LISTADOR DE VARIABLES

Otro de los programas de utilidades que os ofrecemos en este número, es el listador de variables, utilizable a través de nuevos comandos RSX.

Mediante dichos comandos, estaremos en condiciones de listar tanto las variables numéricas como alfanuméricas que se utilicen en un programa.

La rutina en código máquina que realiza dicha función, está ubicada a partir de la dirección hexadecimal $A000 y tiene una longitud de 585 bytes.

Debido a la zona de memoria que ocupa nuestro programa, no podremos trabajar con programas de gran longitud, ya que de lo contrario corromperíamos esas direcciones de memoria. Así pues, la longitud de nuestros programas, no debe superar 39k.

Para que dicha rutina funcione correctamente, deberemos ejecutar el siguiente programa Basic:

```
10 MEMORY &9FFF
20 LOAD"RSX
30 CALL &A000
```

Una vez hecho esto, borraremos todo lo que hay en la memoria mediante el comando:

NEW

y a continuación podremos cargar en memoria el programa con el cual vayamos a trabajar.

La línea 30 del anterior programa Basic, se utiliza para indicar al ordenador que existen nuevos comandos, ya que de lo contario, los ignoraría.

Una vez tengamos en memoria el programa con el que deseemos trabajar, **no deberemos ejecutarlo,** ya que de lo contrario se inicializarían las variables, y la rutina sería incapaz de identificarlas.

A partir de este momento, estamos en condiciones de utilizar nuestros nuevos comandos.

Vamos a indicar a continuación cuáles son dichos comandos y de qué forma funcionan.

El primero de ellos se utiliza para listar las variables alfanuméricas, y se puede usar de dos formas distintas, según la opción deseada.

Este nuevo comando es el siguiente:

IVALFA

escrito de esta forma, dicho comando produciría un listado de todas las variables alfanuméricas, indicando asimismo la línea en la que cada una de ellas se encuentra.

Si por ejemplo existiera en memoria un programa que tuviera la variable alfanumérica A$ en las líeas 20, 40, la variable HOLA$ en las líneas 50 y 100 y la variable B$ en las líneas 20, 50 y 60, tras la ejecución del anterior comando, nos aparecería el siguiente listado:

```
00020 A$ B$
00040 A$
00050 B$ HOLA$
00060 B$
00100 HOLA$
```

Otra manera de utilizarlo sería de la forma que se indica a continuación:

IVALFA, **"HOLA"**

La ejecución de este comando tal como está escrito, produciría el listado de la variable alfanumérica 'HOLA$', indicando al mismo tiempo en qué líneas de programa se encuentra.

Utilizando el ejemplo del hipotético programa citado anteriormente, este comando, nos produciría un listado en pantalla, como el siguiente:

```
00050 HOLA$
00100 HOLA$
```

Este último comando nos será de utilidad cuando se desee conocer en qué partes del programa actúa una variable alfanumérica, y en particular en qué líneas de programa se está utilizando.

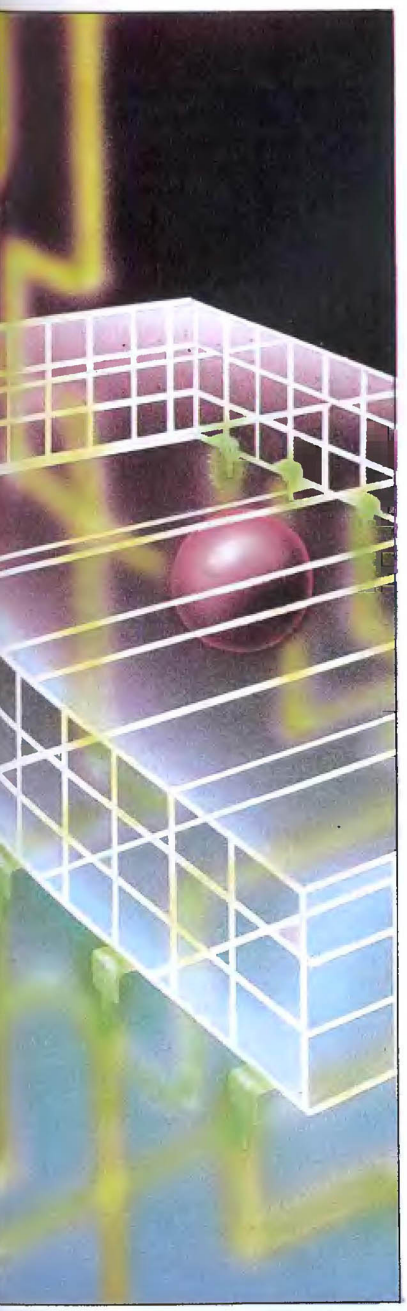
Así pues, cuando se ejecute, se producirá la búsqueda de la variable introducida, y en el caso de ser encontrada se imprimirá en pantalla. En el caso de que no exista ninguna variable de este tipo, no se producirá ningún tipo de impresión en pantalla.

Otro de los comandos de que disponemos, es el que se indica a continuación:

IVNUME

Dicho comando provoca el listado de todas las variables numéricas que contenga nuestro programa, así como los números de línea donde se encuentran.

Como hemos hecho anteriormente, supondremos que existe en memoria un programa que contiene en las líneas 10 y 60 la variable numérica 'POSX', en las líneas 20 y 90 la variable 'YPOS', y en la línea 40 la variable 'DI-



NERO'. La ejecución del anterior comando, produciría un listado semejante al que se indica:

```
00010 POSX
00020 YPOS
00040 DINERO
00060 POSX
00090 YPOS
```

La otra variante de este último comando, nos permitirá buscar una variable numérica en particular, produciéndose únicamente la impresión de dicha variable en pantalla, así como los números de línea en que se encuentra.

Este comando es el siguiente:

IVNUME, **"POSX"**

Su ejecución provoca la busqueda de la cadena introducida entre comillas. En el caso de tener en memoria el programa anteriormente mencionado, provocaría la impresión en pantalla del siguiente texto:

```
00010 POSX
00060 POSX
```

También por impresora

El último comando del que disponemos, y no por ello menos importante es:

IP

el cual nos ofrece la opción de imprimir en pantalla o bien en impresora.

Este comando actúa como una especie de commutador, es decir cuando, se pulsa, activa lo que anteriormente estuviera desactivado.

Así, por ejemplo, si estamos produciendo un listado de variables en pantalla y ejecutando dicho comando, obligaríamos a que el listado saliera por impresora. Si ahora deseamos volver a listar en pantalla, únicamente deberemos ejecutarlo otra vez.

Vamos a ver a continuación cuáles son las principales rutinas de que consta nuestro programa.

En primer lugar se efectúa la instalación de los nuevos comandos RSX en el sistema; para ello se definen dichos comandos y se efectúa una llamada al firmware, que anuncia al sistema su creación.

La primera rutina con que nos encontramos es la que se encarga de activar el comando con parámeteros, o bien sin utilizar ningún parámetro.

Registro A=0 indica que no hay parámetros
Registro A < > 0 indica la existencia de parámetros.

De esta forma, el programa distinguiría las posibles opciones del comando RSX.

Cuando no se utilicen parámetros, el programa saltará inmediatamente a la rutina encargada de buscar todas las variables alfanuméricas que existan en el programa.

Para detectar dichas variables, se ha definido en un buffer la cadena de bytes por la que pueden ser identificadas, y que es la siguiente:

Variable alfanumérica... 3,0,0,nombre

dado que en este caso el nombre de la variable no nos interesa, únicamente se buscará una cadena que contenga los tres primeros bytes.

Una vez encontrada, conoceremos la existencia de una variable alfanumérica en esa dirección de memoria. Para reconocer de qué variable se trata tomaremos los bytes que siguen a continuación, hasta encontrarnos con uno que contenga el valor de un carácter ASCII al que se le ha sumado 128, ya que es de esta forma como el Basic almacena dichas variables.

Veamos por ejemplo cómo se encontraría almacenada en memoria la variable HOLA$:

3,0,0,**"H"**, **"O"**, **"L"**, **"A"**+128

Cuando se utilice el anterior comando con parámetros, en primer lugar se reconocerá qué parámetros se han introducido, y los colocará en un buffer, detrás de los bytes 3,0,0 identificativos de variable alfanumérica.

A continuación se enviará el control del programa a la rutina encargada de buscar cadenas en memorias. Así por ejemplo si el parámetro es la cadena "POSX", dicha rutina intentará localizar la siguiente secuencia de números:

3,0,0,**"P"**, **"O"**, **"S"**, **"X"**+128

Por último nos queda por describir el comando encargado de localizar las variables numéricas. La rutina utilizada, también chequea en primer lugar el contenido del acumulador para averiguar si se han enviado o no parámetros.

En el caso de que no existan dichos parámetros, se llama a la rutina encargada de localizar este tipo de variables, las cuales pueden ser reconocidas por la siguiente secuencia de bytes:

13,0,0

De esta forma, cuando se encuentre la cadena, se conocerá la existencia de una variable numérica en esa dirección de memoria, por lo que tomaremos los bytes que siguen hasta encontrarnos con uno que contenga el valor de un carácter ASCII más 128, para conocer cuál es el nombre de dicha variable.

Como podemos comprobar este tipo de variables se almacenan en memoria igual que las tratadas anteriormente.

Cuando se utilice este último comando con algún parámetro, se tomará dicho parámetro y se colocará en un buffer a continuación de los bytes indicadores de variable numérica, y se llamará a la rutina de búsqueda para que los localice, y una vez encontrados se imprimirán en pantalla.

Una vez dadas las instrucciones de funcionamiento y explicadas las rutinas más interesantes, estamos en condiciones de poder utilizar correctamente el programa.

Para ello deberemos copiar el listado ensamblador que aparece a continuación, o bien teclear el programa cargador. Aquéllos que utilicen esta útlima opción, deberán ejecutar el programa una vez copiado, y en caso de que no dé ningún mensaje de error, se deberá salvar de la forma siguiente:

SAVE **"RSX"**, B, $A000, 585

Cuando se dese ejecutar, deberemos cargarlo en memoria de la forma indicada al principio de ese artículo, sin olvidarnos de efectuar la llamada a la dirección $A000, con el fin de inicializar los nuevos comandos.

```
;VARIABLES
          ORG   #A000
          LD    BC,TABLA
          LD    HL,ESPACE
          JP    #BCD1
TABLA:    DEFW  NAME
          JP    VALFA
          JP    VNUME
          JP    IMPAN
NAME:     DEFM  "VALF"
          DEFB  "A"+#80
          DEFM  "VNUM"
          DEFB  "E"+#80
          DEFB  "P"+#80
          DEFB  0
ESPACE:   DEFS  4
IMPAN:    LD    A,(PRESOR)
          CPL
          LD    (PRESOR),A
          RET
VALFA:    AND   A
          JP    Z,ALFA
          LD    L,(IX+0)
          LD    H,(IX+1)
          LD    A,(HL)
          INC   HL
          LD    E,(HL)
          INC   HL
          LD    D,(HL)
          EX    DE,HL
          LD    DE,DATALF+3
          LD    C,0
          LD    B,A
MOS:      LD    A,(HL)
          LD    (DE),A
          INC   HL
          INC   DE
          INC   C
          DJNZ  MOS
          LD    A,C
          LD    (LONCA),A
          DEC   DE
          LD    A,(DE)
          ADD   A,128
          LD    (DE),A
          CALL  BUSALF
          RET
VNUME:    AND   A
          JP    Z,NUMER
          LD    L,(IX+0)
          LD    H,(IX+1)
          LD    A,(HL)
          INC   HL
          LD    E,(HL)
          INC   HL
          LD    D,(HL)
          EX    DE,HL
          LD    DE,DATNUM+3
          LD    C,0
          LD    B,A
MAS:      LD    A,(HL)
          LD    (DE),A
          INC   HL
          INC   DE
          INC   C
          DJNZ  MAS
          LD    A,C
          LD    (LONCA),A
          DEC   DE
          LD    A,(DE)
          ADD   A,128
          LD    (DE),A
          CALL  BUSNUM
          RET
;
BUSALF:   LD    HL,DATALF
          LD    (VARS),HL
          LD    A,3
          LD    (VARI),A
          LD    A,(LONCA)
          ADD   A,3
          LD    (LONCV),A
          CALL  INIC
          RET
BUSNUM:   LD    HL,DATNUM
          LD    (VARS),HL
          LD    A,2
          LD    (VARI),A
          LD    A,(LONCA)
          ADD   A,3
          LD    (LONCV),A
          CALL  INIC
          RET
ALFA:     LD    HL,VARALF
          LD    (VARS),HL
          XOR   A
          LD    (VARI),A
          LD    A,3
          LD    (LONCV),A
          CALL  INIC
          RET
NUMER:    LD    HL,VARNUM
          LD    (VARS),HL
          LD    A,1
          LD    (VARI),A
          LD    A,3
          LD    (LONCV),A
          CALL  INIC
          RET
INIC:     LD    IX,(VARS)
          LD    HL,#170
          LD    (DIREC),HL
BUCL:     LD    B,(HL)
          INC   HL
          LD    C,(HL)
          DEC   HL
          LD    A,B
          OR    C
          RET   Z
          CALL  BUSC
          LD    HL,(DIREC)
          LD    DE,(LONLIN)
          ADD   HL,DE
          LD    (DIREC),HL
          JR    BUCL
BUSC:     LD    C,(HL)
          INC   HL
          LD    B,(HL)
          INC   HL
          LD    (LONLIN),BC
          LD    E,(HL)
          INC   HL
          LD    D,(HL)
          INC   HL
          LD    (NUMLIN),DE
          XOR   A
          LD    (CONCAN),A
          LD    D,0
BUC:      LD    A,(LONCV)
          CP    D
          CALL  Z,FIN
          LD    A,(HL)
          CP    (IX)
          INC   HL
          JR    Z,INCCO
          LD    D,0
          LD    IX,(VARS)
          JR    NOINC
INCCO:    INC   D
          INC   IX
NOINC:    DEC   BC
          LD    A,B
          OR    C
          JR    NZ,BUC
          RET
FIN:      LD    A,(CONCAN)
          AND   A
          JR    NZ,FINU
          LD    A,13
          CALL  IMPRE
          LD    A,10
          CALL  IMPRE
          PUSH  HL
          CALL  DECIMA
          POP   HL
FINU:     LD    A,(VARI)
          CP    0
          JR    Z,NOBUS
          CP    1
          JR    Z,NOBUS
          LD    A,(LONCA)
          LD    B,A
RESBU:    DEC   HL
          DJNZ  RESBU
NOBUS:    LD    A,(HL)
          BIT   7,A
          JR    NZ,FINIM
          CALL  IMPRE
          INC   HL
          JR    NOBUS
FINIM:    RES   7,A
          CALL  IMPRE
          LD    A,(VARI)
          CP    1
          JR    Z,NOALF
          CP    2
          JR    Z,NOALF
          LD    A,"$"
          CALL  IMPRE
NOALF:    LD    A," "
          CALL  IMPRE
          LD    IX,(VARS)
          LD    D,0
          LD    A,(CONCAN)
          INC   A
          LD    (CONCAN),A
          RET
VARALF:   DEFB  3,0,0
VARNUM:   DEFB  13,0,0
LONCV:    DEFB  3
LONLIN:   DEFS  2
NUMLIN:   DEFS  2
FINPRO:   DEFS  2
DIREC:    DEFS  2
CONCAN:   DEFB  0
VARS:     DEFS  2
VARI:     DEFB  0
LONCA:    DEFB  0
DATNUM:   DEFB  13,0,0
          DEFS  20
DATALF:   DEFB  3,0,0
          DEFS  20
;
;
;IMPRIME NUMEROS DECIMALES
;
```

```
2170 ;
2180 DECIMA:  SCF
2190          LD    HL,(NUMLIN)
2200          LD    DE,10000
2210          INC   HL
2220          LD    A,47
2230 DMIL:    INC   A
2240          SBC   HL,DE
2250          JR    NC,DMIL
2260          CALL  PRINT
2270          LD    DE,1000
2280 MIL:     INC   A
2290          SBC   HL,DE
2300          JR    NC,MIL
2310          CALL  PRINT
2320          LD    DE,100
2330 CIEN:    INC   A
2340          SBC   HL,DE
2350          JR    NC,CIEN
2360          CALL  PRINT
2370          LD    DE,10
2380 DIEZ:    INC   A
2390          SBC   HL,DE
2400          JR    NC,DIEZ
2410          CALL  PRINT
2420          ADD   A,L
2430          CALL  PRINT
2440          LD    A," "
2450          CALL  IMPRE
2460          RET
2470 PRINT:   CALL  IMPRE
2480          LD    A,47
2490          JR    NZ,PAS
2500          INC   HL
2510 PAS:     ADD   HL,DE
2520          INC   HL
2530          RET
2540 IMPRE:   PUSH  AF
2550          LD    A,(PRESOR)
2560          AND   A
2570          JR    Z,PANT
2580 WAIT:    CALL  #BD2E
2590          JR    C,WAIT
2600          POP   AF
2610          CALL  #BD31
2620          RET
2630 PANT:    POP   AF
2640          CALL  #BB5A
2650          PUSH  HL
2660          PUSH  DE
2670          PUSH  BC
2680          PUSH  IX
2690          PUSH  AF
2700          LD    A,66
2710          CALL  #BB1E
2720          CALL  NZ,PAUSA
2730          POP   AF
2740          POP   IX
2750          POP   BC
2760          POP   DE
2770          POP   HL
2780          RET
2790 PRESOR:  DEFB  0
2800 PAUSA:   LD    BC,30000
2810 PAUS:    DEC   BC
2820          LD    A,B
2830          OR    C
2840          JR    NZ,PAUS
2850          CALL  #BB03
2860          CALL  #BB18
2870          RET
```

```
ALFA     A0AE   BUC      A10C   BUCL     A0DF
BUSALF   A080   BUSC     A0F6   BUSNUM   A097
CIEN     A1E7   CONCAN   A194   DATALF   A1B0
DATNUM   A199   DECIMA   A1C7   DIEZ     A1F2
DIREC    A192   DMIL     A1D1   ESPACE   A020
FIN      A12B   FINIM    A15D   FINPRO   A190
FINU     A140   IMPAN    A024   IMPRE    A20F
INCCD    A122   INIC     A0D5   LONCA    A198
LONCV    A18B   LONLIN   A18C   MAS      A06C
MIL      A1DC   MOS      A042   NAME     A014
NOALF    A172   NOBUS    A152   NOINC    A125
NUMER    A0C1   NUMLIN   A18E   PANT     A220
PAS      A20C   PAUS     A23D   PAUSA    A23A
PRESOR   A239   PRINT    A204   RESBU    A14F
TABLA    A009   VALFA    A02C   VARALF   A185
VARI     A197   VARNUM   A188   VARS     A195
VNUME    A056   WAIT     A216

Table used:    600   from   1000
```

```
10 REM LISTADOR DE VARIABLES
20 REM PROGRAMA CARGADOR
30 FOR N=&A000 TO &A249
40 READ A:SUMA=SUMA+A
50 POKE N,A
60 NEXT
70 IF SUMA<>58826 THEN PRINT "ERROR
 EN DATAS"
80 DATA 1,9,160,33,32,160,195
90 DATA 209,188,20,160,195,44,160
100 DATA 195,86,160,195,36,160,86
110 DATA 65,76,70,193,86,78,85
120 DATA 77,197,208,0,0,0,0
130 DATA 0,58,57,162,47,50,57
140 DATA 162,201,167,202,174,160,22
1
150 DATA 110,0,221,102,1,126,35
160 DATA 94,35,86,235,17,179,161
170 DATA 14,0,71,126,18,35,19
180 DATA 12,16,249,121,50,152,161
190 DATA 27,26,198,128,18,205,128
200 DATA 160,201,167,202,193,160,22
1
210 DATA 110,0,221,102,1,126,35
220 DATA 94,35,86,235,17,156,161
230 DATA 14,0,71,126,18,35,19
240 DATA 12,16,249,121,50,152,161
250 DATA 27,26,198,128,18,205,151
260 DATA 160,201,33,176,161,34,149
270 DATA 161,62,3,50,151,161,58
280 DATA 152,161,198,3,50,139,161
290 DATA 205,213,160,201,33,153,161
300 DATA 34,149,161,62,2,50,151
310 DATA 161,58,152,161,198,3,50
320 DATA 139,161,205,213,160,201,33
330 DATA 133,161,34,149,161,175,50
340 DATA 151,161,62,3,50,139,161
350 DATA 205,213,160,201,33,136,161
360 DATA 34,149,161,62,1,50,151
370 DATA 161,62,3,50,139,161,205
380 DATA 213,160,201,221,42,149,161
390 DATA 33,112,1,34,146,161,70
400 DATA 35,78,43,120,177,200,205
410 DATA 246,160,42,146,161,237,91
420 DATA 140,161,25,34,146,161,24
430 DATA 233,78,35,70,35,237,67
440 DATA 140,161,94,35,86,35,237
450 DATA 83,142,161,175,50,148,161
460 DATA 22,0,58,139,161,186,204
470 DATA 43,161,126,221,190,0,35
480 DATA 40,8,22,0,221,42,149
490 DATA 161,24,3,20,221,35,11
500 DATA 120,177,32,226,201,58,148
510 DATA 161,167,32,15,62,13,205
520 DATA 15,162,62,10,205,15,162
530 DATA 229,205,199,161,225,58,151
540 DATA 161,254,0,40,11,254,1
550 DATA 40,7,58,152,161,71,43
560 DATA 16,253,126,203,127,32,6
570 DATA 205,15,162,35,24,245,203
580 DATA 191,205,15,162,58,151,161
590 DATA 254,1,40,9,254,2,40
600 DATA 5,62,36,205,15,162,62
610 DATA 32,205,15,162,221,42,149
620 DATA 161,22,0,58,148,161,60
630 DATA 50,148,161,201,3,0,0
640 DATA 13,0,0,3,0,0,0
650 DATA 0,0,0,0,0,0,0
660 DATA 0,0,0,13,0,0,0
670 DATA 0,0,0,0,0,0,0
680 DATA 0,0,0,0,0,0,0
690 DATA 0,0,0,0,0,3,0
700 DATA 0,0,0,0,0,0,0
710 DATA 0,0,0,0,0,0,0
720 DATA 0,0,0,0,0,0,0
730 DATA 55,42,142,161,17,16,39
740 DATA 35,62,47,60,237,82,48
750 DATA 251,205,4,162,17,232,3
760 DATA 60,237,82,48,251,205,4
770 DATA 162,17,100,0,60,237,82
780 DATA 48,251,205,4,162,17,10
790 DATA 0,60,237,82,48,251,205
800 DATA 4,162,133,205,4,162,62
810 DATA 32,205,15,162,201,205,15
820 DATA 162,62,47,32,1,35,25
830 DATA 35,201,245,58,57,162,167
840 DATA 40,10,205,46,189,56,251
850 DATA 241,205,49,189,201,241,205
860 DATA 90,187,229,213,197,221,229
870 DATA 245,62,66,205,30,187,196
880 DATA 58,162,241,221,225,193,209
890 DATA 225,201,0,1,48,117,11
900 DATA 120,177,32,251,205,3,187
910 DATA 205,24,187,201,0,0,0
```