Talent

Manual MSX Turbo Basic

Talent

MSX Turbo Basic

TELEMATICA S.A.

Chile 1347 – Tel: 37-0051 al 54
1098 – BUENOS AIRES – ARGENTINA
ISBN 950-9688-06-X

Impreso en Argentina
Printed in Argentina

Hecho el depósito que marca la Ley 11.723

Licencia Telemática S.A. de usuario final:

El otorgante concede al usuario final una licencia consistente en el derecho a utilizar el "software" de acuerdo a los siguientes términos:

1. El usuario final solo podrá emplear el "software" en un solo sistema de computadora en cada momento dado.
2. Se permite al usuario final la transferencia a terceros de esta licencia y del "software" siempre que:

(a) Dicho tercero se avenga a todos los términos del presente acuerdo

(b) El usuario final no retenga en su poder copia alguna del "software".

Telemática S.A. no asume ningún tipo de responsabilidad sobre el uso y aplicaciones de este software, o por errores en este manual o en su software. Puede existir, sin embargo soporte adicional y/o garantías extras de parte de Microsoft Corporation o ASCII Corporation sobre este software. Telemática no tiene ninguna relación en ello, y no asume ninguna responsabilidad o garantía relacionada con lo atendido.

Este manual está sujeto a cambios sin previo aviso y no constituye una obligación para Telemática S.A. informar sobre estos cambios.

Indice

CAPITULO 1

INTRODUCCION

Analizaremos en esta introducción la relación existente entre un intérprete y un compilador, utilizando algunas definiciones informáticas en la medida que se necesiten.

1.1) El Intérprete BASIC: descripción general.

Toda computadora digital maneja, por definición, exclusivamente códigos binarios, los cuales son interpretados por la CPU (Central Processing Unit, Unidad Central de Procesamiento) para ejecutar comandos muy sencillos de manipulación de datos y cálculos elementales tales como suma y resta. Estos códigos se representan internamente mediante largas cadenas de unos y ceros.

Sin embargo, la computadora MSX, por ejemplo, maneja muy bien instrucciones tales como:

```
PRINT 3.SQR(4)/36
```

¿De dónde surge esta capacidad de por sí aparentemente incompatible con el funcionamiento de la CPU? De un programa incorporado en la computadora que se denomina intérprete MSX-BASIC.

El intérprete MSX-BASIC es un programa en lenguaje de máquina cuya tarea consiste en traducir al lenguaje máquina las instrucciones que reciba en BASIC. Una vez que las instrucciones están en código de máquina, la CPU ya puede trabajar con ellas y, por lo tanto, entenderlas.
De lo dicho surge que la CPU no entiende realmente BASIC, por lo que antes de que pueda ejecutar programas escritos en dicho lenguaje hay que traducirle todo a lenguaje de máquina. Es como cuando alguien nos habla, por ejemplo, en francés y no le entendemos; para poder comunicarnos con él necesitamos o bien un buen diccionario, o bien un intérprete o traductor que nos vaya traduciendo el francés a castellano.

Análogamente, la parte del sistema operativo que realiza ese trabajo de traducción se denomina intérprete del BASIC. La "traducción" se realiza cada vez que se ejecutan las sentencias. El tiempo necesario para realizar ésta traducción se suma al tiempo necesario para ejecutar el proceso especificado por las sentencias.

Ya definido qué es un intérprete, pasamos a analizar qué es un compilador, y las ventajas de éste frente al intérprete.

1.2) Compilador vs. Intérprete.

El hecho de que se tengan que traducir las instrucciones en BASIC antes de poder trabajar con ellas explica el porqué es más lento utilizar programas escritos en BASIC que los escritos en lenguaje de máquina, por ejemplo. El proceso de traducir un lenguaje a otro lleva tiempo, y a menudo las instrucciones en código de máquina resultantes no son tan eficientes para el tipo de función que se diseñó.

Sin embargo, un programa escrito en lenguaje de máquina posee varias desventajas frente al intérprete, siendo las más destacadas las siguientes:

a) Los programas en lenguaje de máquina son difíciles de leer y de detectar los errores que tengan.

b) Son específicos para cada tipo de computadora.

c) Se utiliza una gran cantidad de instrucciones.

d) Los programas que usan aritmética complicada son difíciles de programar.

Por eso como solución intermedia surge el compilador.

El proceso de traducción de un programa BASIC a código de máquina utilizando el intérprete se realiza instrucción por instrucción, de manera que en el siguiente programa:

```
10 FOR I=1 TO 100
20 PRINT I;
30 NEXT I
```

La línea 20 es traducida 100 veces, ya que aunque ya se tradujo PRINT I, la siguiente vez que se ejecuta, el intérprete "olvidó" su traducción previa.

El compilador, en cambio, cumple una tarea similar a la del intérprete, pero la realiza una sola vez.

Como el intérprete, el compilador traduce una tras otra las instrucciones BASIC. Pero la gran diferencia está en que el compilador traduce todo el programa en una única operación, generando una versión completa del mismo en código de máquina y almacenándola como versión "objeto".

Siguiendo el ejemplo anterior, el compilador traduce la instrucción PRINT I una sola vez, evitando tener que "retraducirla" 99 veces.

Sin embargo, todo tiene su precio. La ventaja de hacer una traducción única en un compilador tradicional se diluye al tener que realizar varios pasos hasta lograr que un programa se pueda ejecutar.

Los pasos a seguir en un compilador tradicional son los siguientes:

a) Utilizar un editor de texto para cargar el programa fuente y grabarlo (generalmente en disco).
b) Correr el programa compilador que analizará el programa fuente e indicará si hubo errores. En este caso deberá ir al paso a) nuevamente hasta que no haya errores de compilación (por ejemplo, errores de sintaxis).

c) Una vez que el compilador tradujo exitosamente el programa fuente, genera otro archivo denominado "código intermedio", que es un programa que está más cerca del código de máquina, pero todavía falta incorporar ciertas definiciones ("linkedición" en lenguaje informático. Consiste en resolver en direcciones de memoria definitivas las que genera el compilador, generalmente no fijadas sino de manera relativa).

d) Una vez ensamblado el programa, se ejecuta el mismo para verificar si no hay errores de lógica. En caso de existir, se deberá

corregir el programa fuente y repetir todo el proceso de compilación hasta lograr que se ejecute correctamente.

O sea, para el usuario del programa es mejor el compilador por su mayor eficiencia y velocidad, pero el programador debe pasar la "odisea de la compilación" si utiliza un compilador tradicional.

La utilización del TURBO BASIC evita estas complicaciones.

1.3) TURBO BASIC vs. Compilador Tradicional.

El TURBO BASIC es un compilador que se puede invocar desde un programa en MSX-BASIC.

La principal ventaja que tiene el compilador TURBO BASIC versus los compiladores tradicionales es que tiene la flexibilidad del Intérprete para edición y depuración (corrección) de programas, con la velocidad de los programas compilados.

Esto se logra gracias a la gran flexibilidad que tiene el sistema MSX, que permite agregar comandos al intérprete que acceden al compilador en forma directa, sin tener que pasar por el paso intermedio de la edición y ensamblaje.

Con el TURBO BASIC, basta ingresar el programa como lo hace habitualmente en su MSX, y luego, ejecutando un comando, el TURBO BASIC toma el control, traduce todo el programa y lo ejecuta, con la única -y gran- diferencia en la velocidad de ejecución.

En el siguiente capítulo describiremos cómo utilizar el TURBO BASIC, con ejemplos aclaratorios y comparaciones con el intérprete BASIC.

CAPITULO 2

COMO UTILIZAR EL TURBO BASIC

Vamos a describir algunas de las técnicas de programación utilizando el compilador BASIC.

2.1) Métodos para utilizar el TURBO BASIC.

Existen dos formas de utilizar el compilador BASIC a saber:

1) Compilar todo el programa BASIC en una sola operación y ejecutarlo.

2) Mientras se está en la ejecución normal del programa BASIC, compilar y ejecutar las líneas comprendidas entre los comandos CALL TURBO ON y CALL TURBO OFF. La ejecución del programa BASIC continua a partir de la instrucción siguiente al CALL TURBO OFF.

2.1.1) Si se compila la totalidad del programa:

En este caso se programa en forma normal (con algunas restricciones que luego detallaremos) en BASIC y luego simplemente con la instrucción CALL RUN (o _RUN), compila todo el programa y lo ejecuta. La velocidad de ejecución es de 20 a 30 veces menor que la del MSX-BASIC.

2.1.2) Si se compila parte del programa.

Esta opción se utiliza en caso de ser imprescindible utilizar alguna sentencia del intérprete BASIC que este compilador no acepta (tal como manejo de archivos).
En este caso, a las líneas de programa que se quieran compilar se les antepone una línea con la sentencia CALL TURBO ON (o _TURBO ON) Y al final de las líneas a compilar se le coloca ,la sentencia CALL TURBO OFF (o _TURBO OFF).

Hay que tener en cuenta que estas sentencias deben ser únicas en la línea, o sea que no se pueden utilizar multisentencias en las líneas que figuren CALL TURBO ON o CALL TURBO OFF.

Finalmente, se ejecuta la sentencia RUN y el programa comenzará su ejecución. Cuando el intérprete BAS1C encuentra la sentencia CALL TURBO ON, le pasa el control al compilador TURBO BASIC, que se encarga de compilar todo el programa hasta que encuentra la sentencia CALL TURBO OFF y ejecuta las rutinas compiladas.

Cuando termina de ejecutar el bloque compilado, el TURBO BASIC devuelve el control al intérprete BASIC y éste continúa la ejecución de las sentencias siguientes al CALL TURBO OFF, en forma normal.

Hay algunos puntos que debemos destacar:

A) LAS VARIABLES UTILIZADAS DENTRO DE LAS RUTINAS COMPILADAS DESAPARECEN AL TERMINAR SU EJECUCION.

B) LAS VARIABLES FUERA DEL BLOQUE COMPILADO SON INDEPENDIENTES DE LAS VARIABLES DENTRO DEL BLOQUE, AUN SIENDO HOMONIMAS.

Sin embargo, muchas veces necesitamos que la parte compilada "conozca" las variables de la parte interpretada, por lo cual deberemos pasar parámetros entre ambas partes.

Esto se puede interpretar como que el contenido de ciertas variables se hace coincidente tanto dentro como fuera del bloque de compilación, o sea que estas variables son compartidas por el TURBO BASIC y el

intérprete. En el TURBO BASIC, se puede lograr esto sólo para variables enteras simples u ordenamientos (arrays) del mismo tipo con subindice.

Para pasar parámetros al compilador, basta escribir en lugar de CALL TURBO ON, la sentencia CALL TURBO ON (XX,XX,XX ...) donde XX son nombres de variables que van a ser compartidas entre la parte compilada y la interpretada.

Por ejemplo:

```
CALL TURBO ON (A%,B())
```

En este caso, la variable A% Y el vector B() contienen los mismos valores, tanto dentro como fuera de la parte compilada del programa.

Veamos un par de ejemplos ilustrativos:

Programa 1:

```
10 ' Programa No. 1
20 '
30 DEFINT A-Z
40 SCREEN 2,0:COLOR 15,1,1
50 SPRITE$(0)=STRING$(S,255)
60 X=255\2:Y=191\2:VX=1:VY=1:XMX=255-9:YMX=191-9
70 LINE (7,0)-(255,191),13,B
80 ' Ciclo principal
90 X=X+VX:Y=Y+VY
100 A$=INKEY$
110 IF X<8 OR X>XMX OR Y<1 OR Y>YMX THEN GOSUB 160
120 PUT SPRITE 0,(X,Y),12,0:PSET (X+4, Y+4),13
130 IF A$=CHR$(13) THEN END
140 GOTO 90
150 ' Rutina Ref1exion
160 IF X>7 AND X<=XMX THEN VY=-VY:Y=Y+VY:RETURN
170 IF Y>O AND Y<=YMX TREN VX=-VX:X-X+VX:RETURN
180 VY=-VY:VX=-VX:Y=Y+VY:X=X+VX:RETURN
```

Programa 2:

```
10 ' Programa No. 2
20 '
30 DEFINT A-Z:DIM C(20,20)
40 SCREEN 2:COLOR 15,1,1:CLS
50 '
60 _TURBO ON
70 FOR X=l TO 20
80   LINE(X*8,0)-(X*8,160)
90 NEXT X
100 FOR Y=0 TO 20
110   LINE(8,Y*8)-(160,Y*8)
120 NEXT Y
130 _TURBO OFF
140 '
150 FOR X=1 TO 19
160   FOR Y=0 TO 19
170     C(X,Y)=RND(1)*13+2
180   NEXT Y
190 NEXT X
200 '
210 TURBO ON (C ())
220 FOR Y-O TO 19
230   FOR X=1 TO 19
240     LINE(X*8+1,Y*8+1)-(X*8+7,Y*8+7),C(X,Y),BF
250   NEXT X
260 NEXT Y
270 _TURBO OFF
280 '
290 GOTO 150
```

Analicemos el primer listado: es un ejemplo en el cual podemos compilar todo el programa con la sentencia CALL RUN.

En la línea 30, con la sentencia DEFINT A-Z hemos definido como entero a todas las variables que se van a utilizar.

En el resto del programa, todas las sentencias que se utilizaron son sentencias comunes del BASIC, por lo tanto se pueden correr con la

sentencia RUN, pero su velocidad de ejecución será de 20 a 30 veces mayor.

Ahora veamos el listado 2; en este programa se ha utilizado compilación en bloques.

Lo que está fuera de compilación es al solo efecto de demostrar cómo se particionan los bloques.

Para el segundo bloque se ha pasado como parámetro el vector C().

2.3) Tipos de variables y precisión.

El tipo de variable numérica por defecto es punto flotante. A diferencia del intérprete, la representación del número se hace mediante 3 bytes. Pueden tomar valores desde 2.939E-39 a 1.7013E+38 y la precisión de los cálculos es de 4.5 cifras significativas.

La representación interna es la siguiente:

<exponente> <mantisa menos significativa> <mantisa más significativa>

Ejemplo:

0.5	->	80h,00h,00h
-1	->	81h,00h,80h

TURBO BASIC maneja la aritmética de enteros en forma análoga al MSX-BASIC.

Los nombres de las variables pueden ser de cualquier longitud pero sólo los dos primeros caracteres son significativos. Los ordenamientos pueden tener cualquier dimensión.

Debe tenerse en cuenta que no se pueden realizar operaciones con variables alfanuméricas muy complejas. Para estos casos es recomendable hacerlo fuera del bloque de compilación, método utilizado en el listado 2.

2.4) Directivas especiales del TURBO BASIC.

Mediante las directivas especiales se le puede indicar al TURBO BASIC que realice tareas especiales que no están previstas por el intérprete. Las directivas se componen de una línea que comienza con "comentario" (REM o ') seguida de la directiva.

Estas directivas son las siguientes:

2.4.1) DIRECTIVA #i

Cuando '#i' se encuentra al principio de una línea de comentario, los valores ubicados a continuación en la línea se ubican directamente en el código objeto (por byte). Un número precedido por el símbolo '@' especifica la dirección de una línea (palabra de 2 bytes), y un nombre de una variable especifica la dirección de la variable (palabra de 2 bytes).

Por ejemplo:

```
10 A%=0
20 IF A%>100 THEN END
30 '#I &H2A,A%,&H23,&H22,A%,&HC3,@10
```

En este caso, la línea 30 es completamente idéntica a:

```
30 A%=A%+1:GOTO 10
```

Obsérvese que los valores indicados en la línea 30 en hexadecimal corresponden al código de máquina del microprocesador Z80 y por lo tanto, para usar esta directiva se deberá tener conocimiento del tema.

En el ejemplo, en lenguaje ensamblador la línea 30 se habría expresado:

```
LD  HL.(A%)
INC HL
LD  (A%),HL
JP  @10
```

2.4.2) DIRECTIVA #C

La directiva '#c' se utiliza para habilitar/deshabilitar el ajuste de coordenadas. Cuando se especifica '#c+', la coordenada Y (vertical) se ajusta si se produce desborde. '#c- suprime el ajuste. El valor por defecto es '#c+.

Ejemplo:

```
10 SCREEN2
20 REM #C-
30 L1NE (0,0)-(255,255)
40 ' #C+
50 LINE (0,0)-(255,255)
```

El ajuste de coordenadas consiste en verificar los límites de la pantalla respecto a las coordenadas indicadas por el usuario. Si se excede de dichos límites, el ajuste consiste en ignorar las coordenadas fuera de rango.

2.4.3) D1RECTIY #N

La directiva '#N activa la verificación de desborde de números enteros. Cuando, por ejemplo, se hace un bucle FOR-NEXT con variables enteras de la forma:

```
10 FOR I%=0 TO &H7FFF
20 NEXT I%
```

Bajo el entorno del MSX-BASIC, esta rutina generará un error de "Desborde"; pero si se la ejecuta con el TURBO BASIC, entrará en un loop infinito.

Para evitar este inconveniente se provee de la directiva especial '#N+ (precedido por REM) Que le indica al compilador que vigile si los cálculos se van fuera del rango de los enteros.

Sin embargo es conveniente no abusar de esta opción, ya que consume más memoria y disminuye la velocidad de ejecución.

Para cancelar esta función se utiliza la opción '#N-.

2.5) Manejo de Interrupciones: sentencias ON... GOSUB…

TURBO BASIC soporta el manejo de interrupciones. Sin embargo cuando se utilizan estas rutinas se genera un código compilado largo y lento, debido a que la rutina de verificación de interrupción se llama cada vez que se ejecuta una sentencia.

El manejo de alfanuméricos se mantiene igual al MSX-BASIC, con ciertas limitaciones que se explicitan en el siguiente capítulo, donde también veremos las principales diferencias entre el TURBO BASIC y el intérprete MSX-BASIC.

CAPITULO 3

DIFERENCIAS ENTRE TURBO BASIC y MSX-BASIC.

3.1) Diferencias de uso.

Veamos algunas observaciones que se tienen que tener en cuenta al programar para el TURBO BASIC.

1) Todos los ordenamientos deben ser dimensionados previamente, aun los que vayan a utilizar subíndices menores a 10.

2) La sentencia DIM sólo puede estar precedida por las sentencias DEFINT, DEFSNG, DEFDBL, REM, DATA Y DIM; de lo contrario generará el error "DIM ya usado".

3) Cuando se llame a rutinas en código de máquina, los parámetros que van a continuación de la sentencia USR sólo podrán ser valores enteros.

4) Cada variable alfanumérica toma 256 bytes de memoria, por lo tanto su uso indiscriminado provocará el mensaje "Memory full" (Memoria completa).

5) Las variables que se utilizan como distribuidor en las sentencias ON GOTO y ON GOSUB son tomadas para la ejecución con el valor que surge del resto de la división de las mismas por 256.

6) No se verifican los límites de los parámetros que se pasan a los funciones ni los subíndices de los ordenamientos.

7) La precisión numérica es de hasta 4,5 dígitos, y el rango numérico va desde +2,939E-39 hasta +1,7012 E+38.

8) Cuando se imprimen números con la sentencia PRINT, las cifras mayores a 10000 serán puestas en notación científica (con el exponente E).

9) Dentro del TURBO BASIC, tanto las variables de doble precisión como de simple precisión son tratados de igual forma, por lo tanto su distinción dentro del programa (por ejemplo A# con A!) son inválidas.

10) Los resultados de operaciones que no sean divisiones o potencias, normalmente se resuelven como enteros. Esto puede traer la siguiente consecuencia:

```
10 A%=100
20 PRINT A%*A%
```

En MSX-BASIC, el resultado será el número real 40000; pero para el TURBO BASIC será el número entero -25536.

Para evitarlo, cuando se sabe "a priori" que el resultado puede salir fuera de rango de los enteros, es necesario anteponer el indicador de número de punto flotante, como por ejemplo:

```
PRINT 1! *A% *A%
```

Con esto dará el resultado correcto, pero a su vez provoca demora en la ejecución y aumento en el consumo de memoria.

11) No se hace ningún tipo de validación de subíndices de los ordenamientos, excepto el número de dimensiones.

12) No se verifican los rangos de los parámetros en _TURBO ON

13) Se debe especificar la variable de la sentencia NEXT (en esta versión del TURBO BASIC).

Ejemplo:

```
10 FOR I=1 TO 10:NEXT I (Correcto)
10 FOR I=1 TO 10:NEXT (Incorrecto en TURBO BASIC).
```

14) Sólo se admite una variable por sentencia INPUT.

15) Los números. de línea de la directiva #i no se renumeran.

3.2) Listado de sentencias BA5IC y su implementación en TURBO BA5IC.

En este apartado brindamos una tabla ordenada alfabéticamente de las sentencias, funciones y operadores del MSX-BASIC referenciada a su utilización en TURBO BASIC.

```
ABS     Igual que MSX-BASIC.
AND     Igual que MSX-BASIC.
ASC     Igual que MSX-BASIC.
ATTR$   N.D. (No Disponible)
ATN     Igual que MSX-BASIC.
AUTO    N.D.
BASE    N.D.
BEEP    Igual que MSX-BASIC.
BIN$    Igual que MSX-BASIC.
BLOAD   N.D.
BSAVE   N.D.
CALL    N.D.
CDBL    N.O.
CHR$    Igual que MSX-BASIC.
CINT    N.D.
CIRCLE  No se admite <excentricidad> ni <angulo inicial/finaL>.
CLEAR   N.D.
CLOAD   N.D.
CLOSE   N.D.
CLS     Igual que MSX-BASIC.
CMD     N.D.
COLOR   Igual que MSX-BASIC.
CONT    N.D.
COPY    Sólo se admite COPY de gráficos (en MSX2).
COS     Igual que MSX-BASIC.
CSAVE   N.D.
CSNG    N.D.
CSRLIN  Igual que MSX-BASIC.
CVD     N.D.
CVI     N.D.
```

CVS	**N.D.**
DATA	**Igual que MSX-BASIC.**
DEF	**Sólo se acepta 'DEF USR'.**
DEFDBL	**Igual que MSX-BASIC.**
DEFINT	**Igual que MSX-BASIC.**
DEFSNG	**Igual que MSX-BASIC.**
DEFSTR	**Igual que MSX-BASIC.**
DELETE	**N.D.**
DIM	**Igual que MSX-BASIC, con la diferencia que DIM debe ejecutarse antes de cualquier sentencia ejecutable, y la declaración debe realizarse utilizando constantes enteras.**
DRAW	**N.D.**
DSKF	**N.D.**
DSKI$	**N.D.**
DSKO$	**N.D.**
ELSE	**Igual que MSX-BASIC.**
END	**Igual que MSX-BASIC.**
EOF	**N.D.**
EQV	**N.D.**
ERASE	**N.D.**
ERL	**N.D.**
ERR	**N.D.**
ERROR	**N.D.**
EXP	**Igual que MSX-BASIC.**
FIELD	**N.D.**
FILES	**N.D.**
FIX	**Igual que MSX-BASIC.**
FN	**N.D.**
FOR	**Igual que MSX-BASIC.**
FPOS	**N.D.**
FRE	**N.D.**
GET	**N.D.**
GO TO	**Igual que MSX-BASIC.**
GOSUB	**Igual que MSX-BASIC.**
GOTO	**Igual que MSX-BASIC.**
HEX$	**Igual que MSX-BASIC.**
IF	**Igual que MSX-BASIC.**
IMP	**N.D.**
INKEY$	**Igual que MSX-BASIC.**
INP	**Igual que MSX-BASIC.**
INPUT	**Una sola variable por INPUT.**

INSTR Igual que MSX-BASIC.
INT Igual que MSX-BASIC.
IPL N.D.
KEY Igual que MSX-BASIC, excepto 'KEY <n>, <alfanumérico>' y 'KEY LIST'
KILL N.D.
LEFT$ Igual que MSX-BASIC.
LEN Igual que MSX-BASIC.
LET Igual que MSX-BASIC.
LFlLES N.D.
L1NE Igual que MSX-BASIC, excepto 'LINE INPUT'.
L1ST N.D.
LLIST N.D.
LOAD N.D.
LOC N.D.
LOCATE Igual que MSX-BASIC, excepto que deben indicarse simultáneamente x e y, mientras que el <activador de cursor> no se admite.
LOF N.D.
LOG Igual que MSX-BASIC.
LPOS Igual que MSX-BASIC.
LPRINT Igual que MSX-BASIC.
LSET N.D.
MAX N.D.
MERGE N.D.
MID$ Igual que MSX-BASIC.
MKD$ N.D.
MKI$ N.D.
MKS$ N.D.
MOD Igual que MSX-BASIC.
MOTOR N.D.
NAME N.D.
NEW N.D.
NEXT Siempre necesita especificación de variable.
NOT Igual que MSX-BASIC.
OCT$ Igual que MSX-BASIC.
OFF Igual que MSX-BASIC.
ON Igual que MSX-BASIC.
OPEN N.D.
OR Igual que MSX-BASIC.
OUT Igual que MSX-BASIC.
PAD Igual que MSX-BASIC.
PAINT Igual que MSX-BASIC.

PDL Igual que MSX-BASIC.
PEEK Igual que MSX-BASIC.
PLAY N.D.
POINT Igual que MSX-BASIC.
POKE Igual que MSX-BASIC.
POS Igual que MSX-BASIC.
PRESET Igual que MSX-BASIC.
PRINT Igual que MSX-BASIC, excepto que SPC y TAB no se aceptan, el espaciado por zonas (',') actúa diferente, y los números no se alinean para conformar la línea (es decir, no se envían al siguiente renglón si exceden al mismo, simplemente se cortan y siguen abajo).
PSET Igual que MSX-BASIC.
PUT Sólo se admite 'PUT SPRITE'
READ Igual que MSX-BASIC.
REM Igual que MSX-BASIC.
RENUM N.D.
RESTORE Igual que MSX-BASIC.
RESUME N.D.
RETURN Igual que MSX-BASIC.
RIGHT$ Igual que MSX-BASIC.
RND Igual que MSX-BASIC.
RSET N.D.
RUN Igual que MSX-BASIC, excepto que las variables no se inicializan.
SAVE N.D.
SCREEN Sólo se admite <modo pantalla> y <tamaño sprites>.
SET Sólo se admite SET PAGE (en MSX 2).
SGN Igual que MSX-BASIC.
SIN Igual que MSX-BASIC.
SOUND Igual que MSX-BASIC.
SPACE$ Igual que MSX-BASIC.
SPC(N.D.
SPRITE Igual que MSX-BASIC.
SQR Igual que MSX-BASIC.
STEP Igual que MSX-BASIC.
STICK Igual que MSX-BASIC.
STOP Igual que MSX-BASIC.
STR$ Igual que MSX-BASIC.
STRIG Igual que MSX-BASIC.
STRING$ Igual que MSX-BASIC.
SWAP Igual que MSX-BASIC.
TAB(N.D.

```
TAN     Igual que MSX-BASIC.
THEN    Igual que MSX-BASIC.
TIME    Igual que MSX-BASIC.
TO      Igual que MSX-BASIC.
TROFF   N.D.
TRON    N.D.
USING   N.D.
USR     Igual que MSX-BASIC.
VAL     Igual que MSX-BASIC.
VARPTR  Igual que MSX-BASIC.
VDP     Igual que MSX-BASIC.
VPEEK   Igual que MSX-BASIC.
VPOKE   Igual que MSX-BASIC.
WAIT    Igual que MSX-BASIC.
WIDTH   N.D.
XOR     Igual que MSX-BASIC.
'       Igual que MSX-BASIC.
>       Igual que MSX-BASIC.
=       Igual que MSX-BASIC.
<       Igual que MSX-BASIC.
+       Igual que MSX-BASIC.
.       Igual que MSX-BASIC.
*       Igual que MSX-BASIC.
/       Igual que MSX-BASIC.
^       Igual que MSX-BASIC.
\       Igual que MSX-BASIC.
```

En el siguiente capítulo veremos cómo lograr programas más eficientes y los listados de ejemplo que acompañan este manual.

CAPITULO 4

CONSIDERACIONES FINALES Y EJEMPLOS.

4.1) Algunos trucos para acelerar todavía más los programas.

Existen algunos trucos para que los programas compilados por el TURBO BASIC sean más rápidos:

a) Tratar de utilizar división modular o multiplicación ("\" o "*"), en vez de división o potencia ("/" o "^"). De esta forma. Aunque el programa pueda resultar un poco más largo, se acelera notablemente.

b) Las divisiones por números enteros, tratar de hacerlas con cifras en potencias de 2.

   Por ejemplo:

   A=1000/2/16 es más rápido que A=1000/32

c) Las multiplicaciones por enteros, tratar de hacer que sean por múltiplos de 2^N o las cifras 3, 5, 6, 7, 9, l0, 20, 25, 50, 80, 100, 200, 256 y 257.

4.2) Programa de ejemplo:

Programa No. 3 - Plot 3d:

Al ingresar diámetro y altura, siendo los valores ideales para MSX 1 50 y para MSX 2 100 con altura para ambos de 20, se dibuja una figura en forma de rosquilla en un gráfico de 3 dimensiones en perspectiva. Luego de finalizado, se pulsa la tecla RETURN y finaliza el programa. Ejecutar con RUN. Si pulsa RETURN en cualquier momento se suspende la ejecución del programa.

```
10 'Programa No. 3
20 'Titulo: PLOT 3D
30 '
40 SCREEN 0:WIDTH 40:KEY OFF
50 COLOR 15,1,1
60 _TURBO ON
70 DEFINT A,I
80 DIM A(300),I(300)
90 'Ingresa los. parametros de la figura
100 INPUT "DIAMETRO ";R1
110 INPUT" ALTURA ";R2
120 'Testea si es MSXl o MSX2
130 IF PEEK(&HFAFC) THEN 70
140 L1=85:L2=120:L3=120:L4=120:L5=2:L6=50
150 SCREEN 2
160 GOTO 200
170 Ll=140:L2=160:L3=240:L4=240:L5=1:L6=90
180 SCREEN 7
190 'Rutina principal de dibujo
200 FOR I=0 TO 300
210   A(I)=19l:I(I)=0
220 NEXT I
230 B1=5O:B2=100:Sl=12:H=0:Z=0
240 FOR Y=-L1 TO L2 STEP 6
250   Bl=Bl-L5:B2=B2-1:H1=B1:Vl=B2
260     FOR X=-L3 TO L4 STEP 2
270       IF INKEY$=CHR$(13) THEN 410'si pulsa
<RETURN>, interrumpe el dibujo
280       H1=H1+1
290       IF INT((X*16*16)/S1)=INT(X/S1)*16*16 THEN
V1=V1-1
300       MM=R1-SQR(X*X+Y*Y)
310       ZZ=R2*R2-MM*MM
320       IF ZZ<0 THEN Z=0 ELSE Z=SQR(ZZ)
330       H=H1:V=V1-Z
340       IF H<0 THEN 400
350       IF V<=I(H) AND V>=A(H) THEN 390
360       IF V>=I(H) THEN I(H)=V
370       IF V<A(H) THEN A(H)=V
380       PSET (H+L6,V+50),15
```

```
390     NEXT X
400   NEXT Y
410 _TURBO OFF
420 'Fin del dibujo
430 OPEN"grp:"FOR OUTPUT AS 1
440 PRESET (10,10),1
450 PRINT'#l,"pulse <RETURN>"
460 CLOSE
470 I$=INKEY$
480 IF I$=CHR$(13) THEN CLS:END
490 IF I$="" THEN 470
```

Programa No. 4 - Curva del Dragón:

El nombre de este programa se debe a que las curvas dibujadas siguiendo ciertas reglas se asemejan mucho a las representaciones de dragones que realizan los chinos. Estas reglas se representan en este programa por unos y ceros. Si es uno, se gira a la derecha y si es cero, gira a la izquierda. Si la regla es al azar, se generan curvas un poco deformadas. En este programa hemos previsto 11 reglas que permiten dibujar curvas con cierta armonía. Si elige el No. 12, podrá ingresar su propia regla de curva. Ejecutar con RUN.

```
10 'programa No. 4
20 'Titulo: Curva del dragon
30 '
40 _TURBO ON
50 DIM A(125),B$(11)
60 GOSUB 230
70 S=2:GOSUB 430
80 'calcula la curva
90 N=l
100 A=N+50:B=N+75
110 M=A(A)
120 IF A(M)=-l THEN 170
130 R=A(B)*(2*A(M)-1)
140 W=P:P=R*Q:Q=-R*W
150 A(A)=A(A)+A(B)
160 GOTO 70
170 A(A)=A(A)-A(B)
```

```
180 A(B)=-A(B)
190 N=N+1
200 IF INKEY$=CHR$(13) THEN 880
210 IF N<=K THEN 100
220 GOTO 520
230 SCREEN 0
240 GOSUB 550
250 FOR I=1 TO 100
260 A(I)=1:D$=MID$(B$,I,1)
270 IF I<51 THEN IF D$="0" THEN A(I)=0
280 NEXT I
290 A(0)=-1:A(LEN(B$)+1)=-1
300 PRINT
310 INPUT"ORDEN ";K
320 P=0:Q=-1
330 'Testea si es MSXl o MSX2
340 IF PEEK(&HFAFC) THEN 380
350 L1=128
360 SCREEN 2
370 GOTO 400
380 L1=256
390 SCREEN 7
400 X=L1:Y=96
410 RETURN
420  'Dibuja una curva
430 FOR L=1 TO S
440 PSET(X,Y)
450 X=X+P:Y=Y+Q
460 IF Y=-1 THEN Y=191
470 IF Y=192 THEN Y=0
480 IF X=-1 THEN X=255
490 IF X=Ll*2 THEN X=0
500 NEXT L
510 RETURN
520 A$=INKEY$: IF A$="" THEN 520
530 GOTO 60
540  'Ingresa la forma del dibujo
550 PRINT "1)   1"
560 PRINT "2)   110"
570 PRINT "3)   1100"
580 PRINT "4)   1011"
```

```
590 PRINT "5)   1101100"
600 PRINT "6)   1000110"
610 PRINT "7)   1101011"
620 PRINT "8)   1100011"
630 PRINT "9)   10001110"
640 PRINT "10)  11000101"
650 PRINT "11)  11011010"
660 PRINT "12)  ORIGINAL"
670 B$(l)="1"
680 B$(2)="110"
690 B$(3)="1100"
700 B$(4)="1011"
710 B$(5)="1101100"
720 B$(6)="1000110"
730 B$(7)="1101011"
740 B$(8)="1100011"
750 B$(9)="10001110"
760 B$(10)="11000101"
770 B$(11)="11011010"
780 PRINT
790 INPUT"ELIJA EL NUMERO";N
800 IF N<1 OR N>12 THEN 790
810 IF N=12 THEN 830
820 B$=B$(N):RETURN
830 CLS:PRINT:PRINT
840 PRINT"<< ORIGINAL >>":PRINT
850 PRINT "INGRESE 0 ó 1"
860 INPUT B$
870 RETURN
880 '
890 _TURBO OFF
900 CLS
910 END
```

Programa No. 5 - Simulación de pulga:

Este programa simula el movimiento de un conjunto de pulgas en una caja cerrada. Normalmente, asimilamos al concepto de pulga ("bug") a un error en la programación, así que podría decirse que este programa esta plagado de "bugs". El cálculo de las parábolas de salto se realiza únicamente con sumas y restas. Se sale del programa pulsando la tecla

RETURN. Para ejecutar este programa, utilice el comando RUN, y para suspender la ejecución, pulse la tecla RETURN.

Si quiere comparar velocidades, borre las líneas 40 y 120. En vez de pulgas tenemos tortugas saltarinas…

```
10 'Programa No.5
20 'Titulo: SIMULACION DE PULGA
30 '
40 _TURBO ON
50 DIM X1(5),Y1(5),CX(5),CY(5)
60 DIM XD(5),YD(5),C(5)
70 COLOR 15,1,7
80  'Testea si es MSX1 o MSX2
90 IF PEEK(&HFAFC) THEN 130
100 SCREEN 2
110 L1=254:L2=193
120 GOTO 160
130 SCREEN 7
140 L1=512:L2=212
150  'Rutina principal
160 Y=191
170 FOR I=0 TO 5
180  X1(I)=RND(1)*256+1
190  Y1(I)=Y
200  CX(I)=X1(I)
210  CY(I)=Y1(I)
220  XD(I)=RND(1)*10+1
230  YD(I)=RND(1)*15+2
240  C(I)=I+10
250 NEXT I
260 FOR I=0 TO 5
270  PRESET(CX(I),CY(I))
280  PRESET(CX(I),CY(I)-1)
290  PRESET(CX(I),CY(I)-2)
300  PSET(X1(I),Y1(I)),C(I)
310  PSET(X1(I),Y1(I)-1),C(I)
320  PSET(X1(I),Y1(I)-2),C(I)
330  CX(I)=X1(I):CY(I)=Y1(I)
340  X1(I)=X1(I)+XD(I)
```

```
350  Y1(I)=Y1(I)-YD(I)
360  YD(I)=YD(I)-1
370  IF X1(I)<1 THEN XD(I)=-XD(I):X1(I)=0
380  IF X1(I)>L1 THEN XD(I)=-XD(I):X1(I)=L1+1
390  lF Y1(I)>L2 THEN GOSUB 460
400 NEXT I
410 'Si pulso <RETURN>, interrumpe
420 K$=lNKEY$
430 IF K$=CHR$(13) THEN 620
440 GOTO 260
450 '
460 Y1(I)=Y
470 XD(I)=(RND(1)*5+l)*SGN(XD(I))
480 YD(I)=RND(1)*20+l
490 GOSUB 530
500 RETURN
510 '
520  'Efecto sonoro
530 SOUND 6,RND(1)*3l+l
540 SOUND 7,&HF7
550 SOUND 8,16
560 SOUND 11,0
570 SOUND 12,RND(1) *7+1
580 SOUND 13,0
590 RETURN
600  '
610  'Fin del programa
620 _TURBOOFF
630 CLS
640 END
```

Programa No. 6 - Cálculo del número e.

El número e es la base del logaritmo natural o neperiano. Es un número real irracional, y por lo tanto posee infinita cantidad de decimales. De los cuales calculamos 500.

```
10 'Programa No. 6
20 'Calculo del número e con 500 decimales
30 _TURBO ON
40 DEFlNT A-Z
```

```
50 DIM E(255)
60 FOR I=260 TO 2 STEP -1
70 A=1
80 PRINT I
90 FOR K=1 TO 255
100  A=(A MOD I)*100+E(K)
110  E(K)=INT(A/I)
120 NEXT K
130 NEXT I
140 PRINT "e=2.";
150 FOR K=1 TO 250
160 PRINT CHR$(INT(E(K)/10)+48);
170 PRINT CHR$(E(K) MOD 10+48);
180 NEXT K
190 _TURBO OFF
200 END
```

Programa No. 7 - Explosión:

Se simulan las explosiones de fuegos artificiales. Existen 3 tipos de fuegos artificiales. Algunos tienen la pólvora un poco húmeda y por eso su explosión no es tan espectacular. La cantidad de "chispas" es de 26. Programa ideal para las fiestas. Ejecutar con RUN. Se suspende la ejecución pulsando la tecla RETURN.

```
10 'Programa No. 7
20 'Titulo: EXPLOSION
30 '
40 COLOR 15,1,7
50 _TURBO ON
60 DIM X(25),Y(25),XD(25),YD(25)
70 DIM CX(25),CY(25),D(25),C(25)
80 'Testea si es MSX1 o MSX2
90 IF PEEK(&HFAFC) THEN 130
100 L1=200
110 SCREEN 2
120 GOTO 160
130 L1=400
140 SCREEN 7
150   'Rutina principal
```

```
160 X=INT(RND(1)*L1)+28
170 D=INT(RND(1)*3)+1
180 SOUND 6,D*10:SOUND 7,&HE7
190 SOUND 8,16:S0UND 9,16
200 SOUND 11,0:SOUND 12,20-D*5
210 SOUND 13,0
220 FOR I=0 TO 25
230  X(I)=X:CX(I)=X
240  Y(I)=191:CY(I)=191
250  XD(I)=INT(RND(1)*10)-5
260  YD(1)=INT(RND(1)*12)+5
270  C(I)=1NT(RND(l) *13)+2
280 NEXT I
290 P=0
300 FOR I=0 TO 25
310  PRESET(CX(I),CY(I))
320  PRESET(CX(I),CY(I)+1)
330  IF Y(I)>191 THEN P=P+1:GOTO 400
340  PSET(X(I),Y(I)),C(I)
350  PSET(X(I),Y(I)+l),C(I)
360  CX(I)=X(I):CY(I)=Y(I)
370  X(I)=X(I)+XD(I)
380  Y(I)=Y(I)-YD(I)
390  YD(I)=YD(I)-D
400 NEXT I
410 IF P=26 THEN 160
420 K$=INKEY$
430 IF K$=CHR$(13) THEN 470
440 GOTO 290
450 '
460  'Fin del programa
470 _TURBO OFF
480 CLS
490 END
500 '
```

Programa No. 8 - Sort:

Este programa genera un vector de 255 elementos al azar y luego los reordena, con arrastre de otro vector que almacena su posición anterior

al ordenamiento. También se imprime el tiempo de ejecución en segundos. Nótese la diferencia de velocidades entre el TURBO BASIC y el intérprete MSX-BASIC: el tiempo con el programa 'compilado es de 20 segs. aproximadamente, mientras que con el intérprete tarda 190 segs. Ejecutar con CALL RUN.

```
10 ' Programa No. 8
20 ' Titulo: Sort
30 DIM D(255),Y(255)
40 SCREEN 0:KEY OFF
50 PRINT "Generando numeros"
60 FOR I=1 TO 255
70  D(I)=INT(RND(l)*3000)
80  Y(I)=I
90 NEXT I
100 PR1NT "Ordenando…":TIME=0
110 L=10
120 L=INT(L+1)/2
130 FA=1
140 FOR I=1 TO 255-L
150  IF D(I)>D(I+L) THEN SWAP D(I),D(I+L):SWAP
Y(I),Y(I+L) :FA=0
160 NEXT I
170 IF FA=0 THEN 130
180 IF L>1 GOTO 120
190 T1=T1ME/60:FOR I=1 TO 255
200  PRINT " Cantidad=";D(I);" No.";Y(I)
210  IF STRIG(0)<>0 THEN INPUT I$
220 NEXT I:PRINT "Tiempo insumido :";TI;"segs."
230 END
```

Talent MSX

CENTROS DE ASISTENCIA AL USUARIO DE TALENT MSX

Al comprar una computadora de la linea Talent MSX, Ud. tiene el derecho de recibir atención gratuita ante cualquier duda que se le presente, obtener respuesta a sus inquietudes y evacuar todo tipo de consultas técnicas.

Además, Ud. tiene derecho a tomar un curso introductorio que le será de gran ayuda y que le permitirá desempeñarse sin mayores inconvenientes en el medio informático.

El curso de introducción y manejo de Talent MSX tiene como finalidad facilitarle al comprador del equipo su primer contacto con la computadora y demostrarle sus múltiples aplicaciones.

Estos servicios se ofrecen a los usuarios a través de un sistema de **Centros de Asistencia al Usuario de Talent MSX**, estos centros son instituciones que Telemática S.A. ha evaluado como las más capacitadas para ello y son de reconocido prestigio en el dictado de cursos y carreras cortas de informática aplicada.

CAPITAL FEDERAL

Centro Cultural de la Ciudad de Buenos Aires
Taller Logo de computación
Junín 1930
Martes a Sábados de 15 a 19.30 horas

Fundación de Informática y Educación
Centro de Computación Clínica
Asistencia al Usuario Discapacitado
Ramsay 2250 - Pabellón F
Tel. 784-2018
Lunes a Viernes de 8 a 17 horas

Barrio Norte
Uriburu 1063 - Tel. 83-6892/826-6692
Lunes a Viernes de 9 a 21 horas
Sábados de 9 a 12 horas

Belgrano
Mendoza 2728 - Tel. 781-2271
Lunes a Viernes de 15 a 22 horas

Centro
Av. Córdoba 654 - Tel. 392-5328/7611/8043/8051/8251
Lunes a Viernes de 12 a 21 horas
Sábados de 9 a 13 horas

Flores
Gral. Artigas 354 - Tel. 612-3902
Lunes a Viernes de 14 a 20 horas
Sábados de 10 a 13 horas

Palermo
Guatemala 4733 - Tel. 71-4124
Lunes a Viernes de 14 a 21 horas
Sábados de 9 a 13 horas

San Telmo
Chile 1345 - Tel. 37-0051 al 54
Lunes a Viernes de 10 a 13 y de 14 a 19 horas

GRAN BUENOS AIRES

Lanús
Caaguazú 2186 - Tel. 247-0678
Lunes a Viernes de 9 a 13 y de 16 a 20 horas
Sábados de 9 a 13 horas

Morón
Belgrano 160 - Tel. 629-3347
Lunes a Viernes de 9 a 13 y de 14 a 21 horas
Sábados de 9 a 13 horas

Ramos Mejia
Bolívar 55 - ler. piso - Te!. 658-4777
Lunes a Viernes de 9 a 13 y de 14 a 21 horas
Sábados de 9 a 13 horas

San Isidro
Av. Centenario 705 - Te!. 743-9678/747-6094
Lunes a Viernes de 9 a 21 horas
Sábados de 9 a 12 horas

Vicente López
Av. Maipú 625 - Te!. 797-6720
Lunes a Viernes de l0 a 19 horas

INTERIOR DEL PAIS

La Plata - Pcia. de Buenos Aires
Calle 48 No. 529 - Te!. (021) 249905 al 07
Lunes a Viernes de 9 a 21 horas
Sábados de 9 a 13 horas

Bahía Blanca - Pcia. de Buenos Aires
Gral. Paz 257 - Te!. (091) 31582
Lunes a Viernes de 9 a 12 y de 16 a 20 horas

Córdoba - Pcia. de Córdoba
9 de Julio 533
Lunes a Viernes de 8 a 12 y de 16 a 20 horas

Villa Maria - Pcia. de Córdoba
Corrientes 1159 - 2do. piso - Tel (0535) 24311
Lunes a Viernes de 16 a 23 horas
Sábados de 8 a 12 y de 15 a 18 horas

Mendoza - Pcia. de Mendoza
Rivadavia 76 - ler. piso - Te!. (061) 291348/293151
Lunes a Viernes de 8 a 13 y de 16 a 20 horas
Sábados de 8 a 13 horas

Santa Fé - Pcia. de Santa Fé
Rivadavia 2553 Loc.22 - (042) 41832
Lunes a Viernes de 9 a 12 y de 16 a 19 horas
Sábados de 9 a 12 horas

Rosario - Pcia. de Santa Fé
Barón de Mauá 1088
Lunes a Viernes de 8 a 12 y de 15 a 19 horas
Sábados de 9 a 12 horas

S.M. de Tucurnán - Pcia. de Tucurnán
Bolivar 374 - Te!. (081) 245007
Lunes a Viernes de 9 a 18 horas

CURSO DE ORIENTACION y MANEJO DE TALENT MSX

Debe ser presentado en el Centro de Asistencia donde se tome el Curso Gratuito al que tiene derecho por compra.

Nombre: _____

Dirección: _____

Código Postal: _____ Localidad: _____

Centro de Asistencia: _____

Talent
MSX 2

Accesorios

Los Accesorios son el software incorporado a su MSX-2 que le permite en todo momento contar con las siguientes aplicaciones:

Un panel de control.
Una calculadora.
Un reloj.
Un juego.
Un calendario perpetuo.

Permiten además generar en cualquier momento un caracter o símbolo gráfico a partir de su código decimal. Para esto mantenga presionada la tecla <CTRL> y a continuación digite el código decimal correspondiente al símbolo que desea generar, al soltar la tecla <CTRL> el símbolo será generado.

Modo de acceso:

Presionando las teclas <SHIFT> y <CTRL> simultáneamente, se ingresa a los Accesorios siempre que el modo gráfico de la aplicaci6n que le encuentra corriendo sea menor o igual a SCREEN 3. Al salir de los Accesorios, la pantalla es restaurada y la aplicación continúa su ejecución.

Los Accesorios no restan memoria ni destruyen ningún parámetro de la aplicaci6n que este corriendo cuando son invocados.

El software de los Accesorios ha sido diseñado respetando estrictamente la norma MSX, de manera que cualquier aplicación que haya respetado estas normas no entrará en conflicto con los mismos. Sin embargo, estos pueden ser anulados si al encender la máquina se mantiene presionada la tecla <CODE>.

Manejo del cursor:

Al ingresar a los Accesorios aparecerá una flecha (que a partir de ahora denominaremos cursor) en el extremo superior izquierdo de la pantalla y una serie de símbolos gráficos que representan, en orden, cada una de las aplicaciones mencionadas anteriormente.

El cursor puede desplazarse usando las teclas de cursor, el mouse o joystick.

Inicialmente los Accesorios están preparados para desplazar el cursor usando mouse o teclado, si se desea usar joystick debe especificarse en el panel de control como se detalla más adelante.

El mouse puede ser conectado en cualquier momento a la entrada de joystick 1 o 2.

Procedimiento para abrir una aplicación:

Posicione el cursor sobre el símbolo gráfico que desea abrir y presione la barra espaciadora, el botón izquierdo del mouse o el disparador del joystick, según el dispositivo en uso para controlar los Accesorios. Para cambiar de una aplicación a otra basta con repetir el procedimiento anterior. El último símbolo gráfico retorna a la aplicación desde donde los accesorios fueron invocados.

Descripción de las aplicaciones:

Panel de control:

El panel de control tiene como propósito alterar una serie de parámetros, los cuales no se pierden al apagar la máquina. Estos parámetros comprenden a seis grupos a saber:

Ajustar pantalla:

Permite cambiar la ubicación del display en la. pantalla. Posicione el cursor en las flechas verticales u horizontales hacia arriba o abajo de acuerdo a la posición final que desea obtener.

Fecha y hora:

Permite alterar día, mes, año, hora, minutos y segundos. Para esto posicione el cursor en la unidad que quiere cambiar, Presione la barra espaciadora (si esta usando teclado) y finalmente con las flechas hacia arriba o abajo lleve este valor al que usted desea dejar establecido.

Color de frente, fondo y borde:

Deja establecido los colores para SCREEN 0 y SCREEN 1. Al posicionar el cursor sobre el cuadrado que contiene una "a" y presionar la barra espaciadora se accederá a una paleta de colores, dando la posibilidad de elegir el color de frente.

Repita la operaci6n para establecer los colores de fondo y borde posicionando el cursor en el segundo y tercer cuadrado respectivamente. El cuadrado de la derecha se pinta del color elegido.

Los colores harán efectivos recién cuando se vuelva a encender la máquina.

Tono del beep:

Existen cuatro tonos de beep (BEEP es una instrucción de BASIC). Pruebe posicionando el cursor sobre cada uno de ellos y elija el que sea de su agrado.

Joystick o mouse:

Establece que periférico va a usarse para el manejo de los accesorios.

Volumen del beep:

Existen cuatro niveles de beep. Elija el que sea de su agrado.

Calculadora:

Permite el cálculo de las operaciones básicas en aritmética de punto flotante, cálculo de funciona trascendentes y dispone de una memoria a la cual se le pueden sumar o restar valores.

Para accionar el teclado de la calculadora lleve el cursor hasta la tecla deseada y presione la barra espaciadora (si está usando teclado). Las teclas de la calculadora pueden accionarse directamente desde el teclado según se especifica más adelante.

Reloj:

Da la hora, el día y número de día actual. Para poner en hora el reloj acceda al panel de control.

Juego:

Se ha implementado e1 conocido juego de 15 inventado por Lewis Carroll. Para correr las fichas del juego posicione el cursor en la ficha que desea mover y luego presione la barra espaciadora (si está usando el teclado). Las fichas pueden ser movidas directamente desde el teclado según se especifica más adelante.

Calendario:

Este es un calendario perpetuo que se abre en el día, mes y año corriente. Posicione el cursor en las flechas que se encuentran debajo del mes y del año de acuerdo a lo que desea cambiar y presione la barra espaciadora (si está usando teclado) para ir recorriendo los distintos meses.

Salida de Accesorios:

El último símbolo gráfico representa una puerta de salida, posicione el cursor sobre ésta y luego presione la barra espaciadora (si está usando teclado) para salir de los Accesorios y continuar con la aplicación que estaba corriendo en el momento en que estaba corriendo en el momento en que fueron invocados.

Teclas para el manejo de los accesorios:

El juego, el calendario y la calculadora pueden ser manejados con el cursor o directamente desde el teclado. Las teclas equivalentes son las siguientes:

Calculadora	Teclado	Función
0-9	0-9	
CA	C	Borra la última operación realizada.
C	A	Borra toda la secuencia.
+	+	Suma.
*	*	Multiplicación.
/	/	División.
=	= ó CR	Presenta el resultado.
EXP	E	Exponencial.
+/-	%	Cambio de signo.
SEN	F1	Seno.
COS	F2	Coseno.
TAN	F3	Tangente.
ASN	F4	Arco seno.
ACS	F5	Arco coseno.
ATN	F6	Arco tangente.
raíz	F7	Raíz cuadrada.
LOG	F8	Logaritmo neperiano.
M+	F9	Memoria más.
M-	F10	Memoria menos.
MR	M	Recuperar de memoria.
Min	I	Entrar en memoria.

Juego	Teclado
Ficha abajo	X
Ficha arriba	E
Ficha derecha	D
Ficha izq.	S

Calendario	Teclado
Mes -	F1
Mes+	F2
Año-	F3
Año+	F4

EXTENSIONES MSX2 BASIC

Se presentan en esta publicación las extensiones MSX BASIC 2.0 al lenguaje BASIC, creadas para soportar la ampliada capacidad gráfica de las nuevas computadoras personales Talent MSX2.
La versión MSX BASIC 2.0 incluye importantes mejoras en estas funciones, manteniendo una total compatibilidad ascendente con las versiones MSX BASIC 1.0. Se han agregado también comandos y funciones especiales para manejar el reloj calendario interno permanente, y el emulador a de disco en RAM.
Con las computadoras Talent MSX2 y el MSX BASIC 2.0 es posible acceder a un nuevo espectro de posibilidades que permiten:

- 5 nuevos modos de pantalla, incluyendo texto en 80 columnas y alta resolución de 512x212 pixels.
- 256 colores disitntos en modo multicolor.
- 512 colores disponibles en modo alta resolución.
- 8 sprites por línea.
- Sentencias para el manejo de reloj calendario interno.
- Sentencias para emular un disco en RAM, similares al Disk-BASIC.

Este Manual incluye solamente las extensiones de la versión MSX-BASIC 2.0. Para tener una referencia completa del lenguaje, debe utilizarse en conjunto con el Manual MSX-BASIC.

Producido en Argentina por TELEMATICA S.A.