normalmente, se ponen los bits a 1 cuando no se elige una dirección o bien cuando no se aprieta el botón de tiro.

```
READY.


7 REM - DESPLAZA UN SPRITE A TRAVES DE LA PANTALLA
8 REM - CON AYUDA DE UN "JOYSTICK" PUESTO EN EL PUERTO 1
10 PRINT"■■"
20 AB=53248:WA=56321
30 POKEAB+32,8:POKEAB+33,5
40 POKEAB+21,1:POKEAB+39,6:POKE2040,13
45 FORG=832TO891:READA:POKEG,A:NEXT
50 X=161:Y=128
60 M=PEEK(WA)
70 PRINT"■ ";M
80 E=0:D=3
90 IF(M AND 16)=0THENE=16:D=5
100 IFM=255-ETHEN60
101 IFM=254-ETHENY=Y-D:GOTO110
102 IFM=247-ETHENX=X+D:GOTO110
103 IFM=253-ETHENY=Y+D:GOTO110
104 IFM=251-ETHENX=X-D:GOTO110
105 IFM=246-ETHENX=X+D:Y=Y-D:GOTO110
106 IFM=245-ETHENX=X+D:Y=Y+D:GOTO110
107 IFM=249-ETHENX=X-D:Y=Y+D:GOTO110
108 IFM=250-ETHENX=X-D:Y=Y-D
110 IFY<29THENY=250
115 IFY>249THENY=28
120 IFX<0ANDT=0THENX=90:T=1:POKEAB+16,1
125 IFX<0ANDT=1THENX=255:T=0:POKEAB+16,0
130 IFX>255ANDT=0THENX=0:T=1:POKEAB+16,1
135 IFX>91ANDT=1THENX=0:T=0:POKEAB+16,0
140 POKEAB,X:POKEAB+1,Y
150 GOTO60
200 DATA7,255,224,15,225,240,31,255,248,63,255,252,127,255,254,252,60,63,252
210 DATA60,63,252,255,63,252,255,63,255,255,255,255,255,255,0,255,255,0
220 DATA255,126,126
230 DATA126,61,255,188,29,255,184,31,255,248,31,255,248,27,187,184,17,17,16

READY.
```

Correspondencias

Botón de disparo
bit 4 =1 no disparo
bit 4 =0 disparo
Dirección abajo
bit 1 =1 no
bit 1 =0 si
Dirección arriba
bit 0 =1 no
bit 0 =0 si
Dirección izquierda
bit 2 =1 no
bit 2 =0 si
Dirección derecha
bit 3 =1 no
bit 3 =0 si

Para leer direcciones.

Veamos el procedimiento para emplear los mandos de juego, si no se dispone de una explicación sobre ello.

Ante todo los puertos para los mandos se leen por los bits 3-0 y 4 de las direcciones 56321 (para el A) y 56320 (para el B). Los bits 3-0 proporcionan las ocho direcciones y el bit 4 el estado del pulsador.

Para la dirección, la línea

DIR =15 — (PEEK DIRECCION AND 15)

dará un valor comprendido entre 1 y 10.

El valor de la variable dirección corresponde a uno de los siguientes movimientos:

DIR= Corresponde a:
0 En reposo
1 Arriba
2 Abajo
4 Izquierda
5 Arriba e izquierda
6 Abajo e izquierda
8 Derecha
9 Arriba y derecha
10 Abajo y derecha

Para el control del pulsador:

Fuego = PEEK (DIRECCION) AND 16

Cuando Fuego vale 16, el botón no está pulsado y si Fuego es nulo sí lo está.

Notas:

Si tiene problema, invierta las direcciones (56320 =A, 56321 =B)

No es necesario comprar los mandos propios del CBM 64.

Si tiene una consola de juegos, pruebe sus mandos (¡economía!)

La dirección izquierda tiene el mismo efecto que la teda CTRL.

Ralentiza la presentación en la parte baja de la pantalla. ¡Asegúrese de que no ralentiza más que eso!

El programa 2 prueba la eficacia del procedimiento.

Normalmente conecte el mando en el acceso A.

Thierry Pauly

```
10 PRINT"■DEMOSTRACION JOYSTICK"
20 FORT=0TO10
30 READDIR$(T):REM LEE LOS NOMBRES DE LAS DIRECCIONES
40 NEXT
50 DATA,ARRIBA,ABAJO,,IZQUIERDA,ARRIBA E IZQUIERDA,ABAJO E IZQUIERDA,,DERECHA
55 DATARRIBA Y DERECHA,ABAJO Y DERECHA:REM NO OLVIDAR LAS 9 COMAS
60 DIR=15-(PEEK(56321)AND15):REM LEE EL VALOR DE LA DIRECCION
70 FU=PEEK(56321)AND16:REM LEE EL PULSADOR DE FUEGO
80 IF FU=0THEN PRINT "FUEGOOO!!!",:REMVERIFICA SI EL BOTON ESTA APRETADO
85 IFDIR$(DIR)=""THEN60:REM VERIFICA SI SE HA DADO UNA DIRECCION
90 PRINTDIR$(DIR):GOTO60:REM MUESTRA LA DIRECCION CORRESPONDIENTE

READY.
```

Reloj parlante

Este corto programa le permitirá la presentación permanente de un reloj en la esquina superior derecha de su fiel 64.

El programa cargador está en Basic, lo que permitirá aprovecharse de este utilitario a los que no dispongan de un ensamblador. Los 177 octetos del programa Reloj se implantan en memoria en el lugar del tampón casete (que empieza en $0330, 828 en decimal).

Se pueden colocar en otro lugar de la memoria. Para hacerlo hay que modificar algunos octetos. Le ayudarán la imagen de me-

Programa reloj

```
1 REM -----PROGRAMA RELOJ----GUY BIDI--
2 DATA120,173,20,3,162,89,141,224,3,142,20,3,173,21,3,162,3,141,225,3,142
3 DATA21,3,88,96,173,11,220,170,41,15,24,105,48,141,67,4,138,16,4,162,16,16
4 DATA 2,162,1,142,77,4,162,32,41,16,240,2,162,49,142,66,4,173,10,220,170,41
5 DATA15,105,48,141,70,4,138,74,74,74,74,24,105,48,141,69,4,173,9,220,170,41
6 DATA15,105,48,141,73,4,138,74,74,74,74,24,105,48,141,72,4,173,8,220,105
7 DATA48,141,75,4,169,32,141,65,4,141,76,4,141,79,4,162,15,157,24,4,202,208
8 DATA250,169,58,141,68,4,141,71,4,169,46,141,74,4,169,13,141,78,4,169,1,162
9 DATA13,157,65,216,202,208,250,76,0,0,120,173,224,3,141,20,3,173,225,3,141
10 DATA21,3,88,96
11 FORI=832TO 1000:READA:POKE I,A:NEXT
12 REM---- PUESTAEN HORA DEL RELOJ-----
13 INPUT"■■■■HORA";H
14 INPUT"■■MINUTOS",M:P=0:IFH>12 THEN H=H-12:P=1
15 IF H>9 THEN H=H+6
16 IF P=1 THEN H=H+128
17 POKE 56331,H:POKE 56329,0:M=INT(M/10)*6+M:POKE 56330,M:POKE 56328,0
18 PRINT"SYS 832=PUESTA EN MARCHA"
19 PRINT"SYS 994=PARAR EL RELOJ"
20 SYS 832
21 END

READY.
```

Desensamble del principio del programa

Modificación de la rutina IRQ. Los asteriscos indican los octetos que hay que modificar si desplaza el programa en la memoria.

```
.D  0340 78       SEI
.D  0341 AD 14 03 LDA  0314     Vector IRQ (Dirección inferior)
.D  0344 A2 59    LDX #59       * Principio programa reloj (dirección inferior)
.D  0346 8D E0 03 STA  03E0     * Dirección de almacenamiento IRQ
.D  0349 8E 14 03 STX  0314
.D  034C AD 15 03 LDA  0315     Vector IRQ (Dirección superior)
.D  034F A2 03    LDX #03       * Principio programa reloj (dirección superior)
.D  0351 8D E1 03 STA  03E1     * Dirección de almacenamiento IRQ)
.D  0354 8E 15 03 STX  0315
.D  0357 58       CLI
.D  0358 60       RTS
```

Imagen de memoria del programa en el tampón casete

Los 177 octetos del programa en el tampón casete. Los octetos subrayados representan valores que hay que cambiar en caso de un desplazamiento del programa.

```
.:  0340 78 AD 14 03 A2 59 8D E0 03 8E 14 03 AD 15 03 A2
.:  0350 03 8D E1 03 8E 15 03 58 60 AD 0B DC AA 29 0F 18   < 0359 Principio
.:  0360 69 30 8D 43 04 8A 10 04 A2 10 10 02 A2 01 8E 4D
.:  0370 04 A2 20 29 10 F0 02 A2 31 8E 42 04 AD 0A DC AA
.:  0380 29 0F 69 30 8D 46 04 8A 4A 4A 4A 4A 18 69 30 8D
.:  0390 45 04 AD 09 DC AA 29 0F 69 30 8D 49 04 8A 4A 4A
.:  03A0 4A 4A 18 69 30 8D 48 04 AD 08 DC 69 30 8D 4B 04
.:  03B0 A9 20 8D 41 04 8D 4C 04 8D 4F 04 A2 0F 9D 18 04
.:  03C0 CA D0 0A 00 3A 8D 44 04 8D 47 04 A9 2E 8D 4A 04
.:  03D0 A9 0D 8D 00 00 00 01 A2 0D 9D 41 D8 CA D0 FA 4C
.:  03E0 00 00 78 AD E0 03 8D 14 03 AD E1 03 8D 15 03 58
.:  03F0 60 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

■

moria de las direcciones afectadas y del desensamble de la rutina llamada del reloj, que figuran a continuación.

En caso de desplazamiento deberán modificarse un total de diez octetos.

El funcionamiento del reloj se efectúa mediante SYS 832 y la parada mediante SYS 994 (o STOP/RESTORE). Es preferible parar el reloj antes de cualquier acceso a casete o disquete bajo pena de conseguir efectos lamentables, como la pérdida de control del ordenador...

G. Bidi

Accidente en fichero

Quizá le haya ocurrido al abrir un fichero secuencial (SEQ, USR o PRG) escribir en él datos o olvidarse de cerrarlo (o sacar el disquete demasiado pronto del lector o tener una avería en la corriente en un mal momento). Cuando más tarde, trata de abrir este fichero en lectura, el Sed le manda el mensaje WRITE FILE OPEN, el fichero está inutilizable.

Sin embargo, la solución es muy sencilla. Como todos saben, la sintaxis de OPEN es:

OPENfn, dn, sa, «dr:fn, ft, modo»

(ver manual VC1541); siendo los parámetros W (escritura), R (lectura) o A (extensión de un fichero existente para añadir datos y, por tanto, ampliarlo). No figura en ningún manual, pero basta indicar M como modo para abrir el fichero accidentado en lectura.

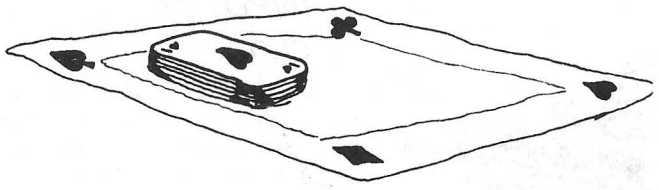
Ejemplo:

OPEN 2, 8, 2, «Nombre, S, M»

abre el fichero «Nombre» que no se había cerrado tras la escritura.

La mejor solución es leer de este modo todo el fichero, volverlo a escribir en el disquete con otro nombre, sin olvidarse de cerrar correctamente este nuevo fichero. Después, hay que borrar con la instrucción VALIDATE el fichero original que quedó abierto, porque es imposible cerrarlo bien.

Rafl Winzer



El grito del teclado

Este pequeño programa gustará a los que tengan un Vic 20, ya que su teclado resulta un poco atontado para un empleo demasiado largo. En esta máquina, es muy fácil emplear la rutina de interrupción para ejecutar un programa (escrito en el lenguaje de máquina) de una forma totalmente transparente para el usuario. El listado que presentamos saca partido de esta propiedad para crear un bip en el teclado que funcionará permanentemente sin perturbar el funcionamiento de sus programas habituales.

La rutina de interrupción (IRQ o INTERRUPT REQUEST) está desviada mediante la modificación de la dirección relativa contenida en 788/789. En estas direcciones colocamos la dirección de partida de la rutina bip, que se ejecutará unas sesenta veces por segundo.

El programa Basic es un cargador, que introduce en memoria los pocos octetos de nuestra rutina. Tras la ejecución, resonará un bip desde que se pulse una tecla en el teclado. Pulsando STOP/RESTORE se interrumpe el funcionamiento. Para volverlo a poner en marcha, hay que teclear SYS 848 y después POKE 36878,15. Finalmente, para emplear el magnetófono es indispensable para el funcionamiento del bip. Por supuesto, este pequeño programa es sólo un ejemplo de las posibilidades vía TRQ.

Basándose en este modelo, puede crear su propia rutina; por ejemplo, para cambiar el color de pantalla mediante pulsado de una tecla de función, etc.

Thierry Ponsada

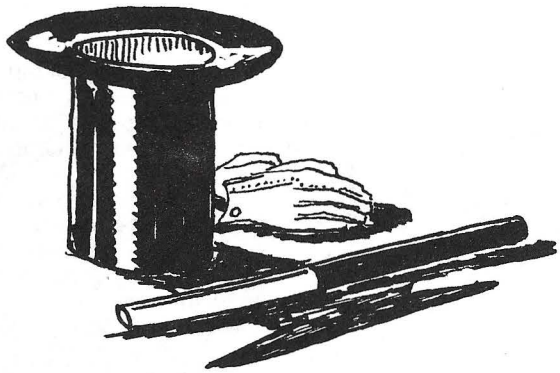
Rutina IRQ

```
100 DATA 120,169
110 DATA 93,141,20,03,169
120 DATA 03,141,21,03,88
130 DATA 96,173,203,0,201
140 DATA 64,240,13,169,242
150 DATA 141,10,144,160,255
160 DATA 136,208,253,140,10
170 DATA 144,76,191,234
180 :
200 FOR L=0 TO 35:READ A
205 POKE 848+L,A:NEXT L
220 :
300 POKE 36878,15:SYS 848

READY.
```

Rutina BIP Desensamblada

```
.,  0350 78       SEI
.,  0351 A9 5D    LDA #$5D     modifica vector IRQ
.,  0353 8D 14 03 STA $0314    en 788
.,  0356 A9 03    LDA #$03     y
.,  0358 8D 15 03 STA $0315    789 ($035D)
.,  035B 58       CLI
.,  035C 60       RTS
.,  035D AD CB 00 LDA $00CB    si tecla pulsada
.,  0360 C9 40    CMP #$40     entonces
.,  0362 F0 0D    BEQ $0371
.,  0364 A9 F2    LDA #$F2     bit
.,  0366 8D 0A 90 STA $900A    y
.,  0369 A0 FF    LDY #$FF     temporización
.,  036B 88       DEY
.,  036C D0 FD    BNE $036B    después
.,  036E 8C 0A 90 STY $900A    para sonido
.,  0371 4C BF EA JMP $EABF    y vuelta a IRQ normal
.,  0374 00       BRK
.,  0375 00       BRK
```

■



¡A la línea buena!

Es posible una tabulación vertical en CBM 64 con:

POKE 214, Y: PRINT"
[crsr abajo] TEXTO"

Y representa el número de la línea en la que hay que posicionar el texto. El descenso del cursor es necesario para que el texto se presente en la línea deseada, sin ello sólo se posicionará el cursor parpadeante en la línea correcta.

Combinado con un TAB(), este POKE permite simular el PRINT AT de algunas máquinas:

POKE 214, Y: PRINT
TAB (X) "[crsr abajo]
TEXTO"

Christophe Thomas

Rep, rep, repite

Al enchufar el CBM 64 algunas teclas son repetitivas y otras no. En realidad, existen tres posibilidades relacionadas con el valor atribuido al octeto de la dirección 650 (decimal):

1. 0-63: estado al enchufar.
2. 64-127: ninguna tecla repetitiva.
3. 128-255: todas las teclas repetitivas.

Estas posibilidades indican que los dos bits de mayor peso de la dirección 650 son fundamentales.

Si los bits 6 y 7 están a 0, el funcionamiento es normal.

Si sólo el bit 6 está en 1, ninguna tecla dispone de repetición.

Si el bit 7 está a 1, todas las teclas se convierten en repetitivas.

J. F. Brioux

No visto, no cogido

Para impedir el listado en pantalla de un programa Basic, basta con incluir en ese programa una línea como:

10 REM (shift/L)

Cuidado: (shift/L) indica el caracter obtenido pulsando ambas teclas a la vez.

En el momento en que se intente listar, el ordenador presentará:

10 REM ? SYNTAX ERROR

Evidentemente, puede elegir otro número de línea; lo más prudente no es colocar el REM de protección al principio del programa.

Christophe Thomas

INPUT de choque

Si se inspira en el programa que sigue, podrá introducir sin problemas en sus respuestas a INPUT todos los signos que habitualmente rechaza esta instrucción (",,). Además, bajo PETSPEED, el envío de una cadena vacía como respuesta no le meterá en READY.

P. Parro

Programa

```
5 REM SUPER INPUT
10 PRINT "SU RESPUESTA ";
20 GOSUB 1000
30 PRINT R$;LEN(R$)
40 GOTO 10
999 :
1000 FOR J=512 TO 600
1010 POKE J,0
1020 NEXTJ
1030 SYS 44025
1040 R$=""
1050 FOR J=512 TO 600
1060 IF PEEK(J)<>0 THEN R$=R$+CHR$(PEEK(J))
1070 NEXT J
1080 RETURN
1090 END

READY.
```

Tres pequeños SYS

RESET

Un RESET del CBM 64 que provoca diversas reinicializaciones con pérdidas de datos en memoria (programa y variables), puede obtenerse con SYS 58260. Es menos eficaz que el SYS 64738 más conocido, que también efectúa una reinicialización total del sistema.

SCROLL

El scrolling vertical de la pantalla (desfile de las líneas hacia arriba) es fácil de conseguir mediante un sencillo SYS 59626. La llamada a esta rutina desplaza la pantalla sólo en una línea. La inserción de esta instrucción en un bucle le permitirá «scrolear» a su gusto...

Ligeramente más difícil de realizar, veamos la clave a emplear:

POKE 781,Y: SYS 59903

Fórmula en la que Y representa el número de la línea a borrar. (La primera línea lleva el número 0).

Con estos conocimientos pruebe su opinión con este programa.

Oliver Carré

Programa

```
100 PRINT"■":POKE 53280,0:POKE 53281,0
110 FORT=1 TO 44:READ X
120 PRINTCHR$(X);:NEXT T
130 DATA 18,69,76,32,79,82,68,69,78,65,68,79,82,32,80,69,82,83,79,78,65,76
140 DATA 146,32,69,83,13,85,78,65,32,66,85,69,78,65,13,82,69,86,73,83,84,65
150 FOR T=0 TO 1000:NEXT T:PRINT"□□"
160 POKE 781,2:SYS59903
170 FOR T=1 TO 8:READ X
180 PRINT CHR$(X);:NEXT T
190 DATA 76,65,32,77,69,74,79,82
200 PRINT:PRINT"■■■■ERROR CORREGIDO"
210 FOR T=0 TO 1000:NEXT
220 FOR T=1 TO 10:SYS 59626:NEXT T
230 SYS 58260
240 END

READY.
```

BOXER 12

high resolution monochrome monitor 12”

NEW 85
NOVEDAD 85

ELECTRICAL ENVIRONMENTAL CHARACTERISTICS

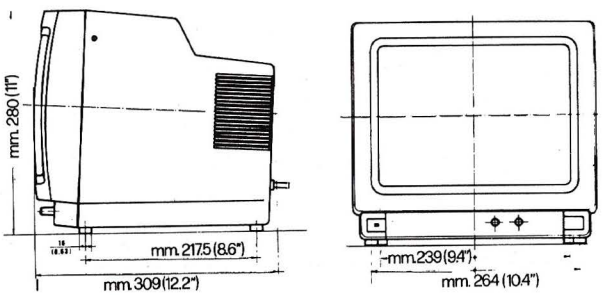
CRT	SIZE	12”
	DEFL. ANGLE	90°
DISPLAY FORMAT	CHARACTERS	2000 (80 × 25)
VIDEO	INPUT SIGNAL	COMPOSITE VIDEO
	VIDEO SIGNAL	1 Vpp pos.
	RISE/FALL TIME	⩽ 30 ns
	BANDWIDTH	20 MHz
	CENTRE RESOLUTION LINES/IN	1000
	INPUT RESISTANCE	75 Ohm
BLANKING TIME	HORIZONTAL	⩽ 8 μs
	VERTICAL	⩽ 700 μs
COMP. SYNC.	H. SYNC.	15.650-15.750 KHz
	V. SYNC.	50-60 Hz
EHT	(Ib = 0)	13 KV
POWER SUPPLY	INPUT VOLTAGE	min. 180 max. 264 Vac
	CONSUMPTION	30 VA
GEOMETRY	RASTER DISTORTION	max 1 %
	SCAN LINEARITY	max 10 %
	FOCUS	internal control
	V. AMPLITUDE	internal control
	V. FREQUENCY	internal control
	V. UPPER AND LOWER LINEARITY	internal control
	H. AMPLITUDE	internal control
	H. FREQUENCY	internal control
	H. LINEARITY	internal control
	H. PHASE	internal control
ENVIROMENTAL	AMBIENT TEMPERATURE	0° C + 40° C
	AMBIENT HUMIDITY (not condensed)	5-90 %
	STORAGE TEMPERATURE	40° C + 65° C
	STORAGE HUMIDITY (not condensed)	5-90 %
WEIGHT	GROSS/NET	5,7/6,6 Kg.

● *audio optional*

TRATTAMENTO SCHERMO: SCURO - ANTIRIFLETTENTE
SCREEN TREATMENT: DARK GLASS - ETCHED

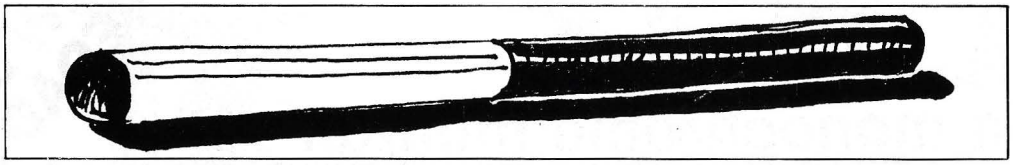
FOSFORO - P31 - VERDE MEDIO-BREVE
PHOSPHOR - P31 - GREEN MEDIUM-SHORT

DATI MECCANICI
MECHANICAL DATA



HANTAREX®
Electronic Equipment Manufacturer
QUALITY . RELIABILITY . SERVICE

Aragón, 210, 1°, 1ª - Barcelona 11 - teléf. (93) 3232941 - telex 98017



Rutina pantalla total

La pantalla habitual está desplazada de la dirección 7680 a la dirección 7168. Se reserva el lugar en MEV en la línea 1: «CLR» = reinicialización de los punteros.

```
0 CLR
1 POKE 52,28:POKE 56,28:CLR
2 A=PEEK(36866) AND 128 : B=PEEK(36867)AND 129
3 POKE 648,28 : SYS58648
4 POKE36864,8:POKE36865,22:POKE36866,260R A:POKE36867,640R B
```

Lo que efectivamente desplaza la pantalla y previene al sistema es la línea 3, mediante la llamada a la rutina Kernal en $E518 (58648) cuyo objeto es ése precisamente. La línea 4 define el tamaño de pantalla, que puede ser diferente de las 32 líneas de 26 caracteres que se manejan aquí.

Cualquiera que sea el tamaño elegido, el procedimiento cuesta 0,5 Ko., porque la RAM pantalla sólo puede desplazarse por intervalos de 512. No hay que preocuparse por la RAM color, que siempre depende del lugar de la pantalla: bien +30720, o bien +33792 en función de la configuración elegida.

Toda la pantalla es accesible por POKE, pero sólo la pantalla normal (23 x 22 máximo) es accesible por PRINT. Aquí la pantalla accesible por PRINT, colocada en 7168 llega hasta 7673 y la pantalla suplementaria empieza en 7674 para acabar en 7999. La RAM color está en EC+30720.

Estos valores son válidos para Vic básico y +3 Ko. Si quiere desplazar el generador de caracteres mejor (128 caracteres programables), sólo quedan 2 Ko. disponibles para el Basic en versión básica (POKE 52 y 56,24). Por tanto, es esencial cortar el programa en varias partes; sabiendo que el tamaño del programa activo más sus variables no podrá exceder de 2Ko. Para configuraciones con +8Ko. o más, es necesario gestionar las direcciones de forma diferente para conservar el máximo de memoria Basic continua posible. En este caso, es interesante colocar la pantalla al principio de MEV y hacer arrancar al Basic más alto, mediante POKE en 44, 46, 48 y 50 (valores opcionales a determinar).

Michel Piperaud

Logicomanía

Inspirado por una particularidad de la SHARP PC-1500, he descubierto un medio de utilizar los test lógicos de comparación (= < >). En efecto, el VIC atribuye a cada proposición un número: 0 si la proposición es falsa y –1 si la proposición es verdadera.

De esta forma PRINT (3 = 2 + 1) nos da –1.

Es posible evaluar series de comparaciones:

PRINT (3 > 2 = 0 – 1) PRINT (–1 = 0 = –1) PRINT (0 = –1) todos dando como resultado 0.

Para acabar digamos que los signos aritméticos son prioritarios. (3 < 2 * 2 = – 1) dará –1.

Esperamos que esto sirva de nueva herramienta de trabajo para los «fanáticos» de la optimización.

L. Nichon

¡Buenos días, programa! Rutina «UN-NEW»

Los veintidós octetos de esta rutina pueden ubicarse en cualquier lugar de la memoria.

```
LDA #$08
STA $0802
JSR $A533
CLC
LDA $22
ADC #$02
STA $2D
LDA $23
ADC #$00
STA ❹2E
RTS
```

Si tiene la precaución de salvaguardarlos con un monitor, le permitirán recuperar un programa perdido por un NEW intempestivo o por un RESET efectuado a continuación de una «plantación». La puesta en marcha se hace mediante SYS XXXXX, siendo XXXXX la dirección del principio de este programa en memoria.

Hervé Lemarchand

Caracteres inversos

Este programa permite invertir todas las letras, símbolos y todas las cifras a excepción de los caracteres gráficos.

Las líneas de la 3 a la 5 analizan las matrices de cada caracter (de 0 a 63), mientras que las líneas 1, 6 y 8 preparan la generación, y 9 y 11 producen los nuevos caracteres invirtiendo las matrices.

PROGRAMA

```
 1 POKE 56,28:POKE 52,28:CLR
 2 G=32768:DIM Q(512)
 3 FOR R=0 TO 63
 4 FOR L=0 TO 7
 5 X=PEEK(G+B*R+L):W=W+1:Q(W)=X:NEXT L,R
 6 POKE 36869,255
 7 FOR I=7168 TO 7679
 8 POKE I,PEEK(I+25600):NEXT:W=0
 9 FOR R=7168 TO 7679 STEP 8
10 FOR L=7 TO 0 STEP -1
11 W=W+1:POKE R + L,Q(W): NEXT L,R
```

Marc Simonnet

Pescar la línea

Se pueden escribir 25, 26 o 27 caracteres por línea. No obstante, como el número de posiciones en la pantalla está fijado en 506, habrá una disminución del número de líneas.

- Para 25 columnas y 20 líneas, o sea, 500 posiciones: POKE 36864,9 : POKE 36865,42 : POKE 36866,153 : POKE 36867,40.
- Para 26 columnas y 19 líneas: POKE 36864,9 : POKE 36865,44 : POKE 36866,154 : POKE 36867,38.
- Para 27 columnas y 18 líneas: POKE 36864,8 : POKE 36865,46 : POKE 36866,155 : POKE 36867,36.

Se obtiene una imagen que ocupa toda la amplitud de la pantalla.

```
10 POKE 52,28: POKE 56,28: CLR
20 A=36864:E=38628
30 POKEA+5,255:POKEA+14,128:POKEA+15,200
40 FOR C=7168 TO 8185:POKEC,0:NEXT
60 GOSUB 500
70 FOR G=7176 TO 7295:READF:POKEG,F:NEXT
75 E=E-30720
80 GOSUB 500
90 GOTO90
500 FOR X=E TO E+4:READF:POKEX,F:NEXT
510 FOR X=E+22  TO E+26:READF:POKEX,F:NEXTX
520 FOR X=E+44 TO E+48:READF:POKEX,F:NEXTX
530 RETURN
1000 DATA11,11,11,11,11
1100 DATA11,11,11,11,11
1200 DATA11,11,11,11,11
1400 DATA 0,0,85,64,79,78,78,78,0,0,85,0,255,170,85,64
1500 DATA 0,0,85,0,255,170,85,0,0,0,85,0,255,170,86,6
1600 DATA 0,0,84,4,196,196,196,196,78,78,78,78,78,78,78,78
1700 DATA 79,78,78,78,78,78,78,79,255,170,86,70,70,86,170,255
1800 DATA198,198,198,198,198,198,198,198,196,196,196,196,196,196,196,196
1900 DATA78,78,78,79,64,85,0,0,64,85,170,255,0,85,0,0
2000 DATA0,85,170,255,0,85,0,0,6,86,170,255,0,85,0,0,196,196,196,196,4,84,0,0
2100 DATA1,2,3,4,5,6,7,8,9,10,11,12,13,14,15
```

Bernard Petrisot

Gráfismos de alta resolución

Los poseedores de Super Expander se interesarán por los dos programas que siguen.

El primero, que necesita una extensión de memoria de 16 Ko. sirve para transferir a memoria una imagen de pantalla de alta resolución y la restituye instantáneamente.

Una parte del programa está en lenguaje de máquina. Los 3200 octetos de la imagen de pantalla se transfieren a partir de la dirección 1280 (decimal) mediante las líneas 250 a 270. Seguidamente se lleva a la memoria de pantalla por las líneas 300 a 320.

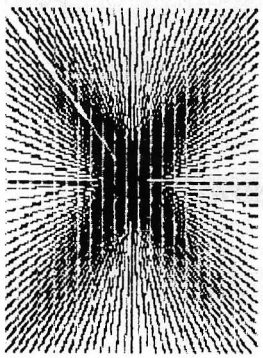
El programa 2 pide una impresora gráfica como la Seikosha GP 100. Permite obtener en papel una copia de la pantalla en alta resolución. Su principal defecto es la lenta ejecución. ¿Algún lector ha hecho algún programa de copia más rápido?

Daniel Christy

Programa

```
100 REM*** TRANSFERENCIA DE GRAFICOS ***
110 POKE 54,79:POKE 56,80
120 FOR I=0TO 46:READ A:POKE 1280+I,A:NEXT I
130 :
140 DATA166,140,240,19,166,140,160,0
150 DATA177,1,145,141,200,208,249,24
160 DATA230,2,230,142,202,208,239,166
170 DATA139,240,11,160,0,177,1,145
180 DATA141,200,196,139,208,247,169,72
190 DATA 133,1,169,210,133,2,96
200 :
210 GRAPHIC 2:COLOR 3,5,6,7
220 FOR I=0 TO 256 STEP 16
230 CIRCLE 2,512,512,I,I/.7
240 NEXT I
250 POKE 1,0:POKE 2,16:POKE 139,128
260 POKE 140,12:POKE 141,0:POKE 142,80
270 SYS 1280
280 FOR I=0 TO 5000:NEXT I
290 SCNCLR
300 POKE 1,0:POKE2,80:POKE139,128
310 POKE 140,12:POKE141,0:POKE 142,16
320 SYS 1280
330 FOR I=0 TO 5000:NEXT I
340 SCNCLR:GRAPHIC 0
350 END

READY.
```

Ejemplo de ejecución.

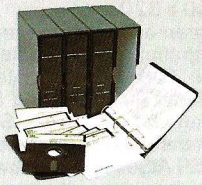


```
READY.


10 GRAPHIC2:COLOR2,4,3,6
20 FOR I=0TO 1023 STEP 32
30 DRAW 2,0,I TO1023,1023-I
40 DRAW 2,I,0 TO 1023-I,1023
50 NEXT I
60 GOSUB 1000
70 GRAPHIC 0
80 END
90 :
1000 OPEN4,4:CO=RCOLR(0)
1010 FOR ZZ=0 TO 21
1020 FOR ZX=0 TO189:XE=ZX*1024/190:ZA=0
1030 FOR ZY=0 TO 6:YE=ZZ*49+7*ZY
1040 IF YE>1023 THEN 1060
1050 IF ROOT(XE,YE)<>COTHEN ZA=ZA2↑ZY
1060 NEXT ZY
1065 ZA=ZA+128
1070 ZA$=CHR$(8)+CHR$(27)+CHR$(16)+CHR$(0)+CHR$(ZX)+CHR$(ZA)
1080 PRINT #4,ZA$;
1085 NEXT ZX
1090 PRINT #4,CHR$(13);
1100 NEXT ZZ
1110 PRINT #4,CHR$(15):PRINT#4:CLOSE 4
1120 RETURN

READY.
```

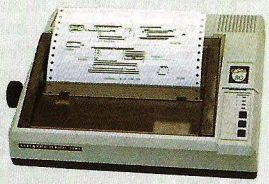
Aquí tiene los productos más buscados...

BONDWELL 12/14/16 - Ordenadores transportables con software incluido.

MODEM BONDWELL -Modem telefónico para comunicaciones.



SOFTWARE BONDWELL Programas de aplicación para ordenadores BONDWELL.

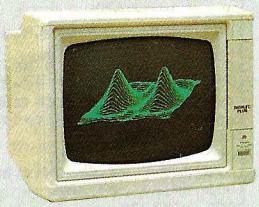


SHINWA CPA-80 -Impresoras matriciales 100 cps (serie o paralelo).

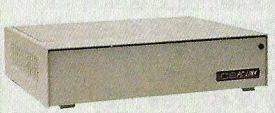
DAISY JUNIOR - Impresora margarita con caracteres españoles.

BONDWELL 2 - Ordenador portátil con unidad de disco incorporada y software incluido.

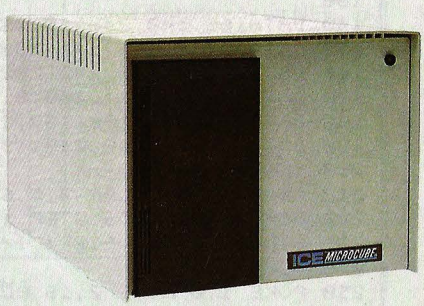
PC-88 - Ordenador de gestión 16 bits MS/DOS.



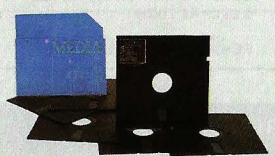
DATALEC PLUS - Monitor monocromo alta resolución.



ICE - PC/LINK - Red local ICE de puesta en marcha instantánea.



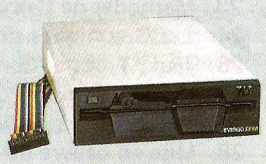
ICE-MICROCUBE - Sistema de disco duro compatible con los principales ordenadores del mercado.



MEDIA TECH - Diskettes de alta calidad.

SOFTWARE ELITE -Programas de aplicación para ordenadores ELITE y compatibles.

ELITE V - Ordenador de gestión compatible.



SS-5B - Unidad de disco flexible muy perfeccionado.

TARJETAS ELITE - Tarjetas de expansión para ordenadores ELITE y compatibles.

ELITE I - Ordenador de gestión compatible.

Véalos en nuestros distribuidores autorizados

SITELSA, importa y distribuye a nivel nacional una línea de productos informáticos altamente competitiva. Todos los productos están soportados tanto técnicamente como a través de desarrollos específicos y documentación para el usuario.

SITELSA

C/. Muntaner, 44 - Tel. (93) 323 43 15
08011 Barcelona - Telex 54218

Rogamos nos indiquen los productos de su máximo interés para poder enviarles mayor información y lista de precios.

DRAGON

Trucos para Dragón-32

Ya por la Edad Media los caballeros se divertían con los dragones, y de camino lograban el amor de sus doncellas preferidas. ¡Seguro que encuestras algún POKE con el que asombrar a la tuya!

Miscelánea para Dragón 32

Recogemos, en miscelánea, una serie de indicaciones que serán de utilidad:

EXEC 33823: RUN

POKE 65495,0: aumenta la velocidad del Basic en un 50 %, ya no son válidas las entradas y salidas.

POKE 65494,0: vuelve a la velocidad normal.

POKE 65497,0: aumenta la velocidad del Basic en un 100 %, anulada sincronización vídeo.

POKE 65496,0: vuelve a la velocidad normal.

PEEK (135): código ASCII de la última tecla pulsada.

POKE 25,6: recupera 31 Ko. de MEV tras NEW, se inutilizan las páginas gráficas.

PEEK (25) x 256 + PEEK (27): puntero del principio del Basic.

PEEK (157) x 256 + PEEK (158): dirección de puesta en marcha de un programa, ensamblador transmitido mediante CLOADM.

PEEK (487) x 256 + PEEK (488): dirección de principio de un programa, ensamblador transmitido por CLOADM.

PEEK (126) x 256 + PEEK (127) – 1: dirección de final...

POKE 65281,50: inhibe el teclado.

POKE 65281,181: suprime la inhibición.

EXEC 32786: inicializa la posición de los mandos de juego.

PEEK (346): posición en X.

PEEK (347): posición en Y, mando derecha.

PEEK (348): posición en X.

PEEK (349): posición en Y, mando izquierda.

EXEC 41194: espera el pulsado de una tecla (A$=INKEY$:IF A$="").

EXEC 46004: equivalente a RESET.

POKE 114,25: tras RESET, presenta el mensaje de puesta en marcha, no se borran las páginas gráficas.

POKE 359,60: listado lento.

POKE 359,57: listado normal.

PEEK (31) x 256 + PEEK (32): puntero de final del Basic.

PEEK (274) x 256 + PEEK (275): valor del reloj.

CHR$ (PEEK(55)) + CHR$ (PEEK(56)): nombre de la última variable empleada.

PEEK (136) x 256 + PEEK (137): dirección del cursor.

EXEC 41126: espera la pulsación de cinco teclas.

POKE 151,X: la lentitud del teclado aumenta con X.

Eric Boucher

Inversion video

Este programa para Dragón 32 permite efectuar la inversión video, texto o gráfico. Simplemente sirve para cargar el lenguaje máquina y para controlar que no existe ningún error en los DATA. Pueden omitirse todas las líneas REM ('); en consecuencia, una vez cargado el lenguaje máquina, se puede borrar el programa.

Algunas explicaciones. El último valor en cada línea de DATA es la suma de los demás valores.

$31-$32 (línea 210) es el número de la línea de DATA que se está leyendo

$BA (línea 510) es el principio de página gráfica.

$B7 (línea 560) es el final de la página gráfica.

Las rutinas están en las direcciones 32700 para texto y 32715 para gráfico.

Para llamarlas, basta hacer EXEC 32700 o EXEC 32715.

Por otra parte, hay una pequeña astucia que, quizá, le permita encontrar cosas interesantes. Haga:

10 POKE 65480,1:GOTO 10

y después RUN.

A partir de ese momento puede visualizar los 512 primeros octetos. Para volver a modo normal, simplemente BREAK.

Marc Dutendas

Programa para Dragón 32

```
1 '*****************************
2 '***   INVERSION VIDEO     ***
3 '***                       ***
4 '***  (C) EL AUTOR  Y      ***
5 '***                       ***
6 '***      EL  O.  P.       ***
7 '*****************************
10 'INVERSION VIDEO
20 CLEAR200,32700
30 FOR T=32700 TO 32725 STEP5
40 C=0
50 FOR TI=0 TO 4
60 READ A$
70 A=VAL("&H"+A$)
80 POKE T+TI,A
90 C=C+A
100 NEXT TI
110 READ B
120 IF C<>B THEN 210
130 NEXT T
140 PRINT "FIN"
150 END
200 ' LINEA DE DATOS CON ERROR
210 X=PEEK(&H31)*256+PEEK(&H32)
220 PRINT"ERROR EN LA LINEA";X
230 END
250 DATA 8E,04,00,A6,84,444
260 DATA 88,40,A7,80,8C,635
270 DATA 06,00,26,F5,39,346
280 DATA 9E,BA,EC,84,43,779
290 DATA 50,ED,81,9C,B7,782
300 DATA 26,F6,39,39,39,455
400 'PROGRAMA EN ENSAMBLADOR
410 'INVERSION TEXTO
420 'LDX #$0400
430 'LDA ,X
440 'EORA #$40
450 'STA ,X+
460 'CMPX #$0600
470 'BNE F5
480 'RTS
500 'INVERSION GRAFICA
510 'LDX $BA
520 'LDD ,X
530 'COMA
540 'COMB
550 'STD ,X++
560 'CMPX $B7
570 'BNE F6
580 'RTS
```

FIRST S.A.

Ordenadores más Personales!

C/ Aribau, 62. 08011 BARCELONA
Tel. (93) 323 03 90
Tlx. 53947 FIRS E

SEGUIMOS BUSCANDO DISTRIBUIDORES

METHAMORPHIC

RAM: 48 Kb, ampliable hasta 192 Kb.
ROM: 12 Kb.
TECLADO: ASCII, tipo máquina de escribir con teclado numérico con signos matemáticos (ideal para sus hojas de cálculo electrónicas). Alta fiabilidad de teclado. Instrucciones directas BASIC por una sola tecla, desde teclado (auténticas macroinstrucciones).
Editor de programas. Movimiento del cursor en los cuatro ejes. Tecla de repetición.
MAYUSCULAS Y MINUSCULAS.
40 × 24 caracteres por pantalla (opcional 80 × 24, 128 × 32, 132 × 24 y 160 × 24).
GRAFICOS: Media resolución y alta resolución (280 × 192 puntos).
Hasta 15 colores (255 por software).

COMPATIBLE con los programas del mundo de

EXPANSION: Por medio de 8 conectores, se abre al mundo de: Diskdrives, discos winchester, PAL color, 80 columnas, Z-80, plotter, impresoras (margarita y matricial), tablero gráfico, koala pad, modem, IEE, RS 232, analógico-digital, robots, etc.
JUEGOS: Conexión para Joystick, Paddle y Trackball.
CONEXION: Para cassete (comunica con cassette, amplificador, RTTY).
SISTEMAS OPERATIVOS: Dos 3.2, dos 3.3, Prodos, Pascal, CP/M.

LENGUAJES: BASIC incorporado, opcional: Pascal, Cobol, Fortran, Assembler, Logo, Pilot, Forth, Modula.
Incluye: Manual completo en castellano y en la versión Disco.
WORLD PACK (R) (tratamiento de textos, base de datos, hoja de cálculo electrónica, tutor de BASIC, editor de programas y utilidades, paquete de juegos).

METHAMORPHIC
79.500 pts.

★ **145.500 pts.**

METHAMORPHIC + *Disco COMPUPRO + Monitor 12" Fosforo Verde con Controlador*

MICRO AGE *Incluye:*

Diskettes

10 Diskettes +GRATIS 1 Diskette

CAJA PROTECTORA DE PLASTICO

DD/SC 3.995 PTAS. DD/DC 4.550 PTAS.
METHAMORPHIC
II+
IIe
COMMODORE 64 *y compatibles*
PC XT *y compatibles*

FIRST, S. A. IMPORTADOR PARA ESPAÑA DE **MICRO AGE**

COMPUPRO - FLOPPY DISK DRIVE PARA II,//e, METHAMORPHIC

NUEVO PRECIO

disk drive Silencioso, ágil y manejable.

★ **32.500 pts.**

FIRST IMPRESORAS MAS PERSONALES *star*

GEMINI 10 X
VERANO FIRST **pts.**

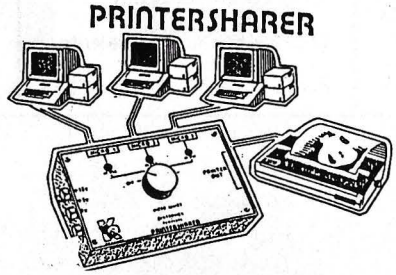
INCLUYE LA INTERFACE MICROANGELO Y CONECTORES Y AHORA...
TAMBIEN MICROANGELO PARA COMMODORE 64
CON GRAFICOS Y SIGNOS COM64, INCREIBLE!!

MICROANGELO

Es la más PODEROSA interface paralelo para su impresora Epson, Star, etc., **existente en el mercado mundial.** Sus habilidades están tanto en el texto como en sus funciones gráficas logrando directamente con su teclado el volcado de sus gráficos en cualquier página y a sus tamaños, rotación, etc. Efectiva y sin sofisticaciones se convierte en la que Vd. precisa en el momento de decidirse a conectar su ordenador a su impresora. Siga nuestro consejo. Exija siempre MICROANGELO.

15.000 pts.
COMPRE SIEMPRE EN FIRST

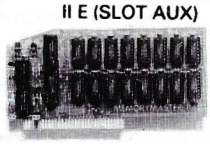
PRINTERSHARER



Conecte hasta tres ordenadores, incluso distintos, a una sola impresora (con ello ahorrará la compra de hasta dos impresoras), se gobierna sólo moviendo un botón.
Y también printershare, un sólo ordenador se conectará a dos impresoras a la vez. Ejemplo, a una impresora matricial y otra margarita. Ud. decide cuálimprimirá en cada momento, un simple giro de su mano ejecuta su orden.
First, ponemos tecnología al alcance de su mano.

PRINTERSHARER 29.063 ptas.
FIRST, S.A. IMPORTADOR PARA ESPAÑA DE

TARJETA 80 COLUMNAS II E (SLOT AUX)

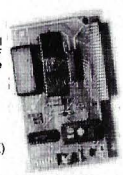


Ahora puede tener auténticas 80 columnas. Manual en español de como trabajan, ideal para AW TIE, OF PFS IIE, etc. FIRST, S.A. importador para España de KEYZONE LTD.

TAR. 80 COL. APPLE II E (AUX)
14.000
TAR. 80 COL. +64K RAM APPLE II E (AUX)
24.550 pts.

FIRST, S.A. IMPORTADOR PARA ESPAÑA DE COMPUPRO

THE TIMEMASTER



- Compatible dos 3.3 y prodos en methamorphic, II+, IIE y compatibles.
- Tiempo en h., min., seg. y milisegundos (única).
- Formato militar 24 h. o 12h. formato AM/PM.
- Fecha con año, mes, día de la semana y salto anual.
- La de más fácil programación en basic, documentación completa y software applesoft, cod. máquina, CP/M y Pascal.
- Reloj que le permitirá cualquier intervalo hasta 48 días hasta el más próximo milisegundo.
- Batería recargable de níquel-cadmio, con cobertura 20 años.

Finalmente un reloj que lo hace todo.

26.457 ptas.

SUPER JOYSTICK

COMPATIBLE CON COMMODORE 64 y VIC 20
SINCLAIR (precisa interf)
ATARI
SPAECTRAVIDEO

1.950 !!

RBG COLOUR CONVERTER
by Trevor Hallatt APPLE//c

MAS ALTA DEFINICION, SATURACION Y BRILLO. NO INFLUYE EN SU SOFTWARE. CONECTABLE DIRECTO A SU UD. DESDE Y COMO NO.. TAMBIEN EL COLOR DE SUS LETRAS. FIRST LE OFRECE LO INCREIBLE. SOLICITE AMPLIA INFORMACION (SOLO PARA PROPIETARIOS DE IIC, IND!
QUE FECHA Y LUGAR DE COMPRA)

FIRST, S.A. IMPORTADOR PARA ESPAÑA DE KEYZONE LTD. **22.950**

Z-80 PLUS!

*TOTALMENTE COMPATIBLE CON TODO EL SOFT CP/M.
*LA ÚNICA Z-80 CON UN CHIP ESPECIAL DE -CP/M DETECTOR-
*COMPLETAMENTE COMPATIBLE CON LOS DISCOS MICROSOFT (NO REQUIERE PRE-BOOTS).
*ESPECIALMENTE CONCEBIDA PARA OPERACIONES DE ALTA VELOCIDAD EN IIE, AL IGUAL QUE IN.
*CORRE DBASE II, NEVADA COBOL, FORTRAN, ETC.
*VUELE A TRAVES DEL CP/M AUN MUY BAJO CONSUMO DE ENERGIA. IMITADO DE TAMAÑO QUE OTRAS, Y USAMOS LA Z-80A DE 4MHZ).
*HACE LO MISMO QUE OTRAS Z-80, MAS INTERRUPCIONES Z-80.
No confunda Z-80 PLUS con otras tarjetas Z-80. La Z-80 PLUS es mucho más sofisticada. Con ello accedera a un gran soft.

Z-80 PLUS!
28.710 pts.

CAJA GRANDE GUARDA DISKETTES 1.950 pts.

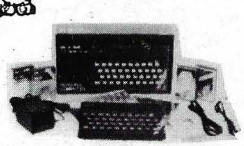
CAJA 2500 HOJAS PAPEL	4.456 pts.
1.000 RECIBOS NEGOCIABLES	2.500 pts.
1000 ETIQUETAS ADHE. 2XLIN	828 pts.
10 CINTAS CASSETE 20 H.	1.100 pts.
CINTA IMP. STAR GEMI-SG10	180 pts.
CINTA IMP. EPSON 80	850 pts.
RECAMBIO CINTA EPSON 100	820 pts.
CINTA IMP. STAR POWER-PLAS	1.500 pts.

LOS PEDIDOS DE PAPEL SE MANDAN A PORTES DEBIDOS.

AIREADOR 9.500 pts.

WORDSTART

PARA TRATAR SUS TEXTOS, MANEJARLOS, MODIFICARLOS, MOVERLOS, UNIRLOS, TENEMOS SU PROGRAMA, SOLICITE AMPLIA INFORMACION.



ZX SPECTRUM 48K
FIRST **ptas.**

★ ZX Spectrum Plus ***FIRST*** ptas.

PRECISION *Paddle Pair*

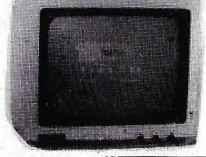
SENCILLAMENTE, CONECTE LOS NUEVOS MINIPADDLES KRAFT A SU METHAMORPHIC, //e, //E O COMPATIBLES. ADAPTELOS PERFECTAMENTE A LA PALMA DE SU MANO, MANEJELOS CON SU SUAVIDAD Y PRECISION, PULSE SU BOTON. SOLO KRAFT LIDER U.S.A. PODRIA HACERLOS PARA UD. A SU MEDIDA
MINIPADDLES KRAFT **11.080 pts.**

FIRTS. S.A. IMPORTADOR PARA ESPAÑA DE **KRAFT**

KRAFT

CP/M ✱ + 80 COLUMNAS 24.000 ptas.

PHILIPS
23.000 ←

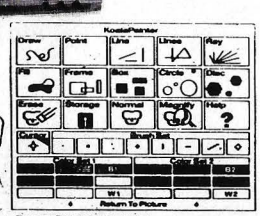


Amstrad CPC 464
FOSFORO VERDE COLOR ***VERANO FIRST***
INCLUYE: - ORDENADOR - MONITOR - CASSETE

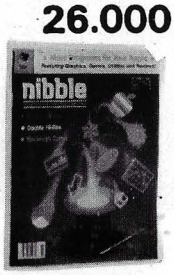
The Monitor **19.000 pts.**

ErgoVisión
(Base móvil) ★
26.000 PTAS.

COMMODORE 64
Amplia gama de SOFT & HARD & LIBROS. Novedades cassette para COM.



nibble



NIBBLE VOL. 16, 17	850 Pts./U.
NIBBLE VOL. 2.2, 2.3, 2.4, 2.6, 2.8	850 Pts./U.
NIBBLE VOL. 3.3, 3.4, 3.5, 3.6, 3.7, 3.8	850 Pts./U.
NIBBLE VOL. 4.2, 4.3, 4.4, 4.5, 4.8	850 Pts./U.
NIBBLE VOL. 5.1, 5.2, 5.3, 5.4, 5.5, 5.6	850 Pts./U.
NIBBLE VOL. 5.7, 5.8, 6.9, 5.10, 5.11, 5.12	850 Pts./U.
NIBBLE VOL. 6.1, 6.2, 6.3, 6.4, 6.5, 6.6, 6.7, 6.8, 6.9	850 Pts./U.
NIBBLE EXPRESS	
VOL. I, VOL. II	4.375 Pts./U.
VOL. III, VOL. IV	4.825 Pts./U.

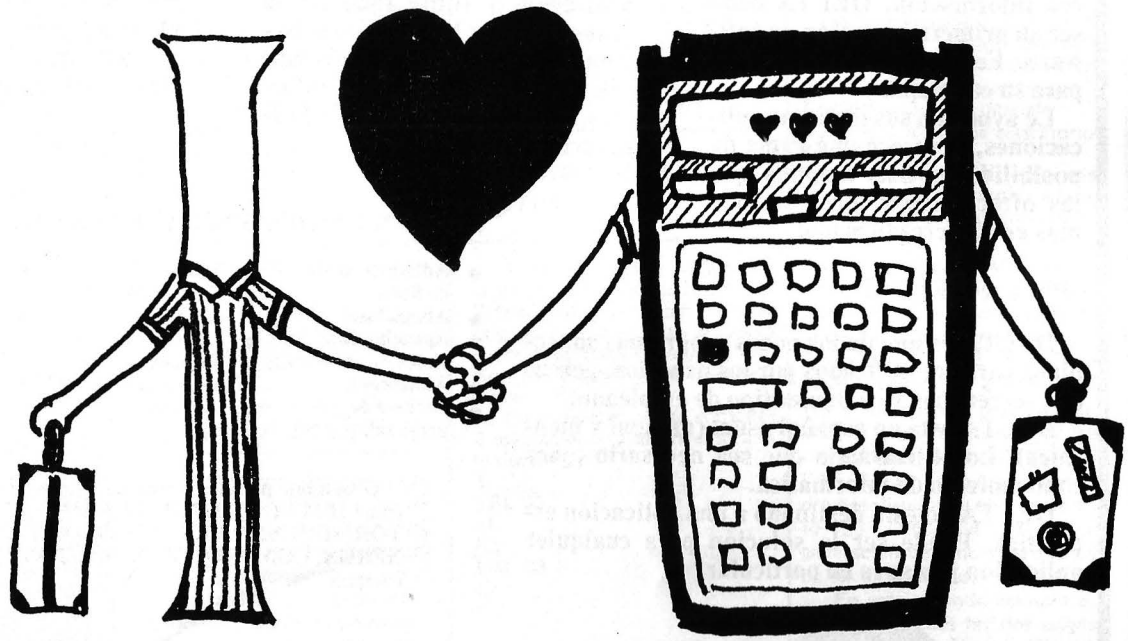
FIRTS, S. A.
IMPORTADOR PARA ESPAÑA DE **NIBBLE**

KoalaPad LA POTENCIA DE UN PAQUETE GRAFICO 19.500 pts.
KOALA PAD para II + IIE y METHAMORPHIC
FIRST, S. A. IMPORTADOR PARA ESPAÑA DE Koala Technologies Corporation

SOLICITE NUESTRO CATALOGO
* CATALOGO DE SFOT, HARD Y LIBROS.
* TRES CATALOGOS EN UNO
VEA NORMAS EN VENTA POR CORREO

VENTA POR CORREO
Mande su pedido, pago talón confirmado o giro postal. Pedidos inferiores a 4.500 pts. sume 180 pts. gastos envio. Libros no sume gastos. Catálogo completo mande 250 pts. gastos envio. La presente lista es susceptible de ser modificada sin aviso previo.
ENTREGA INMEDIATA A PROVINCIAS

HP-41



Los amantes de la "Paka" son incapaces de viajar sin ella, de engañarla con otra, de tocar otras teclas sin su permiso. Con un caracter tan fuerte. ¿Quién conoce sus secretos, sus enfados, sus pasiones? Nosotros, humildes, hemos penetrado hasta donde hemos podido y sin recelo te mostramos algo que te permita comprenderla un poco más. ¡Quizás así podais viajar más unidos!

Como evitar que el hermano pequeño juegue con la HP

El método consiste en impedir que la calculadora se encienda. Para ello hacemos este pequeño programa:

```
01*LBL 01
02 SF 11
03 OFF
04 GTO 01
05 .END.
```

Una vez introducido el mismo lo ejecutamos y la máquina se apagará. Cada vez que intentemos encenderla pulsando ON, gracias a la bandera 11 de ejecución automática, la rutina será ejecutada de nuevo y por tanto se apagará, no siendo posible ponerla en marcha.

Para salir del bucle pulsamos ON e inmediatamente R/S con lo que la tendremos funcionando de nuevo.

Rotulos móviles en la pantalla

Cuando se produce un error con la bandera 25 puesta sucede lo siguiente:

- La función causante de error no es realizada.
- El programa no se detiene.
- La bandera 25 es quitada.
- La bandera 50 anunciadora de mensaje es puesta.

Gracias a esta última acción que no nos es descrita en el manual, podemos hacer desplazar un rótulo en la pantalla, la rutina que sigue ilustra la forma de conseguirlo.

```
01 AVIEW
02 SF 25
03 SF 90
04*LBL 01
05 SIN
06 GTO 01
07 .END.
```

El texto que haya en el registro alfa se desplazará en sustitución del pato que aparece normalmente.

Si en un programa queremos sustituir el pato por un mensaje moviéndose en la pantalla escribiremos el mensaje, AVEW, SF 25, SF 90 y a continuación nuestro programa.

Programas privados sin lector

Mas de uno habrá pensado en hacer programas privados sin necesidad del lector de tarjetas magnéticas. Esto es fácil gracias a la programación sintética.

Veamos como hacerlo.

1-Colocarse en el END del programa a privatizar.
2-STO 01.
3-PACK
4-SST
5-Modo de ejecución.
6-BJ (*)
7-Modo de programación.
8-HMS +
9-SST
10-[←]
11-Pulsar SST tantas veces como sea necesario para llegar al STO 01 introducido en 2 y borrarlo.

Aparecerá el mensaje PRIVATE y el programa estara privado como si se hubiera hecho con el WPRV de la lectora.

NOTA IMPORTANTE: No podrá hacerse BST ni GTO. I m n en sustitución de los SST de 11 pues nos aparecerá private y no podremos borrar el STO 01.

Si el programa es demasiado largo puede ser pesado el hacer un número elevado de SST para llegar hasta el STO 01. Esto podemos evitarlo haciendo sustitución de 11; XEQ "END", GTO. "nombre del programa", BST, BST, [←], SST, [←] y tendremos nuestro programa privado.

(*) El BJ de 6 corresponde al byte de salto XROM 05, 01, código F141 que se supone sabe el lector asignarlo a una tecla.

J.A. Deza.

DELTA

Base de datos esencial para su microordenador

Si una tarea de su microordenador es almacenar y tratar mucha información, DELTA debe ser su primera inversión en software. Es un éxito garantizado para su compañía.

Le ayuda en sus distintas aplicaciones, le ofrece una gama de posibilidades más amplias que las ofrecidas por otros programas en el mercado actual.

El éxito de DELTA está principalmente en su simplicidad de utilización y sobre todo en su gran potencia. Le permite seleccionar su información, efectuar cálculos, imprimir listas, informes, etiquetas adhesivas, y hasta cartas personalizadas!

Si Vd. utiliza Wordstar, Spellbinder, Lotus 1, 2, 3, Peachtext, Visicalc o Multiplan, además necesita a DELTA que puede intercambiar todo tipo de datos con ellos.

¿Por qué DELTA?

DELTA es uno de los pocos programas concebidos para ser utilizados por los usuarios, gerentes, secretarias y cualquier tipo de empleado.

DELTA está en español usual (manual y mensajes). Lo utilizará sin que sea necesario tener conocimiento de informática.

DELTA no está destinado a una aplicación específica. Puede ser la solución para cualquier aplicación y la suya en particular.

EJEMPLOS DE APLICACIONES DE DELTA:

- Administración de fincas.
- Abogados.
- Agencias de viajes.
- Almacenes.
- Archivo de personal.
- Bancos.
- Control de coste de obras.
- Facturación.
- Farmacias.
- Hospitales.
- Librerías.
- Mantenimiento y limpieza.
- Médicos, dentistas, veterinarios.
- Seguros.
- Vídeo club...

Disponible para los ordenadores con MSDOS o PCDOS como IBM PC y XT, HP 150, RAINBOW 100/100+, VICTOR/SIRIUS, APRICOT, OLIVETTI M24, RANK-XEROX, COMPAQ, ITT XTRA, TOSHIBA, ZENITH y compatibles.

EN CASTELLANO

NO PIERDA MAS TIEMPO, ¡INFORMESE!

ORDENADOR	DISTRIBUIDOR
IBM PC y XT	Red de concesionarios autorizados de IBM España, S.A.
Apricot	D.S.E. Tel.: (93) 323 00 66
HP 150	Hewlett Packard. Tel.: Madrid 637 00 11
Rainbow 100/100 +	Digital Tel.: Madrid 734 00 52
Víctor/Sirius	Otesa Tel.: Madrid 754 33 00

Compsoft PLC, Compsoft Manor, Farncombe Hill, Godalming Surrey, England GU7 2AR

Teléfono: (07 44 4868) 25925
Télex: 859210 CMPSFT G
Contacto: Louise KILLICK

Visualización de programas privados

¿Quién dijo que los programas privados eran en realidad privados?. Bien es verdad que lo son hasta que se conoce el método de desprivatizarlos. Veamos dos formas de conseguirlo.

La primera sólo sirve si el programa privado no es el primero en memoria. Método a seguir:

- Ir hasta el END del anterior programa en memoria.
- Modo PRGM.
- STO 03.
- PAK.
- SST.
- Modo RUN.
- BJ (XROM 05, 01).
- Modo PRGM.

Aparecerá Imn LBL "nombre del programa".

Si sólo queremos verlo SST, SST. . . (No hacer BST ni GTO. porque aparecerá PRIVATE y tendremos que empezar de nuevo).

Si se trata de desprivatizar será suficiente llegar hasta el END mediante SST y borrarlo.

El segundo método no requiere el uso del BYTE DE SALTO (XROM 05, 01), y vale también aunque el programa privado sea el primero en memoria. Consiste en lo siguiente:

- Modo PRGM.
- CAT 1
- Inmediatamente R/S
- SST hasta el LBL o END siguiente al programa a visualizar.
- ALPHA
- ←
- ALPHA

Aparecerá 4094 END. Podremos borrarlo para desprivatizar y a continuación XEQ "END".

Si simplemente queremos verlo SST, SST, . . .

Superprivado

El nombre de superprivado es debido a que no aparece el .END. final en memoria y por tanto no es válido el segundo método de desprivatización descrito anteriormente. El primer método sigue siendo válido salvo que esté privatizada toda la memoria, en cuyo caso ni uno ni otro servirán.

Para hacer este tipo de privatización puede proceder como sigue:

Poner al final de programa:

- STO 01
- STO IND Z
- STO IND 64
- "P"
- BST
- BST
- Modo RUN
- BJ
- Modo PRGM
- ←
- ←
- SST
- ←
- Modo RUN
- GTO . .

Hacer CAT 1. El .END. no aparecerá. A veces haciendo GTO. . de nuevo es colocado por el procesador. El que esto ocurra depende de la situación dentro del registro absoluto de memoria. En este caso, se desprivatizará por el segundo método descrito y se repetirá el proceso colocando en lugar de "P" "PPPP".

Tiene el problema de que si toda la memoria está privatizada, no podremos introducir nuevos programas.

Por último decir que también hay una forma fácil de desprivatización. ¿Sabría el lector como hacerlo?.

Ejecución indirecta

De todos es sabido que no es posible la ejecución indirecta de las funciones de la calculadora. No ocurre así con las de los periféricos y extensiones, ya que todas, salvo las no programables, son indirectamente ejecutables.

Veamos un pequeño ejemplo:

```
01*LBL 01
02 "PRA"
03 ASTO X
04 "⊢ INDIR
ECTO "
05 "⊢EJECUT
ADO"
06 XEQ IND
X
07 .END.

PRA INDIRECTO EJECUTADO
```

J.A. Deza.

FIBONACCI

La siguiente rutina permite calcular los términos de la sucesión de Fibonacci y la razón aúrea (número de oro del juego de las cerillas). utilizando únicamente el STACK.

```
01*LBL "FI"
02 CLST
03 1
04*LBL 00
05 +
06 STO Z
07 PSE
08 GTO 00
09 .END.
```

La sucesión es $a_n = a_{n-2} + a_{n-1}$ con $a_1 = a_2 = 1$ y la razón aúrea el

$$\lim_{n\to\infty} \frac{a_n}{a_{n-1}} = \frac{\sqrt{5}+1}{2}$$

Para calcular aproximaciones a esta última pulsar: R/S, X⟨⟩Y, ÷ Después es necesario volver a iniciar el programa.

David Fernández Vergara

Asignaciones de 2 Bytes

Tanto la función ASN como la PASN del módulo X FUNCTIONS, sólo permiten realizar asignaciones de 1 byte, salvo cuando se trata de alguna función de los periféricos (son de 2 bytes).

Cada registro puede contener 2 asignaciones y su estructura es la siguiente:

T
LBL 03
a1
K1
LBL 03
a2
K2

Donde a1, a2 corresponden a la primera y segunda asignación de 1 byte dentro del registro y K1, K2 identifican el código de tecla de cada asignación. El T es el comienzo de cada registro de asignación y los LBL 03 rellenan los 2 bytes restantes.

Es posible hacer asignaciones de 2 bytes según veremos. Para ello debemos hacer mediante ASN una asignación de una función que no lleve posfijo (1 solo byte), si el número de las ya hechas es impar; y 2 si tenemos hechas un número par de ellas.

Puede resultar más cómodo borrarlas todas y hacer 2 a las dos teclas donde queramos asignar las funciones de 2 bytes.

Una vez hecho esto, seguiremos el siguiente método para llegar hasta la zona de asignaciones:

- Modo RUN
- CAT 1 e inmediatamente R/S
- RTN
- Modo PRGM
- XEQ "END"
- CAT 1 e inmediatamente R/S
- ALPHA
- ←
- ALPHA
- GTO . 001

NOTA IMPORTANTE

Los registros de asignaciones no son empaquetados salvo que las dos asignaciones correspondientes a un registro sean borradas mediante ASN ALPHA ALPHA, por lo que para introducir cada byte ha debido de ser borrado previamente el mismo número de ellas, y es preciso colocarlos en el mismo espacio, es decir dentro de los bytes nulos consecuencia del borrado. Si no procedemos así, se producirá un MEMORY LOST como consecuencia del desplazamiento de una constante que hay dentro de uno de los registros de estado de la máquina.

A continuación SST hasta encontrar la función asignada. Borramos esta y el LBL 03 anterior y los sustituímos por la función de 2 bytes deseada.

Veamos 2 ejemplos aclaratorios del método.

El primero consiste en asignar el B J (byte jumper) a la tecla 11 (Σ+) Suponiendo que no hay ninguna asignación hecha. Para ello, asignar PACK (es útil cuando se hace sintética) a la tecla −11 (Σ−) y otra cualquiera de 1 sólo byte, por ejemplo el Σ+ a la tecla 11 (Σ+) y en este mismo orden. A continuación seguir el método descrito para acceder a los registros de asignaciones. Hacer SST, SST, ← ← "A", GTO . . con lo que ya tendremos disponible el byte de salto BJ (XROM 05, 01) para generación de funciones y líneas de texto sintéticas.

En este segundo ejemplo suponemos hechas las dos asignaciones anteriores, y tratamos de asignar el TONE 1 a la tecla 72 (1) y el TONE 2 a la tecla 73 (2). Para ello asignar por ejemplo el + a la tecla 73 y el − a la 72. Ir hasta al zona de asignaciones como ya sabemos. A continuación hacer GTO . 009, ← ←, TONE 1, SST, SST, SST, ←, ←, TONE 2, GTO ..

Por este procedimiento podemos disponer de un pequeño teclado musical, asignar FIX 4, STO 05, etc.

Felices asignaciones, y no olvidar la nota anterior. Una falsa maniobra puede ser causa de MEMORY LOST.

J.A. Deza

Acceso a los registros de asignaciones con BJ

Si tenemos asignado el byte de salto, para llegar a los registros de asignaciones, será suficiente hacer GTO.. y a continuación pulsar la asignación en modo RUN, poner modo PRGM, hacer SST y aparecerá 01T correspondiente al principio de la zona de asignación. Si la memoria de programa está vacía, será necesario introducir una función cualquiera en memoria y seguir el método anterior.

J.A. Deza

Las no programables de los periféricos

Leemos en los manuales de los periféricos que ciertas funciones de los mismos no son programables. Esto no es del todo cierto. Para introducirlas como línea de programa, asignar a una tecla la no programable deseada, retirar el periférico de la calculadora, pulsar la asignación en modo PRGM, y tendremos el XROM correspondiente a la función como línea de programa. Al volver a conectar el periférico aparecerá el nombre de la función.

Algunas como el PRP os darán el mensaje de NON EXISTENT porque necesitan como posfijo el nombre del programa a listar. El LIST colocado en una parte del programa provocará el listado en la impresora o monitor, según el caso, desde la siguiente línea hasta el final del programa; la función VER os pedirá la introducción de una tarjeta magnética para su verificación, etc.

Calcular X módulo 360

A menudo es necesario reducir un ángulo a módulo 360. El siguiente programa lo hace en 5 líneas.

S. Saada.

```
01 SIN
02 LASTX
03 COS
04 R-P
05 X<>Y
```

Este registro contiene información sobre las direcciones absolutas de memoria del primer registro estadístico, del registro 00 y de la posición del .END.

El programa coge los bytes 0 y 1 del registro C y sustituye los 4 bits de mayor peso del byte 1 por 0011_2 para obtener en la línea 15 un número de la forma 00 00 00 00 00 3X Y Z (expresado en Hexadecimal) siendo X y Z la dirección del END. contenida en el C. En la línea 16 este número es introducido en los bytes 0, 1 del registro b, bytes que contienen la dirección del puntero de programa con lo que conseguimos poner el puntero en el byte 3 del registro que contiene el .END. final.

J.A. Deza

Intercambio de programas

Cuando tuve en mis manos el converter HP-ILGPIO, lo primero que se me ocurrió fue que ya que el intercambio de datos alfanuméricos entre dos HP 41 es posible, ¿porqué no intentar el intercambio de programas? Hice dos cortos programas, uno el emisor y otro el receptor que lo consiguen con ayuda del módulo X FUNCTIONS.

La solución que os propongo sólo necesita del X FUNCTIONS. Consiste en guardar mediante SAVEP el programa que deseamos cambiar, en la memoria extendida; sacar el módulo y colocarlo en la calculadora a la que queremos transferir el programa guardado, y ejectuar GETP o GETSUB, con lo que tendremos disponible el programa en la memoria del usuario, también mediante 2 instrucciones.

El método funciona porque a pesar de ser volátil la memoria extendida, conserva la información durante un tiempo suficiente para hacer el cambio.

¿Se le ocurre a alguien un método más fácil que no utilice soporte magnético?

J.A. Deza

Posicionamiento sobre el END

Normalmente cuando estamos poniendo a punto un programa, este se encuentra el último en memoria. Si ejecutamos otro programa o vamos a otra zona de memoria, puede resultar pesado llegar hasta el último mediante CAT 1 o mediante GTO «nombre del programa».

El siguiente programa colocado en cualquier parte de la memoria, os permitirá posicionar el puntero de programa sobre el .END. final.

```
01*LBL "GE"
02 RCL c
03 STO [
04 "⊢♦♦♦♦♦"
05 RCL [
06 X<> d
07 CF 00
08 CF 01
09 SF 02
10 SF 03
11 X<> d
12 CLA
13 STO [
14 "⊢AB"
15 RCL \
16 STO b
17 END
```

La estructura del registro c es la siguiente:

ΣREG		1 6 9		ROO		.END.
6	5	4	3	2	1	0

Conversión Décimal-Binario

Los dos programas siguientes nos permiten mediante el uso del módulo X FUNCTIONS y utilizando los registros alfa, la conversión a binario de números decimles hasta el $2^{24} - 1$ (el registro alfa solo almacena 24 caracteres).

El «B24» coloca el número binario en alfa de izquierda a derecha, es decir como se escribe normalmente (digito de mayor peso a la izquierda). En el «IB24» el número parece invertido (dígito de mayor peso a la derecha). Tanto en uno como en otro programa puede aparecer ceros no significativos. En el «B24» aparecerán a la izquierda y en el «IB24» a la derecha (no os lleve esto a confusión).

La conversión se hace byte por byte (de 8 en 8 dígitos binarios). La rutina 05 hace la conversión de 1 byte, mientras que las líneas 04-18 hacen algo equivalente a la conversión en base 256.

Para la rutina de 1 byte se utiliza X< >F, instrucción que permuta el contenido de X con el valor equivalente binario en las banderas 0 a 7. Por ejemplo 202 pone las banderas 7, 6, 3 y 1 (1100 1010).

Ya sólo deciros que cuando tenemos el resultado de un número grande, el desplazamiento de los números por la pantalla puede no ser fácil de ver. En estos casos puede ejecutarse PRA si se dispone de impresora o sino hacer ALPHA, APPEND y borrar de 1 en 1 copiándolos previamente.

J.A. Deza.

```
01*LBL "B24
"
02 CLA
03 RDN
04*LBL 04
05 R↑
06 RCL X
07 256
08 X>Y?
09 GTO 05
10 /
11 LASTX
12 X<>Y
13 INT
14 RDN
15 MOD
16 R↑
17 XEQ 05
18 GTO 04
19*LBL 05
20 X<>Y
21 X<>F
22 -7
23*LBL 01
24 FS? IND
X
25 "⊢1"
26 FC?C IND
X
27 "⊢0"
28 ISG X
29 GTO 01
30 -8
31 AROT
32 .END.
```

```
01*LBL "IB2
4"
02 CLA
03 RDN
04*LBL 04
05 R↑
06 RCL X
07 256
08 X>Y?
09 GTO 05
10 /
11 LASTX
12 X<>Y
13 INT
14 RDN
15 MOD
16 X<>Y
17 XEQ 05
18 GTO 04
19*LBL 05
20 X<>Y
21 X<>F
22 .007
23*LBL 01
24 FS? IND
X
25 "⊢1"
26 FC?C IND
X
27 "⊢0"
28 ISG X
29 GTO 01
30 .END.
```

modo TRACE. Si no tenemos ninguno de estos periféricos, pero si el módulo X FUNCTIONS y la ampliación de memoria suficiente para poder almacenar en ella el programa cuya longitud en bytes queremos saber, la siguiente rutina nos permitirá averiguarlo.

```
 01*LBL "BYT
ES"
 02 "NAME PR
GM?"
 03 AON
 04 STOP
 05 AOFF
 06 SAVEP
 07 RCLPT
 08 PURFL
 09 .END.
```

El programa es cargado en la memoria extendida. A continuación mediante el RCLPT, por tratarse de un archivo de programas, el registro de la memoria extendida destinado a almacenar la posición del puntero y otras informaciones como longitud del archivo en número de registros y tipo del mismo, en la zona del puntero contendrá la longitud en bytes que obtendremos en el registro X. Seguidamente el programa es eliminado de la memoria extendida mediante PURFL quedando libre el espacio de memoria inicial.

El proceso es algo lento pero no disponiendo de otro medio más eficaz, siempre es mejor que andar contando paso por paso la longitud del programa o utilizar el sistema de hacer un RCL b en la cabecera del programa, otro al final, pasar a decimal las posiciones en memoria obtenidas y luego restarlas.

Función DSL

En algunas ocasiones será interesante disponer de una función DSL (decrementa y salta si menor) la solución propuesta no es totalmente equivalente pero sirve siempre que se trate de un direccionamiento indirecto (IND) ya que el signo no es tenido en cuenta. Consiste en utilizar el ISG con un número negativo. Por ejemplo si queremos examinar el estado de las banderas 7 a la 0 en orden decreciente podemos hacer:

```
-7
LBL 00
FS? IND X
Opcion 1
Opcion 2
ISG X
GTO 00
```

Mediante la función DSE, el estado de la bandera cero no habría sido examinado dentro del bucle.

Conversión de datos a programas

El registro C cuya estructura ya ha sido comentada con anterioridad, es uno de los registros de estado más interesantes por la información contenida en él.

De forma parecida a la propuesta para el posicionamiento sobre el .END. en esta ocasión se trata de convertir todos los datos en memoria de programa. Es decir desplazar la denominada cortina.

```
 01*LBL "D-P
"
 02 "ö*×iö"
 03 RCL [
 04 RCL c
 05 STO [
 06 "⊢*****"
 07 RCL [
 08 X<> d
 09 CF 00
 10 CF 01
 11 CF 02
 12 CF 03
 13 X<> d
 14 STO [
 15 X<> Z
 16 STO \
 17 "⊢AB"
 18 RCL \
 19 CLA
 20 STO c
 21 .END.
```

La línea sintética 02 depende del número de módulos de memoria colocados cuando se trata de una 41C. Como ejemplo, en el caso de una C con dos módulos, la línea 02 tendrá por código exadecimal F5 18 00 01 69 18 y en el caso de una CV o una C con 4 módulos será F5 20 00 01 69 20. A continuación es sacado el contenido del registro c y manipulado para obtener la dirección del .END. y colocarla al final de la línea anterior para volver a introducirla en el registro. Al final de la ejecución de la rutina aparecerán en las zonas más altas de memoria lo que anteriormente eran datos como si de un nuevo programa se tratara.

Tras esta manipulación será necesario borrar una de las primeras líneas (y opcionalmente vuelta a escribir) y hacer un PACK o un GTO... para que el procesador pueda recalcular las nuevas direcciones de los LBL's y END's.

El interés de la rutina es convertir un programa escrito en la zona de datos mediante algún procedimiento como hacer un STO b con la dirección del comienzo de la zona de datos y programar allí directamente o bien mediante una rutina de codificación y el uso de STO.

Ecuación de 2º grado en el Stack

La solución propuesta por el manual para la resolución de ecuaciones de segundo grado no es la más brillante en cuanto a espacio ocupado en memoria se refiere.

Normalmente cuando se piensa en un determinado programa, la primera versión de este no está demasiado optimizada. Una vez tenemos el programa funcionando podemos hacer algunas mejoras para obtener una versión más compacta. Aunque no siempre es conveniente proceder así ya que esto puede tener dos problemas. Uno es el dificultar su estructura de tal forma que si es tratado de seguir por otra persona, no será tan facilmente comprensible. El otro problema es que no siempre un ahorro de memoria traerá consigno una disminución de su tiempo de ejecución.

En esta ocasión trataremos de dos pequeños programas de resolución de ecuaciones de segundo grado que han sido optimizados y que no utilizan registros de datos adicionales (sólo el stack).

```
01*LBL "X"
02 X<> Z
03 CHS
04 ST/ Z
05 ST+ X
06 /
07 ENTER↑
08 ENTER↑
09 X↑2
10 R↑
11 +
12 SQRT
13 ST- Z
14 +
15 END
```

El primero de ellos ocupa 17 bytes sin contar el LBL ni el END y obtiene las raíces reales previa introducción de los coeficientes de la ecuación $ax^2 + bx + c$ en la forma a ENTER b ENTER c. Si los coeficientes son introducidos en el orden c, b, a, el X Z inicial puede suprimirse quedando así un programa de sólo 15 bytes. Al final de su ejecución una de las raíces se encuentra en el registro X. Haciendo X<>Y obtendremos la otra.

En el caso de que las raíces sean complejas, al hacer la raiz cuadrada, la calculadora nos dará el mensaje de error DATA ERROR.

```
01*LBL "XC"
02 X<> Z
03 ST/ Z
04 ST+ X
05 CHS
06 /
07 ENTER↑
08 ENTER↑
09 X↑2
10 R↑
11 CF 01
12 X>Y?
13 SF 01
14 -
15 ABS
16 SQRT
17 ST- Z
18 X<>Y
19 FC? 01
20 +
21 .END.
```

Este segundo programa obtiene tanto las raíces reales como las complejas. La forma de introducción de los coeficientes es la misma que en el anterior. Si las raíces son reales, una de ellas aparecerá en el registro X y la otra en el Y. En el caso de ser complejas nos aparecerá la bandera indicadora 01 anunciándonoslo y quedando la parte real de las raíces en X y su parte imaginativa en el registro Y.

J.A. Deza

Decodificación del registro ALPHA y X

Dos pequeñas rutinas que permiten la decodificación del registro ALPHA (DCA) y la del registro X (DCX). La DCA utiliza a la DCX como subrutina. Al final de su ejecución se obtienen los códigos hexadecimales del contenido de X o de ALPHA según el caso. El programa DCX tiene algunas ventajas respecto de otros programas de decodificación hexadecimal: ocupa poco espacio en memoria, deja intacto el contenido de la pila (salvo el registro T) y es de rápida ejecución. El único pequeño inconveniente (al que uno se acostumbra muy pronto) es que los dígitos mayores que 9 (A-F) son representados respectivamente por: , , , < , = , > y ?

El programa DCA decodifica los 21 caracteres a la derecha del registro ALPHA, es decir, los registros M, N y O.

Un método a seguir por aquellos que no sepan crear las líneas sintéticas puede ser el siguiente:

Colocar en programa:

```
 1
STO IND Z
STO IND XX
YY
```

Donde XX representa el valor decimal del prefijo de la función sintética de dos bytes a crear, y el YY el subfijo de la misma según se indica a continuación

GEMINI 10X : 80 COLUMNAS, 120 cps.
GEMINI 15X : 132 COLUMNAS, 120 cps.

Delta 10 : 80 columnas, 160 cps.
Delta 15 : 132 columnas, 160 cps.

star IBM compatible

IMPRESORAS star

Radix 15 : 80 columnas, 200-38 cps.
Radix 15 : 132 columnas, 200-38 cps.

Powertype : 110 - 132 - 165 columnas, 18 cps.

De venta en establecimientos especializados.

IMPORTADO POR

SCS
COMPONENTES ELECTRONICOS, S. A

Consejo de Ciento, 409, 08009-Barcelona
Tfno.: 231 59 13
Télex 50204 SCS

```
1                    XX
STO IND Z   RCL   16
STO IND 78  STO   17
CLX         ARCL  27
            X<>   78
```

Por ejemplo para el caso X<>0 poner:

```
     YY
M  RDN
N  LASTX
O  CLX
P  X=Y?
```

Seguidamente, ejecutar el salta bytes BJ en la posición del STO IND Z, poner modo PRGM y borrar el T⊠, 1, y el STO OO y tendréis la función sintética deseada.

Es preciso señalar para los que no lo sepan que los subfijos M, N, O y P son representados por la impresora según la siguiente equivalencia:

```
M   [
N   \
O   ]
P   ↑
```

que es la forma en que aparecen en los listados. Por otra parte, la línea⊢◊◊◊◊ (⊢, 4 bytes nulos) puede conseguirse introduciendo en el punto adecuado del programa:

```
STO 01
⊢ABCD
BJ
SST
DEL 004
```

De forma análoga, variando el número de caracteres, se harán las demás líneas de nulos.

J. A. DEZA

Los dos programas siguientes permiten la decodificación del registro X y de los registros M, N y O de ALPHA.

El programa DCX decodifica el contenido del registro X en 1'4 segundos conservando los valores x, y, z de la pila. Después de la ejecución, el registro ALPHA contiene el código hexadecimal de X.

```
01◆LBL "DCX
"
02 ENTER↑
03 FIX 9
04 CLA
05 X<> [
06 "⊢◆◆◆◆"
07 X<> [
08 "⊢◆◆"
09 X<> [
10 ARCL \
11 "⊢◆"
12 X<> ]
13 "⊢◆"
14 ARCL X
15 X<> ↑
16 X<> ]
17 X<> \
18 CLX
19 FIX 4
20 ARCL X
21 STO ↑
22 X<> ]
23 X<> \
24 X<> [
25 RDN
26 PROMPT
27 END
```

El programa DCA visualiza los códigos hexadecimales de los 21 últimos octetos del registro ALPHA en grupos de siete octetos (catorce cifras); después de cada grupo hacer R/S para obtener el siguiente

```
01◆LBL "DCA
"
02 RCL [
03 RCL \
04 RCL ]
05 XEQ "DCX
"
06 RDN
07 XEQ "DCX
"
08 RDN
09 XEQ "DCX
"
10 END
```

RAMON C. MACIA

Asignación con delirios de grandeza

¿Qué pasa en una HP 41 cuando un usuario descuidado destruye un registro de asignación sin modificar consecuentemente el registro de índice de asignaciones (registros ⊢ o e para las teclas con SHIFT)?

Es una situación habitual en programación sintética, cuando se ha torturado la zona de memoria denominada "tabla de asignaciones" que la HP 41 en modo USER siga convencida que la tecla correspondiente está asignada aún.

Si se pulsa la tecla, la 41 busca en la tabla de aspiraciones el código de la función a ejecutar.

Después explora de arriba abajo la memoria de programa examinando cada LBL global para saber si está asignado a esta tecla. Sin éxito, claro está, puesto que hemos hecho desaparecer ilegalmente las "trazas" de la asignación.

La HP 41 decide salir de esta situación patológica utilizando el registro en que se encuentra como un registro de asignación. Determina, con el cuarto octeto del LBL global y su dirección en el registro, el código de una función ficticia a ejecutar.

En este momento, es suficiente modificar el SIZE para modificar al mismo tiempo la dirección de memoria del primer LBL global y sus octetos: haciendo variar el SIZE de 0 a 319, se obtienen sucesivamente en la misma tecla las 256 funciones de un octeto de la tabla de la HP 41.

Entre estas funciones, más de la mitad son sintéticas. Se obtiene en SIZE 010 el BJ, en SIZE 228 el QLOADER, en SIZE 050 los LBL locales deslocalizados así como los XEQ y GTO respectivos en SIZE 225, 226.

Señalemos aún el eG0 BEEP ——— (SIZE 088) y el TE (text enabler) en SIZE 223. Tantas funciones que anteriormente llenaban el teclado de los aficionados y sus tarjetas magnéticas. Para obtenerlas, consultad el cuadro adjunto.

El END y el LBL global son necesarios al principio de memoria para formatear correctamente la dirección del octeto que codifica la función. Se necesita realizar una compilación; y un GTO.. puede ser destructor. Conservad, por tanto, el programa ASS para generar esta falsa asignación.

Para obtener 256 asignaciones en una sola tecla

```
MEMORY LOST
PRGM
XEQ "END"
LBL "ASS"
RCL
STOP
STO
CLX
PRGM
XEQ "PACK"
ASN "ABS" Σ +
XEQ "ASS"
ASN "" Σ +
R/S
```

RCL	Se obtiene con el BG	RCL IND 16 SIGN
STO	Se obtiene con el BG	RCL IND 17 SIGN

Os dejamos el placer de descubrir las potentes funciones disponibles —bajo nombres bizarros— en la tecla que hayamos elegido.

M. Benaïm
J.A. Deza

Byte Grabber

El Byte Grabber (BG) o ladrón de octetos es una asignación muy útil en la fabricación de líneas sintéticas de programa.

Se trata de la asignación F73F (entre otras) que podreis obtener de la siguiente forma:

1) Borrar todas las asignaciones existentes y empaquetar (PACK)
2) Asignad una función a la tecla LN (15)
3) Asignad PACK a la tecla LOG (IY), o a otra tecla cualquiera
4) Acceder a los registros de asignaciones mediante el método descrito en anteriores trucos.
5) GTO. 005
6) DEL 003
7) ALPHA
8) T? AAAAAA (6 Aes)
9) ALPHA
10) GTO..

Si no teneis el módulo X FUNCTIONS o la CX, en la introducción de la línea de texto T? AAAAAA os encontrareis con T? A..... ya que las direcciones de memoria de estos caracteres no existiran, y por tanto las 5 últimas A caerán al vacio. La ? (3F) de la línea de texto posee el mismo código que el GTO 15 (función de dos octetos) y la 1ª A (41) colocada en el lugar de la asignación correspondiente al código de tecla equivale en este caso a la tecla LN. Si en lugar de querer la asignación en el LN la quisieramos en la TAN, deberíamos poner una B; o una C para el SST, etc. A partir de este momento disponeis de una potente asignación XROM 28, 63 que os permitirá la introducción de programas con líneas sintéticas, y que completará a la ya conocida BJ (salta octetos).

Al introducir una función en memoria de programa, la 41 piensa que no puede tener más de 3 octetos. Si no hay espacio suficiente, se libera 1 registro (7 octetos), lo que en principio es suficiente para colocar los supuestos 3 octetos. El BG tiene dos características importantes, por una parte es una función de 3 octetos (2 del GTO 15 y uno del TEXTO 7), y por otra, el primer octeto es el indicador de una cadena de 7 caracteres. Al ejecutar BG en programa, será introducido F7 003 F en el correspondiente registro, pero como el primer obteto es un TEXTO 7, y sólo los dos primeros le son proporcionados, los 5 restantes serán tomados de la memoria de programa para así completar la cadena.

Veamos mediante un ejemplo como aprovechar esta caracteristica. Introducir en PRGM

```
01 LBL'R
02 STO IND 31 (91 9F)
03 LASTX (76)
GTO .001
_BG(LN)
```

En programa podreis ver:

```
01 LBL'R
02 '♦?♦♦♦Q (F7003F0000
000091)
03 TONE N (9F 76)
```

Como entre el LBLTR y el STO IND 31 no había espacio para introducir la asignación, es liberado un registro (7 octetos), pero la línea de texto necesita 8 octetos y el TEXT 7 (F7) robará el primero (91) del STO IND 31. Esto hará que quede libre el 9F (TONE) que necesita a su vez de un octeto (ya que se trata de una función de dos) y será tomado de la siguiente posición de memoria (el 76 correspondiente al LASTX) constituyéndose así el TONE N (9F 76). Ya sólo será necesario borrar la línea de texto y eventualmente empaquetar el programa.

A modo de resumen y como chuletario para aquellos que no conozcan los códigos de las funciones de la 41 a continuación podreis ver una tabla que os permitirá la creación de las funciones sintéticas más comunes.

PREFIJOS	XX
RCL	16
STO	17
ST+	18
ST−	19
ST*	20
ST/	21
ISG	22
DSE	23
VIEW	24
ΣREG	25
ASTO	26
ARCL	27
FIX	28
SCI	29
ENG	30
TONE	31
SF	40
CF	41
FS?C	42
FC?C	43
FS?	44
FC?	45
GO/XQ I	46
X<>	78

POSFI.	YY
T	CLΣ
Z	X<>Y
Y	PI
X	CLST
L	R↑
M	RDN
N	LASTX
O	CLX
P	X=Y?
Q	X≠Y?
⊢	SIGN
a	X<=0?
b	MEAN
c	SDEV
d	AVIEW
e	CLD
IND T	TEXT00
IND Z	TEXT01
.	.
.	.
.	.
.	.
IND d	TEXT14
IND e	TEXT15

Para crear una función, introducir en programa

```
01 LBL'R
02 STO IND XX
03 YY
```

donde XX corresponde al valor del prefijo según la tabla anterior e YY es la función equivalente al postfijo según la otra tabla.
Por ejemplo, para hacer VIEW Q poner:

```
01 LBL'R
02 STO IND 24
03 X≠Y?
GTO .001
BG (LN)
```

Otro ejemplo X ≤ IND M.

```
01 LBL'R
02 STO IND 78
03 'ABCDE
GTO .001
BG (LN)
```

Advertencia para los curiosos desconocedores de la sintética: cuidado con el STO c.
Espero que con lo dicho ya no tengais problemas en la introducción de los programas que utilicen funciones sintéticas, salvo en la creación de líneas de texto. Pero esto ya es otra historia.

Etiquetas globales

A veces es deseable disponer de etiquetas globales (los LBL que aparecen mediante CAT 1) de la A a la J y de la a a la e. Estas etiquetas son normalmente locales (no son catalogadas) y ejecutables mediante la pulsación de la correspondiente tecla en modo USER.

Los que no dispongan de un programa de asignaciones, podrán introducirlas en programa utilizando el BG descrito anteriormente.

```
01 LBL'R
02 STO IND 64
03 LBL 00
04 'PA
GTO .001
BG (LN)
```

Otro ejemplo LBL тe
Poner en PRGM

```
01 LBL'R
02 STO IND 64
03 LBL 00
04 'Pe
GTO .001
BG (LN)
```

Los GTO y XQ se crearán de forma parecida.
Ejemplo GTO тB
Poner en PRGM

```
01 LBL'R
02 STO 29
03 'B
GTO .001
BG (LN)
```

Otro ejemplo XEQ т a.

```
01 LBL'R
02 STO 30
03 'a
GTO .001
BG (LN)
```

Espero que con estos ejemplos haya quedado claro el método (al menos es lo que se pretendia). Y como una imagen (en este caso un ejemplo) vale más que mil palabras...

J.A. Deza

Tarjeta mágnetica CIKEYS

Los desafortunados no poseedores del módulo X FUNCTIONS en más de una ocasión habrán echado de menos una instrucción que borre de una sola vez todas las asignaciones (con el citado módulo es suficiente ejecutar la instrucción CLKEYS). El método de borrar las asignaciones una a una, aparte de ser engorroso, requiere conocer las teclas que se encuentran asignadas para poder borrarlas mediante ASN.

Si disponéis de un lector de tarjetas HP 82104 A,. podréis realizar esta tarea también de forma fácil. Para ello podréis seguir el método descrito a continuación:

— Aprovechando un temido MEMORY LOST, hacer una única asignación a una tecla cualquiera y borrarla seguidamente, mediante ASN.

— Sin hacer PACK ni GTO.., ejecutar WSTS y pasar en el lector las dos bandas magnéticas requeridas para almacenar el estado de la máquina.

Una vez hecho esto, dispondréis de una tarjeta de estado que podréis bautizar «CLKEYS» y que os servirá para borrar todas las asignaciones.

Para ello, será suficiente introducir las dos pistas registradas mediante el método descrito anteriormente. Tras un empaquetado (XEQ «PACK» o un GTO.. tendréis disponibles los registros ocupados por las asignaciones para almacenamiento de datos o programas.

J. A. Deza.

Asignación con GØBEEP

La función eGOBEEP, tiene por código hexadecimal A7 lo que en decimal es 167. Los «UNN» (usuarios no normalizados) o los que posean un programa de asignaciones, no tendrán problemas para disponer de ella en una tecla cualquiera.

Gracias a ella, pueden conseguirse todas las funciones (códigos XROM) del HP-IL sin necesidad de tenerlo conectado para introducirlas en un programa. Las XROM 28 y 29 obtenidos dependen del valor introducido en los dos «prompts» visualizados detrás del nombre. Estos dos «promps» no aceptan ni el IND ni los X, Y, Z, T, L, de la pila, pero si los argumentos ALPHA, en cuyo caso la función obtenida será un LBL global o local dependiendo de que introduzcáis una cadena de caracteres o las letras A-J.

Para obtener las funciones del lector de casetes (XROM 28,00 a 28,24) se dará como valores de 00 a 24, para las del controlador del blucle de 25 a 41 y para la impresora, de 64 a 89. Los valores 00, 25, 26, 64 y 90 no tienen ninguna utilidad ya que sólo visualizan los títulos de los catálogos (-MASS ST 1H, - -, -CTL FNS, -PRINTER 2E,--).

Los valores 42,91 y 92 son divertidos, os dejo el placer... También pueden obtenerse las funciones no programables NEWM (03), LIST (71), PRP (77), si bien su utilidad es mínima al no poder escoger en programa el argumento de las mismas (la 41 toma como argumento el contenido de uno de los registros del microprocesador que coincide con el valor de - -). No os aconsejo su uso en programa sobre todo con el NEWM y un lector de casetes colocado (obtendréis NEWM 03, y el que avisa...).

El caso de PRP es diferente puesto que los argumentos alfanuméricos se guardan en el registro Q. Este registro es fácilmente accesible mediante un LBL por ejemplo. Receta: haced eGOBEEP ALPHA NOMBRE ALPHA y eGOBEEP 77.

Hay otras funciones interesantes. Pero, eso ya es otra historia...

J. A. Deza.

New Brain

Lo bueno sí breve, dos veces bueno ...

Trucos

Aqui tienen un truco para su New-Brain
Para todos los felices propietarios de este ordenador con teclado Azerty,el manual(para teclado Qwerty) no conviene siempre. El Shift/Escape se reemplaza por Control/0.Para parar un Verify, Load, Save "*" será reemplazado por " ".
Para verificar el resto de memoria disponible, hay que hacer: ? FREE (lo que no está tan evidente en el manual).
El mejor modo de visualizar los 512 carácteres disponibles es de escribir este pequeño programa

```
10 FOR I=TO 255
20 PUT 27,i,26
30 NEXT I
40 END
```

Saldrá el primer tipo de carácteres.
Despues teclear CONTROL/W y b (segúndo tipo de carácteres).
Luego teclear CONTROL/W y H (tercer tipo de carácteres).
En fin teclear CONTROL/W y J (cuarto tipo de carácteres).

Otras cosas :
-En modo gráficos se obtiene un circulo haciendo: PLOT ARC(D,O). D es igual al valor del arco y O a la rotación del ángulo en el centro.
-La instrucción PEEK(X)(OX 65535)fúncciona muy bien con un PRINT o "?"delante del PEEK.

Mayúsculas o minúsculas...

Ejecutando la instrucción POKE 43,1 el teclado solo admite caracteres en mayúsculas; si se incluye una instrucción INPUT A$ justamente detrás, se deberá de teclear forzosamente en mayúsculas, ya que si no es asi nos dará un error.

Si la Instrucción que ejecutamos es POKE 43,0 lo que hará es admitir entradas del teclado solo en letras minúsculas; analogamente una instrucción del tipo del INPUT A$ solo funcionará bien si está escrita en minúsculas. Ambas instrucciones son las equivalentes a apretar las teclas CTRL/L y CTRL/0.

Notemos que la respuesta a las instrucciones:

OPEN # 3,5: INPUT #3,A$
U

OPEN # 3,6: INPUT #3,A$

Será introducido siempre en minúsculas salvo los caracteres obtenidos por [SHIFT], ya que el teclado direccionado como periférico de tipo 5 o 6 no tiene en cuenta el haber introducido previamente [CTRL/I].

Laurent Laloum.

Conocimientos asegurados

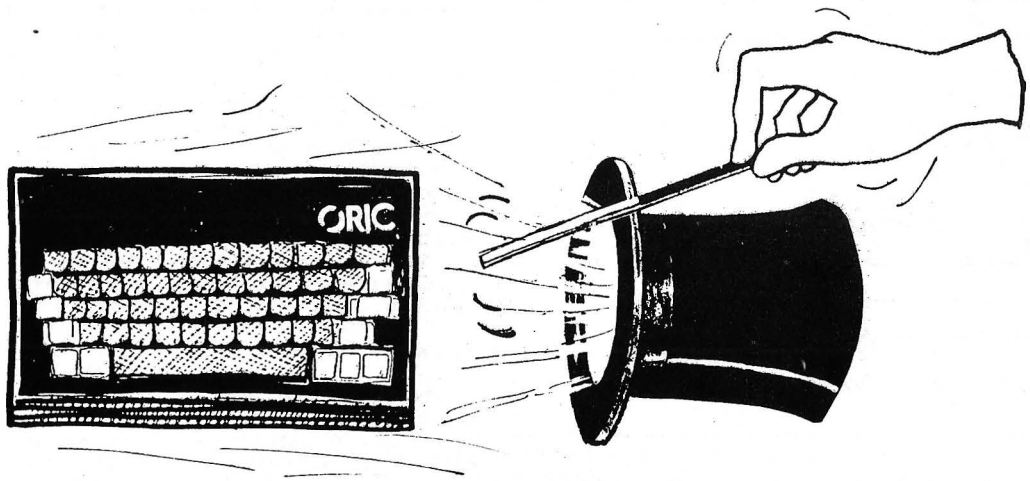
Resultados de investigaciones procesados a largo plazo.
Ideas maduras y conocimientos detallados.
Retenidos con seguridad y accesibles en cualquier momento.
Los soportes de datos Maxell de los que se puede fiar.
Un producto sometido a las más duras pruebas y largas investigaciones.
En los disquetes Maxell están asegurados sus conocimientos.

SISCOMP S.A.

Rosselló, 184. 4rt, 3a
08008 – BARCELONA.
Telèfon 323 45 65
Telex 98251 SCMP E

Maxell Europe GmbH · Emanuel-Leutze-Straße 1 · D-4000 Düsseldorf 11 · Tel.: 07 49-2 11/59 51-0 · Tx.: 8 587 288 mxl d

maxell®
soportes de datos
la fiabilidad



Al igual que la baraja en el bolsillo del mago, no podía faltar los trucos del Oric en la chistera del Ordenador Personal. ¡A ver si tu nos envías los tuyos, o una postal!

Extensiones al Basic Oric 1

El corto programa que publicamos implanta en memoria el código necesario para crear los siguientes comandos:

!F: teclado rápido,
!M: teclado medio,
!N: teclado normal,
!S: teclado lento,
!H: paso a modo HIRES,
!T: paso a modo TEXT,
!E: generación de un EXPLODE.

Todos los demás ! generan un SHOOT.

El código máquina se implanta a partir de la dirección # 400 y ocupa 86 octetos. El sobrepasar la dirección # 420 indicada en el manual como un límite no parece molestar al Oric.

Los comandos de velocidad del teclado modifican el contenido del «DATA REGISTER» del contador que genera las interrupciones.

!F multiplica la frecuencia a la que se interrumpe el 6502, por un factor 3 más o menos, acelerando la repetición automática, lo que se puede apreciar en el transcurso de la edición. El inconveniente es que la ejecución de los cálculos se ve muy retardada (alrededor de 2 veces).

!S, por el contrario, disminuye la frecuencia de las interrupciones, dejando así al 6502 el tiempo suficiente para ejecutar los cálculos.

Cuidado: si su programa ejecuta WAIT, el teclado debe estar en normal (!N) para evitar un plante.

Los restantes comandos llaman a las correspondientes rutinas del Basic.

André Thévenin

```
10 : AD=#400 'Direccion de comienzo
20 : REPEAT : REM Colocacion de los codigos maquina
30 : READ DA:POKEAD,DA:AD=AD+1
40 : UNTIL DA=#7F
50 : AD=#2F5:DA=#400:DOKEAD,DA
100 :STOP
500 :
510 : REM Comienzo del codigo maquina
520 :
530 :DATA #A0,#00      '    LDY #$00
540 :DATA #B1,#E9      '    LDA ($E9),Y
550 :DATA #E6,#E9      '    INC  $E9
560 :DATA #D0,#02      '    BNE   X1
570 :DATA #E6,#EA      '    INC  $EA
580 :DATA #C9,#46      'X1 CMP #"F"
590 :DATA #F0,#0F      '    BEQ   X2
600 :DATA #C9,#4D      '    CMP #"M"
610 :DATA #F0,#12      '    BEQ   X3
620 :DATA #C9,#4E      '    CMP #"N"
630 :DATA #F0,#15      '    BEQ   X4
640 :DATA #C9,#53      '    CMP #"S"
650 :DATA #F0,#18      '    BEQ   X5
660 :DATA #4C,#3D,#04  '    JMP   X6
670 :DATA #A9,#00      'X2 LDA #$00
680 :DATA #A2,#0E      '    LDX #$0E
690 :DATA #4C,#36,#04  '    JMP   X7
700 :DATA #A9,#88      'X3 LDA #$88
710 :DATA #A2,#13      '    LDX #$13
720 :DATA #4C,#36,#04  '    JMP   X7
730 :DATA #A9,#10      'X4 LDA #$10
740 :DATA #A2,#27      '    LDX #$27
750 :DATA #4C,#36,#04  '    JMP   X7
760 :DATA #A9,#FF      'X5 LDA #$FF
770 :DATA #A2,#FF      '    LDX #$FF
780 :DATA #8D,#06,#03  'X7 STA  $0306
790 :DATA #8E,#07,#03  '    STX  $0307
800 :DATA #60          '    RTS
810 :DATA #C9,#54      'X6 CMP #"T"
820 :DATA #D0,#03      '    BNE   X8
830 :DATA #4C,#A9,#E9  '    JMP   TEXT
840 :DATA #C9,#48      'X8 CMP #"H"
850 :DATA #D0,#03      '    BNE   X9
860 :DATA #4C,#BB,#E9  '    JMP   HIRES
870 :DATA #C9,#45      'X9 CMP #"E"
880 :DATA #D0,#03      '    BNE   XA
890 :DATA #4C,#18,#F4  '    JMP EXPLO.
990 :DATA #4C,#15,#F4  'XA JMP SHOOT
4000 :
4500 :
5000 :DATA #7F 'FIN DEL PROGRAMA
```

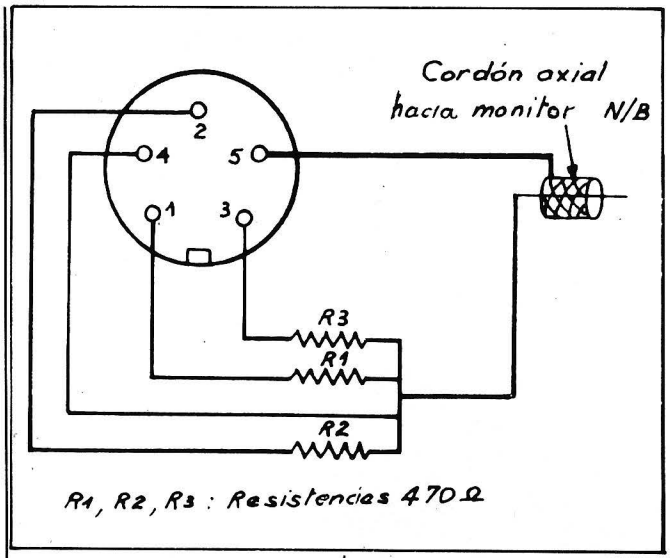
Reanimar los programas comatosos

El monitor Basic contiene una rutina que reajusta las direcciones de las líneas (dirección colocada al principio de cada línea, que indica el principio de la línea siguiente). Es la rutina # C56F. El Basic la emplea para la inserción o supresión de una línea, pero puede utilizarse para recuperar un programa en el que aparezca la línea: «21845 UUUUUUU». Este problema se debe a una carga inexacta de las direcciones de las líneas. La rutina mencionada lo arreglará.

Con un NEW lo único que hace (a excepción de los punteros) es colocar el valor 0 en # 501 y # 502; basta, para recuperar dicho programa, hacer POKE # 502,1 y después CALL # C56F. POKE # 502 es necesario, porque ésta rutina detiene su trabajo a partir de que encuentre tres ceros en la continuación.

Conexiones a monitor blanco y negro

Presentamos el esquema correspondiente a la acometida de un ordenador Oric 1 a un monitor blanco y



negro. En realidad, el aviso del Oric 1 no indica la forma de hacerla y todavía no existe en el comercio el cordón que permite hacer este enlace.

Este montaje, que funciona correctamente cuando es en blanco y negro, proporciona unos contrastes malos cuando los colores elegidos son diferentes de 0 ó 7. En realidad, los valores de las resistencias no están en la relación que proporciona el blanco absoluto para un aparato de televisión.

Guillaume Dorbes

Punteros de memoria del Basic del ORIC

Estas informaciones son muy útiles en los casos de errores del tipo «out of memory» y cuando es imposible introducir una línea en un programa mal cargado.

• Puntero inferior de la memoria del Basic (memoria programa). La dirección está en los octetos #9A y #9B. En principio está posicionada en #501 como indica el mapa de la memoria del manual.

Para modificarla (para, por ejemplo, colocar un programa Basic en la parte superior de la memoria), colocar el valor 0 en el octeto anterior a la nueva dirección (se puede verificar que el octeto #500 es igual a cero) y después hacer NEW para ajustar los demás punteros.

Seguidamente no olvide el necesario HIMEM, porque si no lo hiciera le ocasionaría un «OUT OF MEMORY ERROR».

• Puntero superior de la memoria programa. Se encuentra en los octetos #9C y #9D. Tras la puesta en marcha o tras un NEW, la dirección contenida en estos octetos es #503 porque en #501 y #502 se encuentra la dirección de la próxima línea del programa (en concurrencia la dirección 0 tras un NEW). A causa de este puntero surgen los problemas tras haber cargado una parte de la memoria desde cássette. En efecto, tras la carga este puntero se reajusta a la cima de la zona de memoria cargada (la última en fecha). Para restablecer el puntero y por tanto el programa Basic hay que anotar la dirección del puntero (con PRINT DEEK (#9C)) antes de cargar la zona de memoria y después de haberla cargado, restablecer el puntero con DOKE #9C,x; en donde x es el valor anterior a la carga. Después CLEAR reajustará los punteros de las variables y todo quedará como antes.

• Punteros variables y tablas. El puntero de la dirección del principio de la zona «variable» está en la dirección #9E. Es modificado por un NEW, un CLEAR, o un RUN (en estos tres casos, su valor se hace el que tiene el puntero en #9C, ya que las variables se vuelven a poner en cero; el vértice de las variables se reúne, pues, con el vértice del programa).

El puntero de la dirección del vértice de la zona «variable tablas de variables» está en la dirección #A0. Funciona siguiendo el mismo principio que el puntero anterior; pero el valor de #0 es siempre igual o superior al de #9E, porque en el Oric las variables simples se colocan siempre antes que las tablas de variables en la memoria.

Veamos ahora el modo de restablecer programa y variables tras un NEW o un CLEAR.

La acción del comando NEW es la siguiente: coloca la dirección 0 en #501 (dirección de la próxima línea del programa) y ajusta los punteros #9C, #9E y #10 a la dirección #503. Para restablecer un programa tras un NEW, se ha debido anotar en alguna parte los valores contenidos en estos cuatro punteros y restablecerlos con DOKES después de haber hecho NEW. Tras un LIST o un RUN el programa todavía está presente.

Del mismo modo, tras un CLEAR basta restablecer los punteros #9E y #A0 a su valor anterior a CLEAR (el valor de #9C no se altera por un CLEAR).

Señalemos que en este artículo que trata de punteros, hay que dar por supuesto que cuando se trata del valor de la dirección x, nos referimos al valor de x +256 veces el valor de (x + 1). Este valor se visualiza (modifica), con un PRINT DEEK ( ); (DOKE X,x).

Denis Sebbag

Valoración de expresiones en Basic

La función VAL del Basic permite transformar una serie alfanumérica en un número, pero no puede valorar una expresión del tipo COS(X)*5 y aún menos una variable. Esto sería muy útil, sobre todo en programas de tipo trazado de curvas o recogida de datos.

En realidad, por ejemplo, en un programa de trazado de curvas hay que interrumpir el programa e introducir a mano la línea que contiene la función.

Esta pequeña rutina va a remediarlo sustituyendo la línea especificada por el contenido del tampón teclado (zona que acumula los caracteres a medida que se pulsan en el teclado), tras haber codificado su contenido como si fuera una línea de programa Basic. Es muy sencillo, porque después de un INPUT A$ la serie introducida se encuentra en ese tampón.

Ante todo, determine la línea a cambiar y llénela de espacios tecleando:

XX (número de línea) REM (seguido de tantos espacios como sea posible para que la nueva línea no invada la siguiente).

Después, en el lugar en que quiera que el usuario introduzca la línea, teclee:

INPUT A$: iXX (número de línea a sustituir). El principio es muy sencillo: la línea 2007 contiene un JRS#C60A. Es la rutina del intérprete Basic que busca las palabras clave del Basic y las sustituye por su código.

En la línea 2013, la rutina busca la dirección de la línea especificada por el !, después el resto de la rutina coloca la línea codificada a partir de la dirección determinada por la rutina #C6E4. Finalmente, el programa llena los espacios no empleados de la línea con el código 32 (justamente el espacio) para obtener una línea permitida.

Adaptación para Oric Atmos: puede ver las tres líneas del programa que hay que cambiar para conseguir el funcionamiento en Atmos.

Observación: una vez que se haya ejecutado el programa, puede salvaguardarlo en su forma de lenguaje de máquina, tecleando:
CSAVE«», A#400, E#42F

Para emplearlo, tras cargar, teclee:
DOKE#2F5,#400
para que le sirva el signo de exclamación como llamada de la rutina.

PROGRAMA

```
999 REM  ****************************
1000 REM *      MODIFICACIONES DE    *
1005 REM *        LINEAS   BASIC     *
1008 REM *                           *
1010 REM * AUTOR.-     D. SEBBAG     *
1012 REM *                           *
1014 REM *  (C)     EL AUTOR    Y    *
1018 REM *  EL ORDENADOR PERSONAL    *
1019 REM ****************************
1020 DIREC=#400
1030 REM == RUTINA DE IMPLANTACION ==
1040 RESTORE
1050 FOR I=DIREC TO DIREC+47
1060 READ VA:Q=Q+VA
1070 POKE I,VA
1080 NEXT I
1090 DOKE #2F5,DIREC+14
1100 IF Q=6908 THEN END
1110 PRINT "ERROR EN LOS DATOS"
2000 REM
2001 REM== LISTADO DATOS+NEMONICOS ==
2003 DATA #A5,#E9      'LDA #E9
2004 DATA #48          'PHA
2005 DATA #A9,#35      'LDA %#35
2006 DATA #85,#E9      'STA #E9
2007 DATA #20,#0A,#C6 'JSR #C60A
2008 DATA #68          'PLA
2009 DATA #85,#E9      'STA #E9
2011 DATA #20,#77,#CE 'JSR #CE77
2012 DATA #20,#67,#D8 'JSR #D867
2013 DATA #20,#E4,#C6 'JSR #C6E4
2014 DATA #A2,#00      'LDX %#00
2015 DATA #A0,#04      'LDY %#04
2016 DATA #B5,#35      'LDA #35,X
2017 DATA #F0,#06      'BEQ #06
2018 DATA #91,#CE      'STA (#CE),Y
2019 DATA #E8          'INX
2020 DATA #C8          'INY
2021 DATA #D0,#F6      'BNE #F6
2022 DATA #B1,#CE      'LDA (#CE),Y
2023 DATA #F0,#07      'BEQ #07
2024 DATA #A9,#20      'LDA %#20
2025 DATA #91,#CE      'STA (#CE),Y
2026 DATA #C8          'INY
2027 DATA #D0,#F5      'BNE #F5
2028 DATA #60          'RTS
```

Adaptación para el Atmos

```
2011 DATA #20,#03,#CF 'JSR#CF03
2012 DATA #20,#22,#D9 'JSR#D922
2013 DATA #20,#B3,#C6 'JSR#C6B3
```

PROGRAMA DE APLICACION

```
10 HIRES
20 PRINT"INTRODUCIR LA FUNCION DE LA FORMA:"
30 INPUT"Y=COS(X)*4+EXP(X)   POR EJEMPLO    ";A$
40 !60
50 FOR X=0 TO 239
60 REM ESTA LINEA ES NECESARIA PARA EVITAR SUSTOS EN LA SIGUIENTE ....
70 CURSET X,Y,1
80 NEXT
```

Soluciones a sus problemas de gestión.

DYNADATA INFORMATICA

SPECTRAVIDEO

SVI•728 MSX

Compatibilidad Universal.

Mientras la guerra de precios y la confusión reinan en torno nuestro. Spectravideo trabaja estableciendo estándares por los cuales otros ordenadores personales, de compañías conocidas mundialmente; entre los que podemos citar, entre otros, Fujitsu, Toshiba, Sanyo, Hitachi, General, etc... MSX es el último ejemplo de cómo Spectravideo está afianzando –y configurando– la industria del ordenador personal.

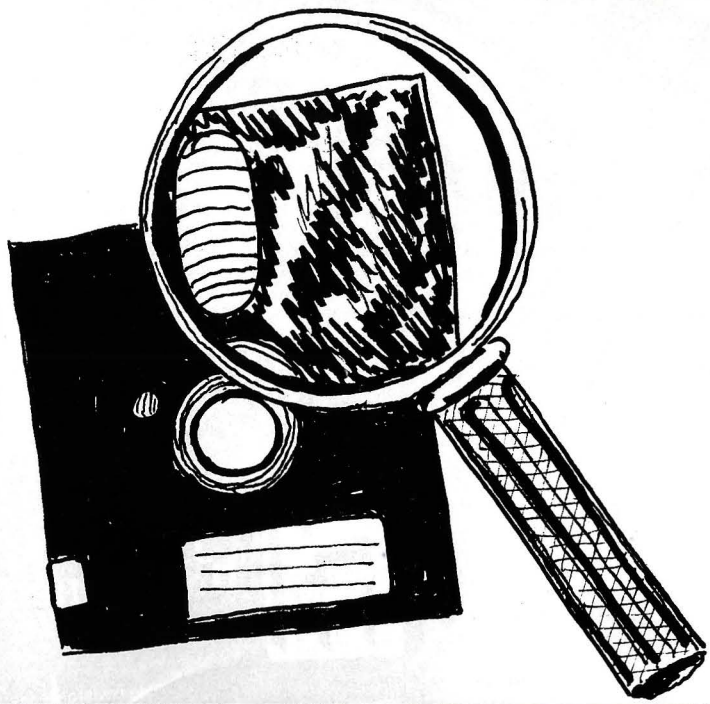
El 15 de junio de 1983 Spectravideo Inc. junto con las más importantes firmas de electrónica japonesas, lanzó al mercado el MSX; el más trascendental estándar en la historia de la Informática personal. MSX es la denominación dada a una configuración específica de Hardware/ Software que hace posible la intercambiabilidad de productos.

Spectravido considera muy estimable su participación en MSX, pero se enorgullece mucho más por un hecho; ha sido su ordenador SV-318 el utilizado como prototipo para el diseño del MSX. Hay dos consecuencias importantes en ésto: Primero, todo Hardware MSX –ordenadores, periféricos y otros dispositivos– desarrollado en el futuro, estará basado en diversos aspectos clave del diseño del SV - 318. ¿Qué significa ésto para usted, el usuario? Una gran ventaja, porque cuando usted compra un SV-318, no sólo tiene usted la posibilidad de utilizar el software y el hardware desarrollado por Spectravideo, sino que además puede disponer de los más destacados equipos diseñados por los otros fabricantes que han participado en MSX. Además en el aspecto software, el MSX está ampliamente inspirado en la construcción lógica del SV-318.

Se pueden conectar consolas de MSX en las Redes de Area Local con hasta 32 unidades.

DYNADATA INFORMATICA

DISTRIBUIDOR: Sor Angela de la Cruz, 24 - 28020 Madrid. Telfs. (91) 279 21 85 - 279 28 01 - 270 01 93 - 270 76 75
DELEGACION: Aribau, 61, entlo. 08011 Barcelona Telfs. (93) 254 73 04 - 254 73 03



Cada vez hay que apretarse más el cinturón para soportar las embestidas de la vida, y siempre viene bien tener unos "ahorrillos". Para que os los gasteis estas vacaciones, tomar 2 K (y gastarlos en cosas productivas. ¡Que está la vida muy dura!

OSBORNE Gane 2K octetos en los disquetes simple densidad de su Osborne 1

Todos los disquetes del Osborne 1 incluyen un programa llamado AUTOST.COM cuyo fin es conseguir la carga automática de otro programa al poner en marcha el sistema. Así, por ejemplo, el disquete del Basic carga automáticamente el MBASIC en memoria y lo deja listo para su uso. El único inconveniente que tiene este programa es que ocupa 2K octetos, lo cual no deja de ser mucho espacio si consideramos que la capacidad real de un disquete de simple densidad es de 90K octetos.

¿Cómo conseguir que se siga cargando automáticamente un programa, el MBASIC por ejemplo, sin tener necesidad de utilizar el programa AUTOST.COM?. Muy sencillo, sobre todo si tenemos en cuenta que el AUTOST.COM no es más que un programa que se ejecuta automáticamente porque se ha introducido un cambio en el sistema CP/M, y cuyo único fin, el del AUTOST.COM se entiende, es mostrarnos por pantalla el logotipo de la Osborne Computer Corporation y cargar a continuación otro programa (en nuestro ejemplo vamos a utilizar el ya citado MBASIC). Así pues, lo que vamos a hacer es indicarle al CP/M que en vez de cargar y ejecutar automáticamente al poner en marcha el sistema el programa AUTOST.COM, nos cargue directamente el MBASIC, con lo cual nos ahorraremos los 2K octetos que ocupa el AUTOST.COM en el disquette del Basic, además de los tiempos de carga y ejecución de este programa.

Lo primero que tiene que hacer es obtener una nueva copia de su disquete de Basic por medio del programa COPY (incluido en el disquete CP/M System), y otra copia de su disquete de utilidades CP/M por el mismo procedimiento. Si ya las tiene vamos a pasar a continuación a describir ordenadamente los pasos que tiene que realizar.

1.— Oprima el botón RESET, introduzca su disquete de utilidades CP/M en la unidad A y el del Basic en la B y pulse RETURN. (**RECUERDE**, las copias que acaba de hacer, **NUNCA** los originales).

2.— Después de que el sistema le informe del contenido de su disquete de utilidades CP/M haga lo indicado en el siguiente listado en el mismo orden en que aparece. Tenga en cuenta que sólo debe teclear lo

FIGURA 1

```
A>movcpm 60 * <RET>  (<RET> → TECLA RETURN)

CONSTRUCTING 60k CP/M vers 2.2
READY FOR "SYSGEN" OR
"SAVE 39 CPM60.COM"
A>save 39 b:cpm60.com <RET>
A>ddt b:cpm60.com <RET>
DDT VERS 2.2
NEXT  PC
2800 0100
-d2008,2017 <RET>
2008 E5 01 07 41 55 54 4F 53 ...AUTOS
2010 54 20 30 31 32 33 34 35 T 012345
-s200a <RET>
200A 07 08 <RET>
200B 41 4d <RET>
200C 55 42 <RET>
200D 54 41 <RET>
200E 4F 53 <RET>
200F 53 49 <RET>
2010 54 43 <RET>
2011 20 00 <RET>
2012 30 . <RET>
-d2008,2017 <RET>
2008 E5 01 08 4D 42 41 53 49 ...MBASI
2010 43 00 30 31 32 33 34 35 C.012345
-g0 <RET>

A>sysgen <RET>

        Operating System Generation Program
          OSBORNE COMPUTER SYSTEM ONE
                Revision 1.5

SOURCE drive (A or B) <RET>
-----------------------------------------------
DESTINATION (A,B  or  RETURN to exit) b <RET>
Put DESTINATION diskette in B, then press RETURN
        System copied successfully.

-----------------------------------------------
DESTINATION (A,B  or  RETURN to exit)<RET>
A>
```

7 líderes en su sector

REDACCION Y PUBLICIDAD:
C/ Ferraz, 11 - MADRID-8.
TEL.: 241 34 00

que está subrayado, pues lo demás es información que le aparecerá por pantalla como consecuencia de las instrucciones que Vd. le vaya dando al sistema. En realidad lo que hemos hecho es teclear ^P (^→tecla de control →CTRL) para que el CP/M nos liste en la impresora todo lo que salga por pantalla (incluidos nuestros comandos), por lo tanto, si Vd. realiza exáctamente todos los pasos indicados en este listado sus cambios funcionarán correctamente, pues son una transcripción real de lo realizado en otro sistema idéntico al suyo y equipado con una impresora.

Si está preparado, vamos a la figura 1.

Ahora ya puede sacar los dos disquetes y poner el del Basic en la unidad A. Oprima el botón RESET y a continuación pulse RETURN, verá como se carga su MBASIC y queda listo para trabajar. (Si me lo permite le aconsejo que borre ahora mismo todos los ficheros de su nuevo disquete de Basic, a excepción del MBASIC y, si lo prefiere, deje también el XDIR).

Por cierto, si se equivoca en algún paso, o si algo falla, vuelva a comenzar desde el principio, es decir, sacando las copias de sus disquetes de utilidades CP/M y de Basic.

Unas palabras mas. Todo lo aquí indicado sirve para cualquiera de los disquetes que le han sido suministrados con su Osborne 1, con la diferencia de que en vez de cambiar en el CP/M el AUTOST por el MBASIC lo tendría que variar por el nombre del programa que quiere que se cargue y ejecute automáticamente al poner en marcha el sistema. Por supuesto también puede incluir programas suyos, así, por ejemplo, si Vd. tiene en su disquete Basic el programa NOMINA.BAS puede conseguir que este se cargue y ejecute automáticamente al conectar el sistema. Para ello tenga en cuenta que al trabajar con el DDT en la dirección 200A (–S200A) ha de introducir en hexadecimal la longitud del nombre del programa o comando que quiera que se ejecute automáticamente al poner en marcha el equipo (la longitud mas 1, y contando el 00 hexadecimal que se introduce al final), y en las siguientes posiciones el valor hexadecimal de cada uno de los caracteres que componen su nombre de programa o comando, seguido de un 00 hexadecimal. Si tiene dudas acerca de la utilización del DDT consulte los manuales que le entregaron con su Osborne 1.

Luis de Cáceres Muñoz.

Consiga que sus PRINT aparezcan en la impresora en lugar de en la consola

Partiendo del supuesto de que usted tiene un sistema CP/M standard con consola e impresora, y que esta última está conectada a un interface asignado al dispositivo LST, voy a mostrarles cómo pueden conseguir que las sentencias PRINT de sus programas MBASIC-80 actúen como si se tratasen de sentencias LPRINT, con lo que los datos serán mostrados en la impresora en lugar de en la consola del sistema.

Para ello basta con que observe el pequeño programa adjunto, y si no sabe como funciona el IOBYTE del CP/M, consulte algún buen manual sobre este sistema operativo.

No ofrezco más explicaciones para no convertir este truco en algo más que un artículo de «regular» tamaño. Sólo añadiré que la «astucia» ha sido probada en un Osborne 1 y que, por supuesto, ha funcionado.

Luis de Cáceres Muñoz

```
100
110 ' Ejemplo cambio IOBYTE
120 '
130 PRINT: PRINT: PRINT
140 PRINT "Seleccion del dispositivo de SALIDA"
150 PRINT
160 PRINT "        -C- para la Consola"
170 PRINT "        -I- para la Impresora"
180 PRINT
190 PRINT "-----------------------------------"
200 PRINT
210 INPUT "Teclee opcion : "; OPCION$
220 CONSOLA = PEEK (3)
230 IF OPCION$ = "C" OR OPCION$ = "c" THEN 280
240 IF OPCION$ = "I" OR OPCION$ = "i" THEN 260
250 GOTO 210
260 IMPRESORA = CONSOLA AND 252 OR 2
270 POKE 3,IMPRESORA
280 PRINT
290 PRINT "  Tabla de multiplicar por 2    "
300 PRINT "--------------------------------"
310 PRINT
320 FOR N = 1 TO 10
330     PRINT "  2  por "; N; " = "; 2 * N
340 NEXT N
350 POKE 3,CONSOLA
360 GOTO 130
```

Gane 2K octetos en los disquetes doble densidad de su Osborne 1

Vamos a hacer lo mismo con los de doble densidad.

Como las diferencias son muy pequeñas, lo primero que tiene que hacer es leer el truco anterior. Una vez lo haya entendido tome buena nota de cuáles son esas diferencias.

1.– Siempre que se haga referencia al disquete de utilidades CP/M sustitúyalo por el disquete CP/M System & Utility (En la versión doble densidad no existe el disquete de utilidades CP/M ya que todos sus programas han sido incluidos en el disquete CP/M System & Utility).

2.– En el paso 2 lo que el sistema hará ahora es cargar el programa HELP. Salga de él pulsando la tecla Escape y cuando le aparezca en pantalla A>proceda a realizar todo lo indicado en el listado adjunto (Recuerde, sólo tiene que teclear la parte subrayada). Asegúrese una vez más de que está trabajando con copias y NO con los disquetes originales y manos a la obra.

No hay más diferencias. Si compara ambos listados observará que hemos estado trabajando con diferentes pisiciones de memoria (201D en lugar de 200A), tenga esto en cuenta al leer la última parte del artículo dedicado a la simple densidad si quiere realizar por su cuenta otro tipo de cambios. También notará que hemos construído un sistema CP/M de 59 K octetos en vez de las 60 anteriores, esto es debido, entre otras cosas, al nuevo tamaño de los sectores de sus disquetes doble densidad del Osborne 1 (1.024 caracteres contra los 256 anteriores), y trae como consecuencia inmediata la pérdida de 1K octetos de memoria libre de usuario (TPA).

Luis de Cáceres Muñoz.

```
A>movcpm 59 * <RET>  (<RET> → TECLA RETURN)

CONSTRUCTING 59k CP/M vers 2.2
READY FOR "SYSGEN" OR
"SAVE 39 CPM59.COM"
A>save 39 b:cpm59.com <RET>
A>ddt b:cpm59.com <RET>
DDT VERS 2.2
NEXT  PC
2800 0100
-d2018,2027 <RET>
2018 00 00 00 00 01 07 41 55 ......AU
2020 54 4F 53 54 20 30 31 32 TOST 012
-s201d <RET>
201D 07 08 <RET>
201E 41 4d <RET>
201F 55 42 <RET>
2020 54 41 <RET>
2021 4F 53 <RET>
2022 53 49 <RET>
2023 54 43 <RET>
2024 20 00 <RET>
2025 30 . <RET>
-d2018,2027 <RET>
2018 00 00 00 00 01 08 4D 42 ......MB
2020 41 53 49 43 00 30 31 32 ASIC.012
-g0 <RET>

A>sysgen <RET>

        Operating System Generation Program
            OSBORNE COMPUTER SYSTEM ONE
              Rev 2.1     (c) 1982 OCC
SOURCE drive (A or B) <RET>
-------------------------------------------------
DESTINATION (A,B  or  RETURN to exit) B
Put DESTINATION diskette in B, then press RETURN <RET>
       System copied successfully.

-------------------------------------------------
DESTINATION (A,B  or  RETURN to exit) <RET>
A>
```

AMSTRAD CPC-464 CON MONITOR Y MAGNETOFONO INCORPORADO

AMSTRAD: EL ¡increíble ORDE

Estamos viviendo la era del ordenador personal. Más de un millón de personas comprarán equipos informáticos en los próximos años: estudiantes, empresarios, educadores, profesionales, comerciantes, los utilizará como herramienta imprescindible en sus actividades. Usuarios cada vez mejor informados, más selectivos y exigentes para los que AMSTRAD, gigante británico de la industria electrónica ha fabricado el ordenador idóneo.

Si en la primavera de 1984 AMSTRAD conmocionó al mundo informático con un modelo CPC 464, la aparición ahora de CPC 664 –en el que el magnetófono ha sido sustituído por una unidad de disco de 3" (180 K) incorporada– vuelve a despertar el entusiasmo de especialistas y público. El éxito arrollador de ambos modelos encuentra su explicación en la filosofía de diseño de AMSTRAD. Una filosofía que ofrece:

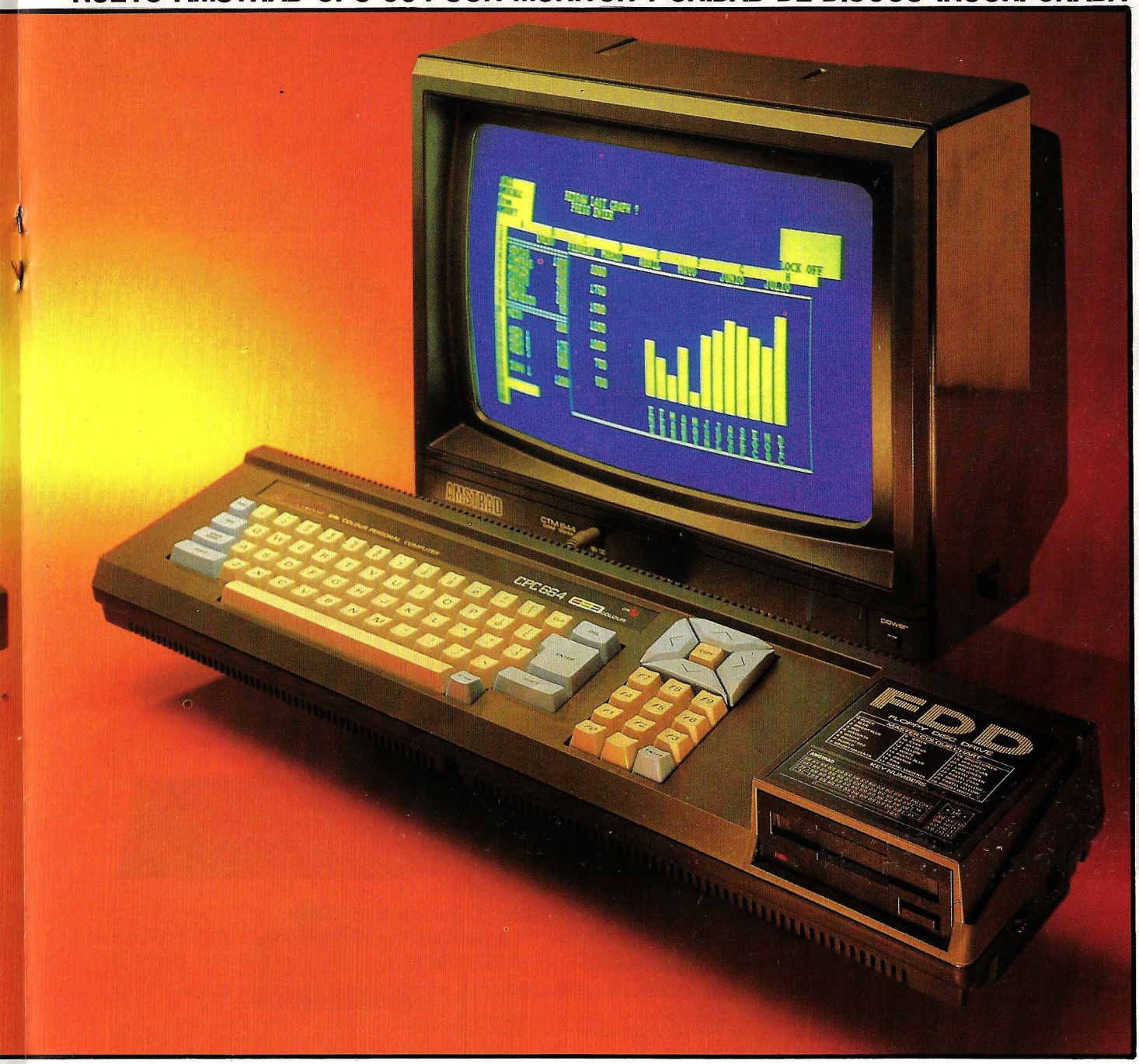
Un sistema completo que incluye la unidad central, el monitor y el magentófono o la unidad de disco. Un equipo compacto, listo para funcionar sin cableados engorrosos ni necesidad de adquirir más periféricos. Sólo requiere desembalarlo y enchufar un cable –un sólo cable– a la red. Con un paquete de **programas de obsequio** y, además, el Sistema Operativo CP/M y el lenguaje LOGO incluídos en el suministro del CPC 664.

Unas prestaciones del más alto nivel, con 64 K de memoria RAM, 32 K de memoria ROM, con resolución de 640 x 200 puntos, 27 colores, 20, 40, u 80 columnas de texto en pantalla, 8 "ventanas" de trabajo, teclado profesional con 32 teclas programables, sonido estéreo con 3 canales y 8 octavas por canal. Y un BASIC super-ampliado y dotado incluso de comando de control del microprocesador (Every, After...).

Una tecnología contrastada y fiable basada en el popular micropocesador Z80A y en una electrónica depurada y con un riguroso control de calidad.

Una extensa biblioteca de programas que se incrementa literalmente día a día y que ya dispone de centenares de títulos para todos los gustos y necesidades: gestión profesional (Contabilidad, Control de Stocks, Bases de Datos, Hojas de Cálculo, Procesadores de Texto,...), educación, lenguajes, y ayuda a la programación

NUEVO AMSTRAD CPC-664 CON MONITOR Y UNIDAD DE DISCOS INCORPORADA



ENADOR PERSONAL

(Ensamblador, Desensamblador, Pascal, Forth, Logo, Diseñador de Gráficos, Diseñador de Sprites...), de toma de decisiones (Proyect Planner, Decisión Maker,...), juegos de habilidad (La Pulga, Manic Miner, Decathlon, Android,...), juegos de inteligencia (Ajedrez, Backgamon,...), juegos de estrategia (Batalla de Midway, II Guerra Mundial, ...), juegos de aventuras (Hobbit, Sherlock Homes,...), juegos de simulación (simulador de Vuelo, Tenis, Billar, Mundial de Fútbol,...).

Una asistencia técnica rápida y eficaz que **AMSTRAD ESPAÑA** garantiza **exclusivamente** a los equipos adquiridos a través de su Red Oficial de Distribuidores y acompañados de la **Tarjeta de Garantía de AMSTRAD ESPAÑA.**

Unos precios increíbles que no admiten comparación con los de cualquier otro ordenador personal de sus características.

* Ordenador CPC 464, con magnetófono incorporado. Manual del Usuario y obsequio del Libro "Guía de Referencia del programador" y de 8 Programas:

- Con Monitor de fósforo verde (12")... **64.900 pts.**
- Con Monitor color (14")..... **93.900 pts.**

* Ordenador CPC 664, con Unidad de Disco incorporada, Manual del Usuario, incluyendo Sistema Operativo CP/M, Lenguaje Logo y **obsequio de cinco programas (Base de Datos, Proceso de Textos, Diseñador de Gráficos, Random Files, Puzzle y Animal, Vegetal, Mineral.**

- Con Monitor de fósforo verde (12")..**109.500 pts.**
- Con Monitor color (14")..... **134.500 pts.**

AMSTRAD (tm)

ESPAÑA

Avda. del Mediterráneo, 9
Tels. 433 45 48 - 433 48 76
28007 MADRID

Delegación Cataluña:
Tarragona, 100 - Tel. 325 10 58
08015 BARCELONA

NOTA: Es muy importante verificar la garantía del aparato ya que sólo **AMSTRAD ESPAÑA** puede garantizarle la ordenada reparación y sobre todo materiales de repuesto oficiales (Monitor, ordenador, cassette o unidad de discos).

EL Einstein DE MICROS

Y POR SOLAMENTE 140.000 Ptas. es puro genio

(INCLUYE, LENGUAJE LOGO, 1 DRIVE Y 6 MESES DE GARANTIA)

Diseñado y producido en Inglaterra por TATUNG (UK), Ltd.

... GENIO EN CASA, EN EL TRABAJO, EN LA ESCUELA...

MEMORIA INCORPORADA DE 80 K
64 RAM + 16 K independiente para pantalla.

UNIDAD DE DISCO INCORPORADO
500 KByte capacidad de disco.

1 Floppy disco drive de 3" incorporado.
Ampliable con un segundo disco drive interno.

16 GRAFICOS DE COLORES INCORPORADOS.
32 sprites - 16 colores.
40 columnas x 24 filas (ampliables hasta 80 c.).

PORTS DE EXPANSION INCORPORADOS.
Un port RS232-C.
Un port de impresora "Centrónic".
Port de usuario de 8 bit.
4 canales analógicos/digitales.
Conector Tatung "pipe".

CON FLEXIBILIDAD INCORPORADA.
Potente BASIC Crystal.
Capacidad de operar programas en CP/M*.
Lenguajes: FORTH, PASCAL, BASIC, COBOL, FORTRAN, LOGO, ASSEMBLY y otros.
Y con teclado tipo máquina QWERTY.

SONIDO VERSATIL INCORPORADO.
3 canales de música con control incorporado.
Altavoz incorporado con regulador de volumen.

EINSTEIN reúne todas estas ventajas y mucho más.
Satisface tanto al principiante en la electrónica como al operador experto, bien sea en casa o en la oficina.

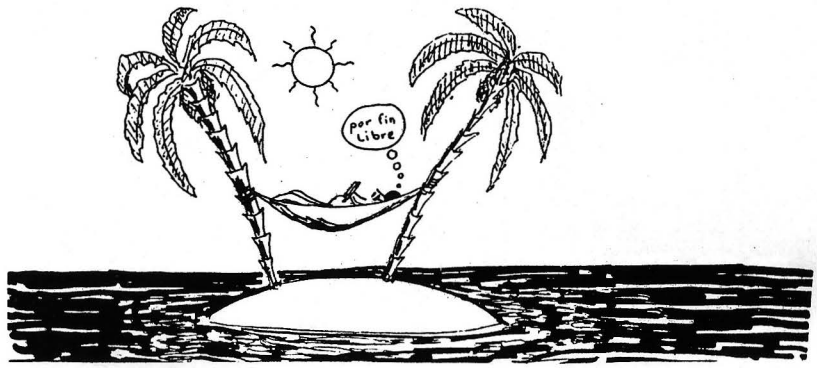
¡Y A QUE PRECIOS!

DISTRIBUIDOR EXCLUSIVO PARA ESPAÑA:
ALPHA MUNDIAL GROUP, Gran Vía Carlos III, 86, 6.ª
Tel. 330 96 51 (télex 52220). 08028 BARCELONA

SE BUSCAN DISTRIBUIDORES

CP/M es una marca registrada de DIGITAL RESEARCH INC.

Trucos de la PC1500



La "ventaja" de tener un ordenador de bolsillo es que no hay manera de quitarselo de encima. En fin, si vos tampoco podeis, al menos practicar con astucia, y si vas a bañarte no olvides sacarlo del bolsillo, que los chips son reacios al agua.

Inversión de Pantalla

En muchas ocasiones sería interesante realizar las salidas de nuestros programas en "inverso", es decir, con fondo negro y caracteres en blanco.

Esto puede realizarse en BASIC, pero es un procedimiento lento y ocupa mucha memoria. Por ello hemos desarrollado esta rutina en lenguaje máquina, que puede ser introducida en cualquier zona de la memoria (recomendamos la zona RESERVE), y al ser llamada invierte la presentación de la pantalla en cuestión de décimas de segundo. El procedimiento seguido es el siguiente: se va leyendo las posiciones de memoria correspondiente a la pantalla (direcciones 7000 a 704E y 100 a 714E), y se realiza la operación XOR entre ellas y FF, que pone a 1 los ceros y a cero los unos (ver tabla de la verdad correspondiente a esta operación). Una vez hecho esto los octetos son devueltos a su lugar. El contador utilizado es BC, para apuntar a las direcciones correspondientes.

Listado de la rutina

```
48 70   LD B,&70
4A 00   LD C,&00
05      LD A,(BC)
BD FF   XOR A,&FF
0E      LD (BC),A
44      INC BC
4E 4E   CP C,&4E
99 09   JR NZ,-&09
48 71   LD B,&71
4A 00   LD C,&00
05      LD A,(BC)
BD FF   XOR A,&FF
0E      LD (BC),A
44      INC BC
4E 4E   CP C,&4E
99 09   JR NZ,-&09
9A      RET
```

Tabla de la verdad de la operación Xor

XOR	1	0
1	0	1
0	1	0

Víctor Manuel Díaz
Iñaki cabrera

El PC-1500 como Calculadora

Uno de los mayores hándicaps que tiene el PC-1500 a la hora de usarlo como calculadora es que el resultado de una operación sólo puede ser usado para otra operación "inmediata" (+, −, /, *, ∧, √), pero para hacerlo argumento de una función (SIN, COS, TAN, etc. . .) o para memorizarlo en una variable es necesario recorrer la operación completa con el cursor e insertar al principio la función o la variable seguida de un =, esto puede resultar muy pesado porque una operación puede tener hasta 78 caracteres de longitud. Existe sin embargo en el excelente intérprete Basic una instrucción que nos puede ayudar: AREAD.

Para aprovecharla se escriben líneas de programa del tipo:

```
60000:"A":AREAD A:
      A=SIN A:
      PRINT A
60001:"S":AREAD S:
      S=COS S:
      PRINT S
60002:"D":AREAD D:
      D=TAN D:
      PRINT D
60003:"F":AREAD F:
      F=ASN F:
      PRINT F
60004:"G":AREAD G:
      G=ACS G:
      PRINT G
60005:"H":AREAD H:
      H=ATN H:
      PRINT H
60006:"J":AREAD J:
      J=LN J:PRINT
      J
60007:"K":AREAD K:
      K=LOG K:
      PRINT K
60008:"L":AREAD L:
      L=1/L:PRINT
      L
60009:"Z":AREAD Z:
      PRINT Z
60010:"X":AREAD X
      PRINT X
60011:"C":AREAD C:
      PRINT C
60012:"V":AREAD V:
      PRINT V
60013:"B":AREAD B:
      B=EXP B:
      PRINT B
60014:"N":AREAD N:
      N=10^N:PRINT
      N
60015:"M":AREAD M:
      M=√M:PRINT M
60016:" ":AREAD Y:
      Y=-Y:PRINT Y
```

60000:"A":AREAD A:A=SIN A:PRINT A (El número puede ser cualquiera).

Con esta línea en memoria y teniendo el resultado en pantalla no hay más que hacer DEF A y aparece el seno del número que había en pantalla, quedando además el valor del seno memorizado en la variable A, para ser utilizado en ulteriores cálculos. De esta forma se pueden asignar todas las funciones en el teclado, y se tiene un funcionamiento parecido a las calculadoras científicas más avanzadas.

Utilizando una de las carátulas del teclado se puede ver en todo momento que función hay en cada tecla. Hay que destacar que esto es diferente a asignar las funciones en modo RESERVE, porque las funciones así asignadas se usan DENTRO de las operaciones, y las que proponemos ahora, actúan sobre el número visualizado en pantalla. Publicamos el listado de un programa que asigna todo el teclado, incluyendo 4 teclas que sirven para memorizar el número, sin efectuar ninguna función. PRECAUCION: el programa está escrito para WAIT infinito, si antes de él hay algún programa que utilice otro WAIT, es necesario terminarlo con END, porque si no ocurrirían cosas muy curiosas. Otra forma de evitarlo es poner en cada línea de nuestro programa la instrucción WAIT.

Victor Manuel Díaz.
Iñaki Cabrera.

...y ahora con **512 K RAM** *por el mismo precio.*

PC-401

Compatible, más completo con el mejor precio.

COMPATIBLE CON IBM-PC Y XT.

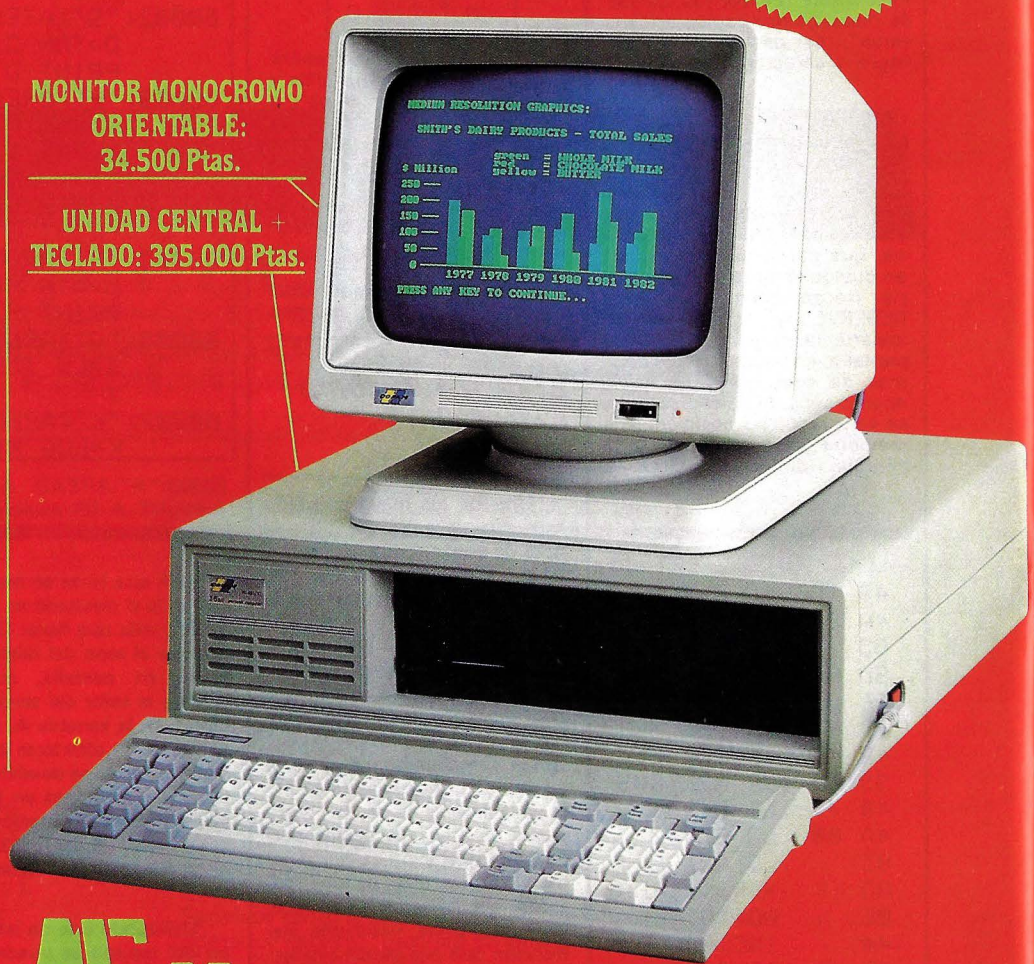
CARACTERISTICAS:

- CPU 8088 (4,77 MHz).
- 8 slots de expansión.
- Multifunción card con:
  RS232 asíncrona para comunicaciones.
  Salida paralelo impresora.
  Opcionalmente otra RS232.
  Reloj/calendario con batería recargable.
  128 K Bytes de memoria RAM, expandible a 512 K RAM
- Tarjeta de color de alta resolución:
  Modo de salida monocroma o de color.
  En modo gráfico hasta 640 × 400 puntos en color y 640 × 704 en monocromo.
  Salida paralelo impresora.
- 2 Unidades de disco de 360 K Bytes por unidad y controlador.
- Teclado tipo IBM, capacitivo.

Accesorios:
Disco duro 10 Mb.
Modem telefónico.
Red local hasta 127 terminales.

MONITOR MONOCROMO ORIENTABLE: 34.500 Ptas.

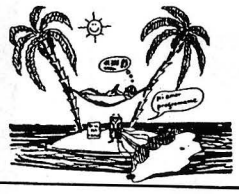
UNIDAD CENTRAL + TECLADO: 395.000 Ptas.



BASE-64A
UNIDAD CENTRAL 64 K RAM, 32 K ROM
P.V.P. 118.500 Ptas.
UNIDAD DISCO tracción directa
P.V.P. 38.000 Ptas.
MONITOR FOSFORO VERDE
P.V.P. 34.500 Ptas.

MICOMPSA

IMPORTADOR PARA ESPAÑA:
General Peron, 32 28020 MADRID. Tel. 455 10 72



Cuando la PC trata los textos

El tratamiento de textos estaba hasta hoy reservado a los ordenadores de mesa y a los ordenadores de bolsillo muy caros como el HP 75 C o el HHC de Matsushita. Pero ahora también existe para el PC 1500 sin extensión de memoria.

Evidentemente, es un programa en lenguaje máquina para la recogida de datos. Para la impresión se ha usado el BASIC por las dificultades de comunicación entre el lenguaje máquina y el BASIC (cómo imprimir en LM?). De todas formas, con la lentitud extrema de la impresora CE-150, no se ganaría demasiado en tiempo de ejecución.

Digamos ahora algo sobre el programa en lenguaje máquina. Antes de cargarlo hay que hacer NEW 4600. Esto reserva los 511 octetos utilizados por el programa en BASIC: y cuando usted reserve los 44 octetos utilizados por la variable T$(0) no quedará ningún octeto libre.

El texto está almacenado como un programa en lenguaje máquina para protegerlo de maniobras suicidas como un RUN que borraría las tablas dimensionadas con consecuencias que se imaginan.

Sin extensión de memoria el texto puede tener una longitud 865 caracteres. Esto es un poco justo pero suficiente para pequeños textos.

Su interés reside en la compilación de los datos gracias a las cualidades de impresión de la CE-150.

Si se dispone de una memoria mayor, se puede prolongar el texto jugando con los contenidos de las direcciones 40C9 y 40CA ya que ellas contienen la dirección del final del texto.

Es necesario modificar además las direcciones 427A y 427C que contienen respectivamente el octeto fuerte y débil de la dirección de la cadena de caracteres T$(0). Si es la primera variable dimensionada la dirección es STATUS 3+7.

En el programa de recogida de datos utilizo una astucia encontrada por un amigo: CALL E267 llama una rutina en ROM que da el código ASCII de la última tecla apretada, teniendo en cuenta los atributos como SHIFT, DEF y sus representaciones en pantalla, con antirebote incorporado.

Otra astucia: para visualizar una zona ASCII se introduce el comienzo en HL y la longitud en A, y luego se hace SBR 92; ¿fácil no?

Para lanzar el programa principal se hace CALL 426B.

Todos los caracteres accesibles desde el teclado pueden ser utilizados pero algunos tienen funciones particulares.

—ENTER producirá en la impresión un retorno de carro y será visualizado en pantalla con CHR$ (126).

—F3 y F4 visualizan el corchete abierto y cerrado que indican respectivamente el inicio y el fin de la parte de texto a imprimir.

—Las flechas no tienen auto repeat pero otras funciones lo reemplazan suficientemente.

—Las flechas verticales saltan de frase en frase con la ayuda de los signos de puntuación.

—F1 y F2 buscan los espacios entre las palabras.

Para borrar el carácter bajo el cursor no se utiliza la tecla habitual sino CL.

Para insertar un caracter se pulsa MODE y la tecla habitual inserta 16 huecos a la vez. El cursor está situado en el centro de la pantalla, y el texto se mueve bajo él, de esta forma es visible aún en avance rápido.

Está materializado por un bloque en vídeo inverso de 7 por 7 puntos. Al final y el principio del texto están señalados por

```
&40C5:45:C2:42:90
&40C9:55:66:04:AE
&40CD:40:C6:84:AE
&40D1:40:C5:9A:A5
&40D5:40:C5:08:A5
&40D9:40:C6:0A:9A
&40DD:84:A7:40:C7
&40E1:81:1C:89:06
&40E5:04:A7:40:C8
&40E9:81:14:84:A7
&40ED:40:C9:81:0C
&40F1:8B:02:8E:0A
&40F5:04:A7:40:CA
&40F9:8B:02:83:02
&40FD:FB:9A:F9:9A
&4101:BE:40:D4:FD
&4105:5A:44:BE:40
&4109:DD:81:03:F5
&410D:9E:08:B5:20
&4111:1E:9A:A5:40
&4115:C9:18:A5:40
&4119:CA:1A:BE:40
&411D:D4:8E:02:55
&4121:53:56:94:86
&4125:81:08:8B:02
&4129:9E:0B:14:06
&412D:93:0F:9A:9A
&4131:48:71:4A:40
&4135:6A:0D:B5:0F
&4139:0D:41:88:06
&413D:9A:BE:40:D4
&4141:6A:0B:46:88
&4145:03:FD:6A:B5
&4149:1A:CD:92:BE
&414D:41:31:9A:A5
&4151:40:C7:08:A5
&4155:40:C8:0A:BE
&4159:40:CB:B5:7F
&415D:6A:0B:46:43
&4161:88:03:BE:40
&4165:D4:B5:20:41
&4169:BE:40:DD:93
&416D:08:6A:0B:B5
&4171:7F:41:88:03
&4175:9A:05:B7:7B
&4179:8B:0E:B7:7D
&417D:8B:0A:B7:20
&4181:8B:06:B7:7E
&4185:8B:02:B7:7F
&4189:9A:BE:40:D4
&418D:46:BE:41:76
&4191:99:06:9A:BE
&4195:40:D4:44:BE
&4199:41:76:99:06
&419D:9A:05:B7:2E
&41A1:8B:0A:B7:21
&41A5:8B:06:B7:3F
&41A9:8B:02:B7:7F
&41AD:9A:BE:40:D4
&41B1:46:BE:41:9E
&41B5:99:06:9A:BE
&41B9:40:D4:44:BE
&41BD:41:9E:99:06
&41C1:9A:B7:08:89
&41C5:0D:BE:40:D4
&41C9:46:BE:40:DD
&41CD:81:03:BE:40
&41D1:CB:9A:B7:0C
&41D5:89:06:BE:40
&41D9:D4:44:9E:13
&41DD:B7:1F:89:07
&41E1:BE:41:13:B5
&41E5:27:0E:9A:B7
&41E9:18:89:04:BE
&41ED:41:01:9A:B7
&41F1:0B:89:05:BE
&41F5:41:AE:9E:2F
&41F9:B7:0A:89:05
&41FD:BE:41:B8:9E
&4201:38:B7:1C:89
&4205:0B:6A:0F:BE
&4209:41:13:B5:27
&420D:0E:88:08:9A
&4211:B7:11:89:05
&4215:BE:41:8A:9E
&4219:50:B7:12:89
&421D:05:BE:41:94
&4221:9E:59:B7:80
&4225:89:04:BE:41
&4229:50:9A:B7:0D
&422D:89:04:B5:7E
&4231:8E:1B:B7:13
&4235:89:04:B5:7B
&4239:8E:13:B7:14
&423D:89:04:B5:7D
&4241:8E:0B:B7:20
&4245:81:05:B7:7B
&4249:83:02:8E:01
&424D:9A:FD:C8:BE
&4251:40:D4:FD:8A
&4255:0E:44:BE:40
&4259:DD:81:03:BE
&425D:40:CB:9A:BE
&4261:E2:67:B7:0E
&4265:89:01:9A:BE
&4269:41:C2:E9:78
```

```
10: INPUT N$:CSAVE
    MN$;&4290, &45F
    0:END
20: INPUT N$:CLOAD
    MN$:END
30: B=0:CSIZE 1:
    DIM T$(0)*37:
    FOR T=&4290TO
    &45F0:IF PEEK
    T=&7BLET T=T+1
    :GOTO 50
40: NEXT T:END
50: POKE &40C5, T/2
    56, T-INT (T/25
    6)*256:CALL &4
    274:IF LEFT$ (
    T$(0), 1)=" "
    LET T=T+1:GOTO
    50
60: F=0:S=0:IF B=&
    7ELET T$(0)="
    "+T$(0):F=1
70: FOR A=1TO 36:B
    =ASC MID$ (T$(
    0), A, 1)
80: IF B=&7DOR B=&
    7FLPRINT LEFT$
    (T$(0), A-1):
    END
90: IF B=&7ELPRINT
    LEFT$ (T$(0), A
    -1):T=T+A-F:
    GOTO 50
100: IF B=32LET S=A
110: NEXT A:IF
    RIGHT$ (T$(0),
    1)=" "LPRINT
    LEFT$ (T$(0), 3
    6):T=T+36:GOTO
    50
120: IF S<2LPRINT
    LEFT$ (T$(0), 3
    6):T=T+36:GOTO
    50
130: LPRINT LEFT$ (
    T$(0), S-1):T=T
    +S-1:GOTO 50
```

```
&426D:75:00:BE:41
&4271:3E:9E:14:BE
&4275:40:D4:6A:24
&4279:58:47:5A:DB
&427D:F5:88:03:9A
```

3 NOVEDADES

EDICIONES ELISA
Balmes, 151 - Tfno. (93) 217.98.54 - 08008 Barcelona

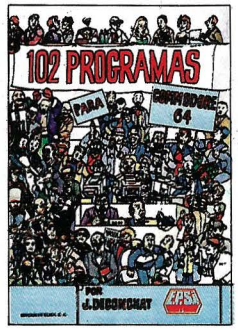
OBRAS PUBLICADAS

Lien: **Diccionario del Basic**
Precio: 3.500 pts.

Breaud-Pouliquen: **Claves para el Apple II, Apple II plus y Apple IIe**
Precio: 1.500 pts.

Deconchat: **102 programas para ZX81 y Spectrum**
Precio 1.950 pts.

David: **El descubrimiento del Commodore 64**
Precio: 1.500 pts.

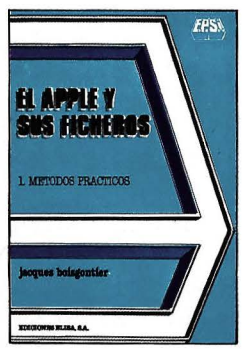


240 págs., 17×24 cm.
rústica.
P.V.P.: 1.900 ptas.
ISBN: 84-7622-006-5.

JACQUES DECONCHAT

102 programas para Commodore 64

El objetivo de este libro es aprender distrayéndose. A lo largo de estos 102 programas de juegos, les guiará en la exploración del Basic Commodore 64. Los programas están clasificados por niveles, cada uno de ellos recurre a nuevos conocimientos y a un mayor dominio del Basic. Cada nivel empieza por una presentación concisa de las nuevas instrucciones utilizadas. Se describen todos los juegos y los programas están abundantemente comentados; se facilita un ejemplo de ejecución para cada versión.

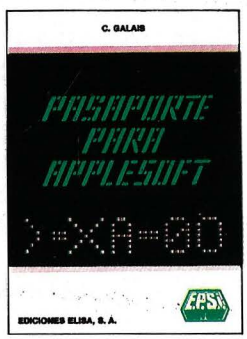


174 págs., 14,5×21 cm.
rústica.
P.V.P.: 1.500 ptas.
ISBN: 84-7622-005-7.

JACQUES BOISGONTIER

El apple y sus ficheros

La obra comienza por una presentación concisa e ilustrada de los comandos del Sistema de Explotación de Disco e Instrucciones del Basic Applesoft. Se describen a continuación las instrucciones de los ficheros secuenciales y de acceso directo, y su empleo se explica con ayuda de programas clásicos de creación, modificación y clasificación de ficheros suficientemente comentados. Métodos prácticos, a menudo mal conocidos, muestran cómo utilizar mejor ficheros de acceso directo. Una veintena de programas ilustran la utilización de estas técnicas.



160 págs., 11,5×16,5 cm.
rústica.
P.V.P.: 1.000 ptas.
ISBN: 84-7622-004-9.

CLAUDY GALAIS

Pasaporte para applesoft

Está dirigido tanto al debutante en informática como al programador experimentado. Es el manual que todo usuario del Basic Applesoft debe poseer. Todas las instrucciones, funciones y comandos están enumerados página por página en orden alfabético. La búsqueda de una definición es, pues, cómoda y rápida.

BOLETÍN DE PEDIDO

Les agradeceré me envíen, contra reembolso, las obras que detallo a continuación:

☐ .....
☐ .....
☐ .....
☐ .....

Don ..... Calle ..... Población .....
Código postal ..... Provincia .....
☐ Talón bancario n.º .....
☐ Contra reembolso ...., a ..... de ..... de 19 .....
(Firma)

Nota: Puede solicitar su pedido a su librero habitual o su envío, contra reembolso (más 100 ptas. por gastos de envío a **EDICIONES ELISA,** Balmes, 151 - 08008 Barcelona.

(Precios al 1 de enero de 1985)

dos bloques negros (CHR$ 127), lo que simplifica también el programa de impresión.

Para terminar se pulsa BREAK o DEF SPACE; esto último nos libera definitivamente del texto. La rutina E267 puede provocar que la máquina quede funcionando hasta 9 minutos antes de pararse (?).

Además del programa maquina se publican tres programas BASIC:

—RUN 10 lanza el programa de salvaguarda donde un «?» nos pide el nombre del fichero.

—RUN 20 realiza la operación inversa.

—RUN 30 lanza la impresión.

Sería conveniente asignar a las teclas de función el CALL 426B y estos comandos.

El programa adolece de ciertas carencias como formateo de párrafos, tamaño de caracteres, subrayado y caracteres programables. Todo esto será posible si se da un ensamblador simbólico en memoria ROM y una extensión de memoria.

Erik Frankenfeld

Neumónicos "oficiales"

Ha aparecido un manual técnico de la PC-1500, en el que se reseñan todas las características (físicas y lógicas) de esta máquina.

En el capítulo de descripción del microprocesador (LH-5801) se exponen las instrucciones de LM, y (por supuesto) los nemónicos no se parecen en nada a nuestro «simil Z-80».

En primer lugar, los registros han «cambiado» su nombre y se llaman:

A = Acumulador
X = Registro de propósito general de 16 bits (antiguo BC), compuesto de XH y XL.
Y = Registro de propósito general de 16 bits (antiguo DE), compuesto de YH e YL.
U = Registro de propósito general (aunque se usa como contador en los bucles) (antiguo HL), UH y UL.
P = Contador de programa.
S = Puntero de la pila.
T = Registro de estado (antiguo F).

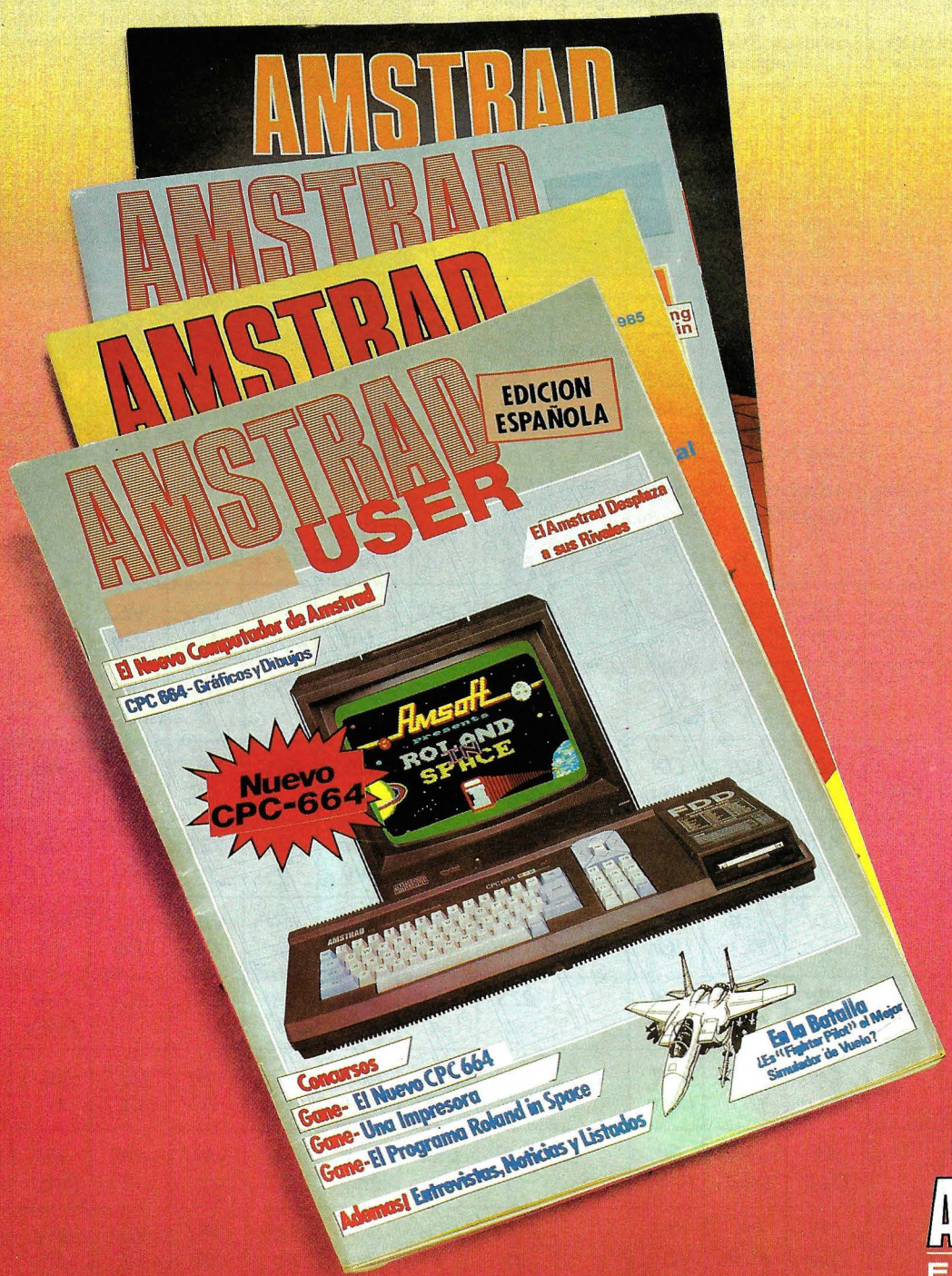
Además de estos nuevos nombres también tenemos nuevos nemónicos, y, al ser los «oficiales», creemos que deberían ser usados a partir de ahora, aunque por claridad mantendremos durante cierto tiempo la antigua notación.

En las tablas adjuntas publicaremos todos los nemónicos del LH-58091. Comparando las dos tablas podreis ver los cambios habidos, por ejemplo, la terminación «I» indica inmediato, mientras que el incremento se indica por «IN» (SIN, LIN...), «LD» ya no es ambivalente, sino que significa siempre cargar el acumulador, excepto en LDX, que significa

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	SBC XL	SBC YL	SBC UL		INC XL	INC YL	INC UL		SBC XH	SBC YH	SBC UH		VEJ CO	VEJ DO	VEJ EO	VEJ FO
1	SBC (X)	SBC (Y)	SBC (U)		SIN X	SIN Y	SIN U		BCR +n	BCR -n	SBC (nn)	SBI N	VCR N	ROR	SPU	AEX
2	ADC XL	ADC YL	ADC UL		DEC XL	DEC YL	DEC UL		ADC XH	ADC YH	ADC UH		VEJ C2	VEJ D2	VEJ E2	VEJ F2
3	ADC (X)	ADC (Y)	ADC (U)		SDE X	SDE Y	SDE U		BCS +n	BCS -n	ADC (nn)	ADI n	VCS n	DRR (X)	RPU	
4	LDA XL	LDA YL	LDA UL		INC X	INC Y	INC U		LDA XH	LDA YH	LDA UH		VEJ C4	VEJ D4	VEJ E4	VEJ F4
5	LDA (X)	LDA (Y)	LDA (U)		LIN X	LIN Y	LIN U		BHR +n	BHR -n	LDA (nn)	LDI n	VHR n	SHR		TIN
6	CPA XL	CPA YL	CPA UL		DEC X	DEC Y	DEC U		CPA XH	CPA YH	CPA UH		VEJ C6	VEJ D6	VEJ E6	VEJ F6
7	CPA (X)	CPA (Y)	CPA (U)		LDE X	LDE Y	LDE U		BHS +n	BHS -n	CPA (nn)	CPI n	VHS n	DRL (X)		CIN
8	STA XH	STA YH	STA UH	NOP	LDI XH,n	LDI YH,n	LDI UH,n		LOP UL,n		SPV	RPV	VEJ C8	VEJ D8	VEJ E8	
9	AND (X)	AND (Y)	AND (U)		ANI (X),n	ANI (Y),n	ANI (U),n		BZR +n	BZR -n	AND (nn)	ANI n	VZR n	SHL	ANI (nn),p	REC
A	STA XL	STA YL	STA UL		LDI XL,n	LDI YL,n	LDI UL,n		RTI	RTN	LDI S,nn	JMP nn	VEJ CA	VEJ DA	VEJ EA	
B	ORA (X)	ORA (Y)	ORA (U)		ORI (X),n	ORI (Y),n	ORI (U),n		BZS +n	BZS -n	ORA (nn)	ORI n	VZS n	ROL	ORI (nn),p	SEC
C	DCS (X)	DCS (Y)	DCS (U)		CPI XH,n	CPI YH,n	CPI UH,n		DCA (X)	DCA (Y)	DCA (U)		VEJ CC	VEJ DC	VEJ EC	
D	EOR (X)	EOR (Y)	EOR (U)		BII (X),n	BII (Y),n	BII (U),n		BVR +n	BVR -n	EOR (nn)	EAI n	VMJ n	INC A	BII (nn),p	2ª tabla
E	STA (X)	STA (Y)	STA (U)		CPI XL,n	CPI XL,n	CPI UL,n		BCH +n	BCH -n	STA (nn)	SJP nn	VEJ CE	VEJ DE	VEJ EE	
F	BIT (X)	BIT (Y)	BIT (U)		ADI (X),n	ADI (Y),n	ADI (U),n		BUS +n	BUS -n	BIT (nn)	BII A.N	VVS M	DEC A	ADI (nn),p	

Una Gran Noticia para los Usuarios de AMSTRAD

A partir del próximo septiembre estará en vuestra tienda de informática, en los quioscos de prensa o —si preferís suscribiros— en vuestro domicilio, la revista AMSTRAD USER. Una publicación mensual, repleta de información, con abundantes listados, trucos de programación, crítica de software y periféricos, noticias y novedades, concursos, etcétera. Para estar al día. Para sacarle aún más partido a tu AMSTRAD.



Para más información:

AMSTRAD
ESPAÑA

Dpto. Publicaciones Avd. del Mediterráneo, 9 28007 Madrid

cargar X. En cambio el hecho de sacar datos del acumulador o de otro registro como el X se identifica con STore o Store, con los sufijos adecuados (IN = incremento).

La instrucción DJC-n pasa a ser LOP-n, lo que indica su vocación de controladora de bucles, el registro contador es UL. Los saltos o subrutinas se llaman VEJ o VMJ o V... por Vector Subroutine Jump, en el caso de saltos referidos a la página FF (antiguos SBR...), y SJP (Subroutine JuMp) para el CALL... Los saltos relativos son BCH (incondicional) y BZ–, BV–, BC–, BH–, donde la última letra indica si es salto si «1» (S) o «0» (R) es decir Branch if Carry Set, etc.

En cuanto a la segunda tabla, además de contener las intrucciones con la segunda página de memoria, contiene un sinfín de instrucciones de control, que están más relacionados con el hardware de la máquina que con el software, en su mayoría son poco utilizables si no se va a hacer alguna «chapuza» seria con la máquina. A reseñar que TTA el LDA, F y ATT es LDF, A.RDP y SDP permiten apagar y encender la pantalla, HLT detiene el incremento, SIE habilita las interrupciones mascarables, RIE las inhabilita, CDV borra un divisor interno del reloj de CPU, AM0 y AM1 permiten cargar un registro especial de la CPU (Timer) de 9 bits con el contenido del acumulador, poniendo a 0 o a 1 el bit de mayor peso. Este registro timer habilita una interrupción especial (Timer interrupt) cuando llega a cero, si IE está a 1, pasando control a la dirección E22C, o a la que contengan las posiciones FFFA, FFFB, que en nuestro caso está en ROM, pero que podrían ser alterados si sustituyéramos esta ROM por una exterior. OFF «apaga» la CPU, y al encender nuevamente la máquina aparece NEWO? CHECK, indicación de que se trata de un arranque en frío.

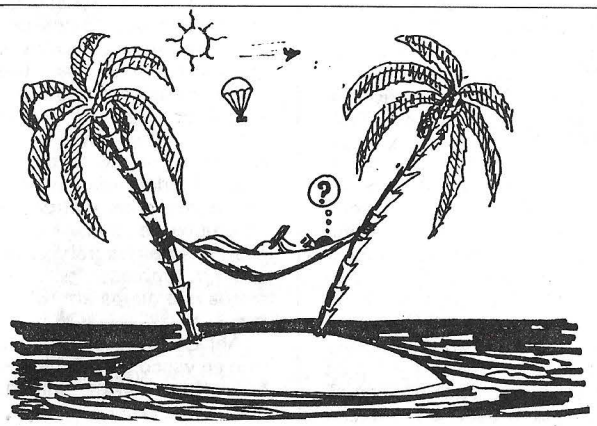
Volviendo a la primera tabla, SPV y RPV ponen a 1 y a 0 un biestable de propósito general, pero que si está la impresora conectada debe estar a 1. Igual función hacen SPU y RPU, pero con el biestable PU. Por último NOP no hace (evidentemente) nada.

Para los que piensan hacer un emsamblador estos nemónicos son muy cómodos, porque todos tienen tres letras, y los diferentes atributos se dan, a igualdad de nemónicos, en el campo de operando, lo que simplifica mucho la tarea del emsamblador. Además se pueden agrupar las instrucciones en grupos de similar funcionamiento (estamos trabajando en este emsamblador, pero intentamos que quepa en 1850 bytes...).

¡Ah! EAI-n no es una expresión en vasco, sino Exclusive or Acumulator with Inmediate data.

Víctor Manuel Díaz
Iñaki Cabrera

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0					INC XH	INC YH	INC UH						RDP			
1	SBC# (X)	SBC# (Y)	SBC# (U)						SIE		SBC# (nn)	HLT	SDP			
2					DEC XH	DEC YH	DEC UH									
3	ADC# (X)	ADC# (Y)	ADC# (U)								ADC# (nn)			DRR# (X)		
4																
5	LDA# (X)	LDA# (Y)	LDA# (U)								LDA# (nn)					
6																
7	CPA# (X)	CPA# (Y)	CPA# (U)								CPA# (nn)			DRL# (X)		
8	LDX X	LDX Y	LDX U		LDX S	LDX P			PSH X	PSH Y	PSH U		PSH A			
9	AND# (X)	AND# (Y)	AND# (U)		ANI# (X),n	ANI# (Y),n	ANI# (U),n				AND# (nn)				ANI# (nn),n	
A	POP X	POP Y	POP U		STX X	STX Y	STX U		POP A	TTA	TTA	ITA	ADR X	ADR Y	ADR U	
B	ORA# (X)	ORA# (Y)	ORA# (U)		ORI# (X),n	ORI# (Y),n	ORI# (U),n				ORA# (nn)				ORI# (nn),p	
C	DCS# (X)	DCS# (Y)	DCS# (U)		OFF				DCA# (X)	DCA# (Y)	DCA# (U)		ATP		ATT	
D	EOR# (X)	EOR# (Y)	EOR# (U)		BII# (X),n	BII# (Y),n	BII# (U),n				EOR# (nn)				BII# (nn),p	
E	STA# (X)	STA# (Y)	STA# (U)		STX S	STX P			CDV		STA# (nn)	RIE	AMØ	AM1		
F	BIT# (X)	BIT# (Y)	BIT# (U)		ADI# (X),n	ADI# (Y),n	ADI# (U),n				BIT# (nn)	B			ADI# (nn),p	



RUTINAS EN CODIGO MAQUINA PARA PC 1500

He aquí unas rutinas en Lenguaje máquina para la PC 1500 que nos ayudarán a programar y corregir programas en lenguaje de máquina.

La primera rutina que conste a su vez de tres subrutinas es la VENTANA HEXADECIMAL que nos ofrece en pantalla un volcado de memoria en hexadecimal de 7 en 7 octetos antecedidos por la dirección del primero de ellos.

Para usarla:

CALL &29,D coloca en la ventana reserve III la dirección d y los valores de los 7 octetos siguientes desde D. Al final de la ejecución D no ha variado.

CALL &C5,D ídem. Sólo varía en que D al final de la ejecución vale D+7; quedando preparado para volcar los siguientes 7 octetos.

Estas dos subrutinas tienen el valor de D limitado a 32767, pero para solventar este problema está la subrutina E6. (CALL &E6,D).

Una vez ejecutada cualquiera de estas rutinas bastará apretar la tecla RCL (en la zona III del RESERVE) y podremos observar los octetos volcados.

La segunda rutina es el POKE RAPIDO, que nos permite, cómodamente, cargar en cualquier zona de la memoria RAM una cadena de octetos. Su utilización es la siguiente:

Ponemos en D la dirección donde queremos empezar a introducir los octetos; después hacemos CALL &E1,D y seguidamente metemos la cadena de octetos en hexadecimal (el límite de longitud de la cadena es la pantalla) y seguido de un ENTER nuestra cadena queda almacenada a partir de la dirección dada. Al acabar la ejecución D=D+ número de octetos introducidos.

CALL&CA,D hace lo mismo, pero no incrementa D.

Como hemos visto sería muy útil el asignar las diferentes llamadas a subrutinas en las teclas correspondientes a la zona RESERVE III.

Las direcciones de los programas están diseñadas para el módulo del 16 K; pero no representa mayor problema el trasladarlas a otra zona de la memoria, teniendo especial cuidado en los saltos que dentro de éstas se producen.

VENTANA HEXADECIMAL

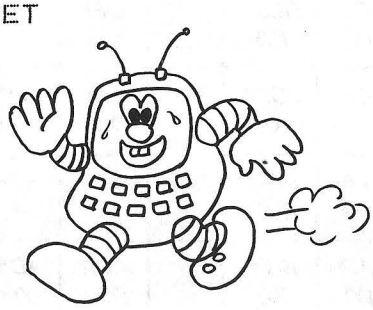
```
0008 58 00        LD D,00
000A 5A 3D        LD E,3D
000C 84           LD A,B
000D F1           NEX
000E BE 00 1B     CALL 001B
0011 84           LD A,B
0012 BE 00 1B     CALL 001B
0015 04           LD A,C
0016 F1           NEX
0017 BE 00 1B     CALL 001B
001A 04           LD A,C
001B 9B 0F        AND A,0F
001D B7 0A        CP A,0A
001F 87 04        JR H,+04
0021 BB 30        OR A,30
0023 51           LDI (DE),A
0024 9A           RET
0025 23 36        ADC A,36
0027 51           LDI (DE),A
0028 9A           RET
0029 BE 00 08     CALL 0008
002C 6A 06        LD L,06
002E B5 20        LD A,20
0030 51           LDI (DE),A
0031 45           LDI A,(BC)
0032 FD 88        PUSH BC
0034 0A           LD C,A
0035 BE 00 15     CALL 0015
0038 FD 0A        POP BC
003A 88 0E        DJC,-0E
003C 9A           RET
*********************************
00C5 BE 00 29     CALL 0029
00C8 FB           SCF
00C9 9A           RET
*********************************
```

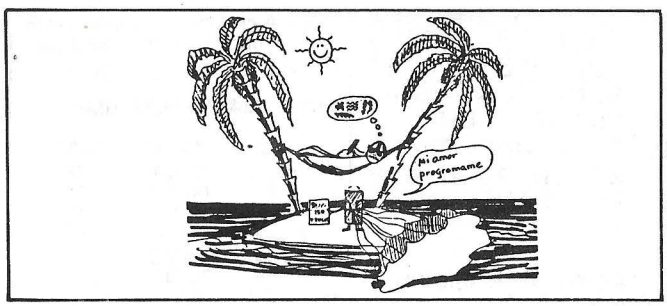
POKE RAPIDO

```
00CA 58 7B        LD D,7B
00CC 5A 60        LD E,60
00CE FD 88        PUSH BC
00D0 FD 18        LD BC,DE
00D2 BE ED 95     CALL ED95
00D5 81 07        JR NC,+07
00D7 FD 5A        LD DE,BC
00D9 FD 0A        POP BC
00DB 41           LDI (BC),A
00DC 9E 10        JR -10
00DE FD 0A        POP BC
00E0 9A           RET
00E1 BE 00 CA     CALL 00CA
00E4 FB           SCF
00E5 9A           RET
*********************************
```

VENTANA HEX (2)

```
00E6 LD B,7B
00E8 LD C,60
00EA CALL ED95
00ED JR NC,+14
00EF LD H,A
00F0 CALL ED95
00F3 JR NC,+0E
00F5 LD L,A
00F6 LD BC,HL
00F8 CALL 0029
00FB LD A,B
00FC LD (0005),A
00FF LD A,C
0100 LD (0006),A
0103 RET
```





Caracteres programables en PC 1500

¿No ha querido nunca programar sus propios caracteres? No sueñe. Esta operación se puede hacer en el PC 1500. No sólo se presentarán en la pantalla ¡sino también en la impresora!

Las direcciones que originan **tantas posibilidades son** &785D y &785E. Intervendrán **legítimamente** cuando se añada un módulo para los caracteres al PC 1500. La dirección & 785D indicará la presencia del módulo tomando el valor & 00 y la dirección &785E indicará el principio de la lista de valores necesarios para formar los caracteres. Podemos modificar el valor de estas direcciones, ¡aprovechémoslo! Los caracteres podrán emplearse por medio de la función CHR$n, que no tiene un argumento superior a 127.

Empecemos por los caracteres programables en pantalla. Simulamos la presencia del módulo citado haciendo POKE &785D, &00. Esta operación deberá hacerse cada vez que se encienda la máquina, porque la dirección se reinicializa automáticamente. Indicaremos el principio de la lista en la que queremos colocar los caracteres. Esta dirección tendrá que ser forzosamente un múltiplo de &100 ya que sólo disponemos de un octeto que contendrá la parte de mayor valor de la dirección.

Eligiendo &4100 como **dirección, tecleamos POKE** &785E, &41. Ahora POKEamos nuestros valores que se calcularán como para un GPRINT. Por ejemplo, el apóstrofo se dibuja mediante GPRINT"000B0700 00"; por **lo tanto haremos POKE** ,&4100,&00,&03,&07,&00, &00 y CHR$ 128 dará el apóstrofo. Pasemos ahora a los caracteres programables en impresora. La organización de los valores es un poco más complicada, porque no todos los caracteres tienen igual número de trazos y se emplea un octeto por trazo o desplazamiento.

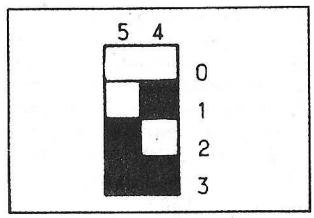
Decidimos colocar los valores que servirán para trazar los caracteres a partir de la dirección &40C5. Por ejemplo; si se necesitan 7 octetos para trazar el primer caracter, el segundo empezará en la dirección &40CC; si este precisa 10 octetos, el tercer carácter comenzará en la dirección &40D6, etc. Todas estas direcciones, que indicarán a la máquina donde comienza cada caracter, deberán colocarse a partir de una dirección, por ejemplo, &4210.

Esta última dirección se colocará en &4100+&200 para su parte más significativa y en &4100+&201 para la menos significativa; por supuesto, con la condición de hacer antes POKE &785E, &41.

En resumen, hacer:
POKE &785D, &00, m
POKE (m+2) * 256, n, p
POKE n * 256 + p, w, x, y, z...
Para el ejemplo anterior:
POKE &785D, &00, &41
POKE &4300, &42, &10
POKE &4210, &40, &C5, &40, &CC...

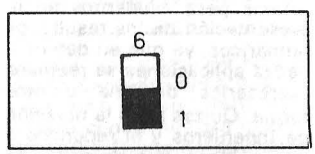
Pero ¿cómo se dibuja un carácter? El bolígrafo se desplaza en una matriz de 6 x 4, pero veremos que podemos llegar más allá. ¡Basta con un octeto por desplazamiento del bolígrafo! Veamos cómo está organizado el octeto (ver figura 1).

- Los bits 0, 1 y 2 contienen la distancia a recorrer, de 1 a 7; ¡porque la distancia 0 hace cosas muy raras!
- Los bits 4 y 5 determinan el movimiento:



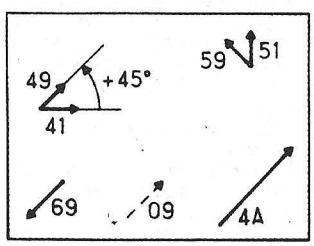
0: desplazamiento hacia la derecha →
1: desplazamiento hacia arriba ↑
2: desplazamiento hacia la izquierda ←
3: desplazamiento hacia abajo ↓

- El bit 6 determina la posición del bolígrafo:



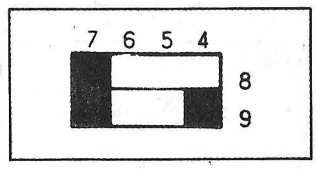
0: bolígrafo levantado.
1: bolígrafo bajado.

- El bit 3, cuando está encendido provoca el trazado de una diagonal en un cuadrado cuyo lado viene determinado por los bits 0, 1 y 2. La diagonal forma un ángulo de + 45° con el movimiento que hubiera tenido el boli si el bit 3 hubiera estado apagado.



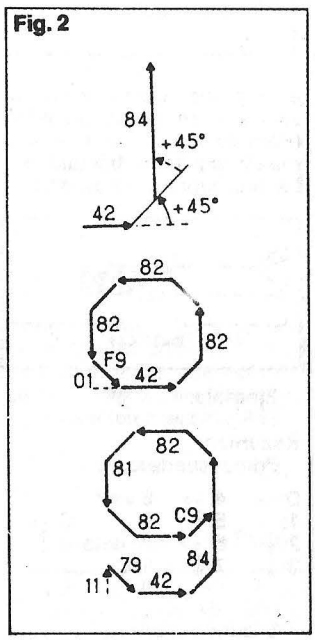
- El bit 7 se emplea para movimientos especiales. (Cuando el bit 7 está encendido, se modifican los efectos de los bits 3, 4, 5 y 6. El bit 3 debe estar apagado.)

Si el bit 7 está encendido:

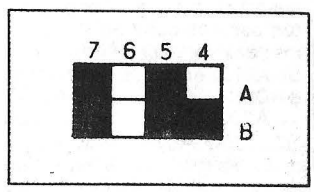


el boli traza una diagonal (en un cuadrado de lado 1). Forma un ángulo de + 45° con el movimiento anterior y después traza un línea (cuya longitud viene indicada por los tres primeros bits), que forma un ángulo de + 45° con la diagonal trazada anteriormente.

Fig. 2



Si los bits 7 y 5 están encendidos:



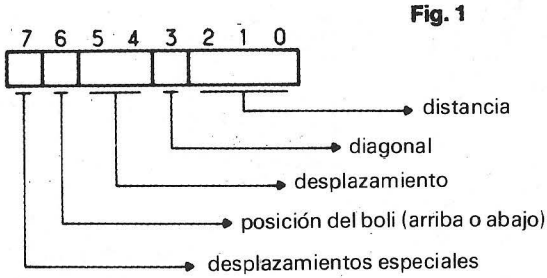
se obtiene el mismo resultado anterior con un ángulo de – 45°.

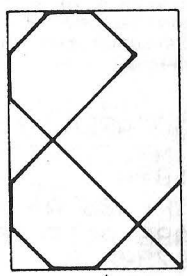
Ejemplos:

Ejemplos (ver figura 3):

Si los bits 7 y 6 están encendidos, indican el final de un carácter: el boli traza el rasgo definido por los dieciséis restantes bits y después se coloca delante de la matriz siguiente (→).

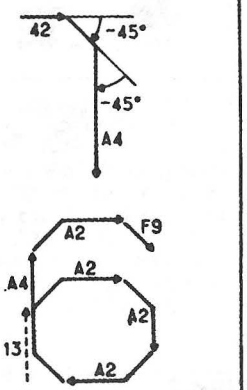
Fig. 1



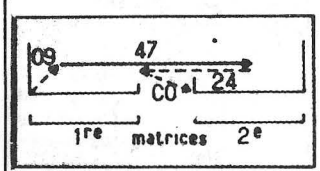

matriz

Observación: Si el boli se desplaza más de cuatro unidades hacia la derecha, tras la instrucción final, no encontrará la matriz siguiente y seguirá su camino sin pararse. Para evitarlo basta con hacerlo volver

Fig. 3



a su matriz de partida sin trazar nada y terminarlo por &CO. (Para conducir al boli ante la matriz siguiente, bastará con hacerle imprimir un espacio.)



Ejemplo:

Resumen:

Primer cuarteto:

0: →	4: →	8 ver	
1:	5: ↑	9	C: →
2: ←	6: ←	A: deta-	D: ↑
3: ↓	7: ↓	lles	E: ←
		B	F: ↓

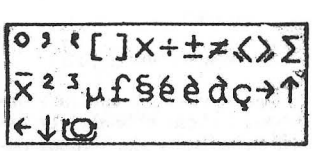
Segundo cuarteto:

1, 2, 3... 7: longitud de los desplazamientos.

9, A, B... F: diagonal de un cuadrado de lado 1, 2, 3... 7.

¡Puede crear caracteres a su gusto! Si quedan algunos puntos oscuros eche un vistazo a los caracteres de la impresora. Se sitúan entre las direcciones &A000 → A01F y &A07F- → A28A; las direcciones &A020 → A07E indican la parte inferior de la dirección del principio de cada carácter y la parte más significativa es &A0, &A1 o A2.

Encontrará a continuación la lista hexadecimal que permite obtener los siguientes caracteres (CHR$ 128 → CHR$ 154):



Para emplearla:

NEW &4302 (o NEW &4244 con la condición de que no se modifiquen las direcciones &4300 y &4301) POKE &785D, &00, &41 POKE &4300, &40, &C6. Después POKEar los valores en las direcciones indicadas en la lista (&40C4 → &4243).

Serge Philipp

Ahorramos papel

Puestas al inicio de la memoria del programa, estas dos pequeñas lineas de Basic, permitiran evitar perder un trozo de papel cadá vez que conectamos el PC–1500 cuando esta conectado al CE–150.

```
1 : ARUN : IF ASC—-O LF-5
2 : END
```

Por supuesto se puede variar el valor de la instrucción LF según queremos que salga mas o menos papel sobre la impresora.

Tratamiento de números

Presentamos dos nuevas rutinas para ayudarnos en la presentación de los resultados numéricos, ya que en determinadas aplicaciones se requiere expresarlos de una u otra forma. Quizas sean la notación de ingenieros y el redondeo a un número determinado de decimales (el llamado en algunas calculadoras FIX) los más importantes y claros ejemplos; que junto con su utilidad en exámenes y proyectos, nos ha inducido a pensar estas dos pequeñas rutinas.

FIX PARA LA PC-1500

Esta rutina simula en BASIC: con algunas limitaciones; la función FIX. Estas limitaciones son:

—solo nos nuestra el numero en pantalla; no es operativo inmediatamente; para conserva el número sin redondear) o con la variable C (número redondeado).

—Sólo sive para un número cada vez.

El redondeo se efectua de igual forma que cualquier calculadora que incorpore la función FIX; pero nos encontramos con un problema, a los numeros que excedian de 10'9 no les afectaba el FIX y a los que no llegaban a 10'-8 les redondeaba invariablemente a 0. De ahí que se nos ocurriera una pequeña astucia en la zona de variables; esperamos pronto poder describir esta zona con todo detalle.

```
10: "A"AREAD A:
    INPUT "FIX ";F
    :B=A
15: IF ABS A>1E9OR
    ABS A<1E-9POKE
    &7908,0
20: Z=B*10^F:C=INT
    Z:Z=Z-C
30: IF Z>0.4555555
    55LET C=(C+1)/
    10^F:POKE &791
    0,PEEK &7900:
    PRINT C:END
40: C=C/10^F:POKE
    &7910,PEEK &79
    00:PRINT C:END
```

NOTACION DE INGENIEROS PARA PC-1500

Esta es otra función que suelen incorporar las calculadoras cientificas y que debido a que su uso esta muy extendido hemos creido que no le podia faltar a nuestra PC-1500.

La notación de ingenieros consiste en expresar los numeros en potencias multiplos y submultiplos de 3 (función ENG en las calculadoras); es decir 1593 se representa 1.593 E3 y 0.000578 se escribe 578 E-6. Además cada potencia tiene un prefijo determinado que evita el engorroso manejo de las potencias. A continuación ofrecemos una tabla con dichos prefijos.

POTENCIA	PREFIJO	SIMBOLO
E 12	TERA	T
E 9	GIGA	G
E 6	MEGA	M
E 3	KILO	K
E —3	MILI	m
E —6	MICRO	
E —9	NANO	n
e —12	PICO	p
E —15	FEMTO	f
E —18	ATTO	a

```
50: "S"AREAD S:E=
    INT (INT LOG
    ABS S/3)*3:U=S
    /10^E:PRINT U;
    " E";E:END
```

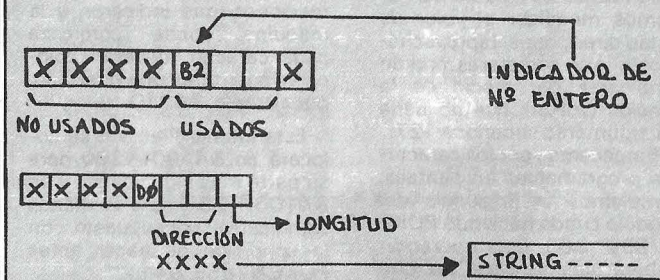
Al ejecutar esta rutina el número no es directamente operable y para trabajar con el se utiliza la variable S donde esta almacenado.

IÑAKI CABRERA
VICTOR DIAZ

Estructuras de las variables

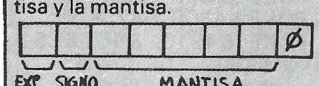
Antes de entrar en la estructura de las variables veamos como representa internamente los números la PC-1500:

También hacen falta 8 bytes, pero 5 no se usan, ya que sólo se necesitan dos bytes para representar el número (en completo a 2), que está comprendido entre 32768 y 32767. El otro byte indica que es un número entero.



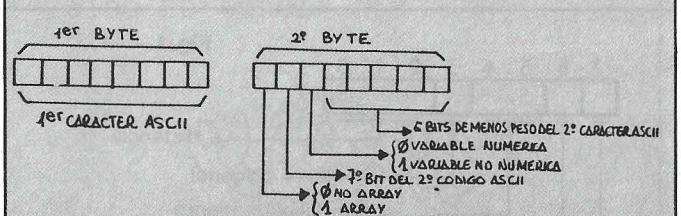
a) Expresión de un número en coma flotante.

Se necesitan 8 bytes para expresar un número en coma flotante, los cuales contienen el exponente, el signo de la mantisa y la mantisa.



El exponente se expresa como un número binario, y si es negativo por su complemento a dos.

b) Expresión de un número entero:



c) Expresión de una cadena de caracteres.

Las fijas ocupan 16 bytes y las que definimos nosotros ocupan esos 16 bytes para los caracteres, más 8 bytes más de los que sólo 4 son datos válidos. Estos 4 bytes indican la direción donde está almacenada la cadena y la longitud de esta.

Ahora veremos la estructura del nombre de una variable:

Consiste en dos bytes, que identifican la variable y el tipo de esta (numérica/no numérica, array o simple).

Vícytor Manuel Díaz
Iñaki Cabrera

Haga copias de su pantalla en la impresora

Este subprograma memoriza la pantalla de la PC-1500 y realiza una copia de ésta en el papel de la impresora.

Utiliza un GOTO y cuatro variables: X e Y son las coordenadas, I es el punto y J representa las potencias de dos.

Las líneas 15 y 75 trazan el recuadro (no imprescindible) y LF 20, en la línea 80, posiciona el papel para poder retirarlo.

Si ponemos C en lugar de COPY y END en lugar de RETURN podemos, con DEF C, hacer una copia de la pantalla directamente a partir del teclado.

El tiempo de copia varía entre uno y dos minutos.

Pierre Guilbert

```
10:"COPY"GRAPH :
   GLCURSOR (180,
   0):SORGN
15:LINE (36,-312)
   -(36,12)-(-30,
   12)-(-30,-312)
20:FOR Y=0TO -620
   STEP -4
30:I=POINT (-Y/4)
   :IF I=0GOTO 70
40:J=.5:FOR X=0TO
   -24STEP -4
50:J=J*2:IF IAND
   JLINE (X,Y)-(X
   +2,Y+2),,,B
60:NEXT X
70:NEXT Y
75:LINE (36,-312)
   -(36,-630)-(-3
   0,-630)-(-30,-
   312)
80:TEXT :LF 20:
   RETURN
```

Rutinas principales de E/S

Para poder escribir programas en Lenguaje Máquina que sirvan para algo es necesario poder comunicarnos con el ordenador, para lo cual hacen falta rutinas que gestionen la entrada-salida. Aquí os presentamos algunas, descubiertas por C. Boyer y Erik Frankenfeld.

1) Rutinas que controlan la visualización:

CALL EE71 (o SBR F2): Ejecuta un CLS. Esta rutina afecta los registros A, H y L, así que hay que cuidarlos.

CALL ED5B (o SBR 8A): PRINT CHR$ (A). En esta rutina BC representa el cursor, B puede valer de 74h a 76h, según en qué parte de la pantalla estemos (4 partes) y C indica la posición dentro de cada parte (00h a 4Ch). Después de la ejecución BC se ve incrementado en 6, excepto en los extremos de cada cuarto de pantalla (cuidado con esto). Además se ven afectados A y HL.

CALL EDF6 (o SBR 88): GRINT (A). BC realiza la misma función que en la anterior, pero se incrementa en 1 cada vez. Afecta a H y A.

CALL ED00 (SBR 92): Esta potente rutina visualiza una zona ASCII, cuyo comienzo debe estar en HL y su longitud en A. Afecta, además de HL y A, a BC. Aquí no hay problemas de cambio de cuarto.

2) Rutinas para leer el teclado.

CALL E42C: INKEY$. Si después del CALL el carry está a 1 quiere decir que no ha sido pulsada, ninguna tecla, si está a cero quiere decir que ha sido pulsada, y su código está en el acumulador. Además de afectar a A (evidentemente), afecta a BC y L.

CALL E451 (SBR A6): Realiza un test de la tecla BREAK. Si después de la ejecución el indicador (flag) Z está a 1, esta ha sido pulsada, y ese indicador queda a cero si no lo ha sido. No afecta ningún registro.

CALL E267: INKEY$ avanzado, tiene en cuenta SHIFT, DEF, SMALL, y no es necesario hacer un test del carry, ya que esta rutina se detiene en espera de la pulsación de una tecla. El código ASCII se almacena en A. Afecta a BC y HL.

3) Para el zumbador.

CALL E66F; BEEP 1,L,BC. Es decir, si L contiene el tono y BC la duración, CALL E66F hará sonar esa nota. Sólo afecta al acumulador.

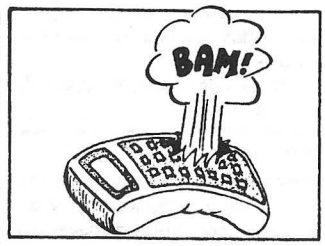
CALL E669: BEEP 1. Corresponde a un CALL E 66F con L=8 y BC=01A0. Modifica el contenido de A, B, C y L.

He aquí una tabla resumen de los registros que son afectados por estas rutinas, así como de su funcionamiento:

CALL	FUNCION	AFECTA A:
EE71	*CLS*	*A,H,L*
ED5B	*PRINT CHR$A*	*A,B,C,H,L*
EDF6	*GPRINT A*	*A,B,C,H,L*
ED00	*PRINT zona ASCII*	*A,B,C,H,L*
E42C	*INKEY$*	*A,B,C,L*
E451	*BREAK?*	*ninguno*
E267	*GET*	*A,B,C,H,L*
E66F	*BEEP 1,L,BC*	*A*

A destacar que la mayoría de estas rutinas se pueden llamar mediante SBR nn, lo que indica que sus direcciones inciales están almacenadas en la última página de ROM (FFOO + nn), luego ppr ahí se puede investigar buscando nuevas rutinas. Adelante.

Víctor Manuel Díaz
Iñaki Cabrera.



Para despistados

He aquí una ingeniosa idea de **Julián Sagredo López**, que evitará más de un «peligroso» enfado contra nuestra querida PC-1500: Quién no ha metido una línea de programa en modo RUN, descubriendo esto último al ver el fatídico ERROR 1 después de dar ENTER? Estareis de acuerdo en que esta situación puede ser peligrosa para la máquina, ya que más de uno de nosotros sentiría deseos de arremeter contra ella. Pero si después de enterarnos de nuestra torpeza insertamos al principio de la línea un (salvador) **POKE 28751,32:** y damos ENTER, veremos que (además de volver a aparecer ERROR 1) la «bandera» RUN pasa a ser PRO, luego no tenemos más que pulsar una de las flechas de desplazamiento para visualizar la línea, borrar el POKE y (por última vez) pulsar ENTER, con lo que nuestra querida línea pasará a formar parte de la memoria programa.

Víctor Manuel Díaz
Iñaki Cabrera

El PC en su 31

¿Nunca ha esperado obtener algunos caracteres, además de los 26, en la pantalla? Este corto programa colmará sus deseos. Sírvase del generador de caracteres situado desde &FCA o hasta &FE7F.

```
10 Y = &FDE5
20 WAIT 0
30 FOR Z = 1 TO 31
40 FOR X = 0 TO 4
50 GPRINT PEEK (Y + X) ;
60 NEXT X
70 Y = Y + 5
80 NEXT Z
90 WAIT
100 GPRINT 0
```

Vea la fórmula que le permitirá obtener la primera de las cinco direcciones que contienen los códigos para imprimir un caracter (en este caso A$)).

PEEK (&FCA0 + ASC A$*5 – 160).

Rodolfo La Pietra

Visualizador de estado

Publicamos un truco que nos presenta en pantalla un histograma mostrando la memoria libre disponible.

La principal característica de este programa es que funciona cualquiera que sea el módulo memoria que tengamos en nuestra máquina.

El funcionamiento es bastante simple:

Línea 10: calcula la memoria total, ocupada o no.

Línea 20: imprime la zona libre, con factor de escala.

Línea 30: imprime la zona ocupada, con factor de escala.

Línea 40: imprime la cantidad de memoria disponible.

El factor de escala se calcula de la siguiente forma:

F = 115 / (STATUS 0 + STATUS 1)

donde 115 es el número de líneas reservadas para dibujar el histograma.

PROGRAMA

```
10: T = STATUS 0 + STATUS 1: T = 115/T: WAIT 0.
20: FOR I = OTO MEM *T: GPRINT «7F» :: NEXT I.
30: FOR I = MEM *TTO 115: GPRINT «41» :: NEXT I:
GPRINT «7F»;
40: WAIT: CURSOR 20: PRINT MEM: END.
```

Víctor Manuel Díaz
Iñaki Cabrera

SHARP MZ80 B

los encantos del sharp

Inhibición total de la función BREACK

La anulación de esta tecla no impide el paro del programa mediante la pulsación conjunta de SHIFT + BREACK, os propongo la forma de inhibir completamente esta función.

1º. Inhibir la tecla de Breack mediante un POKE $0D67,0
2º. Anular la rutina de Breack del Monitor colocando un código de RETURN al inicio ⇒POKE $0571, $C9.

Con esto habremos anulado la función. Para poder parar el programa una vez inhibido el breack se puede pulsar el RESET de la parte posterior de la unidad central con lo que el sistema pasará a modo MON.

Una vez en monitor saltamos a BASIC con el comando JUMP (J) a la dirección $1280. El programa que teníamos almacenado permanece en memoria sin ningún peligro.

CONTROL DEL SALTO DEL TABULADOR: Los saltos que realiza el cursor mediante la pulsación de la tecla TAB se realizan de 5 en 5 caracteres.

Podemos cambiar la tabla de tabulaciones para conseguir que el tabulador salte a las columnas que nosotros programemos.

En la posición de memoria $1140 (4416 Dec.) está almacenada la tabla de los diferentes saltos de tabulación. Los valores normales son los siguientes: (listados secuencialmente a partir $1140).

$1140 - 01 05 0A 0F 14 19 1E 23 28 2D 32 27 3C 41 46

Cada uno de los códigos expresa el número de columna a la que saltar lógicamente los que superan a 39 solo funcionan en CONSOLE C80.

Podemos cambiar estas tabulaciones a base de sucesivos POKE o bien pasando a MON y modificando los datos de las direcciones de memoria correspondientes. Una forma cómoda de entrar nuevas tabulaciones sería la de realizar un programa BASIC que pidiera las tabulaciones o que mediante una serie de GETs se pudiera correr el cursor por una línea y al pulsar determinada tecla almacenar en una variable la posición correspondiente, para finalmente introducir los datos con POKE.

Nota: Lógicamente cada tabulación debe ser un número mayor que el anterior.

Una de las utilidades de la modificación de las tabulaciones es la inclusión de la rutina mencionada en los programas de tratamiento de textos.

Uno de mis últimos descubrimientos sobre MZ80B es una fabulosa rutina que borra todos los sectores libres del disco. Está localizada en $5EB0 (24240 Dec.). Si hacemos un USR ($5EB0) o un JUMP a esta dirección desde monitor, veremos como se puede anular, en segundos, todos los sectores libre de un diskette.

Nota: Los datos del disco siguen enteritos.

PROGRAMA FANTASMA: Podemos hacer que un programa en memoria desaparezca y vuelva a aparecer. Antes de hacer desaparecer el programa si queremos volverlo a tener en memoria debemos anotar el contenido de las direeccciones de memoria $675C y $675D ($505C en Basic cassette).

PEEK ($675C) y PEEK ($675D) visualizará el contenido de estas direcciones.

Como hacer desaparecer el programa: POKE $675C,0 ($505C Cass.) y seguidamente POKE $675D,0 ($505D) provocará que al pedir un LIST no aparezca ninguna línea y si ejecutamos el programa con RUN aparecerá el mensaje de Error 19. (No hay programa).

Pues no!, el programa no ha desaparecido sinó que está en memoria. ¿Qué ha pasado?: Simplemente le hemos anulado los dos primeros Bytes del programa y el ordenador se cree que no hay programa almacenado. (Parece mentira, pero todavía podemos engañar al computador).

Para que el programa se "materialice" nuevamente tendremos que volver a poner en su sitio los datos que hemos cambiado.

Si cambiamos los datos pueden suceder cosas imprevisibles. Desde listar solo una parte del programa hasta listar códigos que no son del programa. (Se acaba siempre pulsando la tecla de IPL).

¿COMO GRABAR UN MONITOR-INTERPRETER EN EL CASSETTE?. Probablemente, se te ha ocurrido alguna vez grabar el MONITOR y el BASIC en un cassette para tener un duplicado y habrás podido constatar que el sistema tiene un magnífico cartel de 'CHECK SUM ERROR'.

Efectivamente, NO se puede grabar el Monitor en cassette (a simple vista, claro) ya que el MONITOR no se puede grabar a sí mismo.

Después de no pocas cavilaciones y gracias al aprendizaje de una fabulosa instrucción del microprocesador Z.80 que permite el movimiento de bloques de datos (LDIR) encontre un sistema bastante potable para conseguir la grabación.

La teoría es fácil: Si no se puede grabar a si mismo, porque no lo duplicamos en la zona libre del usuario y le añadimos una rutina para que una vez cargado se coloque en el lugar correspondiente, limpiando al mismo tiempo la zona del usuario?.

El problema consiste en encontrar la dirección de fin del sistema de forma que no grabemos datos de mas, ni de menos.

Realización práctica (Los datos reales se encuentran con práctica, paciencia y vista, los indicados son a modo de ejemplo, cualquier parecido con la realidad es pura coincidencia).

Supongamos un Sistema (Llamaremos sistema al conjunto de MONITOR + INTERPRETER para abreviar) que tiene su inicio en $0000 y la dirección final es $6000.

1º. En $F000, por ejemplo, hacemos un pequeño programa para mover todo el bloque hacia la zona de usuario:

$7000 21 00 00 LD HL,0000H; (Cargamos en HL la direcc. inicio del bloque origen).

$7003 11 00 70 LD DE,7000H; (Cargamos en DE la direc. inicio del bloque destino).

$7006 01 00 60 LD BC,6000H; (Cargamos en BC el número de Bytes a mover).

$7009 ED B0 LDIR; (Instrucción de mover el bloque).
$700B C3 80 12 JP,1280H; (Salto al BASIC una vez realizado).

Este programa lo entraremos mediante el comando 'M' (Modificación) del MONITOR.

Entrado el programa lo ejecutamos mediante un JUMP (Comando J del Monitor) a la dirección $7000. Como al final del programa le hemos dado la instrucción JUMP, 1280H, finalizado este saltaremos a BASIC y aparecerá el mensaje de READY.

Ahora añadiremos unas cuantas instrucciones para grabarlas juntamente con el sistema que nos servirán para que, una vez cargado en máquina, se sitúe nuevamente en su localización correcta y se limpie la zona de usuario.

$6990 21 00 70 LD HL,7000; Dirección inicio bloque origen.
$6993 11 00 00 LD DE,0000; Dirección inicio bloque destino.
$6996 01 00 60 LD BC,6000; Número de Bytes a mover.
$6999 ED B0 LDIR; movemos el bloque.
$699B C3 20 12 JP,1220; Salto a la primera direc. BASIC.

Sólo queda grabar el programa mediante el comando 'S' (SAVE); Las direcciones que nos solicitará el ordenador son las siguientes:

START ADDRESS =6990
END ADDRESS =D000
JUMP ADDRESS =6990.

Una vez grabado deberíamos verificar la grabación con el comando 'V'. Posteriormente, el programa podemos traspasarlo a un diskette mediante la utilidad 'Filing CMT' del MASTER DISKETTE.

José M. Vidal Lacasa.

Cambio de la velocidad de parpadeo del cursor

En la dirección de memoria $ ϕ6DD (dec. = 1757) está contenida una variable que regula el parpadeo del cursor. Variando el contenido de esta posición de memoria podremos variar la velocidad de parpadeo (El número de destellos/unidad de tiempo) del cursor.

Contenido inicial
=ϕ6DD 4ϕ (Valores Hexa)
1757 64 (" dec.)

1. **Aumento de la velocidad: cambiar $4ϕ por un n.º comprendido entre $ϕ1 y $3F.**
2. **Reducción de la velocidad: Valores comprendidos entre $41 y $FF.**
3. **Cursor fijo, sin parpadeo: Introducir como dato = $ϕϕ.**

Redefinición de las teclas del MZ 80 B

Las teclas del MZ 8ϕB pueden redefinirse asociando a cada una de ellas un nuevo código. Con ello podemos conseguir, entre otras cosas, la posibilidad de disponer en una tecla de una función o caracter que por teclado no estuviera disponible (por ejemplo el caracter del cursor o el semigráfico que tiene por código 147). Otra interesante posibilidad es la de poder inhibir determinadas funciones como el desplazamiento del cursor arriba y abajo en determina-

dos programas (para no salirse accidentalmente de línea en una entrada).
Adjunto un listado del programa "TECLADO" que en su versión inicial permite cambiar la definición de una tecla por la de otra tecla.
Una modificación interesante del programa es la de poder entrar el código ASCII que deseamos tenga la tecla en cuestión. como 4ϕ y 5ϕ del listado.
En ambas versiones puede suprimirse la línea 6ϕ ya que la ejecución es tan rapida que apenas da tiempo de leer el mensaje de "UN MOMENTO!".
Nota = Si se define más de una tecla con el mismo código la que se localizará cuando se solicite un nuevo cambio es la primera. (Primera se entiende por primera en la tabla de direcciones de las teclas, no por primera que se haya modificado).
La ejecución del programa es muy sencilla. Cuando aparece el mensaje "CAMBIAR TECLA =" pulsamos la tecla que deseemos cambiar y a continuación, según la versión del programa, aparecerá "POR..." y pulsamos la tecla correspondiente o bien "CODIGO ASCII? =" e introducimos el código que le queramos asignar.

NOTA FINAL = Cuando se han modificado las instrucciones de BASIC no hace falta modificar los programas; al cargarlos se modifican conforme a la nueva sintaxis. Incluso el programa TRADUCTOR se modifica en su listado conforme se van modificando las instrucciones.

José M.ª Vidal Lacasa

```
 1 REM ---- DEFINICION TECLADO MZ 80 B ----
 2 REM ---- JOSE M. VIDAL LACASA - NOV. 82
 3 REM
 4 REM
10 CONSOLE C40:PRINT CHR$(6);"CAMBIAR TECLA = "
20 X$="":GET X$:IF X$="" THEN 20
30 X=ASC(X$)
35 PRINT:PRINT "POR ... "
40 Y$="":GET Y$:IF Y$="" THEN 40
50 Y=ASC(Y$)
60 PRINT:PRINT "UN MOMENTO !"
65 FOR P=3400 TO 3575
70 IF PEEK(P)=X THEN POKEP,Y:P=3575
75 NEXT P
80 GOTO 10
```

'Trucos' y astucias para el MZ 80 B de SHARP. Versiones de BASIC interpreter SB-6510 y SB-6610.

Nota preliminar : Las posiciones de memoria indicadas deben tomarse como punto de referencia y comprobarse antes de efectuar ninguna modificacion ya que existe la posibilidad de que la direccion o dato indicados se encuentren en las inmediaciones, entre las 5 superiores e inferiores, a la posicion indicada. Para facilitar la localizacion exacta junto con cada direccion adjunto un listado parcial de la zona que la rodea.

Las direcciones vienen expresadas en HEXADECIMAL y su equivalente en DECIMAL. La modificacion del contenido de una direccion de memoria se puede realizar mediante POKE o bien pasando a monitor (MON) y usando el comando de modificacion; Teniendo en cuenta que en este ultimo caso tanto las direcciones como los datos deben expresarse en HEXADECIMAL.

El simbolo '$' que precede a las direcc. o datos indica que estan expresados en forma Hexadecimal , caso contrario en Decimal.

1- Pulsando 'SHIFT' + '↓','→','↑' o '←' el cursor se desplaza de forma continua por la pantalla. Ahora bien, el desplazamiento resulta un tanto lento en la pantalla de 80 caracteres. Vamos a ver como modificar esta velocidad de desplazamiento del cursor :

La posicion de memoria $0725 (1829) contiene el control de la velocidad del cursor. El valor inicial es $40 (64) variando este dato variaremos la velocidad : a) Disminuyendo el valor (de $01 a $3F) aumenta la velocidad. b) Aumentando el valor (de $41 a $FF) disminuye la velocidad.

listado de comprobacion de la zona :

```
          $0724  3E
→ $0725   40
          $0726  32
```

2- CAMBIO DEL CARACTER DEL CURSOR : El caracter del cursor puede ser modificado alterando 2 posiciones de memoria del monitor. Si modificamos solo una de ellas el cursor aparece fijo, sin intermitencias. El contenido inicial de las dos posiciones es $1F (31) es decir el codigo HEX (ASCII) del cursor de origen.

El codigo Hex. o ASCII del cursor que deseemos se debe introducir en las dos posiciones de memoria. Logicamente no se pueden introducir codigos de control como $06 (Borrado pantalla) debe de ser un codigo HEX superior a $1E (o Dec. superior a 30).

Las direcciones son $06D7 (1751) y $06C8 (1736). Listado zona :

```
          $06D6  3E                  $06C7  FE
→ $06D7   1F                 → $06C8  1F
          $06d8  18                  $06C9  20
```

3- CAMBIO DEL CARACTER DE 'RETURN' o 'CR' AUTOMATICO de las teclas definibles (F1 a F10). Las teclas F1 a F10 , como todos sabeis permiten definir para cada una de ellas una instuccion o funcion; si finalizamos la definicion con el signo ↑ (GRPH+SFTLOCK) al pulsar dicha tecla se efectua un CARRIAGE RETURN automatico. Podemos cambiar el simbolo '↑' por cualquier otro simbolo (de Preferencia que no sea numero o Letra del alfabeto). La comprobacion de este simbolo se encuentra en la posicion de memoria $0811 (2065) y su valor inicial es $7F (127) (El corresp. a '↑'). Basta introducir en esta posicion el codigo Hex. o Ascii correspondiente . Asi si introducimos $86 (134) se entendera como retorno automatico el caracter "♥".

Listado zona :

```
          $0810  FE
→ $0811   7F
          $0812  CA
```

4- COMO ABRIR LA PUERTA DEL CASSETTE POR PROGRAMA ? Sencillo utilizando una rutina del monitor ya existente. En Monitor tenemos una rutina (OPEN) que abre el compartimento del cassette. Se puede llamar a dicha rutina desde BASIC con USR o desde 'Maquina' por una instr. CALL ('CD' en Hex). La direccion de OPEN es $048C (1164).

```
Llamada desde prog. BASIC =    USR ($048C) o USR (1164)
    ''   desde prog.  OBJ  =    CD  8C  04    CALL OPEN
```

Listado del inicio de la rutina :

```
$048C  3E  08  D3  E3  CD  17  05 ....
```

5- PARO DEL MOTOR DEL CASSETTE POR PROGRAMA : El bobinado y eebobinado rapido del cassette es programable en BASIC (u OBJ) pero en BASIC no existe el paro del motor del cassette , de forma que si se ejecuta una instr. 'FAST' (Bobinado) o 'REW' (Rebobinado) el cassette se pone en funcionamiento y no para hasta el final de cinta.

Para conseguir un control logico (Hasta el punto en que el cassette lo permite , claro) de la cinta es imprescindile poder parar el cass. en un punto determinado. Tenemos tambien una rutina en Monitor para detener el motor del cassette : es la rut. MSTOP (Motor Stop) localizada en la direccion $04CE (1230). Como ya sabeis se puede llamar desde BASIC u OBJ.

Un ejemplo de utilizacion practica en BASIC :

```
10 REM ---- SALTO CASSETTE SEGUN UN BUCLE ----
20 FAST : REM * bobinado rapido de la cinta
30 FOR N=1 TO 3000:NEXT N : REM * Contador de tiempo de bobinado
40 USR ($04CE) : REM * Para el motor
```

Listado de comprobacion :

```
$04CE  3E  0D  D3  E3  3A  50  11  CB  DF ....
```

6- INHIBICION DE LA TECLA 'BREACK' : La posicion de la tecla 'BREACK' en la tabla de definicion del teclado es $0D67 (3431) el contenido inicial es $0B (11) (Codigo Hex. y ASCII de BREACK resp.) Introduciendo en esta direccion el valor $00 (0) la tecla 'BREACK' queda sin funcion.

7- OTRAS DIRECCIONES DE INTERES para los aficionados al 'lenguaje maquina. a) RUTINAS MONITOR :

-CHR80 (Pone CRT a modo 80 car/linea) equivale a CONSOLE C80 en BASIC. Direccion $0CA6 (3078). Listado zona :

```
$0CA6  3E  10  32  2E  01 ...
```

-CHR40 (Pone CRT a modo 40 car/linea) equiv. a CONSOLE C40. Direccion $0D18 (3352). Listado zona :

```
$0D18  3E  08  32  2E  01  21  36  02 ...
```

-SCRSET (define zona de scroll) Equiv. a CONSOLE S a,b .Debe de introducirse antes de llamar a la rutina , direcc. $0922 (2338), en la direcc. $000B (0011) la linea de inicio scroll y en $000C (0012) la linea de fin de zona de scroll. Listado zona :

```
$0922  F5  C5  D5  E5  3A  0B  00 ...
```

-FFWD (Pone cassette en modo Fast foward) equiv. a FAST (BASIC). la direccion es $04E9 (1257).Listado zona :

```
$04E9  CD  71  05  CD  DA  04 ...
```

-SHORT (Rutina de Delay por corto espacio de tiempo) la direccion es $051D (1309).listado parcial :

```
$051D  F5  3E  0F  D3  E3 ...
```

-LONG (Rutina de Delay por espacio largo de tiempo) Direccion $0539 (1337).Listado parcial :

```
$0539  F5  3E  0F  D3  E3  3E  5A...
```

b) Direcciones importantes :

-DSPXY (Contiene la posicion actual del cursor en el eje horizontal) la direccion es $1151 (4433). la direccion siguiente (DSPXY se carga en reg. de 16 bits) que forma parte de DSPXY nos da la posicion en el eje vertical. Direcc. $1152 (4434). se puede cargar en las direcciones indicadas , por programa, la posicion en que deseemos que aparezca el cursor antes de una entrada (Como si en BASIC utilizaramos la instruccion CURSOR x,y o bien leer los datos contenidos en dichas direcciones para comprobar en que posicion se encuentra el cursor.

Estas direcciones y rutinas OBJ complementan a las que se describe en el manual : MZ 80 B - MONITOR SB - 1510.

Jose Maria Vidal Lacasa

No sucumbas bajo los encantos de tu máquina, haz un esfuerzo y prueba otros encantos más veraniegos, pero si el vicio es tan grande que no te es posible, cogete estas páginas y aumenta su poder seductor con estos trucos... ¡el amor!

Las teclas reservables

Las teclas reservables del ordenador de bolsillo SHARP PC 1211 pueden ser muy útiles y facilitar la programación, a condición de respetar algunos puntos:

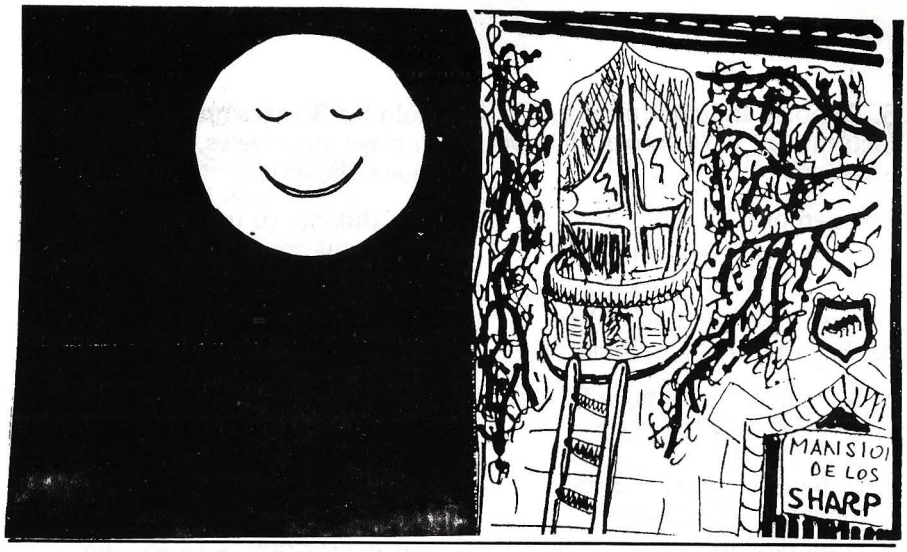
- La secuencia de instrucciones reservada debe ser superior a dos pulsaciones de teclas. Por ejemplo, es inútil reservar una tecla para la instrucción /PRINT/, ya que para acceder a ella, habrá que pulsar (SHFT/ /A/, cuando hubiera sido suficiente con /P/ /./. En cambio, sí podrá reservarse /PRINT "/ que necesita tres pulsaciones, o /GOSUB/ que necesita por lo menos cuatro (/G/ /O/ /S/ /./).

- Reservar sólo las secuencias de teclas muy utilizadas. Es inútil sobrecargar la memoria de reserva con instrucciones que luego sólo se usan de forma muy excepcional.

- Si no se utiliza la plantilla proporcionada, tratar de hacer coincidir la denominación de la tecla con la función que llama. Ejemplos:

/SHFT/ /F/ para /FOR W=1 TO/
/SHFT/ /G/ para /GOSUB/
/SHFT/ /N/ para /NEXT W/.
Etc. . .

Esto facilitará la utilización y la memorización de las teclas reservadas.



Subprogramas para el Sharp PC 1211

Proponemos aquí dos métodos para almacenar y volver a llamar subprogramas sobre cassette.

1) La instrucción CLOAD 1.

Aunque desconocida en el manual SHARP, funciona muy bien. Permite cargar un programa (o un subprograma en nuestro caso) detrás de otro que ya está en la memoria del ordenador.

Sin embargo, es importante señalar que el segundo programa siempre es cargado en memoria después del final del primero, y eso, cualesquiera que sean sus números de líneas.

Ejemplo:

El PC 1211 contiene:	10: INPUT "SU NOMBRE POR FAVOR?";A$ 20: BEEP 3 30: PRINT "BUENOS DIAS"; A$. 40: END.
Se carga (COAD 1) el programa:	20: B$ = "BUENAS TARDES" 30: PRINT B$,A$. 40: END.
El PC 1211 contendrá:	10: INPUT "SU NOMBRE POR FAVOR?"; A$. 20: BEEP 3. 30: PRINT "BUENOS DIAS": A$. 40: END. 20: B$: "BUENAS TARDES". 30: PRINT B$, A$. 40: END.

Por consiguiente, es necesario tomar un mínimo de precauciones para evitar problemas. Como por ejemplo, seguir las recomendaciones del manual Sharp que aconseja numerar los subprogramas con líneas 500 y siguientes. . .

2) El modo "RESERVE".

Otra forma de volver a llamar subprogramas sobre cassettes consiste en utilizar el modo "RESERVE". Método que hay que seguir:

a) Registrar los subprogramas en modo "RESERVE".

b) Durante la programación, en cuanto se necesite escribir un subprograma que ya esté en biblioteca, pasar al modo "RESERVE" y cargar el subprograma deseado a partir del cassette,

c) Volver al modo "PROGRAM" teclear el número de línea en que se quiera colocar el subprograma.

d) Pulsar /SHFT/ y luego la tecla reservada que corresponda al subprograma deseado. Este aparecerá en la línea elegida.

e) Pulsar /ENTER/ y ¡Ya está!.

NOTA: Estando la memoria limitada a 48 pasos en modo "RESERVE", sólo podrán utilizarse de esta forma los programas o subprogramas de longitud inferior a 47 pasos.

Nada impide registrar de forma simultánea varios

SE COMERCIALIZA EN ESPAÑA LA GAMA DE IMPRESORAS RITEMAN

■ Coincidiendo con el auge de la microinformática en nuestro país, son varias las novedades que han aparecido en el mercado español en dicho campo.

■ Sin duda alguna cabe destacar en la línea de periféricos, la aparición de la gama de impresoras RITEMAN, de reconocido prestigio en el resto de Europa, así como en el mercado americano.

■ La gama ofertada es suficientemente amplia, abarcando impresoras conectables a la mayoría de los ordenadores actuales desde los MSX hasta los IBM PC's y compatibles, en conexión paralelo centronic, opcional serie RS 232C.

■ Como características sobresalientes merecen ser tenidas en cuenta las altas prestaciones, pese al reducido tamaño físico externo, además de incorporar los últimos avances técnicos, que sólo equipan impresoras de mucho mayor precio. Y es en el precio donde se aprecia su competitividad, pues aún existiendo más baratas, ninguna consigue mejorar la relación precio/prestaciones conseguidas por RITEMAN, razón por la que se ha logrado introducir con tanta facilidad en nuestro país.

■ Se ofrecen con toda gama de accesorios necesaria, recambios, cables conexión a ordenadores, cintas autoretintadas, etc. El período de garantía es de 6 meses con posterior servicio postventa.

■ Son representadas en España por la firma DATAMON (conocidos por ser fabricantes de monitores), con domicilio en Provenza 385 de Barcelona, tel. (93) 207 24 99, y distribuidos a través de las mejores empresas comerciales especializadas.

■ Como era de esperar la disponibilidad de este tipo de impresoras ha hecho decidir sin dudar a los usuarios de ordenadores personales que esperaban la definitiva oferta de periféricos avanzados, en cuyo caso la espera ha valido la pena: RITEMAN.

subprogramas utilizados a menudo juntos, siempre que su longitud total sea compatible con la capacidad de la memoria de reserva.

Ejemplos de subprogramas:

REDONDEO AUTOMATICO AL CENTIMO

Z = INT /Z* 100 + 5) / 100

GENERADOR DE NUMEROS ALEATORIOS:

Z = Z + π : Z = ZZZZZ : Z = Z - INT Z.

Simplificaciones tiempo-espacio sobre el PC 1211

He aquí todos los pequeños trucos que he podido descubrir en mi PC-1211. El mejor método para optimizar un programa consiste, por supuesto, en optimizar el organigrama, pero si con este método se ganan bastantes pasos, la verdad es que se pierde mucho desde el punto de vista de la claridad.

1. Hay que "condensar" al máximo las líneas (se ganan 2 pasos en cada línea).

2. Utilizar al máximo la tecla exponente. Por ejemplo:
100 se escribe E2.
5000 se escribe 5E3.

3. Utilizar siempre la forma directa sin emplear el signo de multiplicación que siempre es prioritario. Por ejemplo:

INT (6* X) se escribe INT 6X.
3/(100*X) se escribe 3/100X.
O mejor aún 3/E2X.

4. Los paréntesis son facultativos al final de una instrucción (antes de los 2 puntos) o al final de una línea.

Por ejemplo:

10: INPUT A,X : W = COS (2X / (3 + A) + 10 : END.

Se escribe:

10 : INPUT A,X : W = 10+COS (2X/ (3 + A : END.

Por lo tanto, habrá que arreglarse para agrupar los paréntesis al final (en el ejemplo, poner el 10 delante) y suprimir todos los paréntesis delante de dos puntos.

5. También las comillas son facultativas, pero sólo al final de una línea.

Por ejemplo:

10 : PRINT " PC-1211.

Tener cuidado para no dejar "space" invisible que utiliza pasos después del mensaje.

6. A nivel de los tests, la forma:
10 : IF (A = 2)*(B = 3)
LET C = 4.
(Función si A=2 y B=3 entonces (C=4) puede ser reemplazada por:
10 : IF A = 2 IF B = 3
LET C = 4.
lo que hace ganar 4 pasos.

7. Por último, el "LET" es facultativo y el END es, en general, inútil. Pero hay que tener cuidado con los programas que pueden mezclarse.

8. Otro truco, más sutil pero muy interesante para el número de pasos y la velocidad de ejecución de los bucles: Si hay que llamar muchas veces un valor contenido en una tabla, se pueden ganar pasos colocando este valor en una memoria "borrador", por ejemplo:

10 : FOR W = 1 TO 10: A(W) = INT 10A(W) + LOG A(W) + COS (1/A(W)) + A(W):NEXT W

se podrá escribir:

10 : FOR W = 1 TO 10 : X = A(W):A(W)= INT 10 X+ LOG X + COS (1/X) + X : NEXT W.

y se ganarán 9 pasos.

9. Poner los subprogramas al principio de los programas: Se ganan mucho tiempo y bastantes pasos; GOSUB 5 utiliza 2 pasos menos que GOSUB 500 (se ganan 2 pasos cada vez que es llamado el subprograma).

10. Por último, utilizando al máximo W, X, Y, Z, se puede ganar hasta el 5%/o del tiempo de ejecución.

Or y And en el PC 1211 y en el TRS 80 P

Son muchos los usuarios que no saben que las instrucciones OR y AND existen. Y sin embargo, las posibilidades de estas dos funciones son amplias y sólo los 80 caracteres por línea limitan sus niveles, lo que es más que suficiente.

Cuidado, su utilización no es exactamente igual que tratándose de los OR y AND clásicos.

Aquí se utiliza el hecho que:

Verdadero = 1 Falso = 0

Ejemplo: En la mayoría de las máquinas se escribe:
IF A > 6 AND A < 9 THEN. . .

En el TRS 80P, esto nos da:
IF (A > 6) + (A < 9) = 2 THEN. . .

Otro ejemplo:
IF A =12 OR B = 12
da en el TRS 80P: IF (A = 12)* (B = 12) = 1 THEN.

También pueden utilizarse las funciones lógicas en los cálculos X = C* (A < = 20) + B quiere decir que si A es mayor que 20, X será igual a B, sino X será igual a C + B.

Este método permite ahorrar numerosos pasos de programas.

Números aleatorios

Este generador de número seudo-aleatorios se basa en la conocida fórmula (manual de aplicaciones Sharp):

$U_n = FRAC\ (U_{n-1} + \pi)^5$

La fórmula ha sido adaptada al dialecto BASIC del ordenador de bolsillo Sharp PC 1211 (y TRS 80 Pocket).

— Primera versión:

a) V = (U + π) ^ 5: U = V - INT V (16 pasos).

Una primera mejora consiste en utilizar una sola variable perfectamente entre W y Z. (Acceso más rápido para el ordenador). Además, una elevación a la quinta potencia tarda más tiempo que cinco multiplicaciones, ya que el ordenador debe pasar por los logaritmos.

De ahí, segunda versión:

b) Z = Z + π : Z = ZZZZZ;
Z = Z - INT Z
(20 pasos)

— Cotejo de los resultados:

Han sido probadas las dos versiones gracias al programa:

```
10: FOR W = 1 TO 100
20: ..... (primera o segunda
    versión).
30: NEXT W.
40: END.
```

Cada una de las dos versiones ha sido introducida en la línea 20, y luego, al cabo de un minuto de funcionamiento, se ha interrumpido el programa (BREAK). Después de esto se ha apuntado el valor de W.

Resultados:

VERSION	1	2
Número de pasos	16	20
Número de bucles en 1 mn	61	71

Utilización práctica: Un generador de números seudo-aleatorios es algo muy bonito pero, ¿Para qué sirve? y sobre todo, ¿Cómo se utiliza? Pues bien, sirve sencillamente para simular el azar. Que sea para juegos, estadísticas, etc. . ., a menudo se necesitaban números sacados al azar.

El método antes descrito da un número comprendido entre 0 y 1. (Número decimal tipo: 0,6537. . .) A partir de este valor es fácil obtener números comprendidos en cualquier margen.

Para ello, es suficiente con reemplazar, en la línea de BASIC siguiente las letras por unos valores escogidos:

```
Y = A + INT (B*Z).
```

Y: Número aleatorio comprendido entre A y A+B - 1 (entero).
A: Límite inferior del margen elegido.
B: Amplitud del margen (número de términos distintos deseados).
Z: Número aleatorio comprendido entre 0 y 1 (forma decimal).
Ejemplos:

Juego de dados Y = 1 + INT (6*Z)	dará un número entero comprendido entre 1 y 6 (o sea 1 + 6 - 1).
Juego de loto Y = 1 + INT (49*Z)	dará un número entero comprendido entre 1 y 49.
Siglo Veinte Y = 1900 + INT (100*Z)	dará un año comprendido entre 1900 y 1999. . .

EXPRIMA SU SPECTRUM

Termina de destruir las naves enemigas, haz un RESET y ponte manos a la obra con estos trucos que sin duda darán más vida a tus programas y te permitirán aquello que parecía imposible. ¡Puesto a exprimir, que tal unos limones con hielo granizado!

Todo para simular la instrucción "scroll" del ZX81

El ZX Spectrum dispone de un sistema propio para ejecutar scroll en pantalla, normalmente deteniéndose con el informe **scroll**? Si se incluye una instrucción POKE 23692,255 antes de que se llene la pantalla, el programa no se detiene y el scroll se ejecuta automáticamente. En la dirección 23692 hay un contador de scroll (variable del sistema SCR CT) con el número de scrolls a ejecutar más uno.

El ZX81 posee una instrucción SCROLL cuyo efecto es el de desplazar un renglón arriba el texto de la pantalla, aunque esta no se haya llenado, y pasar el último renglón la dirección de impresión. Aunque las posibilidades del ZX Spectrum son superiores, puede interesar disponer de una rutina que haga en éste lo mismo que la instrucción SCROLL hace en el ZX81, particularmente a la hora de adaptar al Spectrum un programa del ZX81. La siguiente rutina se encarga de esta tarea:

```
1000  REM rutina de scroll
1010  POKE 23692,2
1020  PRINT AT 21,0: PRINT
1030  PRINT AT 21,0;
1040  RETURN
```

Para llamarla vale la instrucción GO SUB 1000. Puede probarse añadiendo el siguiente programa:

```
 10   REM "scroll"
 90   LET n =0
100   GO SUB 1000
105   PRINT n
110   LET n =n +1
120   GO TO 100
```

La cantidad depositada en el controlador de scroll por la rutina es 2 en lugar de 255 para que la similitud con el SCROLL del ZX81 sea mayor: sólo se realizará un scroll por cada llamada a la rutina.

UN METODO MAS SIMPLE:

La rutina de la ROM del Spectrum que ejecuta el scroll se encuentra en la dirección (decimal) 3582, de modo que basta una llamada a dicha dirección para obtener el resultado deseado. Por ejemplo:

IF USR 3582 THEN REM

Pero la instrucción SCROLL del ZX81 tiene además el efecto de transferir la posición de impresión a la fila nº 21 de la pantalla. Esto puede conseguirse añadiendo en el Spectrum la instrucción.

PRINT AT 21,0

El efecto combinado de ambas instrucciones puede obtenerse poniendo, por ejemplo:

IF USR 3582 THEN PRINT AT 21,0

que ya es genuinamente equivalente al SCROLL del ZX81.

Miguel A. Lerma.

¿Varios colores en el borde?

Cuando alguien dice que su micro-ordenador no puede hacer algo, en un 99% de las veces, se esta equivocando.

He aquí un método para obtener al mismo tiempo varios colores en el BORDE de la pantalla de Spectrum. Este efecto, se puede utilizar junto con algún otro sonoro en nuestros juegos, por ejemplo en choques o disparos. Tome nota:

¡Como ven todo es BASIC! Cuando se ejecuta aparecen unas franjas negras en la parte superior e inferior de la pantalla delimitando la pila de franjas de distintos colores. Si elimina la línea 110 las franjas ya no permeneceran inmóviles, consiguiendo el efecto mencionado. Con más de ocho bandas el efecto no es tan puro; experimente con sus colores preferidos.

J.M.

```
  1 PAPER 7: INK 0: BORDER 7: C
LS
  5 PRINT AT 10,6;"el ORDENADOR
 PERSONAL"
 10 BORDER 7
 20 BORDER 1
 30 BORDER 2
 40 BORDER 3
 50 BORDER 4
 60 BORDER 5
 70 BORDER 6
 80 BORDER 7
 90 BORDER 0
110 PAUSE 1
120 GO TO 10
```

Ahorrando conectores

Cuando deseemos hacer un Reset en nuestro Spectrum, hay otra posibilidad aparte de desenchufar y enchufar la fuente de alimentación (siempre y cuando tengamos el control por el teclado) y es introducir:

RANDOMIZE USR 0

Lo cual limpia la memoria, anula los Pokes y restablece el RAM TOP, pero sobre todo, alarga la vida del conector de alimentación.

J.M.

¿Print AT 23.0...?

Es posible "imprimir mensajes en la línea 23 de la pantalla, donde normalmente aparecen los textos informativos mediante PRINT # 1: pero debemos situar un sistema de pausa pues si no inmediatamente obtendríamos el informe de ejecución probar la siguiente línea:

```
100! PRINT # 1; "SE PUEDE
   ESCRIBIR AQUI": PAUSE
   4e4
```

J.M.

¿Cuánta memoria queda?

Si desea saber el número de bytes (octetos) libres en su Spectrum de 16 ó 48 K.

Teclee:

```
PRINT 65561-USR 7962
```

También si Ud. desea que su programa se adapte a la memoria del equipo en que se carga puede usar el PEEK 23733, que dará 127 si el equipo tiene 16K. ó 255 para 48 K. de tal manera que por ejemplo en una aventura creemos 100 ó 400 habitaciones según la capacidad del equipo en el que este trabajando el programa.

J.M.

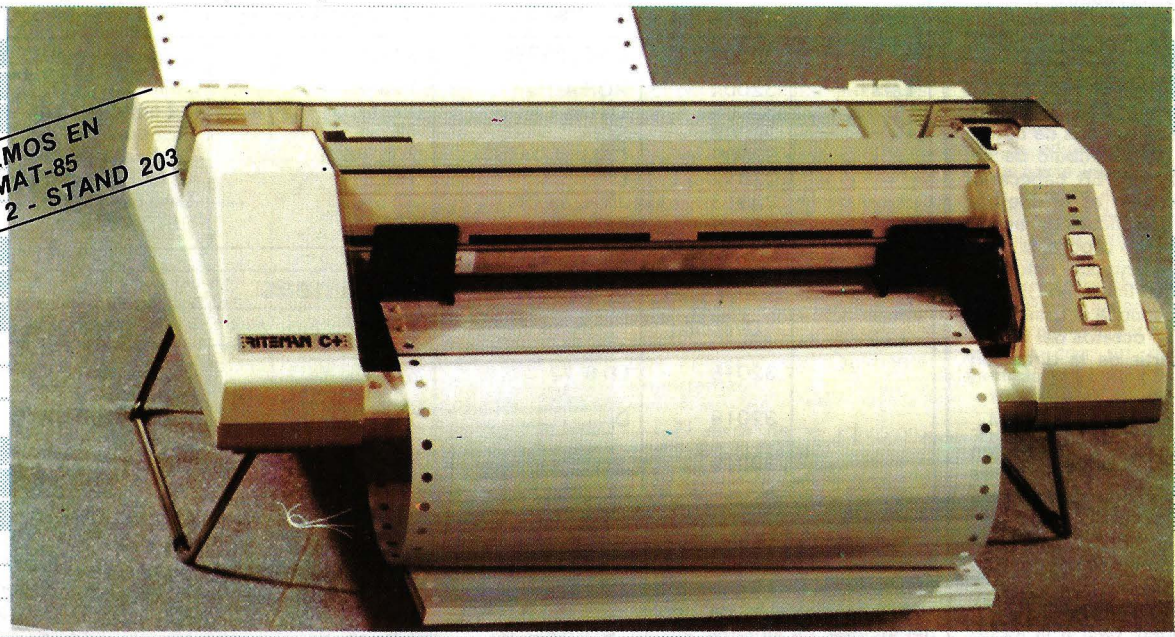
:RITEMAN:

PERSONAL/BUSINESS PRINTER

AMPLIA GAMA

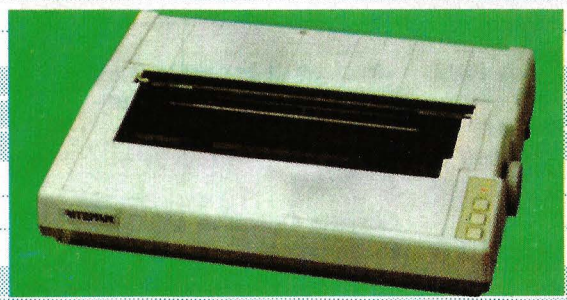
Nuevas impresoras modelos F+ y C+, sin rodillo alimentación horizontal, impresión vertical, tracción y fricción desde 4 a 10", bidireccional optimizada velocidad 105 cps. con soportes de elevación.

ESTAREMOS EN INFORMAT-85 NIVEL 2 - STAND 203

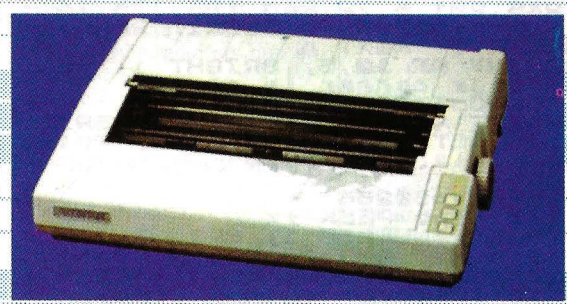


RITEMAN F+: Interface Paralelo Centronics, 2K buffer NLQ **P.V.P. 69.000 pts.**
RITEMAN C+: Especial directa a COMMODORE (cable inc.) **P.V.P. 67.000 pts.**

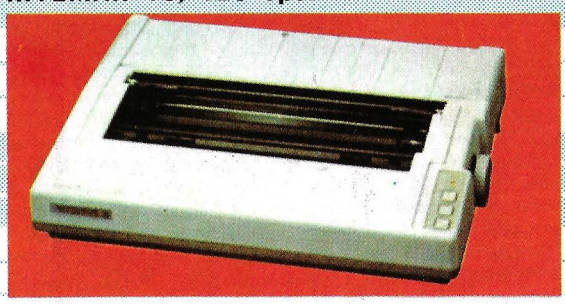
Otros modelos RITEMAN en 80 y 136 columnas, velocidad 120, 140, 160 cps.



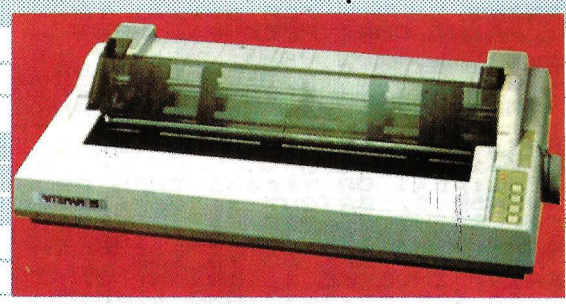
RITEMAN 10, 120 cps. P.V.P. 81.000



RITEMAN 10-IBM, 140 cps. P.V.P. 85.000



RITEMAN 10-II 160 cps. P.V.P. 93.000



RITEMAN 15 160 cps. P.V.P. 155.000

DE VENTA EN LOS MEJORES ESTABLECIMIENTOS ESPECIALIZADOS

DATAMON

DATAMON, S. A.

PROVENZA, 385-387, 6.º, 1.ª
TELÉFONO (93) 207 27 04

REPRESENTACION EN ESPAÑA DE:

:RITEMAN:

-IMPRESORAS PROFESIONALES-

08025 - BARCELONA

* MAYORES PRESTACIONES
* MENOR TAMAÑO
* MEJOR PRECIO

Decodificador de cabeceras

Las grabaciones en cassettes en el ZX Spectrum son más fiables, más rápidas y más diversificadas que en su predecesor, el ZX 81.

Son posibles cuatro tipos de grabaciones (save):

0 = tabla numérica
2 = tabla de caracteres
3 = bloque de octetos.

Previamente a la grabación de estos cuatro tipos de datos, se graba un preámbulo de 17 octetos (o HEADER: cabecera). Este preámbulo es el que vamos a descodificar con ayuda de un pequeño programa mixto Basic + lenguaje máquina, en este caso figura en 32000, pone a la escucha el cassette y pone los 17 octetos del «HEADER» a partir de la dirección 32256.

El primer octeto permite determinar el tipo de grabación que se registra con los valores 0, 1, 2 ó 3 que corresponden a los tipos precitados. Los diez octetos siguientes corresponden a las 10 letras del nombre de la grabación.

Los dos octetos que siguen corresponden, bien a la longitud del programa más las variables, bien a la longitud de las tablas o bien a la longitud del bloque de octetos.

Después vienen dos octetos que corresponden según los cuatro tipos:

- al número de línea en que se autolanzará un programa Basic, tras su carga;
- al nombre de la tabla (numérico o alfanumérico) salvaguardado;
- a la dirección de origen del bloque de octetos salvaguardado.

Los dos últimos octetos corresponden, para el primer tipo de grabación, a la longitud del programa Basic grabado, sin la longitud de las variables.

La segunda rutina en lenguaje máquina, en este caso implantada en 32016 sirve para copiar en la impresora el contenido que presenta la pantalla en las nueve primeras líneas.

Este programa, una vez cargado y tras un RUN, le permitirá realizar un sumario de los valiosos registros que haya grabado en cassettes.

En los dos recuadros se ve el desarrollo de las dos rutinas en lenguaje máquina.

Escucha del cassette

Etiqueta	Dirección	Nemonicos	Códigos decimales	Comentarios
	3200	LD IX, 32256	221, 33, 0, 126	Destination de 17 octets
Lect	32004	XOR A	175	
	32005	LD DE, 17	17, 17, 0	17 octetos a leer
	32008	SCF	55	
	32009	CALL «LOAD»	205, 86, 5	Llamadas a LOAD
	32012	CP D	186	Es un HEADER?
	32013	JR NZ LECT	32, 245	Sino LECTURA
	32015	RET	201	RETORNO

Copia de las 9 líneas

	Dirección	Nemonicos	Códigos decimales	Comentarios
	32016	LD B, 72	6, 72	9 veces 8 mimileas a copiar
	32018	DI	243	Suspender las interrupciones
	32019	JP «CPY»	195, 175, 14	Llamada a COPY

```
  10 CLEAR 31999
  20 DATA 221,33,0,126,175
  30 DATA 17,17,0,55,205,86,5,18
6,32,245,201,6,72,243,195,175,14
  40 FOR n=32000 TO 32021: READ
a: POKE n,a: NEXT n
  50 PRINT AT 10,5; BRIGHT 1; FL
ASH 1;"Deje correr la cinta"
  60 RANDOMIZE USR 32000
  61 PRINT AT 10,5; FLASH 1; BRI
GHT 1;"  DETENGA  ";AT 14,3;"Pul
se una tecla": PAUSE 0
  65 CLS
  70 LET ix=32256
  80 LET tipo=PEEK ix
  90 PRINT INVERSE 1; BRIGHT 1; (
"Programa : " AND tipo=0)+("Tabl
a numerica : " AND tipo=1)+("Tab
la de caracteres : " AND tipo=2)
+("Bloque de bytes : " AND tipo=
3);
 100 PRINT "  ";: FOR n=ix+1 TO
ix+10: PRINT CHR$ PEEK n;: NEXT
n
 110 PRINT AT 2,0; INVERSE 1; BR
IGHT 1;"Long. "+("Prog. + Variab
les :" AND tipo=0)+("Code : " AN
D tipo);
 120 PRINT " ";PEEK (ix+11)+256*
PEEK (ix+12);AT 5,0;
 130 IF tipo=1 OR tipo=2 THEN PR
INT INVERSE 1; BRIGHT 1;"Variabl
es : ";: PRINT "  ";CHR$ (PEEK (
ix+14)-32-64*(PEEK (ix+14)>192))
+("$" AND tipo=2): GO TO 160
 140 PRINT INVERSE 1; BRIGHT 1; (
"Linea de comienzo :" AND tipo=0
)+("Comienzo del bloque : " AND
tipo=3);
 150 PRINT "  ";PEEK (ix+13)+256
*PEEK (ix+14)
 160 PRINT AT 7,0; INVERSE 1; BR
IGHT 1;("Longitud del Programa :
" AND tipo=0);
 170 PRINT "  ";PEEK (ix+15)+256
*PEEK (ix+16)
 175 PRINT "                +++++++++
+++
 180 RANDOMIZE USR 32016
 200 GO TO 50
```

Linea de programa imborrable

Una de las pocas satisfacciones que uno tiene tras haber desallado trabajosamente un programa, después de haber empleado bastantes horas o días en teclearlo es poder colocar una línea parecida a la siguiente:

```
10 REM © Fulano De Tal
```

Pero es bien sabido que esta tarjeta de identidad es fácil de borrar por ese «buen» amigo al que le hemos dado una copia, simplemente introduciendo 10 y ENTER. Lo que quizás Ud., necesite es un método para insertar líneas en el listado que no se eliminen con tanta facilidad.

Uno de los métodos que podemos emplear es utilizar la variable del sistema NXTLIN situada en las posiciones 23.637-8 donde se almacena la dirección de comienzo de la siguiente línea de programa. El manual del spectrum nos dice que cada línea de programa comienza con el número de línea almacenado en dos bytes en el siguiente orden: byte más significativo (MSB) seguido del byte menos significativo (LSB); con lo cual la línea uno tendrá 0,1 y la línea 258 será 1,2 (1 x 256 + 2). Así que pokeando un cero en ambos bytes obtenemos nuestro objetivo de una línea de programa virtualmente imborrable.

He aquí como hacer esa línea en BASIC:

Ejecute (RUN) el programa y listelo (LIST). La primera línea a puesto un cero donde antes teníamos un 2, verá también que las líneas no se han clasificado en su orden correcto. La clasificación sólo se efectua cuando introducimos una línea en un programa y no en este caso.

La línea 1 ya no es necesaria puede eliminarla de forma normal. Si Ud., quiere usar esta línea en todos sus programas puede grabarla y unirla (MERGE) en ellos. También puede salvar la rutina anterior en una cinta con SAVE «Copyright» LINE 1. Lo cual hará que se autoejecute y solamente tendrá que borrar la línea 1 cuando comience un programa. Cuando cree esta línea cero de Copyright, puede aprovechar para hacerlo con PAPER blando, tinta (INK) negra y FLASH u otro que sean llamativos.

J. M.

```
  1 LET l=PEEK 23637+256*PEEK 2
3638: POKE l,0: POKE l+1,0: STOP

  2 REM © EL ORDENADOR PERSONAL
          Justo Maurin

0>REM © EL ORDENADOR PERSONAL
          Justo Maurin
```

Algunos PEEKS y POKES útiles

Para todos aquellos que desean añadir un toque de profesionalidad y sofisticación a sus programas un análisis detenido del capítulo 25 del manual (las variables del sistema), les puede suministrar información útil. Como ejemplo esta relación de direcciones y su uso práctico.

POKE 23561, n.— (REPDEL) Tiempo que debe estar una tecla pulsada para que se produzca la repetición su valor normalmente es 35 pero puede ser cambiado (POKE) por 255 anula prácticamente la autorrepetición.

* POKE 23.609, n.— Duración del clip del Teclado, aumentándolo, el valor de n alrededor de 40, optenemos un pitido más audible.

* POKE 23.692, n.— Contador de Scroll. Si introducimos (POKE) en n el valor 255, la pantalla realiza el Scroll sin consultarle.

Compresor de programas

Los programas Compresores son aquellos cuya misión es reducir por diversas técnicas el espacio que ocupan otros programas BASIC en memoria. Tratando de evitar la desagradable sensación que produce el mensaje **Memoria completa**—. Los hay de varios tipos algunos eliminan las sentencias REM y partes no ejecutables de los programas otros como el aquí expuesto aprovechan las características de almacenamiento de la propia maquina (el ZX Spectrum en este caso).

La subrutina que se da a continuación comprime la mayoria de los programas en BASIC, pudiendo llegar a reducir un programa común al 15% de su tamaño original dependiendo de la cantidad de números que haya en el programa a reducir. Actua cambiando cada número del programa original en su equivalente VAL; por ejemplo el número 125 se cambiaria por VAL «125» de esta forma cada número es almacenado en tres bytes, los necesarios para el CHR$ 14 y los siguientes 5 bytes son eliminados (recuperados para memoria libre).

Como el programa va comprimiendo toda la memoria hasta Eline es movida hacia abajo cada vez que un número se comprime. Se alteran también las variables del sistema VARS y ELINE. Las variables de su programa no son alteradas pero es preferible realizar un CLEAR antes de llamar a esta rutina.

La rutina esta compuesta por los numeros almacenados en las datas que pueden introducirse con cualquier otro cargador. La forma del programa biene dada para evitar pérdidas de tiempo en caso de que nos equivoquemos al introducir un número, introduzca el programa y sálvelo antes de ejecutarlo, una vez grabado lo ejecutaremos habra que contestar afirmativamente varias veces a la pregunta de SCROLL. Cuando finalice el programa (con 0,0 OK) de nuevo salvaremos la rutina con SAVE «COMPRESOR» CODE dirección, 122. Antes de ejecutarla por la misma razón que antes ya que podiamos obtener un CRACK. Borraremos todas las líneas y podremos ejecutarla cuando tengamos un programa en Basic en memoria mediante: RAND USR la dirección (en la que lo situamos).

J.M.

```
  5 REM COMPRESOR
 10 INPUT "direccion ?";dir
 20 FOR d=dir TO dir+121
 30 READ b
 40 PRINT d,b: POKE d,b
 50 NEXT d
 60 DATA 42,83,92,43,237,75,75,
92,35,167,237,66,9,200,35

 70 DATA 35,78,35,70,229,35,126
,254,13,32,3,209,24,231,254

 80 DATA 14,32,243,209,213,11,1
1,11,197,120,18,27,121,18,229

 90 DATA 43,126,254,47,40,19,25
4,46,56,15,254,196,40,4,254

100 DATA 58,48,7,35,35,119,43,4
3,24,231,35,54,176,35,54

110 DATA 34,225,35,35,54,34,35,
84,93,213,35,35,35,229,42

120 DATA 89,92,167,237,82,68,77
,225,237,176,42,75,92,43,43

130 DATA 43,34,75,92,42,89,92,4
3,43,43,34,89,92,225,193,24,155,
0,0,0
```

Los errores del Spectrum

Como ya hemos comentado en anteriores ocasiones, cuando uno lleva algún tiempo trabajando en su O.P. descubre que a veces pasan cosas que no deberían ocurrir. Esto es debido en la mayoría de las ocasiones a los «bugs» o errores en la ROM.

El objeto de estas líneas, es poner en su conocimiento la existencia de estos errores para que los evite al hacer sus programas especialmente si requieren precisión matemática.

El programa monitor de 16K es excelente, pero tiene algunos errores. La siguiente lista detalla doce de dichos errores, de los cuales sólo los dos primeros son realmente importantes.

i. El error de la «división» (Dr. Frank O'Hara).

La posición 32ØØh debería contener DAhen lugar de Elh. Este error en la rutina de división conduce, por ejemplo, a lo siguiente:

Ø.5 tiene la forma F-P: 7F 7F FF FF FF

pero 1/2 tiene la forma F-P: 8Ø ØØ ØØ ØØ ØØ

ii. El error del número «–65536» (Dr. Ian Logan).

En el programa monitor hay un fallo que tiene que ver con este número. En algunas ocasiones se toma como «ØØ FF ØØ ØØ ØØ» mientras que en otras adopta su forma F-P completa.

El mejor ejemplo de este error es el siguiente:

PRINT INT –65536 que da –1

iii. La subrutina «program name».

La subrutina que se encuentra entre las posiciones Ø4AAh y Ø4C1h se aplica al ZX81, y debería haber sido borrada.

iv. El error de «CHR$ 9».

En la rutina PRINT-OUTPUT hay una sección para manejar «CHR$ 9» (cursor a la derecha). Sin embargo el programador olvidó almacenar la nueva posición de impresión, de modo que «CHR$ 9» sólo funciona si la siguiente impresión se hace en un lugar nuevamente definido. Por ejemplo:

PRIN PAPER 2; CHR$ 9; AT 4,Ø;

funciona, pero no sirve de mucho.

v. El error de «scroll?». (También se aplica a «start tape...»).

No es posible responder a un informe con CAPS LOCK, shift y GRAFICS o shift y SYMBOL SHIFT sin que la línea de edición previa se copie en la parte inferior de la pantalla. El error está en la rutina KEYBOARD-INPUT, que no reconoce la situación de «informe».

vi. El error del «cursor de línea» (Paul Harrison).

Es posible obtener una línea editada con un cursor. Por ejemplo, introducir:

1ØØ PRINT (ENTER)
1Ø1 (ENTER)
Shift y EDIT

Aparecerá un cursor en la línea editada porque el número de dicha línea más uno es igual al número de la «línea en curso». El error está en la subrutina de impresión de una línea BASIC.

vii. El error del «espacio delantero».

Hay una inconsistencia en la impresión de espacios antes de señales («tokens»). Por ejemplo:

PRINT CHR$ 255; CHR$ 13; CHR$ 255

incluye el espacio la primera vez, pero se omite en la segunda.

viii. El error del modo K (Chris Thornton).

Cuando el SPECTRUM está en modo K, se imprime una palabra-clave (keyword) cuando se presiona una tecla apropiada.

Desafortunadamente si se mantiene presionada la tecla, la palabra-clave se repite.

El error está en la subrutina «key repeat», que continúa suministrando el mismo código incluso después de que el modo ha cambiado a «L». La

SALTO EN MEDIO DE UNA LINEA

Es posible ir (GOTO) a una sentencia concreta, dentro de una línea multi-instrucción en el Spectrum. Para ello añada esta línea a su programa:

```
9999 POKE 23618,linea-256*INT (l
inea/256): POKE 23619,INT (linea
/256): POKE 23620,sentencia
```

Fije después las variables LINEA y SENTENCIA a las desea transladar el control, mediante línea de programa o de forma directa y realice un GOTO 9999.

Puede probar su funcionamiento, por ejemplo con lo siguiente:

```
 10 PRINT 1: PRINT 2: PRINT 3:
PRINT 4
 20 STOP
```

subrutina debería comprobar que el hit 3 de FLAGS no ha cambiado.

ix. El error de «CHR$ 8» (Dr. Frank O'Hara).

La posición ØA33h debería contener 19h en vez de 18h. El «retroespaciado» (cursor a izquierda) trabaja perfectamente mientras se usa en las líneas 1 a 21. Sin embargo no se puede usar para retroceder del principio de la línea 1 al final de la línea Ø cuando el programador ha usado el límite erróneo. El retroceso desde «Ø,Ø» lleva a varios resultados interesantes.

x. El error de «SCREEN$» (Stephen Kelly y otros).

La posición 257Dh debería contener C9h (RET) en vez de C3h(JP). Como consecuencia de este error, la cadena obtenida usando SCREEN$ se almacena dos veces. Esto se puede mostrar así:

```
1Ø PRINT «123»
2Ø PRINT SCREEN$ (Ø,Ø)+SCREEN$ (Ø,1)
```

lo que da la cadena «22».

Este error puede eludirse usando variables alfanuméricas temporales:

```
2Ø LET S$=SCREEN$ (Ø,Ø)
3Ø LET T$=SCREEN$ (Ø,1)
4Ø LET S$+T$
```

xi. El error de «STR$» (Tony Stratton).

Cuando se manejan números en el rango $-1<n<1$ la rutina PRINT-FP pone un cero extra en la pila del calculador, lo que da más resultados que operaciones. Por lo tanto:

PRINT «A»+STR$ Ø.1 se evalúa como PRINT ««+STR$ Ø.1 y PRINT 1+VAL STR$ Ø.1 como PRINT Ø+VAL STR$ Ø.1 etc.

De nuevo se puede evitar el error usando variables alfanuméricas temporales cuando se manejan parámetros de STR$ que puedan dar errores; o colocando STR$ antes de cualquier operador binario.

xii. El error de «CLOSE» (Martín Wren-Hilton).

Cualquier intento de cerrar (close) los flujos (streams) Ø4 a ØF sin haberlos abierto primero, conducirá a un «restart» del sistema (tal como un salto a la posición ØØØØ) o a la producción de un extraño informe.

La razón de este error está en que la tabla «CLOSE stream look-up» en la posición 1716h no termina con una marca de fin, como es costumbre poner al final de tal tabla.

MIGUEL A. LERMA

Adaptando programas de ZX 81 al Spectrum

Corrientemente se suele leer y oír que cualquier programa del ZX81 que no contega sentencias PEEK o POKE puede funcionar sin dificultad en el ZX Spectrum. Esto no es correcto (y su experiencia propia se lo puede haber descubierto), pues es necesario a la hora de adaptar un programa tener las siguientes precauciones:

• En primer lugar debemos repasar cuidadosamente el programa a adaptar para asegurarnos que no contiene ningún PEEK o POKE ya que los mapas de memoria de ambos son distintos y estas instrucciones no funcionarían de uno para otro.

• Un programa que ocupe cerca de los 16 K. En el ZX81 no cabra en un Spectrum de 16K. ya que, en este último la memoria realmente utíl son alrededor de 9 K. utilizando el resto para almacenar la información de pantalla y las variables del sistema.

• Acerca del FAST y el SLOW (del ZX81) deben ser en general ignorados, aunque los bucles FOR-NEXT utilizados para retardos en el ZX81 deben hacerse unas cuatro veces mayores en el Spectrum para obtener el mismo retraso.

• Las líneas de programa que usen CODE o CHRS deben ser estudiadas debido a las diferencias existentes entre ambos códigos, que suelen ser la principal causa de problemas. Por ejemplo los dígitos del 0 al 9 en el Spectrum tienen sus códigos entre el 48 y el 57 mientras que en el ZX81 están situados del 28 al 37; de igual forma, las letras mayúsculas de la A a la Z se situan en el ZX81 del código 38 al 63, mientras que en el Spectrum del 65 al 90. Como no hay una relación lógica o simple entre los dos juegos de códigos de los caracteres debemos irlos analizando y cambiando uno a uno con ayuda de la lista que adjuntamos a continuación.

• Por último el Scroll deberá ser generado en algunos casos como ya señalamos en un truco: anterior

J.M.

TABLA DE CONVERSION DE CODIGOS

ZX 81	Spectrum	Símbolo	ZX81	Spectrum	Símbolo
Ø	32	espacio	3	131	▗
1	130	▘	4	136	▖
2	129	▝	5	138	▌
6	137	▐	126-127	No Existen	
7	139	▄	128	143	▀
8 a 10	Necesitan ser definidos.		129	141	▚
			130	142	▞
11	34	"	131	140	▙
12	96	£	132	135	▟
13	36	$	133	133	▜
14	58	:	134	134	▛
15	63	?	135	139	▜
16-17	40-41	( )	136-138	Necesitan ser definidos (USR)	
18	62	>			
19	60	<	139-191	No Están	Caracteres inversos
20	61	=			
21	43	+	192	No Existe	
22	45	—	193	22	AT
23	42	*	194	23	TAB
24	47	/	196-211	175-190	
25	59	;	212-215	192-195	
26	44	,	216	94	↑
			217-221	197-201	
27	46	.	222-223	203 204	THEN TO
28-37	48-57	a 9	224	205	STEP
38-63	65-90	de A a Z	225	224	LPRINT
64	165	RND	226	225	LLIST
65	166	INKEY$	227	226	STOP
66	167	PI	228-229	No Existen	Fast/Slow
112	11	↑	230	230	NEW
113	10	↓	231	No Existe	
114	8	←	232-251	232-251	
115	9	→	252	N/E	
			253-255	253-255	
116	N/E	GRAPHICS			
117	7	EDIT			
118	13	ENTER			
119	12	ROBOUT			
120-121	No Existen				

Nota:

Los códigos del 67 al 111 y del 122 al 125 no son usados en el ZX81.

Anular las teclas BREAK

Seguramente a usted le gustaría poder eliminar la posibilidad de que alguien PARE la ejecución de su programa en BASIC y se dedique a hurgarlo por dentro. Puede conseguir desactivar las teclas de Break con POKE 23613, PEEK 23730-5, volviéndolo a activar con POKE 23613, PEEK 23730-3. Aunque si está seguro de que el RAMPTOP está en su dirección habitual (32599 en el modelo de 16 K. y 65367 en el 48 K.), puede cambiar el contenido de los POKEs a 82 y 84, respectivamente.

Este Poke trabaja alterando el byte bajo de la variable ERR-SP que contiene la dirección devuelta cuando ocurre un error. Normalmente ésta señala una rutina que termina la ejecución del programa. Pero el POKE fuerza a apuntar a una rutina que continúa la ejecución del programa. Desgraciadamente existen problemas en el uso de este método que quizás sean la razón por la que no se comenta en el manual.

El anterior sistema trabaja solamente en el 90 % de las veces. Debe también Pokear el contenido de la dirección 23614 y asegurarse que el salto en caso de error es restaurado a su posición normal antes de que termine el programa para evitar que la máquina realice un RESET por sí misma.

Este POKE salta todos los errores menos el «Nonsense in Basic». Puede extraer el tipo de error de la dirección 23610. Y comprobar cuál es, por ejemplo, con IF PEEK 23610= 5 THEN... Si hace esto antes haga POKE 23610,255 para que cualquier error previo sea limpiado. Teniendo presente que si ocurre un error el Spectrum saltará las demás instrucciones si se trata de una línea multisentencia.

Otro peligro es el crak que se produce si a un INPUT numérico se contesta con el comando STOP o una variable no definida es usada en un INPUT. Es mejor utilizar INKEY$ e INPUT LINE para sus entradas de datos y comprobar cada cadena antes de usarla con VAL$.

Esperamos que si usted descubre otros efectos nos los escriba.

Justo Maurin

Un supercatálogo en microdrive

La interfase ZX 1 del Spectrum aporta a esta máquina tres nuevas funciones: conducción de ocho lectores de bandas sin fin, reunión en red de hasta 64 Spectrum, interfase RS 232C.

Lo esencial de ésta son ocho kilo-octetos de ROM, que se añaden a los 16Ko de la memoria muerta del Spectrum básico: por consiguiente, se dispone de 24Ko.

Ahora bien, el espacio de RAM que está entre las direcciones 16384 a 65535 (4000H a FFFFH) queda disponible para el usuario. Este esfuerzo se ha realizado por «hard». La memoria muerta de la interfase ZX 1 se selecciona al encontrar una instrucción Z80: RST 8 (Restart 8). La memoria ROM del Spectrum reaparece en el momento de encontrar una instrución Z80: RET situada en la dirección 0700H de la interfase ZX 1. De este modo, en un momento determinado, está activa una u otra ROM, Pero nunca las dos a la vez.

El DEFB, que sigue a la instrucción RST 8 determina la continuación de las operaciones: para DEFB entre 0 y 26 (tras una vuelta por la ROM de la interfase ZX 1 para verificación) se devuelve el control al Spectrum y se presenta el informe de error que corresponda a DEFB. Los DEFB entre 27 y 50 orientan el programa hacia una función específica de la interfase ZX 1 (apertura de un fichero en lector de bandas, apertura de la red, salida vía interfase RS 232C, etc.).

Por el contrario, los DEFB comprendidos entre 51 y 254 envían hacia un mensaje de error de un nuevo tipo, el «Hook Code Error»: no se ha «enganchado» una función disponible en el sistema.

Si bien se pueden escribir programas en lenguaje máquina con los Hook Code de la interfase ZX 1; sin embargo, existen limitaciones. Por ejemplo, se podría emplear el Hook Code que sirve para abrir un fichero en lector de bandas, después utilizar reiteradamente el que permita la escritura en un sector determinado y finalmente, emplear el Hoock Code que cierra el fichero. Pero no existe otra acción sobre el desarrollo del programa en la interfase ZX 1. Además la función vuelve a seleccionar obligatoriamente la ROM básica del Spectrum al final de la ejecución.

No obstante, para alegría de los programadores (prevenidos), al Hook Code 50 (32H); sencillamente reseñado «reservado para Sincair Research Ltd» por el Dr. Ian Logan en su libro sobre lectores de bandas sin fin; nos permite permanecer todo el tiempo necesario con la ROM de la interfase ZX 1, mientras no se ejecute el RET de la dirección 0700H.

Empleando este Hook Code al principio de una rutina en lenguaje de máquina seleccionamos sólo la MEM de la interfase ZX 1. Sin embargo, se puede llamar una rutina de la memoria muerta básica empleando la instrucción Z80: RST 16d, seguida por dos octetos que indican la dirección del principio de la rutina. Así, por ejemplo, la secuencia:

```
RST 16
DEFB E3H
DEFB 2Dh
```

sirve para imprimir en el periférico de salida el número en coma flotante almacenado en la pila del calculador.

Podemos escribir una rutina de tipo CATALOGO que proporcionará informaciones suplementarias sobre los registros (tipo, nombre, origen y longitud de bloque, líneas de autolanzamiento, etc.).

La rutina, de 621 octetos, se encarga de la búsqueda en el cartucho y de la impresión del catálogo (no necesita interfase Basic). Aquí se ha fijado el origen de la rutina en 31000d. No obstante, si dispone de un ensamblador, podrá modificar el origen de implantación.

Las líneas 80 a 160 nos introducen en la MEM de la interfase ZX 1 salvaguardando la dirección de retorno al Basic. De la 170 a 200, se restaura la dirección de retorno y se devuelve el control al Basic.

«Comi» inicializa el canal del lector de bandas y la zona MAP que afecta a los sectores del cartucho. El contador, empleando una de las variables del sistema que ha dejado libre Sinclair, totalizará el número de tipos de registro de la banda. El bucle identificado como BOUC busca el número del primer sector empleado por cada registro. Estos números se almacenan en la zona DEFB titulada REC 1, situada a continuación de la rutina. Las líneas 610, 620, paran el motor del lector de bandas.

SELP selecciona el periférico de salida, la impresora. Por medio de un sencillo POKE sobre el código de la rutina podrá orientar las salidas hacia la pantalla: POKE 31087,2.

Las líneas 730 a 800 imprimen el nombre de identificación del cartucho.

REC verifica que existe, por lo menos, un registro en el cartucho.

El bucle BOUC 2 («Buc 2») vuelve a tomar uno por uno los sectores inventariados antes para leerlos y disponer de todas las informaciones contenidas en el preámbulo (¡compuesto por 9 preciosos octetos!). Las líneas 1160 a 1800 sirven para interpretar e imprimir esas informaciones.

Otro 1 verifica que no se ha interpretado antes el sector afectado.

FIN calcula e imprime el número de kilo-octetos que quedan disponibles en el cartucho. El canal del lector de bandas se cierra y se selecciona de nuevo la pantalla como periférico de salida.

IMPMES se emplea para imprimir uno de los mensajes, IMPVIR imprime el valor numérico del registro BC. LECT selecciona el primer lector de cartucho (el registro A codifica su número de 1 a 8).

Sin embargo, su interfase ZX 1 puede pertenecer a la segunda serie fabricada por Sinclair (parece que se vende desde hace poco en Gran Bretaña). Es bastante diferente, en especial al nivel de las direcciones: se ha modificado para paliar ciertas «faltas» de la primera versión, que quizá ya haya descubierto.

Si fuera este el caso, la rutina que presentamos aquí no funcionaría sin un arreglo de las direcciones de las rutinas llamadas. En una anexo, un programita le permitiría determinar el tipo de su interfase.

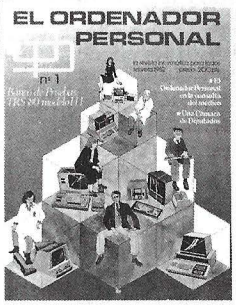
Benoit Thonnart.

PROGRAMA

```
  10          ORG  31000
  20          ENT  31000
  30
  40          RST  8
  50          DEFB #31
  60          EXX
  70          PUSH HL
  80          EXX
  90          LD   HL,COMI
 100          LD   (#5CED),HL
 110          RST  8
 120          DEFB #32
 130          EXX
 140          POP  HL
 150          EXX
C160          LD   A,2
 170          CALL #1601
 180          LD   HL,VERS1
 190          LD   A,(IY+110)
 200          CP   #FF
 210          JR   Z,IMP
 220          LD   HL,VERS2
 230 IMP      LD   A,(HL)
 240          CP   1
 250          RET  C
 260          RST  16
 270          INC  HL
 280          JR   IMP
 290
 300 COMI     LD   A,(#16DA)
C310          LD   (IY+115),A
 320          RET
 330
 340 VERS1    DEFM "ANTIGUA ROM"
 350          DEFB 0
 360 VERS2    DEFM "NUEVA   ROM"
 370          DEFB 0
 380
 390 END
```

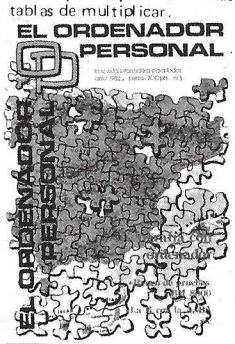
1ª OFERTA

5 números por 1.000 ptas.*

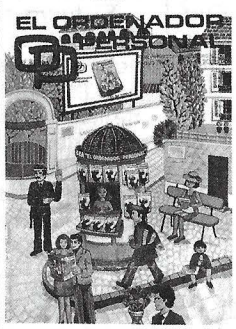


1 Presentación de ADAMICRO • Disquettes. Una tabla de índices para un acceso más directo • El Ordenador y la formación. Simulación y enseñanza asistida • Informática y Sociedad. La Cámara de Diputados • El TRS-80 modelo III en el banco de pruebas • El ordenador trata y mantiene la información en la consulta del médico • Pequeño glosario de informática • Utilización profesional • Juegos y Ordenador. Principios generales • Juegos: La huida con obstáculos • Perfeccionamiento. Para hacer buenos programas: una pizca de estructura y un puñado de módulos.

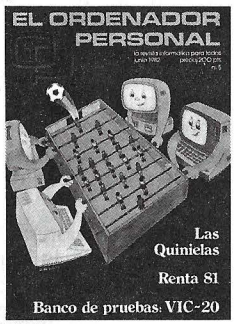
2 La exposición HARAMURI en Tokio • A pequeño comercio, pequeño ordenador • Abajo los prejuicios • Un tuno llamado VIC ronda bajo su ventana • Exploración anatómica y geográfica del ordenador • Banco de pruebas: PIPPLE II • Utilice un ordenador para la gestión de su club • Pequeño glosario de informática • ¿Estará Ud. en forma mañana? • Iniciación a la programación • La arquitectura de los programas de juegos • Gestión familiar • El Apple pelado • ¿Recuerda el día de la semana en que nació? • Avanzadilla de pruebas: SINCLAIR ZX 81 • Las calculadoras programables también sirven para aprender • Las tablas de multiplicar.



3 Convierta las frías tablas en sugestivas curvas • Comencemos con la B con la A, BASIC • ¿Por qué una nueva informática? • Enseñe, al ordenador, Geografía • Iniciación: En la intimidad del 007 • Un servidor que sirve para todo • Los sub-programas • Avanzadilla: Sinclair (Continuación) • El juego del ahorcado • Avanzadilla: Sinclair (Continuación) • Banco de pruebas: CBM 8000 • La Dietética asesorada por calculadora • Encantos del Sharp • Marcador automático con Sharp 1500 • Las Vegas • Gran Premio de Penches • Pequeña música informática.

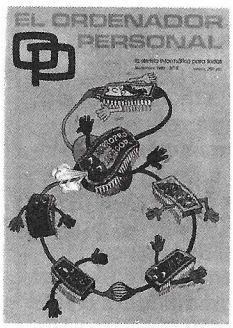


4 El sueño de una noche de invierno. Los 12 trabajos del microprocesador • Ensambladores, compiladores, intérpretes. La historia verdadera de su nacimiento • Las quinielas, relaciones de equivalencia • ¿Qué periféricos conectar a su ordenador? • La informática personal en Japón • Si está perdido, sitúese con un mapa y una calculadora de bolsillo • Banco de Pruebas: Philips P2000 • Una cuestión de método. La programación estructurada • Tres novedades Sony • Avanzadilla de pruebas: TRS80 color de Tandy Radio Shack • Enseñanza: A sumar se ha dicho • Dígalo con flores: Una tesis doctoral • Pasatiempos aritméticos: Los cuadrados mágicos o los crucigramas de la aritmética • Club de usuarios ZX81 Sinclair • Cosillas del ZX80 y 81.

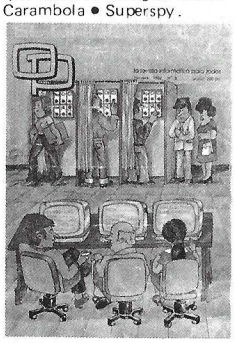


5 Renta 1981 ó como calcular el impuesto • Seguimiento de los mundiales • Las quinielas. El método del potencial • Retrato de Familia • El futuro inmediato. Una vida diferente en la oficina la Ofimática • Banco de Pruebas: VIC-20 • Algunos consejos para una elección correcta de logical de gestión de ficheros • Prueba de periféricos • ¿Quieren Uds. jugar a los juegos del ordenador personal? • Cómo remitir artículos para su publicación • Cosillas del ZX81.

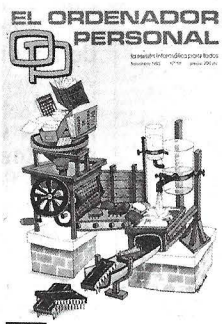
6 ¿El Basic le cansa? • Prueba de periféricos • Las quinielas (III). El símil cristalográfico • Ensayo para Los Angeles 84: 007 emite desde el Valle del Silicio • División de polinomios • Banco de Pruebas: Sharp MZ80-B • ¿Quiere Ud. jugar a los juegos del Ordenador Personal? • Los diskettes y su sistema de explotación • Cosillas del ZX 81 • Pequeña música informática.



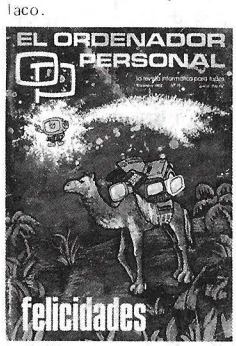
8 San Francisco, siempre la más avanzada (la más hacía el Oeste) • Aplicación profesional: tres analistas de laboratorio • Tertulia de lenguajes. Un lenguaje potente: Forth • La generación de las pantallas planas • Como aprende morse con un Apple • Conectar un ordenador con un periférico: Los problemas de interface • Informática de bolsillo. Cambie de base cuando lo necesite con este programa para calculadora H.P. • Banco de Pruebas: BHP modelo 80-21 D Micral • ¿Quiere Ud. jugar a los juegos del Ordenador Personal • Material y Logical ¿Qué precauciones hay que tomar? • Juegos: Micro-Carambola • Superspy.



9 Programa Electoral: La Ley d'Hont • El programa supervisor coordinador de una circulación completa • Feria de los ordenadores • Música en el TRS-80 • Las quinielas (IV). Geometría dispersa • Las novedades de Texas: TI-88 y TI-57 LCD • Informática y medicina • Ponga un "zoom" en sus gráficas • Los juegos y el ordenador: Cómo programar una partida de dominó • Cuando el tren sigue la vía de su amo (I) • Banco de Pruebas: Atari 800 • Banco de Pruebas Logical: CORP. Generador de programas • Divulgación. Un ordenador doméstico muy perfeccionado. El sistema YIS de Yamaha • Tertulia de lenguajes: Los lenguajes de programación de ordenadores • No descuide la seguridad de los programas y de los datos • ¿Qué es lo que dá vueltas como un disco, es negro como un disco y es a la vez cuadrado? El Diskette • La caja negra • ¿Quiere Ud. programar los juegos del Ordenador Personal?

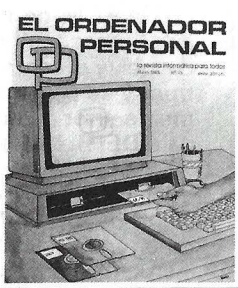


10 Houston: La NCC • ZX Spectrum • Ponga un ordenador en la máquina de escribir y consiga el tratamiento de textos • Respuestas del limón a la manzana • Banco de Pruebas: Olivetti M-20 • El Ordenador en casa: El juego del radar para Sharp PC-1211 • Las novedades del SICOB • Big-Pattern • Banco de Pruebas de Logical: Basi Data • Informática y Sociedad: ¿es de temer la informática? • Informática y Medicina: La informática ayuda al tratamiento del cancer (1ª parte) • Juegue con el Ordenador Personal • Como ganar 140K octetos al menos taladrando un diskette • Las leyes de Golub del reino del ordenador • El encanto del Sharp • Informática de bolsillo: El tejano polaco.



11 Ecuaciones n-cuadradas • Basic y lenguaje máquina • Robots para jugar. Empezamos por una tortuga • Prepare un TRS para activar reles • Un ordenador que domina formas y colores • Ajedrez. Los principios • Banco de Pruebas: FACIT DTC 6522 • El laberinto de Candy. Juegos • Código de Barras, Impresora C. Itoh 8510 • Periféricos. Impresiones sobre impresoras • La cara oculta del Sharp • Avión Espía • Gestión de ficheros • La Informática ayuda al tratamiento del Cáncer 6 ¿Es usted lógico? Un juego sobre HP-41C • Avanzadilla de pruebas: CASIO FX-702P • ¿Quieren ustedes programar los juegos del Ordenador Personal?

12 Apple, IBM y Visicorp • Novedades en Japón • Lenguaje de programación ESCOLAR • Ciencia Ficción (mañana ¿qué ordenadores?) • Los juegos y el ordenador • Avanzadilla de pruebas: EL ACORN-ATOM • EL BASIC BASICO (La B con la A, Basic) • El tratamiento de textos en Japón • Y el Hombre creará el ROBOT (1ª parte) • Traductor - Monitor/Intérprete para MZ 80 B • Banco de Pruebas: EL NEWBRAIN • ELMASTERMIND en Basic del ZX-81 • Programe en lenguaje máquina: ZX-81 - ROMPEMUROS • El secreto de los algoritmos calculadoras • El microordenador en las clínicas • La funcion HIR de la TI 58/59.



BOLETIN DE PEDIDO

Sírvanse enviarme los números atrasados del ORDENADOR PERSONAL que marco con una equis ☒

Nombre ..... Apellidos .....

Dirección ..... Tfno. .....

Población ..... D.P. ..... Provincia .....

FORMA DE PAGO: ☐ Talón adjunto ☐ Giro Postal ☐ Contra reembolso

Deseo recibir los nºs

Siguientes del O.P.

(marque con una equis ☒)

1	2	3	4	5	6	8	9	10	11	12	13

14	15	16	17	18	20	21	22	23	24		

(*) Esta oferta es válida sólo para cinco o más ejemplares, cada ejemplar de más se cobrará al mismo precio de 200 pts.

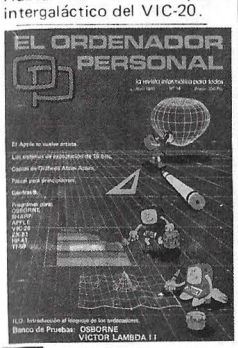
FIRMA:

–REMITIR A: El Ordenador Individual S.A. C/ Ferraz, 11 - Madrid - 8

2ª OFERTA

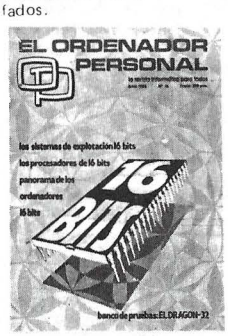
Ahorre 1.000 ptas. al suscribirse

13 Visita a la MICROFAIR • COMPEC-82: La informática Británica • Hewlett-Packard:... Una estrategia diferente • Introducción al sistema CP/M • ¿Cómo seleccionar el software educativo? • Y el hombre creará el robot (parte II) • Avanzadilla de prueba: EL AIM 65/40 • LOGO - Carmela y la tortuga • LOGO - Meta una tortuga en su ZX-81 • Los nuevos antiguos contra los nuevos modernos • Banco de Pruebas: EL SIRIUS-1 • Viaje alucinante a través del INTERPRETER EN UN MZ-80-B • Sistema periódico de los elementos. Gestión de ficheros secuencias en CBM 8032 • Los juegos y el Ordenador (Parte II). Cada vez menos tiempo con el algoritmo Alfa-Beta • Minigolf de Karnak: para TRS y Video Genie • Las quinielas. Clases de equivalencias - Apple II • No jugueis a las cerillas si no estais seguros de poder ganar HP-41 • Laberinto para MZ 80 B • Ficheros para Atom • Había una vez en el espacio intergaláctico del VIC-20.

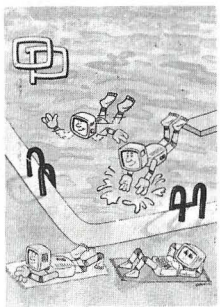


14 Los Sistemas de Explotación 16 bits en guerra • ILO - Introducción al Lenguaje de los Ordenadoras • Ahorre memoria y aumente la velocidad de sus programas Basic interpretados • Banco de Pruebas: EL OSBORNE 1 • GENFRAS 8. Programa generador de frases para el ZX81 • Avanzadilla de pruebas: EL VICTOR LAMBDA II • Alerta. Las naves del Imperio contraatacan. ZX-81 • Recetario BASIC • PASCAL para principiantes • La informática y el diseño asistido: EL APPLE se vuelve artista-1ª parte • Copia de Gráficos en alta resolución sobre impresora. ATOM-ACORN • Las confidencias del PC-1500 - 1ª parte • Un laberinto sin el hilo de Ariana - VIC 20 • Impresión de calendarios optimizada - HP 41.

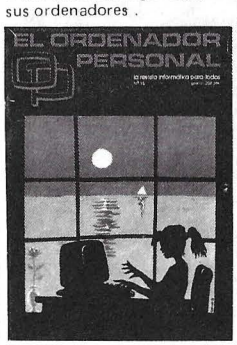
15 Diseño E A O • Médicos Consulten un O.P. • El Ordenador al servicio de las elecciones • Lenguaje máquina y ensamblador (el lenguaje del 6502) • Póngase Ud. al día • Por qué y cómo informatizarse... Consejos y recetas • Examinemos las memorias del Basic • Aprendizaje del Basic en un Instituto de bachillerato • Pascal para principiantes (2ª parte) • Banco de pruebas Basic • Confidencias del P C. 1500 (2ª parte) • Banco de pruebas: ORIC-1 • El Acorn Atom protegiendo a la tierra frente a una terrible invasión • El Apple se vuelve artista (2ª parte) • El juego de Neiscat • Métodos de Montecarlo (P C 1211) • Programa para alta resolución (Z X-81) • Activación y desactivación de sus aparatos domésticos • Como hacer un puente • Producto de Matrices (H P-41) • La astucia y la habilidad hacen más para fundar un club que la fuerza y los enfados.



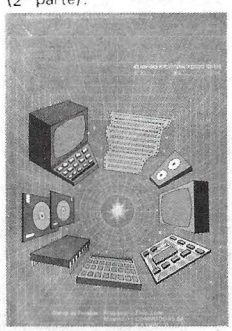
16 El Cebit-83 de Hannnover • Los ordenadores 16 Bits • Los procesadores 16 Bits • Banco de Pruebas: Dragón 32 • Los sistemas de explotación 16 bits • PASCAL para principiantes (III) • Y el hombre creará el ROBOT (III) • Pánico en el fondo del mar • Las carreras de coches, un deporte de Salón • Programas de 1 K para el ZX-81 • Rally de Montecarlo para PC-1500 • Recetario Basic.



17 La 8ª West coast Computer Fair • Selección de equipos con fines educativos • El O.P. no hace al monje • Viaje al país de los juegos • Lenguaje máquina y ensamblador. El ejemplo del 6502 (y II) • 20.000 Leguas de viaje sub-pantalla • Periféricos HP-IL • Periféricos inteligentes para trabajar más rápidos • Sobre dos tipos de "Cracks" misteriosos en el ZX-81 • Primeros pasos del programa en notación algebraica • Síntesis musical • Nuevos usos para viejas calculadoras • ¿Conseguirá el ZX-81 salvar a los náufragos? • Conducir una locomotora • Un microbiólogo habla de sus ordenadores.



18 Memorización de pantalla en el PC-1.500 • Pascal para principiantes (4ª parte) • Un sistema operativo estructurado. Unix • Un medidor de velocidad de cassette para Atom • Un poderoso programa para la correlación de sus datos. Parte I • Como transformar la impresora PC-100 en un trazador de curvas • El Apple se vuelve artista (3ª parte) • Descubrir las artes gráficas gracias a la informática • El ordenador ayuda en la investigación de la paternidad • Recetario Basic • Las cuatro en raya del O.P. • Control informático del tratamiento antibiótico • ¿Quién pagará las cañas, usted o su HP-41? • Eche una carrera con su TI-59 • Y ante todo la música. Práctica de la síntesis musical • Los invasores han vuelto, yo los he encontrado • El ordenador jefe de estación (2ª parte).

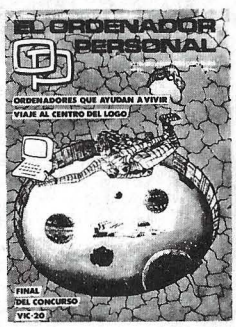


20 SICOB-34 edición • Euromouse-83 • Repertorio de instrucciones del microprocesador Z-80 y Lenguaje Assembler • Banco de Pruebas: KAYPRO II • Pascal para principiantes ficheros, procedimientos y funciones (5ª y última parte) • Banco de Pruebas programas: TIME ZONE: la máquina del tiempo • Vera Molnar o como dominar la casualidad • Creación artística • Banco de Pruebas: EL COMMODORE 64 • Dibujos en Perspectiva en su H P-41 • Le toca a Ud. ahora, fulminar al dragón del VIC-20 • El dibujo animado al alcance de su pantalla ZX81 • Lenguaje máquina para PC-1500 • Gráficos en el OSBORNE I • Cálculo de Velocidad de perfusión de fármacos vasoactivos • Juego de las siete y media para el CASIO FX 702-P • Integración por el método Simpson con ZX SPECTRUM.

21 Estudio detallado de un V.I.A. • Redes locales • Repertorio de instrucciones del microprocesador Z80 y lenguaje Assembler (2ª parte) • Banco de Pruebas: RED OMNINET • Cómo utilizar redes de mil formas razonables • Avanzadilla: Casio PB-300 • La falta de dinero no es tan grave —clubs— • Rutina en código máquina para proteger programas en BASIC ZX81 • Supervisión de cuentas corrientes con un SHARP PC-1211 • Es el momento de esquivar HP 41 • Dump hexadecimal para PC-1500 • A hacer chuletas ZX-81 • Programas de 1 K para el ZX-81 • Tic-tac-tic-tac en Vic y en ORIC • Economice la ocupación de memoria TI-59.

22 Sistema operativo para microprocesador de 16 bits: MS-DOS • Repertorio de instrucciones del microprocesador Z 80 y lenguaje Assembler (3ª parte) • Viaje al centro del LOGO (3ª parte) • Un programa lleno de energía atómica. ATOM-ACORN • Rosas negras • Terrible amenaza a la federación galáctica HP-41 • Banco de Pruebas: MICRO PROFESSOR II • Geografía Espacial. PC 1.500 • Choque elástico. DRAGON 32 • Realizad nuevos caracteres en vuestro VIC-20 • Tratamiento de textos en la FX-702P • Estadística de dos variables para el ZX-81.

23 Viaje al centro del Logo (2ª parte) • Pequeños que casi no temen a los grandes • Realización de un protocolo CENTRONICS • ¿Quién es Richard Paul Jones? • Un laberinto sin el hilo de Ariadna. VIC-20 (Resultado del concurso) • Ordenadores que ayudan a vivir • Espíritu ¿estás aquí? o los fantasmas del Commodore 64 (1ª parte). CBM-64 • La PC-1500 hace música. PC-1500 • BASIÑOL. El Basic español. APPLE • O.P. Defender. ZX SPECTRUM • ¿Dónde se encuentran los planetas? ZX-81 • Combinatoria. VIC-20.



24 Avanzadilla de Prueba: HP-150 • Viaje al centro del Logo (3ª parte) • Cuando el sueño se convierte en tecnología • Banco de Pruebas SORD M-5 • ¿Carecen de voz los O.P.? • Introducción al lenguaje de programación C • La PC - 1500 aprende música • Escalera de color y escalofríos asegurados • Al claro de luna, amigo Pierrot préstame tu O.P. • Música es el arte de combinar sonidos • Cómo evaluar un biorritmo con la HP-41 • ¿Espíritu estás aquí? o los fantasmas del Commodore-64 (2ª parte) CBM-64 • Resolución de ecuaciones de 1er Grado • Rutina para formato de datos en impresión • Cuando el Atom tiene cita con la luna.

10 números al año = 2.500
\+
Guía = 500
\+
2 nos atrasados a elegir = 500
Total = ~~3.500~~
Ahora sólo = **2.500**

Deseo suscribirme a la revista EL ORDENADOR PERSONAL, por un año (10 números) recibiendo además LA GUIA de ORDENADORES PERSONALES y los dos números atrasados que marco a continuación.

MARQUE CON UNA ☒ LOS DOS Nos ATRASADOS QUE DESEA RECIBIR.

1	2	3	4	5	6	8	9	10	11	12	13
14	15	16	17	18	20	21	22	23	24		

Nombre ..... Apellidos .....

Dirección ..... Tfno. .....

Población ..... D.P. ..... Provincia .....

Forma de Pago: ☐ Cheque adjunto ☐ Reembolso ☐ Giro Postal.

Firma Fecha

PROGRAMA

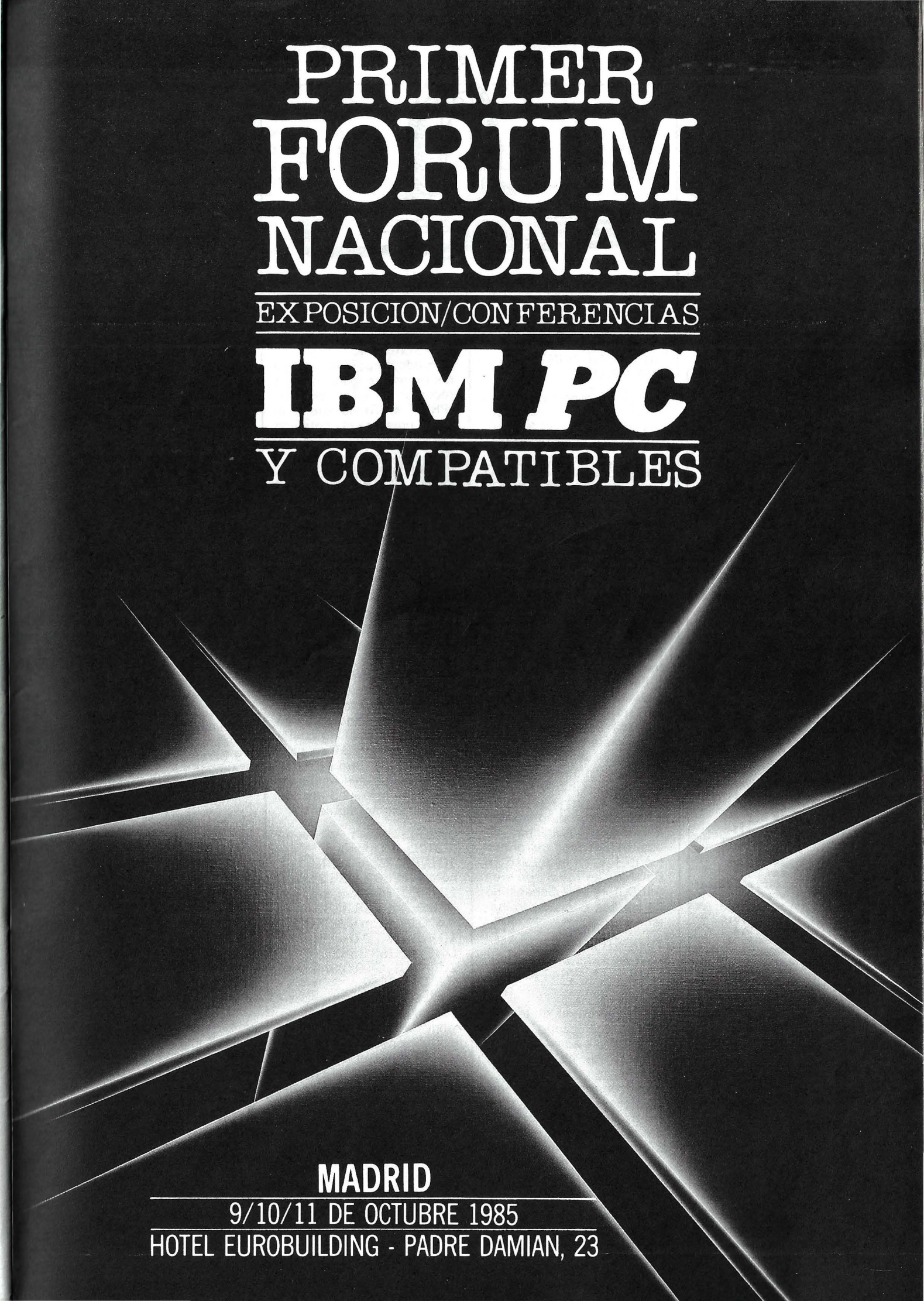
```
  10 ;Super catalog
  20         ORG  31000
  30         ENT  31000
  40
  50
  60
  70 ;entrada en la Rom ZXi1
  80         RST  8
  90         DEFB #31
 100         EXX
 110         PUSH HL
 120         EXX
 130         LD   HL,COMI
 140         LD   (#5CED),HL
 150         RST  8
C 160        DEFB #32
 170         EXX
 180         POP  HL
 190         EXX
 200         RET
 210
 220 ;inicializacion del canal
 230 ;del microdrive, contador
 240 ; y area de trabajo
 250 COMI    CALL #0FE8
 260         CALL LECT
 270         LD   HL,REC1
 280         XOR  A
 290         LD   (IY+118),A
 300         PUSH HL
C 310
 320 ;lectura de todos los
 330 ;sectores
 340 ;busqueda de comienzos de
 350 ;cada grabacion
 360 BOUC    PUSH HL
 370         CALL #12C4
 380         CALL #1E53
 390         POP  HL
 400         JR   NZ,BOUC
 410         LD   A,(IX+69)
 420         OR   (IX+70)
 430         JR   Z,LIBRE
 440         OR   (IX+67)
 450         AND  2
C 460        JR   NZ,ZERO
 470 LIBRE   PUSH HL
 480         CALL #12FE
 490         POP  HL
 500         JR   DECR
 510 ZERO    LD   A,(IX+68)
 520         AND  A
 530         JR   NZ,DECR
 540         LD   A,(IX+41)
 550         LD   (HL),A
 560         INC  (IY+118)
 570         INC  HL
 580
 590 DECR    CALL #1312
 600         JR   NZ,BOUC
C 610        XOR  A
 620         CALL #17F7
 630
 640 ;seleccion del periferico
 650 ;de salida
 660 BELP    LD   A,3
 670         PUSH IX
 680         RST  16
 690         DEFB 1
 700         DEFB #16
 710 ;impresion del nombre del
 720 ;cartucho
 730         LD   HL,CART
 740         CALL IMPMES
 750         POP  HL
C 760        LD   DE,#2C
 770         ADD  HL,DE
 780         CALL #1D50
 790         CALL CR
 800         CALL CR
 810
 820 REC     POP  HL
 830         LD   A,(IY+118)
 840         LD   B,A
 850         AND  A
 860         JP   Z,FIN
 870         LD   A,(HL)
 880         LD   (IY+119),A
 890
 900 BOUC2   PUSH BC
C 910        PUSH HL
 920         LD   (IX+13),A
 930         CALL LECT
 940
 950 LIT     CALL #12C4
 960         LD   A,(IX+41)
 970         CP   (IX+13)
 980         JR   Z,HEAD
 990         CALL #1312
1000         JR   NZ,LIT
1010
1020 HEAD    PUSH IX
1030         POP  HL
1040         LD   DE,#0043
1050         ADD  HL,DE
C1060        CALL #18A9
1070         XOR  A
1080         CALL #17F7
1090         CALL #1341
1100         JR   NZ,ERR
1110         LD   DE,#000F
1120         ADD  HL,DE
1130         CALL #1346
1140 ERR     JP   NZ,OTRO1
1150         BIT  2,(IX+67)
1160         LD   HL,FICH
1170         JR   Z,IMP
1180 SAVE    LD   A,(IX+82)
1190         LD   HL,TYPE
1200         LD   B,4
C1210        CPIR
1220
1230 IMP     CALL IMPMES
1240
1250 IMPF    LD   A,60
1260         CALL #1D66
1270         LD   A,(IX+71)
1280         AND  A
1290         JR   NZ,NONP
1300         LD   A,95
1310
1320 NONP    CALL #1D66
1330         LD   B,95
1340         PUSH IX
1350
C1360 IMPNOM LD   A,(IX+72)
1370         CALL #1D66
1380         INC  IX
1390         DJNZ IMPNOM
1400         POP  IX
1410         LD   A,62
1420         CALL #1D66
1430         CALL CR
1440         BIT  2,(IX+67)
1450         JR   Z,OTRO
1460         LD   HL,LONG
1470         CALL IMPMES
1480         LD   A,(IX+82)
1490         LD   HL,PROGV
1500         AND  A
C1510        JR   Z,CONTI
1520         LD   HL,CODE
1530
1540 CONTI   CALL IMPMES
1550         LD   C,(IX+83)
1560         LD   B,(IX+84)
1570         CALL IMPVIR
1580         LD   HL,ORIGI
1590         CALL IMPMES
1600         LD   C,(IX+85)
1610         LD   B,(IX+86)
1620         CALL IMPVIR
1630         LD   A,(IX+82)
1640         AND  A
1650         JR   NZ,OTRO
C1660        LD   HL,PROG
1670         CALL IMPMES
1680         LD   C,(IX+87)
1690         LD   B,(IX+88)
1700         CALL IMPVIR
1710         LD   A,(IX+90)
1720         CP   #FF
1730         JR   Z,OTRO
1740         LD   HL,LINEA
1750         CALL IMPMES
1760         LD   C,(IX+89)
1770         LD   B,(IX+90)
1780         CALL IMPVIR
1790
1800 OTRO    CALL CR
C1810
1820 OTRO1   POP  HL
1830         POP  BC
1840         INC  HL
1850         LD   A,(HL)
1860         CP   (IY+119)
1870         JR   Z,FIN
1880         DEC  B
1890         JP   NZ,BOUC2
1900
1910 FIN     LD   HL,NOMB
1920         CALL IMPMES
1930         CALL #1D38
1940         LD   A,E
1950         SRL  A
C1960        LD   C,A
1970         CALL IMPVIR
1980         CALL #10C4
1990         LD   A,2
2000         RST  16
2010         DEFB 1
2020         DEFB #16
2030         RET
2040
2050 IMPMES  LD   A,(HL)
2060         CP   10
2070         RET  C
2080         CALL #1D66
2090         INC  HL
2100         JR   IMPMES
C2110
2120 IMPVIR  RST  16
2130         DEFB #2B
2140         DEFB #2D
2150         RST  16
2160         DEFB #E3
2170         DEFB #2D
2180 CR      LD   A,13
2190         CALL #1D66
2200         RET
2210
2220 LECT    LD   A,1
2230         CALL #17F7
2240         LD   BC,#00FF
2250         LD   (#5CC9),BC
C2260        RET
2270
2280 TYPE    DEFB 0
2290         DEFM "Programa "
2300         DEFB 1
2310         DEFM "Tab. numerica "
2320         DEFB 2
2330         DEFM "Tab. de caracteres "
2340         DEFB 3
2350         DEFM "Bloque de octetos "
2360         DEFB 4
2370
C2380 LONG   DEFM "Longitud "
2390         DEFB 0
2400 PROGV   DEFM "Prog. + Var : "
2410         DEFB 0
2420 CODE    DEFM "Code : "
2430         DEFB 0
2440 ORIGI   DEFM "Origen del bloque : "
2450         DEFB 0
2460 PROG    DEFM "Longitud del programa
                   solo : "
2470         DEFB 0
2480 LINEA   DEFM "AUTO Linea n. : "
C2490        DEFB 0
2500 NOMB    DEFM "Koctetos libres : "
2510         DEFB 0
2520 FICH    DEFM "Fichero de datos : "
2530         DEFB 0
2540 CART    DEFM "Cartucho : "
2550         DEFB 0
2560
2570 REC1    END
```

PRIMER FORUM NACIONAL

EXPOSICION/CONFERENCIAS

IBM *PC*

Y COMPATIBLES



MADRID

9/10/11 DE OCTUBRE 1985

HOTEL EUROBUILDING - PADRE DAMIAN, 23

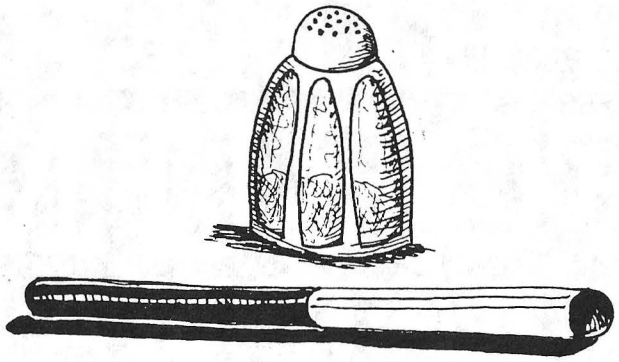
PRINT USING

(NDLR. Un PRINT para el Spectrum, pero esta vez en lenguaje de máquina. Hay que resaltar que no redondea los números).

La rutina que presentamos realiza un PRINT USING. Tiene una longitud de 119 octetos y es totalmente reubicable. Damos el listado fuente así como lo códigos decimales de la rutina.

Programa PRINT USING

```
   10         ORG  64000
   20 USING CALL 7289
   30         CALL 8967
   40         LD   (23546),BC
   50         LD   HL,23296
   60         LD   B,40
   70 U7      LD   (HL),48
   80         INC  HL
   90         DJNZ U7
  100         LD   HL,23320
  110 U2      LD   E,255
  120 U1      LD   A,32
  130         PUSH HL
  140         LD   HL,23688
  150         SUB  (HL)
C 160         LD   B,A
  170         INC  HL
  180         LD   A,24
  190         SUB  (HL)
  200         LD   C,A
  210         PUSH DE
  220         CALL 9528
  230         PUSH AF
  240         LD   A,8
  250         RST  16
  260         LD   A,32
  270         RST  16
  280         LD   A,8
  290         RST  16
  300         POP  AF
C 310         POP  DE
  320         POP  HL
  330         DEC  HL
  340         LD   (HL),A
  350         INC  E
  360         CP   45
  370         JR   Z,U1
  380         CALL 11547
  390         JR   NC,U1
  400         CP   46
  410         JR   Z,U2
  420         RST  16
  430         LD   A,(IY-64)
  440         SUB  E
  450         JR   C,U10
C 460         JR   Z,U10
  470         LD   B,A
  480 U3      LD   A,32
  490         RST  16
  500         DJNZ U3
  510 U10     LD   B,E
  520 U4      INC  HL
  530         LD   A,(HL)
  540         RST  16
  550         DJNZ U4
  560         XOR  A
  570         LD   B,(IY-63)
  580         CP   B
  590         JR   NZ,U6
  600         LD   A,8
C 610         RST  16
  620         JR   U8
  630 U6      INC  HL
  640         LD   (HL),46
  650 U5      LD   A,(HL)
  660         RST  16
  670         INC  HL
  680         DJNZ U5
  690 U8      LD   A,(HL)
  700         SUB  48
  710         LD   C,A
  720         LD   B,0
  730         RET
```



El empleo es bastante sencillo. Tras haber introducido el programa en lenguaje de máquina, teclee, por ejemplo, la siguiente línea:

10 INPUT a: PRINT «A=»; USR 6400,5,3

El 5 representa el número de espacios y de cifras antes de la coma, el 3 el número de cifras tras la coma.

Estas cifras pueden sustituirse por una variable o una expresión del tipo: VAL «2». No obstante, existe una obligación: la variable debe estar precedida por un carácter diferente de una cifra, un punto (.), o un signo menos (–).

Si la primera cifra es menor que el número de cifras anteriores a la coma, la función no la tiene en cuenta, pero respeta la segunda cifra.

Esta función trabaja de la siguiente manera: el Basic presenta la variable en la pantalla. El programa hacia atrás, mete en memoria las cifras, las cuenta y las borra. De vuelta, presenta los, espacios y las cifras en memoria.

Lanzando el programa en lenguaje de máquina, 10 octetos después del principio (o sea, en este caso en 64010 en vez de en 64000), se está dispensado de las dos cifras de formateo.

10 INPUT a: PRINT «A=»; a; USR 64000,5,2
20 INPUT b: PRINT «B=»; b; USR 65410

Los caracteres de tabulación, coma, punto, coma, apóstrofo se admiten a continuación de las dos cifras de formateo.

El programa en lenguaje máquina emplea tres subprogramas del Basic Spectrum. Los dos primeros CALL permiten la valoración de las cifras de formateo que están colocadas en el registro BC. El tercer CALL carga en el acumulador, el código del carácter, el código del carácter situado en la pantalla y precisado por el registro BC (B número de columna, C número de línea). En realidad, es la función SCREEN$.

Puede sorprender el final del programa, pero hay que recordar que al emplear «PRINT USR», la vuelta al Basic presenta el contenido del registro BC. LD A,8 y RST retroceden una posición en la pantalla. SUB 48, LD C,A y LD B,O introducen el último carácter en el registro BC.

Gilberto Richon

Variaciones en INPUT

El Basic del Spectrum tiene la particularidad de reservarse como mínimo, dos líneas en la parte baja de la pantalla para entradas y mensajes de error. Es práctico y muy bueno, pero en algunos casos, gustaría guardar los INPUT en pantalla como ocurre en otros Basic. Es posible utilizando como modelo uno de los tres programas siguientes:

- Se emplea la rutina SCROLL del monitor.

10 INPUT «nom :»; LINE n$' USR 3582, USR 3582
20 INPUT «número: »; n' USR 3582, USR 3582

- O bien con el SCROLL:

10 INPUT AT 0,0 «nom :»; LINE n$' USR 3582
20 INPUT AT 0,0; «número :»; n' USR 3582

- Este programa necesita una pequeña rutina en lenguaje máquina de dos instrucciones de cinco octetos, POKE-ados a partir de 23.530 (pero se puede en cualquier otra dirección libre). La rutina realiza: LD (IY + 49), 2 RET

10 POKE 23530,253
20 POKE 23531,54
30 POKE 23532,49
40 POKE 23533,2
60 INPUT AT 4,0; AT 0,0 «nom :»; LINE n$' «número :»; n' USR 23530

El «AT 4,0» reserva cinco líneas en la parte baja de la pantalla «AT 0,0» presenta la primera de estas líneas. USR equivoca al Basic reduciendo el número de líneas reservadas a dos.

No olvidar los apóstrofes tras las variables en los tres programas.

Gilberto Richon

¿Merge o Load?

Sinclair Research, para proteger sus logicales en cartucho de una eventual piratería, ha modificado el funcioniento MERGE con relación al magnetófono. De este modo, un programa salvaguardado en cartucho con una línea de autolanzamiento no puede cargarse en memoria mediante MERGE. Cualquier tentativa devolverá el informe de error Merge error.

Los creadores de logicales no tienen necesidad de esta protección y podrían encontrar otras más eficaces (diferente formateo de la banda...). Solamente para vosotros programadores aficionados, ¿cómo recuperar uno de sus programas salvaguardado con un línea de autolanzamiento, antes de cualquier inicialización para añadirle modificaciones?

Veamos un programa de 99 octetos, en lenguaje de máquina, cuyo programa fuente de ensamblador se escribió en ensamblador GENS2. Está seguido por códigos hexadecimales para los que no tengan ese ensamblador.

El acceso a la rutina se hace de dos maneras, según se quiera cargar un programa sin autolanzamiento o fusionarlo con otro programa. Este acceso puede hacerse también en modo comando o en modo programa.

No obstante, la llamada debe hacerse en una línea de tres instrucciones separadas por dos puntos: llamada a la rutina mediante RANDOMIZE USR 23296 dos puntos, la palabra clave REM dos puntos, la palabra clave LOAD o MERGE (y no estas palabras con todas sus letras); según la opción elegida, seguido por el nombre del programa entre comillas.

Benoît Thonnart

Códigos hexadecimales

```
2A 5D 5C 23 23 23 7E FE D5 3E E6 20 02 3E F6 32 2D 5B 06 00 23 11 68 5B 23 7E
FE 22 28 05 12 13 04 18 F5 C5 CF 31 C1 D9 E5 D9 FD CB 7C E6 3E 01 32 D6 5C 78
32 DA 5C 21 68 5B 22 DC 5C 21 49 5B 22 ED 5C CF 32 D9 E1 D9 C9 CD 53 07 21 E6
5C 11 DE 5C 01 07 00 ED B0 CD 80 15 21 FF FF 22 ED 5C C3 F2 08
```

Programa

```
  10 ;MERGE o LOAD sin
  20 ;autolanzamiento
  30
  40 ;Se emplea en modo
  50 ;directo o progrma
  60 ;      haciendo:
  70 ;RAND USR 23296:REM:
  80 ; LOAD "Nombre"
  90
 100        ORG  23296
 110        ENT  23296
 120
 130        LD   HL,(#5C5D)
 140        INC  HL
 150        INC  HL
C160        INC  HL
 170        LD   A,(HL)
 180        CP   #D5
 190        LD   A,#E6:'I LOAD
 200        JR   NZ,NOM
 210        LD   A,#F6:SI MERGE
 220
 230 NOM    LD   (FLAG+3),A
 240        LD   B,0
 250        INC  HL
 260        LD   DE,23400
 270
 280 LONG   INC  HL
 290        LD   A,(HL)
 300        CP   31;Comillas
C310        JR   Z,VAR8
 320        LD   (DE),A
 330        INC  DE
 340        INC  B
 350        JR   LONG
 360        _
 370 VAR8   PUSH BC
 380        RST  8
 390        DEFB #31
 400        POP  BC
 410        EXX
 420        PUSH HL
 430        EXX
 440
 450 FLAG   SET  4,(IY+124)
C460        LD   A,1
 470        LD   (23766),A
 480        LD   A,B
 490        LD   (23770),A
 500        LD   HL,23400
 510        LD   (23772),HL
 520        LD   HL,COMI
 530        LD   (#5CED),HL
 540        RST  8
 550        DEFB #32
 560        EXX
 570        POP  HL
 580        EXX
 590        RET
 600
C610 COMI   CALL #0753
 620        LD   HL,#5CE6
 630        LD   DE,#5CDE
 640        LD   BC,7
 650        LDIR
 660        CALL #1580
 670        LD   HL,#FFFF
 680        LD   (#5CED),HL
 690        JP   #08F2
 700
 710        END
```

Tranferencia de fondos

La intrucción MOVE permite transferir datos de un fichero desde el lector de banda sin fin hacia sí mismo o hacia otro conectado a su Spectrum. Pero tal como se indica en el manual (p. 39), la instrucción MOVE sólo funciona con ficheros de datos.

La rutina que presentamos, totalmente escrita en lenguaje de máquina, le aportará el complemento de la instrucción MOVE estándar.

Puede emplearla para hacer copias de salvaguardas de sus programas (comprendidos los que se autolanzan), de sus bloques de octetos, de sus cuadros numéricos o de caracteres, aquí de un micro-drive hacia sí mismo o hacia uno de los siete posibles. Igualmente esta rutina puede servir como un equivalente a la función RENAME. En efecto, el fichero de destino puede tener un nombre diferente al del fichero origen.

La rutina que ocupa 133 octetos hay que posicionarla obligatoriamente en el tampón de la impresora y no es reubicable.

Su empleo necesita una sintaxis de escritura relativamente rígida que debe ser de la forma:

RANDOMIZE USR 2396: REM
"xNOM100 TO "yNOM2".

Así, el comando o la línea de programa:

RANDOMIZE USR 23296: REM
"1ABCD" TO "2EFGH"

transfiere el programa ABCDE registrado en el cartucho del lector 1. Un nuevo programa llamado EFGH se escribirá entonces en el cartucho de lector 2. Hay que resaltar que no es preciso teclear espacios antes y después de la instrucción REM, ni antes o después de la palabra clave TO (es la palabra clave y no las letras T y O).

Aquí le proporcionamos el programa fuente en ensamblador de esta rutina, así como los octetos decimales así como los códigos decimales de los 133 octetos que la componen. Si no dispone de ensamblador puede crear el código en el cartucho por un pequeño programa similar a éste:

Bennoit Thornart

Rutina en ensamblador

```
  10 ;MOVE
  20
  30         ORG  23296
  40
  50 DUPLI   RST  8
  60         DEFB #31
  70         EXX
  80         PUSH HL
  90         EXX
 100         LD   HL,COMI
 110         LD   (#5CED),HL
 120         RST  8
 130         DEFB #32
 140         EXX
 150         POP  HL
C160         EXX
 170         RET
 180
 190 COMI    LD   HL,DATA8
 200         LD   DE,23767
 210         LD   BC,15
 220         LDIR
 230         LD   HL,(#5C5D)
 240
 250 INIT1   CALL DRIVE
 260         LD   (23766),A
 270         LD   DE,23500
 280         CALL NOM
 290         LD   A,B
 300         LD   (23770),A
C310
 320 INIT2   CALL DRIVE
 330         LD   (23774),A
 340         LD   B,0
 350         LD   DE,23510
 360         CALL NOM
 370         LD   A,B
 380         LD   (23778),A
 390
 400 CANAUX  SET  4,(IY+124)
 410         CALL OPEN
 420         LD   HL,(#5C4F)
 430         PUSH HL
 440         CALL #14C7
 450         CALL OPEN
C460         JP   #1402
 470
 480 OPEN    CALL #1B29
 490         SET  2,(IX+67)
 500         XOR  A
 510         CALL #17F7
 520         LD   (#5CDA),IX
 530         RET
 540
 550 NOM     INC  HL
 560         LD   A,(HL)
 570         CP   34
 580         RET  Z
 590         LD   (DE),A
 600         INC  DE
C610         INC  B
 620         JR   NOM
 630
 640 DRIVE   INC  HL
 650         INC  HL
 660         INC  HL
 670         LD   A,(HL)
 680         SUB  48
 690         RET
 700
 710 DATA8   DEFB 0
 720         DEFB 255
 730         DEFB 77
 740         DEFB 0
 750         DEFB 0
C760         DEFB 204
 770         DEFB 91
 780         DEFB 0
 790         DEFB 0
 800         DEFB 255
 810         DEFB 77
 820         DEFB 0
 830         DEFB 0
 840         DEFB 214
 850         DEFB 91
 860
 870 ;......FIN.....
```

Lista de los 133 octetos de la rutina

```
207  49 217 229 217  33  17  91  34 237  92 207  50 217 225 217 201  33 118  91
 17 215  92   1  15   0 237 176  42  93  92 205 111  91  50 214  92  17 204  91
205 101  91 120  50 218  92 205 111  91  50 222  92   6   0  17 214  91 205 101
 91 120  50 226  92 253 203 124 230 205  85  91  42  79  92 229 205 199  20 205
 85  91 195   2  20 205  41  27 221 203  67 214 175 205 247  23 221  34 218  92
201  35 126 254  34 200  18  19   4  24 246  35  35  35 126 214  48 201   0 255
 77   0   0 204  91   0   0 255  77   0   0 214  91
```

■

TRUCOS DE LA TI-59

Para tu TI-59 hemos preparado un Cocktail con trucos y juegos. ¡Porque no solo de ecuaciones vive el hombre!

Test de un registro de tarjeta mágnetica TI-59

En los ordenadores personales, se puede comprobar un registro por medio de un comando, comparando bit a bit con el contenido de la memoria, sin modificarla (Comando CLOAD del TRS, por ejemplo).
En la tarjeta TI 59 no existe esta posibilidad, por lo que siempre queda la duda. Sin embargo, puede comprobar su registro volviéndola a leer inmediatamente después de haberla escrito en una única partición de memoria, por ejemplo: (—) (3) (INV) (WRITE). Si hay un defecto importante, el número lateral de la tarjeta se pondrá en intermitencia. El programa en el interior de la calculadora no se habrá modificado: puede rehacer el registro y volverlo a controlar de inmediato.
Esta operación puede evitarle muchos inconvenientes en el momento del empleo de un programa.

(Xavier de la Tullaye)

Programación diferida TI 58, C 59

Seguramente habrá notado que al modificar una partición de memoria durante un programa, es posible transformar datos almacenados en memoria en paso de programas.
Ensaye este programa, escrito en TI 58, pero que funciona lo mismo en TI 59, o mejor funcionará con una participación de memoria apropiada, cambiando los números de registros utilizados.

Al principio del programa, la participación 5 Op. 17 reserva 50 memorias de datos. Algunas se llenan con números del retorno en partición normal (3 Op. 17), en instrucciones. Ciertamente, el programa producido no es de los más óptimos, pero funciona. Sería necesario, como es lógico, encontrar una regla de codificación simple para utilizar estas posibilidades. Suponiendo, no obstante, que ofrezcan un interés aparte del anecdótico.

Aquí exponemos la lista del programa generado por el programa listado anteriormente. Comienza en el paso 80 y termina en el 111.
Entre en programa fuente . Haga RST, R/S. Su calculadora va a ponerse a contar sola, cuando esto no es en absoluto lo que prevé el programa fuente .
En realidad, la cuenta la realiza el programa objeto que gira en los pasos 80 y siguientes.

(Denis Pivorard)

```
000  05   5     015  05   5      030  03   3
001  69  DP     016  00   0      031  02   2
002  17   17    017  01   1      032  06   6
003  93   .     018  08   8      033  06   6
004  00   0     019  05   5      034  03   3
005  01   1     020  03   3      035  02   2
006  01   1     021  02   2      036  42  STO
007  07   7     022  55   ÷      037  47   47
008  06   6     023  01   1      038  93   .
009  42  STO    024  00   0      039  00   0
010  49   49    025  95   =      040  01   1
011  93   .     026  42  STO     041  01   1
012  03   3     027  48   48     042  42  STO
013  02   2     028  93   .      043  46   46
014  09   9     029  00   0      044  03   3
                                 045  69  DP
                                 046  17   17
```

```
                                 101  32  X:T
075  00   0     088  24  CE      102  66  PAU
076  00   0     089  00   0      103  32  X:T
077  00   0     090  00   0      104  24  CE
078  00   0     091  32  X:T     105  00   0
079  00   0     092  85   +      106  00   0
080  24  CE     093  01   1      107  00   0
081  00   0     094  95   =      108  00   0
082  00   0     095  32  X:T     109  00   0
083  00   0     096  24  CE      110  00   0
084  00   0     097  00   0      111  11   A
085  00   0     098  00   0      112  00   0
086  76  LBL    099  00   0      113  00   0
087  11   A     100  00   0      114  00   0
```

La caza del ratón (juego)

Paso de programa	Código de tecla	Instrucción
000	76	2nd Lbl
001	11	
002	29	2nd CP
003	47	2nd CNs
004	42	STO
005	11	11
006	06	6
007	00	0
008	42	STO
009	10	10
010	71	SBR
011	10	2nd E'
012	43	RCL
013	15	15
014	42	STO
015	00	0
016	71	SBR
017	10	2nd E'
018	43	RCL
019	15	15
020	42	STO
021	01	1
022	43	RCL
023	02	2
024	75	-
025	43	RCL
026	00	0
027	95	- (igual)
028	50	2nd/x/
029	42	STO
030	05	5
031	43	RCL
032	03	3
033	75	-
034	43	RCL
035	01	1
036	95	
037	50	2nd /x/
038	44	SUM
039	05	5
040	25	CLR
041	43	RCL
042	03	3
043	55	:
044	01	1
045	00	0
046	85	+
047	43	RCL
048	02	2
048	02	2
049	95	- (igual)
050	58	2nd Fix
051	91	1
052	91	R/S
053	43	RCL
054	10	10
055	55	:
056	01	1
057	00	0
058	00	0
059	85	+
060	43	RCL
061	05	5
062	95	=
063	58	2nd Fix
064	02	2
065	91	R/S
066	76	2nd Lbl
067	12	B
068	42	STO
069	06	6
070	58	2nd Fix
071	00	0
072	91	R/S
073	42	STO
074	07	7
075	43	RCL
076	06	6
077	50	6
077	50	2nd /x/
078	85	+
079	43	RCL

Trucos de la TI-59

Todos, cuando hemos conseguido tener en nuestras manos la calculadora, después de absorber ávidamente las instrucciones generales, hemos introducido el ejemplo de la página iv-101 del manual: El juego del código secreto.

Una vez tecleado y depurado de los posibles errores de entrada, siguiendo las instrucciones, nos hemos puesto a la tarea de demostrarle a la máquina y a nuestros amigos quién es el dueño de la situación.

Cual no será sin embargo nuestra decepción y la alegría de nuestros amigos al comprobar que a veces este programa falla de una forma lamentable escogiendo como cifra del número secreto el número 10.

Los más animosos, sin embargo no han perdido la fe en su capacidad, y deciden destripar el programa (cosa nada difícil por la claridad con que está desarrollado) y encontrar el malicioso duende. Pero este duende es esquivo, y tras larga cacería no se encuentran rastros de él.

El fallo del programa se debe a que algunas instrucciones tienen efectos secundarios, perfectamente detallados en el manual y que hay que tener en cuenta a la hora de redactar un programa.

En este caso las instrucciones responsables del mal funcionamiento del programa son (FIX) y (EE).

Efectivamente, en el paso de programa 107 encontramos la instrucción (FIX) (1) encargada de dar formato a la respuesta, y en la generación del código secreto tenemos en el paso 20 una multiplicación por diez de la forma (EE) (1).

¿Qué tienen que ver estas dos partes del programa, en principio independientes, y tan alejadas entre sí?.

Se puede observar que el programa sólo falla a partir del segúndo juego.

Efectivamente, cuando vamos a jugar por segunda o sucesivas veces, nos encontramos normalmente con la condición (FIX) (1) activa, y por lo tanto, si en los pasos 16 a 18 el programa 15 del módulo nos devuelve un número mayor que 0,995 tendremos en pantalla 1,0 y al hacer EE 1 cogerá tan sólo las cifras visualizadas y las multiplicará por 10 dando el fatídico 10.

Para eliminar este resultado tan molesto basta eliminar los pasos 20, 21 y 24 y añadir en lugar del 20 antiguo:

020	65	(X)
021	01	(1)
022	00	(0)
023	95	(=)

con lo cual perdemos un paso de programa y el tan molesto efecto.

Atentos todos.

A pesar de lo mucho que lo juran y lo perjuran en cualquier lugar donde lo mires, si que se utiliza el (=) en algún programa del módulo MASTER-1.

Estos son algunos de los ejemplos encontrados:

programa	paso de programa	rutina
02	362	D
	410	E
	435	E
	461	E?
	472	
03	016	
.....		

Gerardo Izquierdo Cadalso.

Aterrizaje Lunar (juego)

Para iniciar el programa pulsar E irán apareciendo en pantalla alternativamente los siguientes datos:
500 Altitud comienzo
-50 Velocidad bajada
120 Combustible que tenemos
20 Tiempo con que contamos para realizar el aterrizaje
0 Intermitente único momento en que podemos parar el programa para introducir los litros de combustible.

Si en este punto introduciesemos 20 litros los parámetros serían:
20 R/S 457'5 Altitud
-35 Velocidad de bajada
100 Combustible
19 Tiempo
0 Intermitencia para introducir nuevos datos pulsando R/S.

Se hace notar que si no se para el programa en el momento de visualizar 0 el programa continúa solo:
420 Altitud
-40 Velocidad de bajada
100 Combustible
18 Tiempo que queda
0 Intermitencia para parar el programa.

El margen para realizar un aluzinaje perfecto está situado entre los siguientes parámetros:
+ — 2 metros
+ — 2 velocidad

Si no estamos dentro de estos parámetros nos estrellaremos y la visualización marcará 99999999 intermitentemente.

José María Yus

Paso de programa	Código de tecla	Instrucción
000	76	2nd Lbl
001	15	E
002	47	2nd CMS
003	05	5
004	00	0
005	00	0
006	42	STO
007	00	00
008	05	5
009	00	0
010	94	+-
011	42	STO
012	01	01
013	01	1
014	02	2
015	00	0
016	42	STO
017	02	02
018	02	2
019	00	0
020	42	STO
021	09	9
022	43	RCL
023	00	0
024	66	2nd Pause
025	66	2nd Pause
026	43	RCL
027	01	1
028	66	2nd Pause
029	66	2nd Pause
030	43	RCL
031	02	2
032	66	2nd Pause
033	66	2nd Pause
034	43	RCL
035	09	9
036	66	2nd Pause
037	25	CLR
038	91	R/S
039	76	2nd Lbl
040	11	A
041	69	2nd Op
042	26	26
043	50	2nd /x/
044	42	STO
045	05	05
046	02	2
047	01	1
048	32	x cambia t
049	43	RCL
050	06	6
051		
67	2nd x = t	
052	16	2nd A'
053	43	RCL
054	05	5
055	22	INV
056	44	SUM
057	02	2
058	29	2nd CP
059	43	RCL
060	05	5
061	75	-
062	05	5
063	95	=
064	42	STO
065	03	3
066	55	:
067	02	2
068	95	=
069	85	+
070	43	RCL
071	01	1
072	85	+
073	43	RCL
074	00	0
075	95	=
076	*42*	*STO*
077	*00*	*0*
078	*43*	*RCL*
079	*03*	*3*
080	*85*	+
081	43	RCL
082	01	1
083	95	=
084	42	STO
085	01	1
086	25	CLR
087	29	2nd CP
088	43	RCL
089	00	0
090	22	INV
091	77	2nd x mayor igual t
092	12	B
093	06	6
094	32	x cambia t
095	43	RCL
096	00	0
097	22	INV
098	77	2nd x mayor igual t
099	14	D
100	43	RCL
101	00	0
102	66	2nd Pause
103	66	2nd Pause
104	66	2nd Pause
105	66	2nd Pause
106	43	RCL
107	01	1
108	66	2nd Pause
109	66	2nd Pause
110	66	2nd Pause
111	66	2nd Pause
112	43	RCL
113	02	2
114	66	2nd Pause
115	66	2nd Pause
116	66	2nd Pause
117	66	2nd Pause
118	69	2nd Op
119	39	39
120	43	RCL
121	09	9
122	66	2nd Pause
123	25	CLR
124	29	2nd CP
125	01	1
126	32	x cambia t
127	43	RCL
128	01	1
129	77	x mayor igual t
130	17	2nd B'
131	25	CLR
132	66	2nd Pause
133	66	2nd Pause
134	66	2nd Pause
135	66	2nd Pause
136	66	2nd Pause
137	61	GTO
138	11	A
139	76	2nd Lbl
140	14	D
141	29	2nd CP
142	01	1
143	32	x cambia t
144	43	RCL
145	01	1
146	77	x mayor igual t
147	17	2nd B'
148	03	3
149	94	+ -
150	32	x cambia t
151	43	RCL
152	01	1
152	01	1
153	22	INV
154	77	x mayor igual t
155	12	B
156	13	C
157	76	2nd Lbl
158	12	B
159	25	CLR
160	35	1/x
161	91	R/S
162	76	2nd Lbl
163	13	C
164	25	CLR
165	43	RCL
166	06	6
167	52	EE
168	91	R/S
169	76	2nd Lbl
170	16	2nd A'
171	02	2
172	00	0
173	65	x
174	65	x
175	91	R/S
176	76	2nd Lbl
177	17	2nd B'
178	25	CLR
179	11	A

080	07	7
081	50	2nd /x/
082	95	=
083	94	+/-
084	85	+
085	43	RCL
086	10	10
087	95	6
088	42	STO
089	10	10
090	32	x cambia t
091	00	0
092	77	2nd X mayor igual t
093	16	2nd A'
094	29	2nd CP
095	43	RCL
096	02	2
097	85	+
098	43	RCL
099	06	6
100	95	=
101	42	STO
102	16	16
103	43	RCL
104	03	3
105	85	+
106	43	RCL
107	07	7
108	95	=
109	42	STO
110	17	17
111	01	1
112	00	0
113	32	x cambia t
114	43	RCL
115	16	16
116	77	2nd x mayor igual t
117	17	2nd B'
118	43	RCL
119	17	17
120	77	2nd x mayor igual t
121	17	2nd B'
122	00	0
123	32	x cambia t
124	43	RCL
125	17	17
126	77	2nd x mayor igual t
127	01	1
128	30	30
129	17	2nd B'
130	43	RCL
131	16	16
132	77	2nd x mayor igual t
133	01	1
134	36	36
135	17	2nd B'
136	43	RCL
137	02	2
138	85	+
139	43	RCL
140	06	6
141	95	=
142	42	STO
143	02	2
144	43	RCL
145	03	3
146	85	+
147	43	RCL
148	07	7
149	95	=
150	42	STO
151	03	3
152	29	2nd CP
153	43	RCL
154	00	0
155	32	x cambia t
156	43	RCL
157	02	2
158	67	2nd x=t
159	01	1
160	64	64
161	61	CTO
162	00	22
163	22	
164	43	RCL
165	01	1
166	32	x cambia t
167	43	RCL
168	03	3
169	67	2nd x=t
170	18	2nd C'
171	61	GTO
172	00	22
173	22	
174	91	R/S

SUBRUTINAS

240	76	znLbl
241	10	E'
242	43	RCL
243	11	11
244	85	+
245	43	RCL
246	12	12
247	85	+
248	04	4
249	03	3
250	09	9
251	01	1
252	04	4
253	07	7
254	95	=
255	50	2nd /x/
256	42	STO
257	12	12
258	01	1
259	85	+
260	52	EE
261	08	8
262	95	=
263	42	STO
264	13	13
265	43	RCL
266	12	12
267	65	X
268	02	2
269	03	3
270	95	=
271	42	STO
272	14	14
273	55	:
274	43	RCL
275	13	13
276	95	=
277	59	2nd Int
278	65	X
279	43	RCL
280	13	13
281	95	=
282	94	+/-
283	85	+
284	43	RCL
285	14	14
286	95	=
287	42	STO
288	12	12
289	55	:
290	43	RCL
291	13	13
292	65	X
293	01	1
294	00	0
295	95	=
296	59	2nd Int
297	42	STO
298	15	15
299	92	INV SBR
300	76	2nd Lbl
301	16	A'
302	25	CLR
303	65	X
304	65	X
305	91	R/S
306	76	2nd Lbl
307	17	B'
308	25	CLR
309	35	1/x
310	66	Pause
311	66	Pause
312	66	Pause
313	61	GTO
314	00	65
315	65	
316	76	2nd Lbl
317	18	C'
318	69	2nd Op
319	24	24
320	01	1
321	00	0
322	44	SUM
323	10	10
324	43	RCL
325	04	4
326	66	Pause
327	66	Pause
328	25	CLR
329	43	RCL
330	04	4
331	66	Pause
332	66	Pause
333	61	GTO
334	00	10
335	10	10
336	91	R/S

Para comenzar el juego en primer lugar se introduce un número cualquier de tantas cifras como se quiera, entero o decimal, pulsando la tecla A la visualización de la pantalla será 0.0 que corresponderá a nuestra posición inicial dentro del tablero de la Fig. 1. Pulsando seguidamente la tecla R/S la máquina calculará la distancia absoluta a la que estamos del «ratón» y la cantidad de «combustible» de nuestro depósito. Para ir de unas casillas a otras deberemos introducir el número de casillas que nos queremos mover (arriba o abajo, a derecha o izquierda) y pulsar R/S.

Pongamos un ejemplo de partida:

Para iniciar el juego un número cualquiera (42345678) y pulsamos A

42345678 A 0.0
Posición inicial dentro del tablero.
R/S 9.60
(9 distancia albsolutal al blanco
60 cantidad de combustible)
4 R/S 4
En el eje de las "X" estamos en el punto 4
5 R/S 4.5
Estamos en el punto 4.5 del tablero
R/S 8.51
(8 distancia absoluta al blanco)
(51 reserva de combustible)
4 R/S 4
(nos hemos movido 4 puntos a la derecha de X)
-4 R/S 1
(Destellante le hemos alcanzado)
8.1
(Posición que ocupamos ahora en el tablero)
(Distancia absoluta al blanco)
R/S 1.53
(53 reserva de combustible)
0 R/S 0
(seguimos estando en el punto 8 de las X)
-1 R/S 8.0
(Posición actual)
(2 distancia al blanco)
R/S 2.52
(52 reserva de combustible)

y así sucesivamente hasta alcanzar el blanco.

El juego se acaba cuando se acaba el combustible, cuando se acierta al blanco el combustible se incrementa en 10 unidades, los desplazamientos de unas casillas a otras van mermando combustible.

Cuando por error se «va» el disparo fuera de los límites marcados por el tablero de 9 × 9 aparece en la visualización unos 9. 99 intermitentes pulsando la tecla CLR se borran y hay que intentar entrar dentro del límite del tablero. Cuando se acaba el combustible aparece un 0 intermitente en la visualización.

José María Yus

Comentario al juego de la caza del ratón

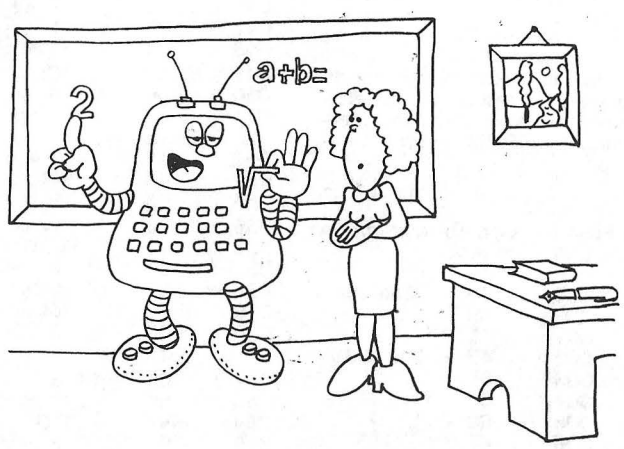
Queremos hacer un comentario a este programa.

La rutina E' genera un número al azar entre 0 y 9 ambos incluidos. Se podría cambiar por

:
LBL
E'
PGM
15
SBR
DMS
X
1
0
=
INT
STO
15
RTN

Teniendo que cambiar en los pasos 74, 00, 107 y 148 la memoria 7 por la 8, y en el paso 5 la memoria 11 por la 9.

Gerardo Izquierdo



Las ideas del ZX81

No maltrates al "abuelo" estas vacaciones y sobre todo ojito con el sol. Toma unas "recetas" que sin duda le darán vitalidad.

Errores trucos y consejos de programación para el ZX 81

Atención los matemáticos: se ha encontrado un error en el ZX81 que quizá se traduzca en otros errores al realizar operaciones matemáticas.

Escriba PRINT 2**32 en el ZX81 y realice la operación con una calculadora. Comprobará que los resultados son bastante diferentes.

Un error de la impresora: ejecute un LPRINT valores entre 0.00001 y 0.0099, lo que salga por la impresora puede resultarle bastante extraño.

Hay una serie de errores matemáticos que se producen frecuentemente en el ZX81, pero sólo los que llevan la primera versión de la ROM, que **tiene 3 bytes incorrectos**. En algunos ZX81 estos bytes quedan deshabilitados por unas chapuzas soldadas al micro-procesador Z80. Estas chapuzas son un par de circuitos integrados de puertas lógicas.

Parece que algunos han tenido dificultades al cargar el programa de música del folleto, pues no podían borrar las líneas sobrantes una vez introducido el código máquina. Sinceramente, no sabemos todavía a qué puede ser debido este fallo.

Parece que muchos usuarios tienen problemas con el cassette. Unos no pueden grabar programas, otros no pueden cargarlos, ni incluso los que han grabado previamente, a otros les aparece siempre la K en vez del 0/0, y a otros el ZX81 les sale "artista" con un poco de "arte abstracto" en la pantalla.

Todos estos problemas son debidos en su mayor parte al aparato reproductor/grabador de cassettes utilizado. Primeramente **no sirven los aparatos estéreo ni los de alta fidelidad**, aunque se puedan conmutar a mono. No decimos esto como un axioma, sino que es lo que ha sido comprobado por norma general, además un aparato de alta fidelidad puede incluso dañar al ZX81.

Segundo: es casi siempre necesario con la mayoría de los aparatos desconectarle el cable que no se usa, es decir, cuando se está cargando un programa al ZX81, desconectar el cable de MIC, y viceversa, pues **hay muchos aparatos que sufren realimentaciones internas a través de los cables.**

Tercero: es importante que los cabezales de grabación y reproducción estén limpios y alineados correctamente. También es posible que no tenga la suficiente potencia. Esto se puede ver porque las líneas que aparecen en la pantalla al cargar un programa al ZX81 **deben tener al menos el mismo grosor las blancas que las negras.** En caso de que las negras sean mucho más delgadas es que no hay suficiente volumen, y si se ve todo negro es que nos hemos pasado.

Para cargar programas al ZX81 que no hayan sido grabados mediante el aparato utilizado normalmente, nos encontramos casi siempre con el problema de hallar los **niveles adecuados de tono y volumen.** Se recomienda normalmente poner el tono al máximo de agudos y el volumen a tres cuartos del máximo. Sin embargo es muy posible que haya que probar, diez, veinte o cien combinaciones antes de hallar la adecuada. Esto se puede ajustar también observando las líneas en la pantalla, pues llega un momento en que por ellas se llega a conocer si se está cargando correctamente un programa.

Si a pesar de todo no se consiguen cargar los programas, mejor cambiar de aparato o llevar a revisar el ZX.

Otra clase de problemas nos los proporciona la fijación del módulo de 16K RAM. A la que se mueve un poquito desaparece el texto de la pantalla y todo el programa (que puede ser muy largo y tal vez unas cuantas horas de trabajo) que había en el ZX81. La primera recomendación es tener el máximo cuidado en mover el ZX81. Pero como esto no resulta muy práctico y todos estamos escarmentados y andamos con pies de plomo, ¿por qué no probar a fijar el módulo de alguna manera? por ejemplo con cinta adhesiva o si alguno se atreve, soldando varias capas de estaño sobre las pistas del circuito impreso del ZX81 donde se conecta el módulo, o fijando todo el conjunto en un tablero o caja. Todo esto suponiendo que no se le quiera añadir ningún otro accesorio al ZX81, claro.

Otro inconveniente que presenta el ZX81 es tener que desconectar la alimentación cada vez que **"se atasca"**. Esto tiene fácil solución: un interruptor entre el alimentador y la red.

Asímismo, también puede pensarse en un conmutador para no tener que andar conectando y desconectando los cables del cassette.

Un truco muy útil para saber donde empieza un programa (o termina), aparte del contador de vueltas, por supuesto: hay algunos cassettes en los que se puede oir al rebobinar o en avance rápido, así resulta mucho más fácil descubrir los huecos entre los programas.

Trucos y consejos de programación

1K de memoria son 2^{10} posiciones de memoria, pero a pesar de que el ZX81 tenga 1K de memoria interna, menos de la mitad están disponibles en la realidad para el usuario. 2^{10} son 1.024, pero aparte de las casi 150 posiciones que necesita el BASIC para conservar sus variables particulares, hay que saber que cada línea de programa, por el sólo hecho de existir, ocupa 5 bytes más las instrucciones y caracteres que contenga.
Estos 5 bytes o posiciones de memoria se reparten así: 2 bytes para el número de línea, 2 bytes para almacenar la longitud total de la línea en número de caracteres más 1 byte de NL. Así pues, conviene poner en una línea todo lo que se pueda. Esto reulta particularmente conveniente en instrucciones PRINT que vayan una detrás de otra. En lugar de escribir:

```
10PRINT AT 0,10;"HOLA"
20PRINT AT 1,8;"SOY EL ZX81"
30PRINT AT 2,8;"SINCLAIR".
```

Escribiremos:

```
10PRINT AT 0,10;"HOLA";AT 1,8;"SOY EL ZX
  81";AT 2,8; "SINCLAIR".□
```

Club Nacional de Usuarios del ZX-81

DATA READ y RESTORE Protección de un programa en BASIC Conversion DEC-HEX

* DATA, READ y RESTORE: Una de las herramientas más útiles en la programación, que se encuentra en el BASIC de otros ordenadores, y que se echa en falta en el ZX81 en muchas ocasiones, es, sin duda alguna, este conjunto de instrucciones. Suele utilizarse para llenar tablas de datos por programa, por lo que la solución que se propone en el manual del ZX81 para suplir esta falta, y que consiste en ir introduciendo los datos uno a uno, resulta muy engorrosa, si no absurda, ya que no representa una solución válida al problema.

La rutina que se propone a continuación, que por su longitud es aconsejable usar sólo con los 16K, pretende proporcionar al ZX81 la posibilidad de disponer de alguna manera de esas instrucciones muy útiles en bastante casos. Los datos, que pueden ser de cualquier tipo y que en otros BASICs se colocarían en líneas de programa con la instrucción DATA, se colocan aquí en líneas REM una detrás de otra a partir de una posición en la zona de memoria reservada a los programas, determinada por la variable INICIO, que debe contener la primera dirección de memoria después del primer REM. Para simplificar, aquí se ha utilizado la 16514, que está en la primera línea de programa. Los datos deben ir separados por comas (,) y al final de cada sentencia REM se coloca otro carácter separador (en este caso se ha escogido "/"). Tras el último dato de la última sentencia REM con datos se coloca otro carácter separador diferente del anterior ("*" por ejemplo).

En el programa se definen mediante unas variables las líneas donde empiezan las subrutinas READ y RESTORE (en el ejemplo 1000 y 2000 respectivamente), para poderlas llamar mediante GOSUBs. Es decir, GOSUB READ cuando se quiera leer un dato, y GOSUB RESTORE. La rutina sólo permite la lectura de un dato a la vez, aunque puede ser fácilmente modificada para leer varios al mismo tiempo. Los datos se tratan como cadenas de caracteres, aunque indudablemente pueden tratarse como números, tratando el dato correspondiente con la función VAL. Cada dato leído queda almacenado en la variable A$. Una vez leídos todos los datos, la rutina se inicializa automáticamente al primero. Mediante GOSUB RESTORE puede re-inicializarse la lectura aunque no se haya terminado de leerlos.

```
 10  REM SEAT, RENAULT, FIAT, OPEL,
     SIMCA, TALBOT, FORD/
 20  REM CITROEN, PEUGEOT, ALFA
     ROMEO, PORSCHE, FERRARI/
 30  REM BMW, MERCEDES, VOLKSWAGEN,
     VOLVO, ROVER, BUICK*
 35  LET INICIO=16514
 40  LET READ=1000
 50  LET RESTORE=2000
1000 REM *READ A$*
1005 LET A$=" "
1010 LET B$=CHR$ PEEK (INICIO)
1014 IF B$="/" THEN LET INICIO-INICIO+7
1016 IF B$="/" THEN GOTO 1060
1018 IF B$="*" THEN GOSUB RESTORE
1019 IF B$="*" THEN GOTO 1060
1020 IF B$="," THEN GOTO 1050
1030 LET A$=A$+B$
1035 LET INICIO=INICIO+1
1040 GOTO 1010
1050 LET INICIO=INICIO+1
1060 RETURN
2000 REM *RESTORE*
2010 LET INICIO=16514
```

```
2020  RETURN
```

Evidentemente, los datos utilizados en las líneas REM forman parte del ejemplo, en el que se insertarían las llamadas a las subrutinas entre las líneas 50 y 1000, según el programa del usuario en que se necesiten. Un par de ejemplos podrían ser los siguientes (el primero va sacando por pantalla los datos leídos, y el segundo sirve para llenar una tabla de datos, se recomienda hacerlo en modo FAST para tablas grandes).

EJEMPLO 1

```
 55  FOR A=1 TO 22
 60  GOSUB READ
 70  PRINT A$
 80  NEXT A
 90  GOSUB RESTORE
100  PAUSE 100
110  CLS
120  GOTO 55
130  STOP
```

EJEMPLO 2

```
 60  FAST
 70  DIM C$(18,10)
 80  FOR A=1 TO 18
 90  GOSUB READ
100  LET C$(A)=A$
110  NEXT A
120  SLOW
130  STOP
```

```
*  RUTINA PARA ENTRAR PAREJAS DE
   DATOS CON UN SOLO NEWLINE:
100  INPUT A$
110  FOR I=1 TO LEN A$
120  IF A$(I)= "!" THEN GOTO 160
130  NEXT I
140  PRINT "FALTA DATO 2"
150  GOTO 100
160  LET A=VAL A$(TO I-1)
170  LET B=VAL A$(I+1 TO)
```

Los datos sólo pueden ser numéricos. En la línea 100 se entran los dos números separados por un espacio, y luego NEWLINE. Si sólo se introduce un dato tiene lugar un aviso. (Joan Sales Roig).

```
*  CLS PARCIAL: Hacer la primera línea
   1 REM t!LN GsTAN
```

(esto es un pequeño programa en código máquina, LN es logaritmo neperiano y TAN, tangente). Para hacerlo funcionar en el programa debe ponerse un POKE 16515,n siendo n el número de líneas a borrar empezando por abajo y contando las 24, RAND USR 16514 ejecuta el CLS parcial según el número de líneas especificado. (Joan Sales Roig).

*** Protección de un programa en BASIC:**

- Hacer la primera línea del programa: PRINT AT 10,12; "PRIVADO".
- Para bloquearlo: POKE 16514,254 o un número mayor que 63. Para desbloquearlo: POKE 1654,0.
- Queda protegido frente a LIST, RUN, y borrado o sobreescrito en líneas, salvo la primera. (Joan Sales Roig).

*** RESTART:** El mismo efecto que desenchufar y volver a enchufar el ZX81, se consigue con RAND USR 0, siempre que el ZX81 no se haya atascado de tal forma que sea imposible entrar cosas por el teclado, claro. (Joan Sales Roig).

*** CONVERSION DEC-HEX.** Gabriel Indalecio Cano (201).

Este programa convierte números decimales al código hexadecimal, tabulándolos en pantalla. Hay que introducir el número de octetos, es decir, el número de caracteres que va a poseer como máximo el número hexadecimal.

```
  5  REM CONVERSION DEC-HEX
 10  PRINT "INTRODUZCA NUMERO MAXIMO
     DE OCTETOS"
 20  INPUT N
 25  CLS
 30  PRINT "INTRODUZCA LOS NUMEROS
     SEGUIDOS DE NL"
 35  PRINT
 40  PRINT "DECIMAL", "HEXADECIMAL"
 50  PRINT "(7G7)","(10G7)"
 60  LET A=16**(N/2)
 70  LET A$=" "
 80  INPUT B
 90  LET D=B
 95  IF A > B THEN GOTO 130
100  LET C=INT(B/A)
110  LET B=B-C*A
120  LET A$=A$+CHR$(C+28)
130  LET A=A/16
140  IF A < 1 THEN GOTO 160
150  GOTO 100
160  PRINT D,A$
170  GOTO 60
```

*** CONVERSION HEX-DEC.** Gabriel Indalecio Cano (201).

Como su nombre indica, el programa convierte los números introducidos en hexadecimal a decimal, no importando la longitud de estos. Cada uno de ellos se introducirá seguido de NL. La pantalla queda de esta forma:

```
INTRODUZCA LOS NUMEROS SEGUIDOS DE
NEWLINE.

HEX                DEC
FF                 255
08                   8
23                  35
...                 ...
```

```
  5  REM CONVERSION HEX-DEC
 10  PRINT "INTRODUZCA LOS NUMEROS
     SEGUIDOS DE NEWLINE".
 20  PRINT
 25  PRINT "HEX","DEC".
 30  PRINT "(3G7)", "(3G7)"
 35  LET C=1
 40  LET B=0
 45  INPUT A$
 50  LET C$=A$
 55  LET N=LEN A$
 60  LET A$=A$(TO N)
```

```
65 LET B$=A$(N TO)
70 IF CODE B$>43 OR CODE B$<28 THEN
   GOTO 120
75 LET B=B+(CODE (B$)-28)*C
80 LET C=C*16
85 LET N=N-1

 90 IF N<1 THEN GOTO 105
100 GOTO 65
105 PRINT C$,B
110 GOTO 35
120 PRINT "ERROR"
130 GOTO 35
```

CLUB NACIONAL DE USUARIOS DEL ZX81

NOTA:

La normalización adoptada es la siguiente:

i :significa "espacio".

Las letras minúsculas significan el carácter gráfico de la tecla correspondiente.

Para indicar los caracteres gráficos de las teclas numéricas, se ponen entre paréntesis, por ejemplo, (G1) significa el carácter gráfico de la tecla 1.

Para indicar varios caracteres iguales seguidos se ponen entre paréntesis, por ejemplo: (12!), significa 12 espacios en blanco. (12G1), significa 12 caracteres gráficos de la tecla 1. (12s), significa 12 caracteres gráficos de la tecla S.

Los caracteres en video inverso se indican subrayándolos; por ejemplo: "ZX81".

Así, en una misma cadena de caracteres, se puede encontrar: "(12!) (12G1)st(G1) ZX81! (12s)", este es ejemplo algo exagerado.

En el truco de "Protección de un programa en BASIC", la posición a POKEar para desbloquear o bloquear el programa es la 16543, y no la 16514 ni la 1654, como aparece en el texto. □

Trucos y consejos de programación

1K de memoria son 2^{10} posiciones de memoria, pero a pesar de que el ZX81 tenga 1K de memoria interna, menos de la mitad están disponibles en la realidad para el usuario. 2^{10} son 1.024, pero aparte de las casi 150 posiciones que necesita el BASIC para conservar sus variables particulares, hay que saber que cada línea de programa, por el sólo hecho de existir, ocupa 5 bytes más las instrucciones y caracteres que contenga. Estos 5 bytes o posiciones de memoria se reparten así: 2 bytes para el número de línea, 2 bytes para almacenar la longitud total de la línea en número de caracteres más 1 byte de NL. Así pues, conviene poner en una línea todo lo que se pueda. Esto resulta particularmente conveniente en instrucciones PRINT que vayan una detrás de otra. En lugar de escribir:

```
10 PRINT AT 0,10; "HOLA"
20 PRINT AT 1,8;"SOY EL ZX81"
30 PRINT AT 2,8;"SINCLAIR"
```

Escribiremos:

```
10 PRINT AT 0,10;"HOLA";AT 1,8;"SOY EL
   ZX81";AT 2,8; "SINCLAIR"
```

Esto puede ser también muy útil al utilizar el evaluador lógico de expresiones del BASIC: si una expresión entre paréntesis es verdadera toma valor 1, y 0 si no lo esl Por ejemplo, un programa que sumase 1 a la variable A si se detecta la pulsación de la tecla "8" en el teclado, y que reste uno si se detecta "5". Este podría ser una rutina a utilizar para mover algo en la pantalla, por lo tanto hay que hacer que no pase de los extremos laterales de la misma. Como la variable A contendrá el valor de la columna donde está la cosa a mover, este valor no puede ser menor que 0 ni mayor que 31.

El programa, sin utilizar el evaluador lógico podría ser una cosa así:

```
10 LET A=15
20 IF INKEYS="5" THEN LET A=A-1
30 IF A<0 THEN LET A=0
40 IF INKEYS="8" THEN LET A=A+1
50 IF A >31THEN LET A=31
60 PRINT AT 12,A;"V"
70 GOTO 20
```

Esta misma rutina, usando el evaluador lógico, queda:

```
10 LET A=15
20 LET A=A - (INKEY$="5" AND A>0) +
   (INKEY$="8" AND A<31)
30 PRINT AT A,12; "V"
40 GOTO 20
```

Como se podrá comprobar, a esta rutina para mover cosas en la pantalla le falta un trozo que borre las posiciones previas de la cosa. Como ejercicio, se recomienda probar a completarla.

Hay otro elemento que contribuye en gran parte a "comerse" la memoria disponible a grandes bocados. Las variables numéricas utilizadas en el BASIC del ZX81 se almacenan en formato de coma flotante. Esto, en la práctica viene a resultar en que se pueden manejar números enteros y también con decimales o en forma exponencial (el ZX80 sólo permitía números enteros). Pero ocurre que por tener estos números tanta precisión, se necesitan muchas posiciones de memoria para almacenarlos. En concreto cada número o variable numérica ocupa 5 bytes más los caracteres que tenga. Por ejemplo:

7 ocupa 6 bytes
10 ocupa 7 bytes
12.34255 ocupa 13 bytes

Así pues, es conveniente aprovechar las variables en distintas partes del programa, ya que **sólo ocupan los 5 bytes cuando se ejecutan en el programa**, es decir, cuando quedan inicializadas por el programa. Por lo tanto:

10 LET A=0 ocupa 15 bytes, pero una vez ejecutado ocupa 5 más en la zona de variables de la memoria.

Si hay varias variables que se inicializan al mismo valor es pues interesante hacerlo de la siguiente forma:

```
1Ø  LET A=Ø
2Ø  LET B=A
3Ø  LET C=A
```

Las líneas 2Ø y 3Ø ocupan sólo 9 bytes hasta que se ejecutan.

Los que ya conocen el BASIC de otros ordenadores habrán echado muy en falta las instrucciones DATA-READ y RESTORE. Dado que normalmente se usaban para llenar tablas de datos, la forma como se deben implementar en el ZX81 consiste en hacer un pequeño programa que vaya almacenando datos entrados por el usuario en la tabla.
Por ejemplo:

```
1Ø  DIM A(5)
2Ø  FOR B= 1 TO 5
3Ø  INPUT A(B)
4Ø  NEXT B
```

Una vez ejecutado este programa y entrados los números que se precisan en el programa como datos. Se escribirá el resto del programa y se ejecutará con un GOTO número de línea, pues si se pone RUN se borran todas las variables anteriores, siendo una de ellas la tabla de datos entrados. Estas tablas de datos pueden ser almacenadas en cassette junto con el programa, para disponer de ellas, teniendo cuidado de no iniciar la ejecución con RUN, sino con GOTO.

Para simular la lectura (READ) de la tabla deberá implementarse en el programa un contador que vaya contando de uno en uno cada vez que se coja un dato de la tabla y que se ponga a cero cuando llegue a 5.

Por ejemplo:

```
1Ø  LET C=1
2Ø  REM READ D
3Ø  LET D=A(C)

4Ø  LET C=C+1
5Ø  IF C=6 THEN LET C=1
6Ø  RETURN (ya que en realidad, se trataría de
    una subrutina en este caso a partir de la línea
    2Ø, para implementar la función READ).
```

Pasemos ahora a emplear el PEEK que representa tanto misterio para muchos usuarios del ZX81. Según el manual de programación, en el capítulo 28 se nos indican las posiciones de memoria empleadas por las variables particulares del BASIC del ZX81, y lo que almacena cada una de esas variables. Vemos que en las posiciones 16396 y 16397 está guardada la dirección de memoria donde empieza la zona de memoria reservada a la pantalla, es decir, donde se almacenan los caracteres que deben salir en la pantalla y que en algunos momentos podrá ser el listado del programa y en otros, los marcianos, los tiros, y el disparador, por ejemplo.

Estas posiciones de memoria nos van a resultar muy útiles para tratar de detectar si un carácter va a ocupar el sitio de otro (por ejemplo, cuando el disparo toca a un marciano).

Según el programa en que utilizaríamos esta rutina, el disparo se movería en la pantalla por ejemplo mediante un bucle con una instrucción PRINT AT. Queremos saber entonces dentro del bucle, si en alguna posición van a coincidir el disparo y el marciano.

Para ello exploramos en ese momento donde empieza la memoria de pantalla, pasando el valor hexadecimal almacenado en las dos posiciones de memoria mencionadas, a decimal:

```
LET P=PEEK 16396+256*PEEK 16397
```

Sabemos ahora que la memoria de pantalla empieza en la posición P, y que dentro de esa memoria de pantalla están el disparo y el marciano. Ahora debemos considerar que en la memoria de pantalla hay 22 líneas de 32 caracteres más NL, es decir 22 x 33 posiciones de memoria. El programa sabe que debe poner el disparo en la posición definida por PRINT AT, por ejemplo, PRINT AT A,B siendo los valores de A y B en esos precisos momentos dentro del bucle A=15 y B=18. Entonces hay que explorar esa posición A=15 y B=18 a ver si hay un marciano ahí antes de colocarle el disparo. Es decir, a P le sumamos (18 x 33) + 15 y vemos lo que hay en la posición de memoria definida por el nuevo P. Si corresponde al carácter que hemos definido como un marciano, es que el disparo le ha alcanzado.

Hemos visto pues que PEEK coge el carácter contenido en la posición de memoria n (PEEK n). Lo que hará POKE n,m es colocar en la posición de memoria n, el caracter con código m.

Se ha tratado de explicar y parece que no muy claramente algunos de los trucos de programación del ZX81. Esperamos mejorar en el próximo boletín.

Hay otra manera de conseguir detectar cuándo se van a superponer dos caracteres mediante el acceso directo a memoria con PEEK y POKE, estudiando las posiciones 16398 y 16399. A ver quién se rompe la cabeza para explicarlo.

El siguiente programa usa el acceso directo a memoria en las posiciones 16396 y 16397 así como a la memoria de pantalla en general, y puede servir para demostrar la utilidad de la explicación anterior.

El juego consiste en guiar una nave esquivando los cuerpos celestes que van apareciendo. Las teclas 5 y 8 controlan el movmiento.

```
1Ø  CLS
2Ø  LET A$="
3Ø  LET A=15
4Ø  LET B=15
5Ø  LET T=Ø
9Ø  FOR N=Ø TO 21
```

```
1ØØ PRINT A$
1Ø5 LET P=PEEK (16396)+256*PEEK (16397)
11Ø FOR M=1 TO 4
12Ø POKE (P+1+(33*N)+INT (RND*3 2) ), 151
14Ø NEXT M
145 NEXT N
15Ø LET K=A+166+(PEEK (16396)+2 56*
    PEEK (16397) )
155 IF PEEK K=19Ø THEN GOTO 165
16Ø IF PEEK K<>128 THEN GOTO 3ØØ
165 PRINT AT 4,B,"□"                    <-
17Ø PRINT AT 5,A, "□"                   <-
175 LET B=A
18Ø LET A=A+ (INKEY$="8" AND A<3 1) -
    (INKEY$="5" AND A>Ø)
22Ø SCROLL
23Ø PRINT A$
235 LET P=PEEK (16396)+256*PEEK (16397)
24Ø FOR N⇔1 TO 4
26Ø POKE (P+694+INT (RND*32)),151
28Ø NEXT N
285 LET T=T+1
29Ø GOTO 15Ø
3ØØ PRINT AT 2Ø,Ø ,"SCORE="; T
31Ø PRINT "OTRA PARTIDA?"
32Ø INPUT B$
33Ø IF B$="S" THEN RUN
```

El carácter en inversa de la línea 170 es una "Y". Los de las líneas 20 y 165 son espacios.

Por último, veamos cómo crear una especie de base de datos con el ZX81. Para conocimiento general, diremos que el ZX81 tiene la posibilidad de almacenar las variables que hayan adquirido unos valores tras la ejecución de un programa, siempre y cuando no se vuelva a ejecutar con RUN, se borre con NEW, o se borren sólo las variables con CLEAR.

Esas variables se pueden pues almacenar, junto con el programa, en cassette, al hacer SAVE "nombre del programa".

Entonces de lo que se trata es de organizar esas variables de manera que estén ordenadas de alguna manera y el programa las pueda buscar, ordenar, clasificar, modificar, etc. cada vez que se carguen del cassette junto con el programa.

El tipo de variables que resulta más apropiado para todo esto serán las tablas, bien numéricas o alfanuméricas (de cadenas de caracteres).

Supongamos que se quieren almacenar los datos de 100 personas: nombre, dirección, ciudad, provincia, teléfono. Pensamos en un máximo de 20 caracteres para el nombre, otros 20 para la dirección, 15 para la ciudad, 15 para la provincia y 7 para el teléfono; total 77 caracteres (es una suposición, todos sabemos que ocuparían más caracteres). Dimensionamos pues una tabla alfanumérica de 100 líneas de 77 columnas, por decirlo de alguna manera, la primera instrucción será DIM X$ (100,77).

A continuación el ZX81 deberá ir preguntando los datos uno a uno para cada una de las 100 personas:

```
PRINT "NOMBRE",
INPUT N$
PRINT N$
PRINT "DIRECCION",
INPUT D$
PRINT D$

etc. . .
```

Luego hay que hacer que cada una de esas variables N$, D$, etc tengan exactamente los caracteres máximos que hemos definido. Por ejemplo:

```
LET N$=(N$+"(2Ø ¡)")(TO 2Ø)
```

y así para cada una de las variables de una misma persona. Una vez que todas las variables de un mismo sujeto tienen la longitud máxima en caracteres que se había asignado, se ponen todas juntas en la "línea" correspondiente de la tabla. Por ejemplo, suponemos que se trata de los datos del sujeto número n:

```
LET X$(n)=N$+D$+. . .
```

Y ya se puede pasar al sujeto n+1.

También se necesitará una rutina de búsqueda para encontrar los datos de un sujeto dentro de la tabla, sabiendo sólo su nombre, por ejemplo. Supongamos que se llama "PEPE". La rutina sería más o menos:

```
FOR A=1 TO 100
IF X$ (A,TO LEN "PEPE")="PEPE" THEN STOP
NEXT A
```

Cuando se ejecute el STOP es que la "línea" A de la tabla están los datos de PEPE.

Podremos dividir la línea en los datos correspondientes dividiéndola de nuevo en trozos de las longitudes asignadas a cada dato, pero teniendo en cuenta que al final de cada dato sobrarán seguramente algunos espacios en blanco. Ejercicio: programa para eliminar los espacios en blanco al final de una cadena de caracteres.□

Club Nacional de Usuarios del ZX-81

Lector de cintas

En algunas ocasiones por las prisas no hemos identificado el contenido de una cinta y no sabemos que programas contiene, este fenómeno se acentúa en aquellas personas que tienen la costumbre de introducir gran cantidad de programas en una sola cinta y no se acuerdan de hacer anotaciones.

El siguiente programa nos permite, sin cargarlos y listarlos uno a uno, que nos vayan apareciendo en pantalla los nombres con los que grabamos los programas.

Instrucciones:

1.— Introducir el programa y grabarlo antes de ejecutarlo.
2.— Pulsar RUN.
3.— Borrar las líneas: de la 5 a la 60 inclusives.
4.— Hacerlo funcionar pulsando RUN 100 o RUN si se han borrado las líneas del punto anterior.
5.— Conexionar el casete en modo LOAD y ponerlo en marcha (PLAY) con la cinta

TARJETA DE SUSCRIPCION A EL ORDENADOR PERSONAL

EL ORDENADOR PERSONAL
Servicio de Suscripciones

Nombre .....
Empresa .....
Calle ..... Tfno .....
Población ..... Código Postal ..... Provincia .....

Se suscribe a El Ordenador Personal por 1 año, aceptando se prorrogue la misma por años sucesivos, salvo aviso en contrario.

Su importe de 2.500 pts. se abonará mediante:

☐ Cheque adjunto ☐ Reembolso ☐ Giro Postal.

La suscripción empezará con el nº .....

España: 2.500 Pts. por 1 año ☐ Firma:
Iberoamérica: (Correo aéreo) 50 dólares ☐
(Correo ordinario) 40 dólares ☐

Señale con una cruz lo que le interese.

Fecha .....

TARJETA DEL SERVICIO DE LECTORES

EL ORDENADOR PERSONAL Nº ... MES .. AÑO...
SERVICIO DE LECTORES

Nombre .....
Dirección .....
C.P. ☐☐☐☐☐ Ciudad .....
¿Es usted suscriptor? ☐ Sí ☐ No.
Profesión .....

Marque con un círculo la(s) página(s) en las que se encuentra el anuncio o anuncios sobre los que desea recibir, sin compromiso alguno, una información más detallada.

1	31	61	91	121	151	181	211	241	271	41	71	101	131	161	191	221	251	281	11	42	72	102	132	162	192	222	252	282	12	43
2	32	62	92	122	152	182	212	242	272	73	103	133	163	193	223	253	283	13	44	74	104	134	164	194	224	254	284	14	45	75	105	135
3	33	63	93	123	153	183	213	243	273	165	195	225	255	285	15	46	76	106	136	166	196	226	256	286	16	47	77	107	137	167	197	227
4	34	64	94	124	154	184	214	244	274	257	287	17	48	78	108	138	168	198	228	258	288	18	49	79	109	139	169	199	229	259	289	19
5	35	65	95	125	155	185	215	245	275	50	80	110	140	170	200	230	260	290	20	51	81	111	141	171	201	231	261	291	21	52	82	112
6	36	66	96	126	156	186	216	246	276	142	172	202	232	262	292	22	53	83	113	143	173	203	233	263	293	23	54	84	114	144	174	204
7	37	67	97	127	157	187	217	247	277	234	264	294	24	55	85	115	145	175	205	235	265	295	25	56	86	116	146	176	206	236	266	296
8	38	68	98	128	158	188	218	248	278	26	57	87	117	147	177	207	237	267	297	27	58	88	118	148	178	208	238	268	298	28	59	89
9	39	69	99	129	159	189	219	249	279	119	149	179	209	239	269	299	29	60	90	120	150	180	210	240	270	300						
10	40	70	100	130	160	190	220	250	280																					

criticar, mejorar, sugerencias ....

TARJETA DE PEQUEÑOS ANUNCIOS GRATUITOS

PEQUEÑOS ANUNCIOS GRATUITOS de EL ORDENADOR PERSONAL

Completa los cuadros en letra de imprenta, utilizando un solo espacio por letra. Atención sólo se admitirán los anuncios que lleven la dirección completa (calle, número, población, código postal, provincia), el apartado postal o teléfono sólo no basta. Para las ventas de material de segunda mano indicar mes y año de compra.

Señalar con una cruz el apartado (uno solo) en que deseen publicar su anuncio.

☐ Clubs ☐ Contactos ☐ Compra de material ☐ Venta de material ☐ Diversos

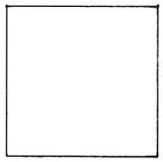
¿De qué ordenador dispone? **¿Es usted suscriptor?** ☐ Sí ☐ No

Nombre .....
Dirección .....
Población C.P. .....

Estos pequeños anuncios gratuitos están reservados exclusivamente a particulares y sin objetivos comerciales: intercambio y venta de material de ocasión, creación de clubs, cambio de experiencias, contactos y cualquier otro servicio útil a nuestros lectores. Los anuncios de venta e intercambio de programa serán rechazados sistemáticamente.

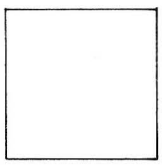
EL ORDENADOR PERSONAL, no garantiza ningún plazo de publicación y se reserva el derecho a rehusar un anuncio sin tener que dar ninguna explicación.

Este anuncio no podrá utilizarse pasados los tres meses de su publicación.



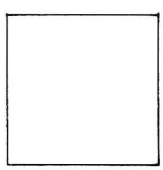
EL ORDENADOR INDIVIDUAL S.A.

Ferraz, 11 - 28008-Madrid (España)
Tels. 247 30 00 y 241 34 00



EL ORDENADOR INDIVIDUAL S.A.

Ferraz, 11 - 28008-Madrid (España)
Tels. 247 30 00 y 241 34 00



EL ORDENADOR INDIVIDUAL S.A.

Ferraz, 11 - 28008-Madrid (España)
Tels. 247 30 00 y 241 34 00

```
    1 REM 12345678901234567890123
456789012345678901234567890123456
7890123456789012345678901234567
8901234567890
    5 FOR I=16515 TO 16609
   10 POKE I,PEEK (I-16550)
   15 NEXT I
   20 POKE 16514,55
   25 POKE 16520,140
   30 POKE 16521,64
   35 POKE 16536,35
   40 LET A$="1114FF2A1040010600
0CD8C407123CB79C01D20F5C9C3A203"
   45 FOR I=1 TO 25
   50 POKE 16549+I,16*CODE A$(2*I
-1)+CODE A$(2*I)-476
   55 NEXT I
   60 STOP
  100 CLEAR
  105 DIM B$(20)
  110 FAST
  115 RAND USR 16514
  120 PRINT B$
  125 PAUSE 100
  130 POKE 16437,255
  135 RUN 100
```

que queremos.

Irán aparecienco en pantalla los nombres de los programas grabados. Una vez leída tòda la cinta o en un momento que nos interese podemos hacer un break para observar por más tiempo la lista.

Auto-Repeat en Soft

Cuando programamos con el ZX-81, nos damos cuenta en seguida que el sistema de edición no ha sido concebido para el descanso del utilizador: en cuanto queremos editar una línea un poco larga, el cursor se vuelve muy lento. Nos cansamos en seguida en manipular las teclas de desplazamiento del cursor y además el teclado se estropea rápidamente.

He aquí un pequeño remedio a esta situación: un programa que permite la repetición automática de la tecla que tenemos apretada.

Instrucciones para la función REPEAT.

1. Introducir el programa.
2. Salvaguardarlo antes de ejecutarlo, por si hemos cometido algún error.
3. Ejecutar un GOTO 10.
4. Borrar las líneas de la 10 a la 80 inclusive.
5. Pulsar RUN o RAND USR 16514.

Podrá ver, en seguida, un cuadrado intermitente en el ángulo superior izquierdo de la pantalla como signo de que el programa funciona.

Mantenga pulsada una tecla, una tecla, una tecla (¡Perdón yo creía que ya estaba en auto-repeat!) unos segundos y tendrá el placer de ver como se repite el caracter mientras mantenga la tecla pulsada.

No obstante hay dos limitaciones:

a) Este programa sólo funciona en modo SLOW.
b) Cada modificación en las líneas de programa detendrá la rutina Auto Repeat.

Para solventar este problema es suficiente pulsar un RUN o un RAND USR 16514 cuando el cuadrado intermitente haya desaparecido.

Funcionamiento del programa: utiliza las interrupciones que en tiempo normal sirven para gestionar la pantalla.

El auto-repeat es una de las numerosas aplicaciones de las interrupciones y modificando este programa podemos fácilmente crear un reloj en tiempo real o prohibir el uso de ciertas teclas y muchas otras cosas que os dejo descubrir

Vicente Baillet.

```
    1 REM AAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
    2 RAND USR 16514
    3 STOP
   10 LET A$="DD227B40DD218B40C9E
1D1C1F1223E4021AB40E52A3E40F5C5"
   20 LET A$=A$+"D5E5E6D3FD2A0C40
CBFCDD2A7B40DDE9DD227B40DD218B"
   30 LET A$=A$+"40E5F53A3440CB5F
28082A0C40237EC680772A2540242C7C
"
   40 LET A$=A$+"5520073E3232140
1815247CB528F43A2140A728063D3221
401804AF322740F1E1C9"
   50 FOR X=1 TO LEN A$ STEP 2
   60 LET A=16*(CODE A$(X)-28)+CO
DE A$(X+1)-28
   70 POKE (16514+(X-1)/2),A
   80 NEXT X
```

Explicación del funcionamiento.

16 514	LD (16 507).IX LD IX.L2 RET	Cuando el ZX-80 recibe una NMI, existe un JP (IX) que lo mandará en L2.
L2	POP HL POP DE POP BC POP AF LD (16 446). HL LD HL.L1 PUSH HL LD HL (16 446) PUSH AF PUSH BC PUSH DE PUSH HL	Cadena de instrucciones destinadas a efectuar un salto en L1, en cuanto la pantalla habrá sido refrescada.
	OR (HL) OUT 253.A	Ninguna significación, sirve para efectuar dieciocho ciclos para la pantalla.
	LD HL. (16 396) SET 7.H LD IX. (16 507) JP(IX)	Salto a la rutina de gestión de la pantalla con HL inicializado.
L1	LD(16 507). IX LD IX.L2	Vuelva a colocar IX para tener un ciclo contínuo.
	PUSH UL PUSH AF	Salvaguarda de los registros utilidos para la rutina "Auto-Repeat".
	LD A. (16 436)	Rutina "Auto-Repeat".
	BIT 3.A JRZ.L7 LD HL. (16 396) INC HL LD A. (HL) ADD A. 128 LD (HL) .A	Intermitencia del cuadrado cada treinta y dos interrupciones (0,32 s).
L7	LD HL. (16 421) INC H INC L LD A.H ORL JR NZ. L3	Si no hay tecla pulsada →L8 Si no →L3
L8	LD A.50 LD (16 417).A JR L4	Si no hay ninguna tecla pulsada, hay que esperar cincuenta ciclos (0,5 s)
L3	INC H LD A.H OR L JR Z.L8 LD A. (16 417) AND A JR Z. L5 DEC A LD (16 417).A JR L4	Si sólo está pulsado el SHIFT →L8. Si la tecla está pulsada menos de 0,5 s →L4 si no →L5.
L5	XOR A LD (16 423).A	Vuelta a 0 del anti-rebote.
L4	POP AF POP HL RET	Restauración de los registros utilizados en la rutina. Vuelta de NMI.

¿Cuánta memoria he ocupado?

El siguiente comando puede ser útil como colofón de un programa para conocer el número de bytes (octetos) ocupados por el programa (sin contarse el mismo).

También se le puede dar un número de línea e incluirlo como una instrucción dentro de un programa.

```
PRINT PEEK 16396+256*PEEK 16397
-16565
```

Los sonidos del ZX-81

Procurese una radio pequeña, de mano y sintonizela a 1.600 Khz. Situandola en la parte posterior de su ordenador, e introduzca el programa siguiente:

```
 5 FAST
10 FOR A=1 TO 200
20 NEXT A
30 FOR A=1 TO 200
40 NEXT A
50 GOTO 10
RUN
```

En este momente la pantalla se vuelve gris y podrá escuchar alternativamente dos notas tipo sirena;que podrá parar pressionando BREAK. Varie a su gusto los valores de la variable A y añada nuevos bucles FOR-NEXT para conseguir diversos sonidos,teniendo en cuenta que el contenido de la linea 50 debera ser suprimido y transladado al final del programa. Que se divierta.

Alain Cupif

Renumeración

Cuando terminamos un programa o deseamos añadirle algo, a menudo necesitamos que los numeros de las lineas de instrucciones crezcan de una forma uniforme o con una separación mayor.Podemos conseguirlo comodamente añadiendo a nuestro programa Seguidamente ejecutamos un RUN 9985 (pulsando NEW LINE).Introduzca a continuación el "Paso" o separación que desea entre los numeros de linea del nuevo listado(no olvide el NEW LINE) y el numero en que desea comienze el nueve listado (no olvide el N.L.),que puede ser igual al antiguo.Espere un momento y cuando aparezca un informe 9/9998 su programa estará renumerado.ATENCION : Este módulo no renumera los numeros que siguen a las instrucciones GOTO y GOSUB; por tanto convendrá tenerlo en cuenta y listar el programa antes de renumerarlo o tomar nota para despues modificar el contenido de estas instrucciones manualmente

```
9985 REM RENUMERACION
9986 REM EL ORDENADOR PERSONAL
9987 REM
9988 PRINT "PASO?, COMIENZO?"
9989 INPUT S
9990 INPUT L
9991 LET A=16509
9992 POKE A,INT (L/256)
9993 POKE A+1,L-256*INT (L/256)
9994 LET A=A+1
9995 IF PEEK A<>118 THEN GOTO 99
94
9996 LET A=A+1
9997 LET L=L+S
9998 IF PEEK A*256+PEEK (A+1)=99
85 THEN STOP
9999 GOTO 9992
```

Trans-Compilador de 1K para el ZX81

Este artículo presenta un programa de 1 K. que de forma instantánea translada las principales instrucciones del BASIC ZX81 en Código Máquina.

Bastantes usuarios del ZX81 quisieran aprender las nociones de Codigo Maquina pero lo encuentran muy dificil o no tienen tiempo; ademas los compiladores que se comercializan requieren normalmente 16 K. de RAM. Como ya hemos dicho este programa solo necesita 1 K. para funcionar y ofrece una buena ayuda para utilizar algunas de las ventajas del Codigo maquina (C.M.).

He aqui dicho programa:

Para hacer posible que entre en 1 K. de RAM se han utilizado tecnicas que ahorran memoria; como el utilizar la función Val y PI como números, etc.

Cuando teclee làs lineas más largas puede serle util limpiar la pantalla antes de empezar. Este sistema se puede tambien utilizar cuando desee editar una linea y el ZX81 no se lo permita por tener ocupada casi toda la memoria, simplemente limpie primero la pantalla (teclee CLS) y luego pulse Edit.

Al ejecutar el programa Ud. vera un IMPUT alfanumerico. Preste mucha atención a las siguientes instrucciones:

— Este transladador puede convertir los siguientes comandos en codigo maquina:

LET
PRINT «cadena»
FOR–NEXT
IF – THEN
INKEYS
GOTO
SCROLL
STOP
NEW

— Recuerde que en Codigo Maquina no existen las cadenas: AS, BS, CS, etc.

— Solamente puede utilizar las variables C,D,E,F y G.

— Lo siguiente es importante; debe teclear todos los comandos y funciones letra por letra incluidos los espacios: LET D = 1Ø se compondra de ocho caracteres separados.

— Cuando use una instrucción condicional despues del IF deben ir dos espacios. Las sentencia IF puede solo comparar una variable con un número y no una variable con otra variable.

— Otra restricción de las instrucciones condicionales es que un THEN debe siempre ir seguido de un GOTO esta es una resctricción del Codigo Maquina y no del programa.

— Solo puede añadir o restar 1 a una variable. Si desea añadir o restar más debera usar el número de sentencias +1 ó –1 que desee.

— PRINT puede solamente ser utilizado de la forma PRINT «cadena» y no PRINT AT, TAB ó PRINT C, D, E, F ó G.

— INKEYS debe solo ser usado de la forma LET C = INKEYS (Donde la variable puede ser C, D, E, F ó G). Nota: INKEYS nos da el codigo de la tecla pulsada.

— No hay numeros de lineas en codigo maquina, asi que introduzca su sentencia como si fuera un comando directo. Esto supone un problema cuando se utilizan comandos como GOTO y NEXT, Ud. encontrara que el programa imprime dos X donde se utilizan estos comandos. Debera trabajar un poco para conseguir ir a donde quiere, pero es realmente sencillo. En Codigo Maquina las unicas instrucciones GOTO son las de «Ir a delante» o «Ir para atras». Tendra que contar el número de posiciones (pasos) a moverse a partir de las dos X hasta dondo quiere ir. Por ejemplo:

Basic	C.M.	
FOR C = 1 TO 8	ø6	ø8
PRINT ««A»»	3E	26 D7
NEXT C	1ø	XX

En este caso vamos 4 pasos para atras (–4) desde XX (fin de NEXT C) hasta 3E (principio del PRINT ««A»») busque –4 en la siguiente tabla y reemplace XX por FB.

-1	-2	-3	-4	-5	-6	-7	-8	-9	-10	-11
FE	FD	FC	FB	FA	F9	F8	F7	F6	F5	F4

Para avanzar utilice la siguiente tabla:

+1	+2	+3	+4	+5	+6	+7	+8	+9	+10	+11
00	01	02	03	04	05	06	07	08	09	0A

— Si encuentra una D entre paréntesis despues de un número significa que este número esta en decimal y debe buscar su equivalente hexadecimal al final de su manual Sinclair.

Haga su programa lo más simple posible las lineas largas no son transladadas correctamente y por supuesto siempre incluya un C9 al final de sus programas en Codigo Maquina ¡Para volver al Interprete BASIC!.

Justo Maurin

```
TRANS-COMPILADOR PARA ZX81 1K.

  10 INPUT R$
  20 IF LEN R$>VAL "4" THEN LET
A=(CODE R$(VAL "5")-CODE "B")*VA
L "2"
  30 IF LEN R$>VAL "11" THEN IF
R$(CODE "£")="$" THEN PRINT "D9C
D8B02444D511428F7CD8D077ED9";" 4
F575F876F";(A TO A+SGN PI)
  40 IF R$(SGN PI)="F" THEN PRIN
T "06";R$(CODE "£");"(D)"
  50 IF R$(SGN PI)="L" AND LEN R
$(VAL "9" THEN PRINT " 0E161E262
E";(A TO A+SGN PI);R$(VAL "7" TO
VAL "8");"(D)"
  60 IF R$(SGN PI)="L" AND LEN R
$>VAL "8" THEN PRINT (" 0C141C24
2C";(A TO A+SGN PI) AND R$(VAL "8
")="+");(" 0D151D252D";(A TO A+SG
N PI) AND R$(VAL "8")="-")
  70 IF R$(SGN PI)="Q" THEN PRIN
T "18XX"
  80 PRINT ("CF" AND R$(PI)="0")
;("C7" AND R$(PI)="U");("CD0E0C"
 AND R$(VAL "2")="C")
  90 IF LEN R$>VAL "4" THEN IF R
$(SGN PI)="1" THEN PRINT "3E";R$
(VAL "7" TO VAL "8");" B9BABBBCB
D";(A TO A+SGN PI);"28XX"
 100 IF NOT R$(SGN PI)="P" THEN
GOTO VAL "10"
 110 FOR N=SGN PI TO LEN R$-VAL
"3"
 120 PRINT "3E";CODE R$(VAL "7"+
N);"(D)D7";
 130 NEXT N
 140 GOTO VAL "10"
```

Del C.M. al Basic

A continuación se expone una forma sencilla de saltar desde un programa en código maquina a una línea de programa en BASIC.

Ld HI, nº línea

JP OE 86

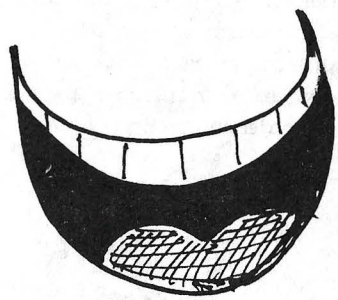
Comentario

Carga en HL el número de línea del programa Basic a la que se va a saltar (byte menos significativo va primero).
Salto a la ROM

¡Es maravilloso! y espero que sea útil a los que profundizan en el ZX81.

Pedro Gruenholz.

sistemas varios



Esta sección por su sofisticación en el tratamiento con la máquina está dirigida **SOLO** a los más experimentados programadores. Sin duda encontrareis el truco más inédito y la más aguzada astucia.

Además por su generalidad lo podeis practicar en cualquier ordenador, doméstico o no.

La redacción no se compromete de las incorrecciones realizadas en el ordenador por supuestos consejos de tan ilustres personajes.

¡QUE NO OS PASE NADA!

PROLOK Casero

Hoy todo el mundo quiere copiar los programas, y si DRAKE levantara la cabeza seguro que abría una tienda de Software en lugar de navegar por las aguas solitarias del Pacífico. Por eso se utilizan métodos diversos para proteger los programas de tan terrible fin.

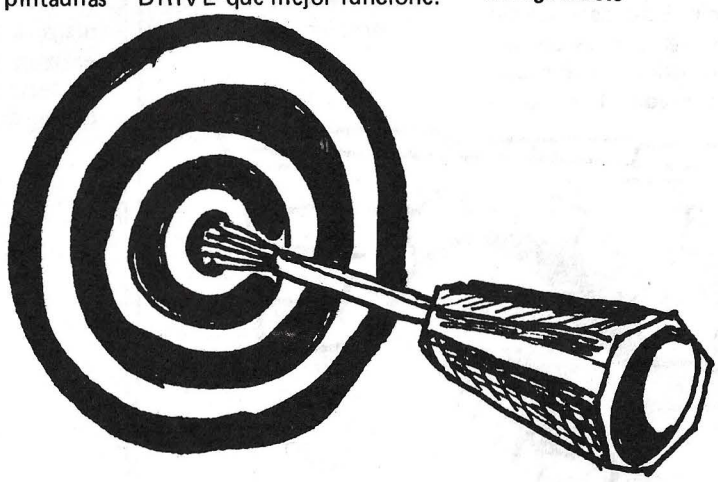
Nosotros te proponemos un sistema PROLOK casero que a buen seguro te protegerá para el resto de sus días tu programa más sofisticado.

Para ello toma el/los discos más caros y el programa que no tengas salvado y pídele a tu hermana el pintauñas 0'7. A continuación (con mucho cuidado no estropees alguna pista) acerca la punta del pintauñas a la pista 2 sector 8° y marca un punto con el esmalte en ese lugar (bit de mayor peso), dejalo secar e inserta (ya seco) el disco en tu DRIVE que mejor funcione.

¡Veras como no puedes copiarlo ni visualizarlo!

Gracias

George Boole



RESET Generalizado

Algunos ordenadores tienen incorporada la tecla de RESET para inicializar todas las variables y todas las banderas, pero desgraciadamente otros no. Esto tiene dos fáciles soluciones. El RESET duro y el RESET "faible" o blando para los más castizos.

El primero de ellos requiere el uso de una tijera, la cual se aplicará a un lado del cable (polo positivo), sin pasarse, ya que podría provocarse un cortocircuito con daños irreparables para el ordenador. Este hecho provocará la puesta a cero de todas las variables del sistema. Para volver a arrancarlas basta con unir de nuevo el cobre, (ver figura 1), o instalar un interruptor de pera, como en la mesilla, con lo cual para un nuevo RESET, deberemos cortar el polo negativo.

El faible es un poco más basto ya que consiste en tirar de ambas polaridades a la vez, justo al borde de la toma de corriente. Para volver al sistema consultar el manual de instrucciones en la sección de "instalación del ordenador".

Charles Babbage

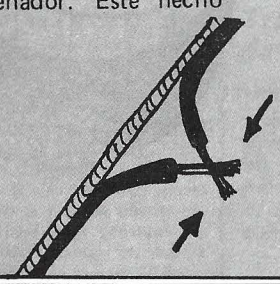


Figura 1

Tarjetas magnéticas doble cara, doble densidad

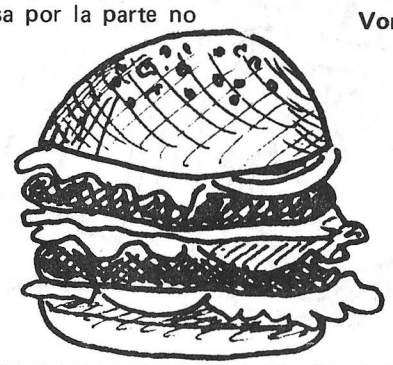
Que duro es introducir 4, 5 ó más tarjetas para copiar/salvar un programa en tu calculadora programable. Tranquilo, nosotros te proponemos un método seguro y eficaz para aumentar dicha capacidad y conseguir resultados sorprendentes en el almacenamiento de tus programas.

Para ello necesitas un imán, una cinta de casete vieja y un poco de pegamento.

Primero (para limar asperezas) pasa por la parte no magnética el imán, de forma que las partículas informativas se alinien para recibir la doble densidad. Posteriormente introducir un trozo de la cinta magnética (sin canciones) sobre dichas partículas, utilizando 0'6 μ de pegamento. Para terminar, una vez aumentada la densidad de todas tus tarjetas, pega por la parte no magnética dos a dos. Obtendrás tarjetas de doble cara y doble densidad.

Si tienes problemas para introducirla por el orificio, consulta a tu proveedor habitual.

Von Neumann

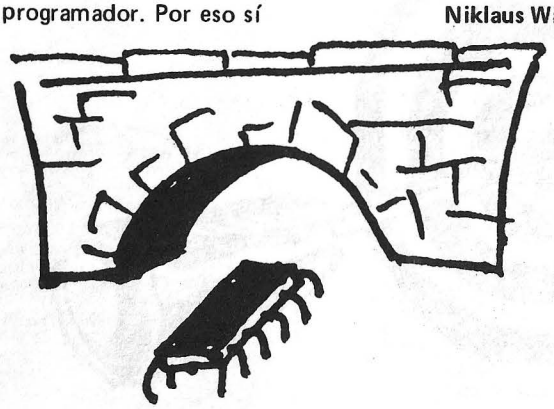


Gane 2 MHz en su Z80 ó 6502

Cuando no se van a utilizar las salidas de impresora ni las de casete se puede aumentar en unos 2 MHz la velocidad del microprocesador Z80 ó 6502. ¿Cómo, por qué? Fácilmente, el controlador de entradas salidas tiene una distancia de unos 10 cm. entre las pistas que tiene que recorrer la señal, y claro al cabo de 1 minuto se hace unos kilómetros entre ir y venir. Esto desgasta la señal a la vez que las pistas, el controlador y la paciencia del programador. Por eso sí no vas a imprimir ni grabar nada, haz un puente entre las patas 20 y 1 para el Z80 y 24 y 3 para el 6502, con una "pera" de interruptor para cuando quieras imprimir o salvar. De esta manera la señal toma el atajo y se ahorra combustible y tiempo. Es evidente que con este tipo de puentes la velocidad aumenta, lo que puede ser algo molesto es cambiar de micro todos los días, por eso además de la pera, poner un fusible de 0,2 amperios.

!Felices vacaciones sin problemas!.

Niklaus Wirth

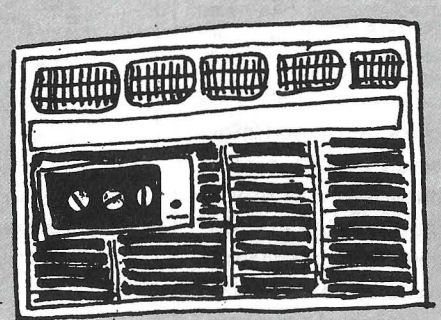


Refrigeración radical para las cabezas lectoras

El calor dilata los cuerpos (Arquímedes). Este principio básico lo es igualmente para nuestro ordenador. El verano llega también a la informática y todos agradecemos la labor que desarrolla la máquina mientras tomamos refrescos en la playa. ¿Pero qué ocurre cuando introducimos un disco y se niega a leerlo? Dilatación, señores, dilatación; la cabeza lectora tiene más calor que el deseado y es por ello que se niega a leer o escribir.

Este genérico error es fácil de solucionar, los físicos saben que al calor se le combate con frío, es por ello que el frío devolverá la cabeza a su posición normal. Para ello, toma una botella de agua fría de la nevera y llena 25 cc. en un vaso alto. A continuación (y con el ordenador encendido, sino no se entera) viértela en la cabeza rebelde. ¡Notarás la diferencia!

Blaise Pascal

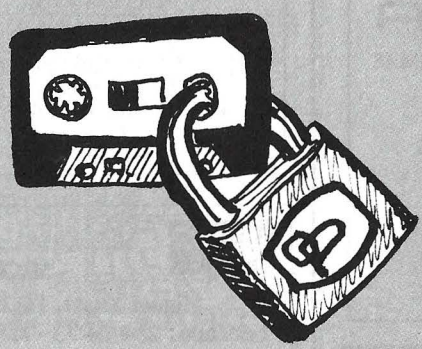


Protección para los casetes

Los discos ya tienen numerosos sistemas de protección pero ¿y los casetes? No todos tenemos la posibilidad de salvar tipo turbo nuestros programas y nos encontramos con la duda de que hacer para ocultar horas de trabajo de manos ajenas.

La respuesta es igualmente (sencilla) pero insual simplemente con cargarse el "HEADER" basta para que nadie pueda usurpar el resto. ¿Cómo? fácilmente, inserta la cinta en la pletina, con el programa al principio, el contador a cero y el tocadiscos en alguna canción folklórica, posteriormente grabar durante 5'8 segundos los tonos bajos del cantante, ¡y ya está! tu programa está libre de manos extrañas.

Michael Faraday



Caracteres extraños por impresora

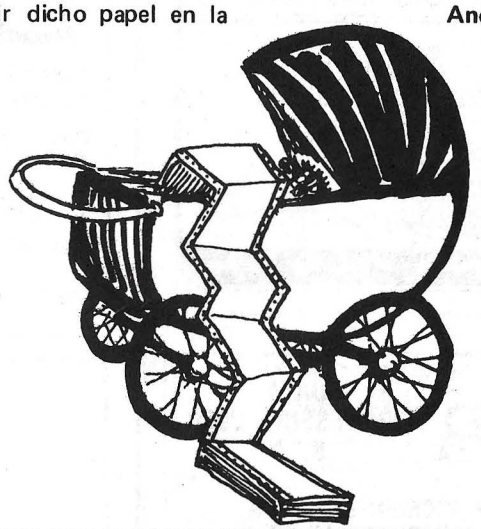
El sistema ASCII ya está muy explotado, y todos sentimos la necesidad de caracteres más expresivos que alegren nuestros programas "tipo sprites". Nuestro truco consiste en ofertar un "taco" de hojas de ordenador al hermano pequeño o sobrinito, junto con la correspondiente caja de rotuladores.

Dos horas más tarde introducir dicho papel en la impresora y teclar este programa en Basic estandar'

```
10 REM CARACTERES
EXTRAÑOS
20 FOR I = I TO 1000
30 PRINT
40 NEXT I
50 END
```

¡Ya vereis que maravillas tipográficas!

Anónimo



Pantalla gráfica alta resolución con sintetizador de voz

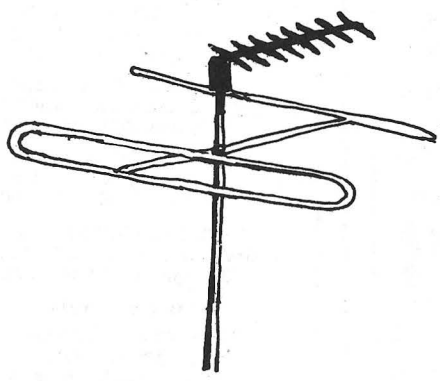
El que más y el que menos ha visto los programitas de vanguardia en los billares donde el guerrero (no del antifaz) busca afanosamente a la doncella para salvarla guiado por su linda voz.

Si posees un ordenador doméstico, hay un truco que no viene en ningún manual pero que te permite la máxima resolución, con pantalla gráfica y un maravilloso sintetizador de voz incorporado.

Conecta habitualmente el ordenador y una vez conseguida la imagen nítida del mensaje indicador, pulse POKE 25472, 625 ó simplemente POKE 25472, 1, a continuación sitúate en el sintonizador variable de la televisión y gíralo suavemente hasta conseguir una magnífica resolución de 625 líneas.

Según la hora se pueden sintonizar programas divertidísimos con voz y todo. Seguro que os resulta más fácil que programar nuestro propio guerrero pixel por pixel.

Eckert y Mauchly



Programación Off-Line

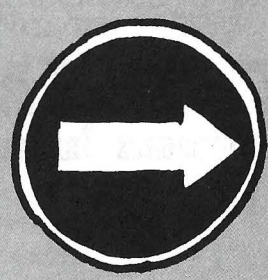
¿Quién no ha tomado una aspirina como método resolutivo de una rutina inacabable? Bueno pues no es la solución, porque no conviene abusar de los medicamentos; como respuesta a esos mareos vómitos y diarreas informáticas tenemos un revolucionario sistema americano de programación Off-Line.

Es tan simple que nadie había caído en ello, hasta que en la Universidad de Colorado se les estropeó la fuente de alimentación de un mini y la cambiaron por la del otro. imaravilloso! durante una semana estuvieron poniendo a punto off-Line sus más difíciles programas.

Manos a la obra:

Cortar el cable de alimentación de su enchufe marcado 9VDC en la fuente y con mucho cuidado cambiar el cable que estaba a la derecha hacia la izquierda y viceversa. Poner atención no equivocaros ya que si no quedaría todo como esta. Una vez unidos correctamente, insertar el pequeño enchufe en el lugar adecuado y esperar unos segundos. El limitador de corriente de entrada sufre un maravilloso cambio de estado que os permitirá la programación Off-Line durante varios días. Todo gracias a que los chips no entienden que la electricidad circula mejor por la derecha que por la izquierda.

¡Ah la electrónica!



DIRECTORIO

Tiendas de Informática.

Alonso Cano, 2
Teléf.: 446 60 18
28010 Madrid.

Brusi, 102 - Entresuelo 3º
Teléf.: (93) 201 21 03
08006 Barcelona.

Distribuidores de: Apple II, Macintosh, Lisa.
Discos rígidos CORVUS de 5, 10, 15, 20 y 45 Megabytes.
Redes de área OMNINET.

MAYBE

Electronic Center Villa

componentes electrónicos y microordenadores
C/. Ntra. Sra. de la Mercé, 41
GAVA (BARCELONA)
Tno.: 662 87 01
Especializados en
INSTALACION DE AULAS INFORMATICAS Y MANTENIMIENTO
SISTEMA DE PERIFERICOS COMPARTIDOS

7000 Sistemas en Kit

ELECTRONICA SANDOVAL S.A.
COMPONENTES ELECTRONICOS PROFESIONALES
TELEVISION RADIO AMPLIFICACION
VIDEO ALTA FIDELIDAD

Sandoval, 4
Tel.:445 18 33 - 445 18 70
MADRID - 10

Micro Ordenadores:
Rockwell
Ohio Scientific
Videogenie
Sinclair

8000 Libros y Revistas

PRODACE
Ferraz, 11 - 3o
Tel.: 247 30 00
MADRID 8

Programación de Ordenadores en Basic.;

El centro MICRO SPOT, especializado en informática, que ofrece la oferta más amplia en microordenadores y una variada gama de periféricos, impresoras, unidades de cassette y disquette, monitores color y F. V., etc. Disponemos de completos listados de software en cinta y disco, para programas técnicos, de aplicación, educativos y juegos. Accesorios diversos, manuales, libros técnicos y revistas especializadas.

Consulte sobre nuestros cursos de BASIC y PASCAL para estudiantes de BUP - COU - Escuelas Técnicas - Universitarios - Profesionales - Empresas y adultos en general.

Por vez primera en España cursos de iniciación y tarifas especiales para amas de casa y para la tercera edad.

MICRO SPOT

Conde de Cartagena, 9 (zona Retiro) - Madrid-7 - Tels. 251 32 04/05/06/07

Micromed
MICROINFORMATICA AVANZADA

Sistemas y Servicios

La única Tienda de Ordenadores especializada en la mecanización de la Pequeña y Mediana Empresa donde en cualquier momento podrá discutir:

- **Análisis Mecanización de su Empresa.**
- **Desarrollo de Programas a Medida.**

HEWLETT PACKARD - HP 150
WANG PC
TOSHIBA T300, T100
VICTOR/SIRIUS

Numerosas instalaciones en empresas nos avalan.
Venta en Provincias Zona Centro
Servicio Técnico Propio
Juan Alvarez Mendizabal, 55. MADRID-8
(En Argüelles, antes Victor Pradera)
Teléfonos: (91) 242 15 57 y 67.

INFO SOFT

Programas específicos para arquitectura, construcción y obra civil, sobre microordenadores Hewlett-Packard.
Pídanos Catálogo gratuito.

SOFT biblioteca de programas
Apartado de Correos, 10.048. Tel. (91) 448 35 40. Madrid.

LOGIMATICA
en
Lagasca, 90
(esquina Ortega y Gasset)
MADRID-6

UN NUEVO CONCESIONARIO DE INVESTRONICA PARA EL ORDENADOR SINCLAIR

SINCLAIR ZX 81: 14.975 Pts.
SINCLAIR ZX SPECTRUM 16 k: 32.000 Pts.
SINCLAIR ZX SPECTRUM 48 k: 41.900 Pts.

Y UN SIN FIN DE PROGRAMAS PARA JUEGOS, EDUCACION Y UTILIDADES/GESTION.

NO PIERDA EL TREN DE LA INFORMATICA

Visítenos portando este anuncio y obtendrá condiciones especiales

Tlnos: 431 60 32
435 52 56

LE ESPERAMOS

LOGIMATICA S.A.

¿Lo hubiera podido comprar más barato...?

La pregunta es lógica, hay tantos precios para los mismos ordenadores y accesorios, que nunca sabe Vd. si lo hubiera podido comprar más barato.

Claro que si hubiese sabido antes que en REGISA es donde se puede comprar al precio más bajo del mercado, y además puede elegir entre una mayor gama de microordenadores y accesorios **(por supuesto todo con garantía),** esta pregunta ya no se la haría.

ventas al mayor

REGISA

Comercio, 11. Tel. 319 93 08. Barcelona

lo mismo y más..., pero al mejor precio.

SAGA
sinclair
AMSTRAD
SPECTRAVIDEO
SEIKOSHA
DK-TRONIC
commodore
HIT BIT
SONY

MONTERO & ASOCIADOS

Establecimientos recomendados: • BAZAR DELHI. Reina Cristina, 11. Barcelona • INTERJOYA. Reina Cristina, 9. Barcelona • BAZAR TAIWAN. Plaza Palacio, 9 (Galerías). Barcelona • LOS GUERRILLEROS. I. Canarias, 128. Valencia • BAZAR KARDIS. I. Canarias, 130. Valencia • BAZAR DELHI. M. Ruano, 5. Lleida • BAZAR TAIWAN. Pujós, 35. Hospitalet.

SENCILLO, ASEQUIBLE, PROFESIONAL

ASI ES EL QL DE SINCLAIR, HECHO PARA NOSOTROS

Para los profesionales que necesitamos un teclado en nuestro idioma, QL nos ofrece, en castellano, su QWERTY standard de 65 teclas móviles.

Para los que deseamos comunicarnos a gran velocidad y capacidad con nuestro ordenador, QL nos presenta su lenguaje SUPER BASIC.

Para los que necesitamos gran margen operativo, ahora disponemos de un ordenador con memoria ROM de 32K que contiene el sistema operativo QDOS, un sistema mono-usuario, multi-tarea y con partición de tiempo.

Para los que deseamos tener perfectamente ordenada nuestra agenda de trabajo, presupuestos, fichas de productos, nuestra correspondencia, estadísticas de venta, archivo... QL viene dotado de cuatro microdrives totalmente interactivados entre sí. QL QUILL de Tratamiento de Textos, QL ARCHIVE Base de Datos, QL ABACUS Hoja Electrónica de Cálculo y el QL EASEL para realización de todo tipo de gráficos.

Para los que nos gustan las cosas bien acabadas, QL

se suministra con su fuente de alimentación, cables de conexión y adaptadores de TV, monitor y red local, cuatro programas de software de uso genérico, cuatro cartuchos en blanco para los microdrives y manual de instrucciones en castellano.

Para los que creemos que lo bien hecho puede tener también el mejor precio, QL el ordenador grande a precio pequeño.

Para los que nos gusta siempre ir bien acompañados, Sinclair —el mayor vendedor del mundo en ordenadores personales— e Investrónica, la mayor red de distribución de España, son nuestras mejores Compañías. Nuestra mejor garantía.

En definitiva, para los que queremos ordenarnos y nunca nos habíamos atrevido.

Con QL ya no hay excusas.

DISTRIBUIDOR EXCLUSIVO **investronica**

Tomás Bretón, 60. Telf. (91) 467 82 10. Télex 23399 IYCO E. 28045 Madrid
Camp, 80. Telf. (93) 211 26 58-211 27 54. 08022 Barcelona