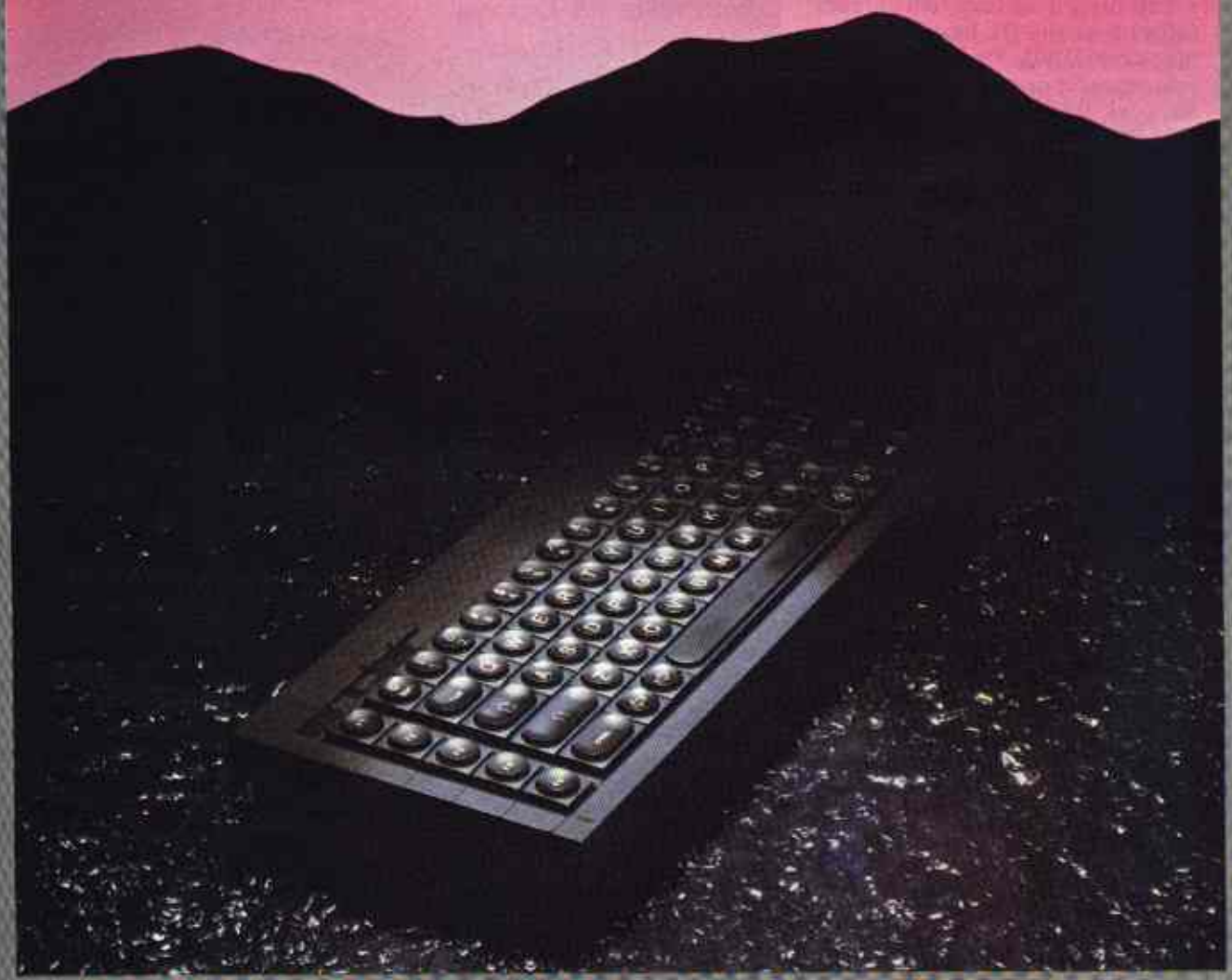
QL MAGAZINE

Suplemento especial

Quién es quién en el mundo del QL



Los BUGS del sistema operativo

Quien es quien en el mundo del QL

Una de las mayores preocupaciones de los usuarios del QL es la escasez de software y hardware en España. Cada vez son más quienes recurren al mercado británico, dado el desinterés de Investrónica. Por ello, ofrecemos una lista de las empresas británicas que trabajan para el QL, especificando los productos que comercializan.

PROSPERO SOFTWARE LTD.
190 CASTLENAU
LODON SW13 9D4

Esta firma desarrolla actualmente software para el QL bajo el sistema operativo QDOS. Prospero es conocida por sus programas para CP/M-80 y el PRO-PASCAL-ISO para IBM PC, primer compilador de Pascal para microordenador que superó las pruebas de homologación del I.S.O.

DIGITAL PRECISION
222 THE AVENUE
LONDON E4 9SE

Prácticamente desconocida hasta ahora, Digital Precision ha sido lanzada al estrellato por el SUPERCHARGE SUPERBASIC COMPILER (Compilador de SUPERBASIC). Otros productos para QL desarrollados por esta empresa son:

Super Forth
Super Sprite Generator
Super Monitor Dissasembler

QJUMP
25 KING STREET
RAMPTON
CAMBRIDGE CB4 4QD

QJUMP fue fundada por Tony Tebby, creador del sistema operativo del QL. Entre al software desarrollado por esta empresa se encuentra el QL TOOLKIT, extraordinario programa completo al sistema operativo. También comercializan los siguientes productos:

QL Toolkit II (ROM)
QL Monitor Debugger
Eprom Programmer
QFLP Upgrade ROM

QJUMP ha escrito el firmware de la mayoría de los controladores de floppy para QL. El firmware de Tony Tebby permite leer, escribir y formatear discos en sectores de 128 bytes a 1024 bytes, lo que cubre la mayor parte de los formatos utilizados. Puede acceder al disco como si fuese un sólo fichero de éste, lo que permite el lujo de tener un floppy con discos formato PC DOS, otro QDOS y otro UNIX, por ejemplo. Y con rutinas adecuadas se podría trabajar con los tres formatos a la vez. Con las rutinas adecuadas, el QL podría ser una solución a la conversión de formatos.

CENTRONICS

Este fabricante de impresoras, que ha dado su nombre al interface paralelo para la conexión de estas, ha lanzado una impresora con el juego de caracteres del QL, terminando con los problemas de *translate*.

COMPUTER ONE
SCIENCE PARK
MILTON ROAD
CAMBRIDGE

Computer One es una empresa muy conocida en el mundo del QL. Dispone de los siguientes programas:

QL Typing Tutor
QL Assembler
QL Forth
QL Pascal

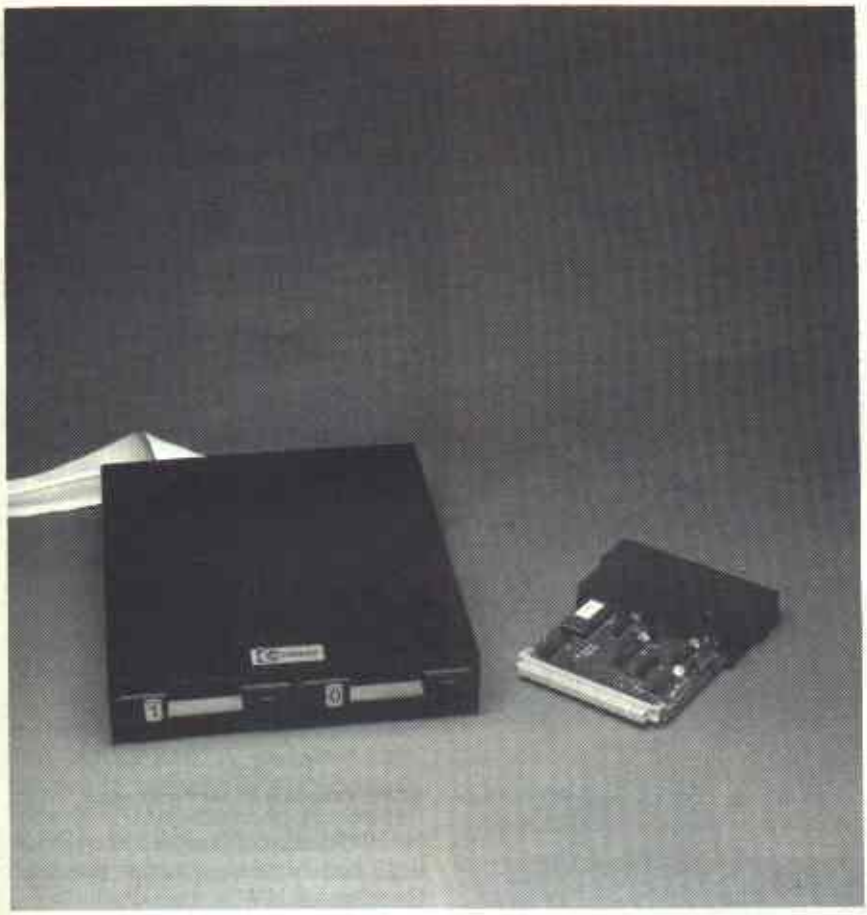
El Pascal es inferior al de METACOMCO, pero a cambio deja más memoria libre para el usuario.

GST COMPUTER SYSTEM LTD.
91 HIGH STREET
LONG STANTON
CAMBRIDGE

De GST destacan el ASSEMBLER y el LINKER, programas adoptados oficialmente por Sinclair.

QUEST INTERNATIONAL LTD.
TEL.: 04215 66488

Destaca la implementación del sistema operativo CPM/68K, origi-



Unidad de discos y controlador de Cumana.

nal de DIGITAL RESEARCH, disponible desde septiembre de 1984. También distribuye paquetes de aplicaciones y juegos, tanto para CPM como para QDOS.

En cuanto a hardware, cuenta con ampliaciones de memoria de 64 a 512 K, unidades de disco de 5"1/4 y un disco Winchester de 7,5 MB.

MICRO APL.
TEL.: 01 622 0395

Esta firma, especializada en intérpretes de APL, distribuye al APL del QL. Este potente lenguaje estaba disponible hasta ahora únicamente para grandes ordenadores.

TANDATA MARKETING LTD.
ALBERROAD NORTH
MALVERN
WORCS WR14 2TL

Distribuye el QCOM, interface de comunicaciones, el QCALL, módulo de llamada, marcado y respuesta automática y el QMOD, modem. Juntos constituyen un sistema completo de comunicaciones para el QL, permitiendo la conexión a bases de datos de grandes ordenadores.

DATA MANAGEMENT
CLARK HOUSE
THE VILLAGE
HAXBY
YORK Y03 8HU

Otra firma que dedica sus esfuerzos al terreno de las comunicaciones con emuladores múltiples de terminal, software de comunicaciones y modems.

MICROPERIPHERAL LTD.
TEL.: (0256) 473232

Fabricante de los discos oficiales de Sinclair. Estos discos han sido criticados por tener un firmware inferior al de otras unidades.

COMPUTAMATE
TEL.: 0768 811711

Una de las primeras casas en fabricar discos de 3,5 y 51/4 para el QL. El software es de QJUMP. También realiza consolas de expansión.

Adaptador serie-Centronins de Miracle Systems.

PCML
ROYAL MILLS
ESHER, SURREY KT10, 8AS

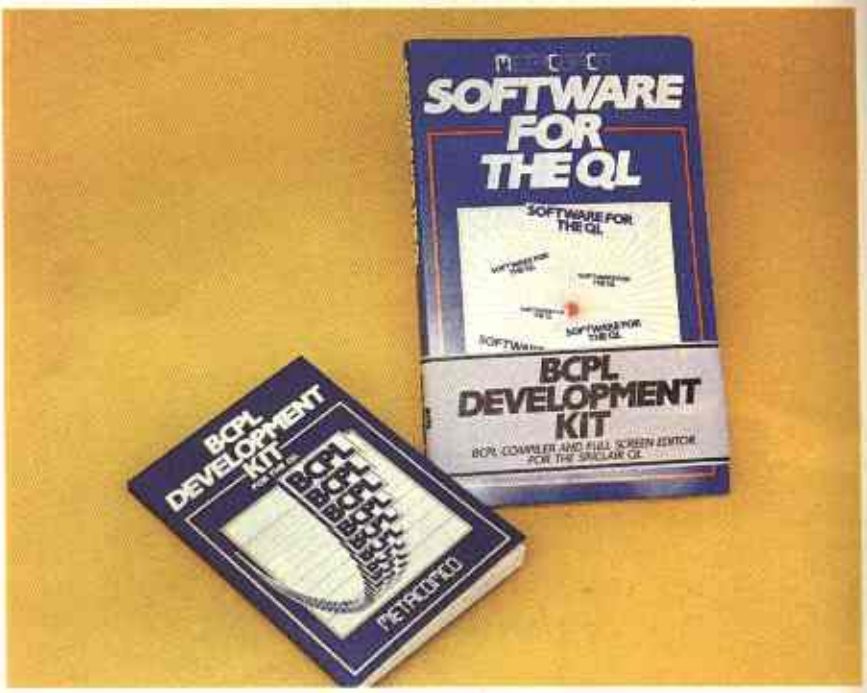
Dispone de expansiones de memoria, interfaces de floppy con RAM adiccional y de una tarjeta con un procesador Z-80, lo que permite la posibilidad de utilizar CPM 80.

TDI, LTD.
BRISTOL

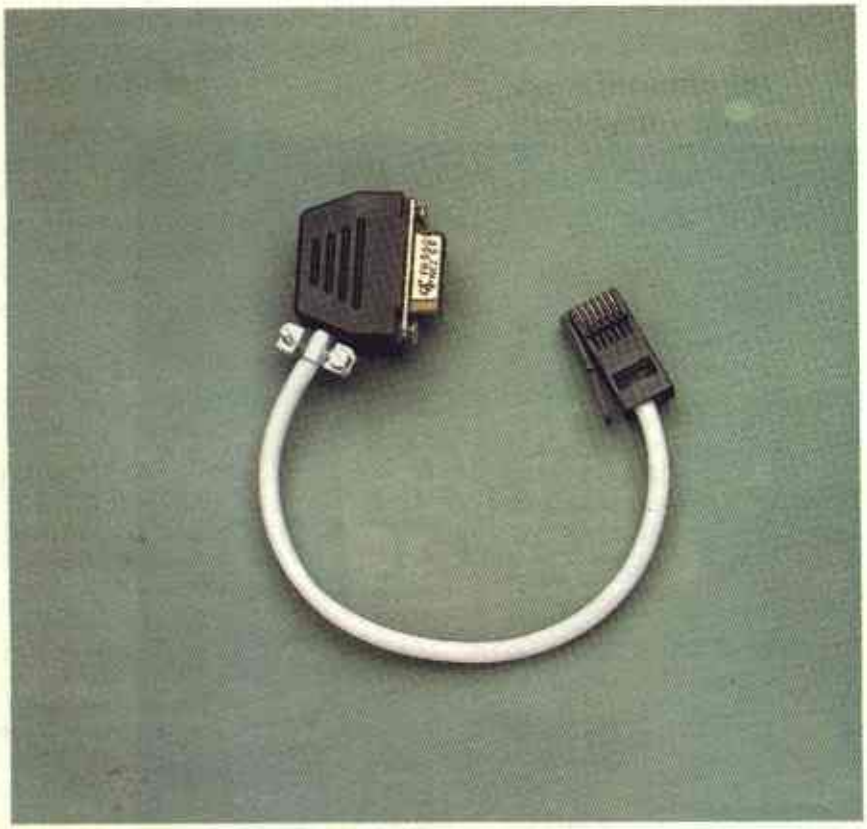
TDI ha desarrollado el sistema operativo P-SYSTEM o UCSD-PL, creado en la Universidad de California –San Diego. Para este sistema se han lanzado inicialmente los populares UCSD-PASCAL y FORTRAN-ANSI 77, así como diversas aplicaciones.

MIRACLE SYSTEMS LIMITED
UNIT 37a, AVONDALED WORKSHOPS
WOODLAND WAY, KINGSWOOD
BRISTOL BS15 1QL

Produce ampliaciones de memoria de 256 K y 512 K, con un conector que permite la conexión simultánea del disco. También comercializa convertidores serie paralelo Centronics.



BCPL, desarrollado por Metacomco.



Cable adaptador de joysticks, de Miracle System.

CAMBRIDGE SYSTEMS TECHNOLOGY (CST)
30 REGENT STREET
CAMBRIDGE CB2 1DB

Entre sus productos destaca el superprofesional interface IEEE-488, creada por el Instituto de Ingenieros Eléctricos y Electrónicos y adoptado como standard mundial por la Comisión Electrototécnica Internacional (IEC).

METACOMCO
26 PORTLAND SQUARE
BRISTOL BS2 8RZ

Junto con Computer One, fue una de las primeras empresas en producir software para el QL.
Cuenta con los siguientes programas:

Assembler
BCPL
LISP
Pascal-ISO
Lattice-C

JOSE M. GUZMAN

LOS BUGS DEL SISTEMA OPERATIVO

AUNQUE el QL tiene un número muy pequeño de bugs, es conveniente conocerles, para saber evitarlos.

En primer lugar vamos a citar los bugs comprobados en la versión española, que son tres únicamente, y dos de ellos en el WHEN ERRor.

1 La instrucción GOSUB dentro de un bucle corto FOR o REPeat actúa con fin del bucle.

Forma de evitarlo: utilizar la forma larga, o utilizar las más racionales procedures.

2 Hay un indicador de error que cuelga el QL.

Forma de evitarlo: no utilizar indicadores, es mejor utilizar el ERNUM y una variable. Ejemplo INDIC=ABS(ERNUM) y luego if o select con la variable INDIC. Los valores de INDIC están en el manual en ERRORES, la función ERNUM los da en negativo, de ahí el ABS.

3 Cuando está el WHEN ERRor activado, y al llamar a una función con una expresión en su argumento se produce el error de desbordamiento, al calcular ésta se cuelga el QL.

Forma de evitarlo: utilizar una variable intermedia de forma que no se llame a las funciones con expresiones dentro del paréntesis.

d=sqrt(3*X/12−10) puede colgar el QL con WHEN ERRor.

d=3*x/12−10:d=sqrt(d) no lo cuelga nunca.

Creo que por esos dos pequeños bugs, no debemos de abandonar el uso del estupendo WHEN ERRor.

Hay detalles que no son bugs, sino actuaciones racionales de los diseñadores del QL:

El mensaje de "error en expresión" o de "nombre incorrecto" ocurre siempre que se encuentra con una variable no inicializada, procedures o funciones no definidas, etc. Por cierto, el ANSI Basic exige este mensaje de error para variables no inicializadas.

No admite el dimensionamiento implícito, al igual que el ANSI BASIC, hay que definir siempre las dimensiones de las matrices.

En un WHEN ERRor no se admiten errores, para evitar bucles de autotratamiento de errores, y pone el mensaje de:
"durante una ejecución WHEN"
para volver a la normalidad pulsar directo desde el teclado END WHEN.

El «break» no es atrapado por WHEN ERRor, para facilitar la corrección de programas que lo utilicen.

Debido a que se trazan por «stipples» bloques de cuatro puntos, al trazar puntos traza dos en vez de uno, es fácil definirse una función punto con BLOCK, o solicitar la rutina de Investrónica para un solo punto.

En procedures y subrutinas no se admite MRUN y MERGE para evitar que se pueda desordenar la pila de retornos de subrutinas.

RENUM es peligroso desde un programa, puede liar al intérprete si cambia el número de línea en que se solicita, también DLINE es peligroso desde programa, si borra la linea que se está ejecutando, los dos no se admiten desde procedures.

Hay una combinación extraña de teclas que «cuelga» al QL sino se tiene un software adecuado, es debido a que genera una interrupción nivel 7. Sirve pra que los programadores puedan intentar levantar «cuelgues» de programas en código máquina. Programas monitores especiales aprovechan esta *fantástica facilidad del QDOS.*

Para evitar lios, la radiodifusión

de red local funciona sin comprobación, sólo para cortos mensajes, debe de utilizarse sólo como canal de órdenes, y disponer de la propia comprobación por programa de los usuarios. Viene documentado en el Manual de usuario. Máxima longitud de mensaje en Radiodifusión por red local 255 bytes. Que nadie hable de Bugs si falla en mensajes largos. Para mensajes largos utilizad la red normal tal y como indica el manual.

Por último, antes de hablar de un nuevo bug, consultar, es muy fácil que sea error de programación.

Veamos ahora la «TERRIBLE» lista de bugs, algunos muy oscuros, de las versiones anteriores. No existen en la versión española, excepto algunos no comprobados. Aquellos que tengan QLs antiguos, que cambien a las nuevas ROMS.

1 Una muy molesta para los programadores en C.M.: el QL se colgaba al utilizar CALL desde un programa de SuperBasic que ocupase más de 32K. No se ha comprobado, pero Sinclair daba una rutina que lo corregía. Habitualmente se cargan extensiones en C.M. en el programa BOOT, que es corto. No es bueno cargar en el programa principal las extensiones, si se para y se vuelve a lanzar, va rellenando cada vez la memoria de rutinas en C.M. y se puede forzar un fuera de la memoria por mala programación.

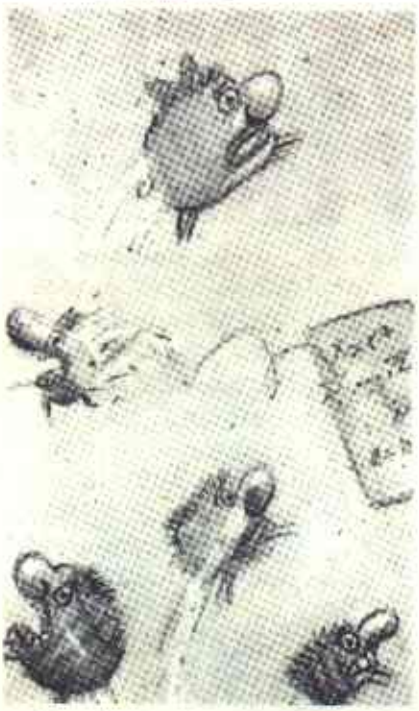
2 No reconocía más que una ROM de periféricos. Aunque no se ha comprobado experimentalmente, no hay errores en las rutinas de chequeo de ROM de periféricos, situadas entre $4A62 y $4AC2. Sinclair también proporcionaba una rutina de corrección.

3 List desde un programa daba "no implementado".

4 Si con DLINE se borra la última procedure de un programa, se llama a la procedure desde el teclado, y después se hace clear, se colgaba el QL.

5 Si se coloca una expresión entre paréntesis en los data, ignoraba el resto de la línea.

El QL admite expresiones en los DATA, y si se colocan las constantes alfanuméricas sin comillas o



apóstrofes da el correcto "error en expresión" al no encontrar la variable.

6 dir mdv8__ y después dir mdv2__ no funcionaba bien.

7 Al editar una procedure abortada, a veces editaba una línea cualquiera. Se corregía con un simple list u otra cosa y después editar.

El mensaje "PROC/FN limpiado" es correcto, siempre que se edite, o se dé un comando directo distinto de CONTINUE O RETRY, y el programa se haya parado en medio de una procedure o función, si no se hiciese así, se desordenaría la pila de retornos.

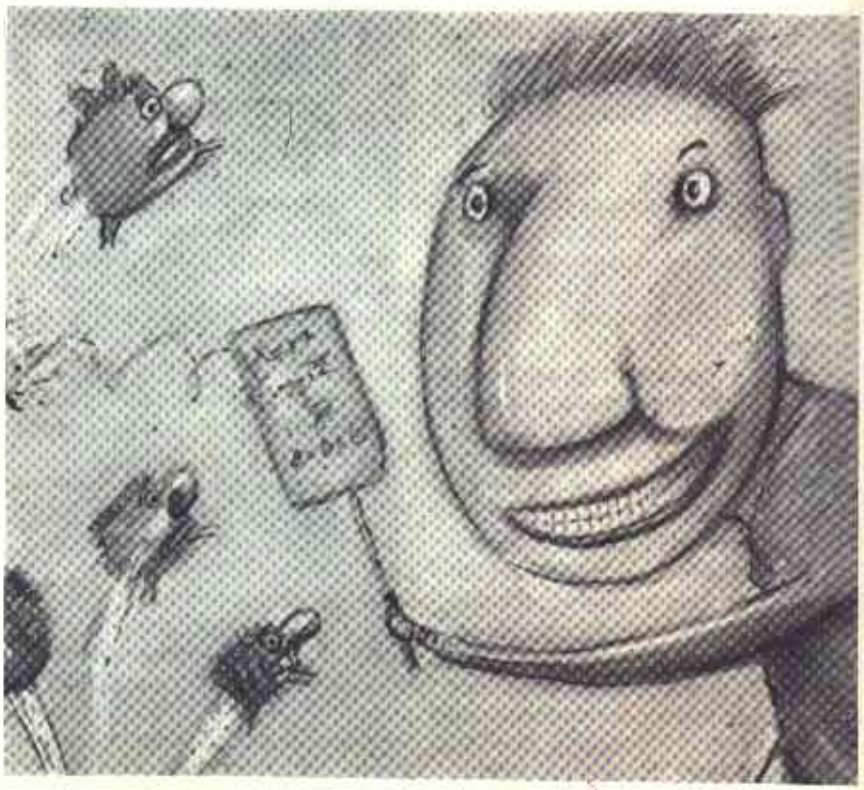
8 Cuando se declaraban más de 9 nombres de variables locales, originaban problemas.

9 No era conveniente hacer troceado de matrices de cadenas, era mejor utilizar una variable intermedia.

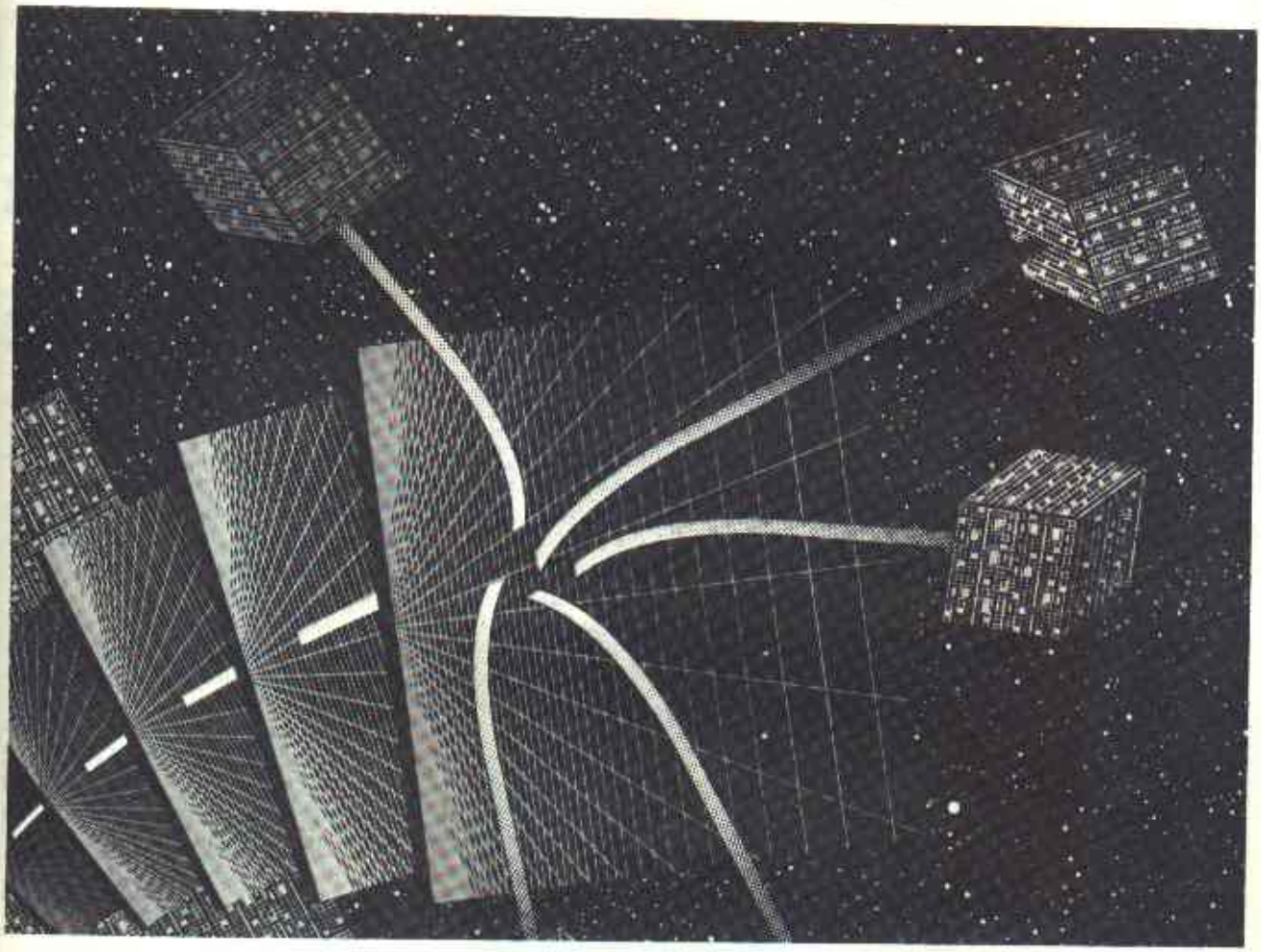
10 En la ROM JS, de todos los argumentos de la función, sólo admitía el SELECT con el último.

Además hay que destacar que los bugs son en su mayoría del intérprete y no del QDOS, por cierto sería genial una rutina que aprovechase la interrupción de nivel 7, para «levantar» los cuelgues del intérprete. Aunque lo bueno es compilar.

José M. Guzmán



SISTEMAS OPERATIVOS MULTITAREA



DESDE el lanzamiento del QL se habla mucho de la multitarea y del mítico UNIX. Vamos a explicar un poco la historia de los sistemas operativos multitarea.

Cuando la Informática estaba empezando, y los equipos eran muy costosos, una máquina con las posibilidades de un Spectrum costaba millones de pesetas y a nadie se le ocurría poder tenerla para él solo.

El sistema que se utilizaba para aprovechar los costosos equipos era el *batch*, o procesado por lotes. Los usuarios del ordenador depositaban sus programas y datos en el Centro de Cálculo, los empleados de centro introducían los programas y datos y recogían los resultados, luego el usuario recibía los resultados.

Naturalmente el menor error del programa y de los datos, obligaba a repetir el proceso, con la consiguiente pérdida de tiempo. El desarrollo de programas era laboriosísimo.

Como era económicamente imposible el que un usuario pudiese tener un ordenador para él sólo, en las universidades los programadores, cansados de entregar un programa, recogerlo al día siguiente con errores, volverlo a entregar, etc., tuvieron la genial idea de que, dado que el ordenador es mucho más rápido que el usuario, si se le hacía dedicar una pequeña fracción de su tiempo a cada usuario, a este le parecería tenerlo sólo para él, pero como el ordenador es rápido, podría atender a la vez a muchos usuarios.

Esta genial idea es el sistema de multitarea por partición de tiempo, «time-slicing», el más utilizado hoy en día.

Naturalmente, desde entonces han progresado los sistemas operativos multitarea, añadiéndose más funciones, pero la básica es ésta, la partición del tiempo, y vamos a dar una idea de cómo se realiza.

El corazón de la multitarea es el repartidos de tiempos de ejecución, el *sheduler*, es una rutina que se llama por las interrupciones hardware cada cierto tiempo, y decide qué programa va a ser ejecutado, salvando los valores necesarios para poder volver a comenzar el progrma anterior, y colocando los nuevos del pro-

grama a ejecutar.

Para aquellos que quieran examinar con un desensamblador el *scheduler* del QL, ofrecemos una somera indicación de las direcciones (hexadecimales) de las rutinas más importantes:

— $352 entrada de todo tipo de interrupciones nivel 2, comprueba si son de *microdrives*, disco, etc.

— $90A entrada de interrupciones de temporización y *scheduler*.

— $936 Comprobación de tareas de altísima prioridad (modo sistema). Si se está en ellas no se ejecuta el *scheduler*.

— $940 a $966 el *scheduler* propiamente dicho.

— $9DE a $A14 la rutina que salva todos los valores y datos necesarios para volver a reanudar la ejecución del programa suspendido.

— $A16 a $A80 la rutina que elige el programa que le corresponde ejecutarse.

— $A82 a $AA6 la rutina que coloca los valores necesarios para reanudar la ejecución de la tarea.

— $AA8 a $AC8 una interesante rutina de llamada a las listas encadenadas de tratamiento de interrupciones.

Las direcciones indicadas son las de la ROM española MGE, en otras ROM pueden cambiar.

El *scheduler* es el corazón del sistema operativo multitarea, pero un sistema multitarea necesita mucho más que eso. El hecho de que estén ejecutándose varios programas a la vez puede dar lugar a muchos problemas, por ejemplo, tomado del libro de Dickens, es fácil imaginarse un listado en que varios programas escribiesen por impresora a la vez.

El sistema operativo gestiona la utilización de los recursos del ordenador, evitando, en lo posible, que unos programas «machaquen» a otros, controla la utilización de ficheros, *interfaces*, etc. También, desde el Multics, se incorporan sistemas de comunicación intertareas, llamados *pipes*, para facilitar el uso del equipo. Los *pipes* están disponibles en el QDOS, pero no en SuperBASIC. Hay rutinas de Qunata que permiten utilizar las *pipes* desde el SuperBASIC.

Por otro lado, en los modernos sistemas multitarea, cuando un programa está esperando la realización de una operación de entrada/salida o el envío de la comunicación de un dato por otro programa, este programa se detiene para evitar que malgaste el tiempo del procesador.

Como vemos son muy complejas y variadas las labores que debe realizar un sistema operativo multitarea, por lo que se suelen implementar en ordenadores potentes. Además, al ser complejos, su desarrollo es costoso, por lo que normalmente sólo se utilizan máquinas caras. Pero tio Clive nos puso, como siempre, lo inasequible a precios razonables.

Naturalmente sistemas operativos multitarea hay muchos, empezando por el mítico UNIX y sus variantes, siguiendo con el PICK, OASIS, los Concurrent CPM 86, 286 y 68K, el Tripos del Amiga, el 68KOS y nuestro QDOS.

Y si hablamos de miniordenadores y grandes Ordenadores *(mainframes)*, hoy en día no se concibe un sistema operativo para ellos sin ser multitarea, simplemente por lo menos para el *spool* de impresora (esto es imprimir en ficheros de disco, y un programa en multitarea se encarga de imprimir mientras el ordenador sigue trabajando). Realmente sería absurdo tener estos equipos parados esperandod el acabar de imprimir.

Hay muchas variantes de sistemas operativos multitarea, de tiempo real, multiusuario, etc. Pero vamos a ir empezando poco a poco.

Para aquellos que quieran introducirse más a fondo en el QDOS, el libro recomendado es la **Guía del Usuario Avanzado del QL** por **Adrian Dickens**, publicado por **RAMA.**

José M. Guzmán