12

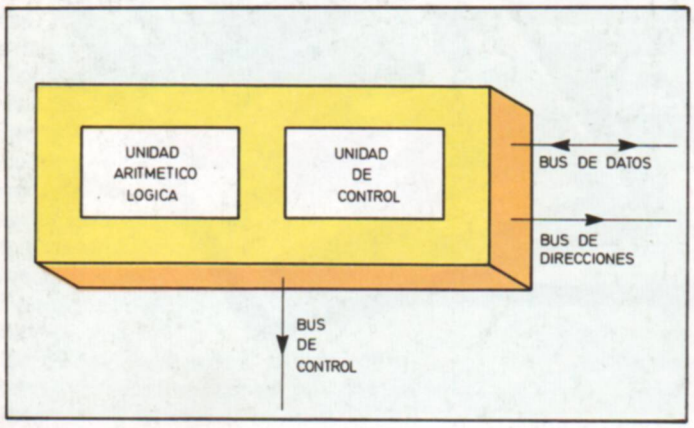
Enciclopedia Práctica de la

INFORMATICA

FUNCIONAMIENTO DEL MICROPROCESADOR

HARDWARE: **EPSON QX-10 / EL LENGUAJE FORTRAN**

EL GOBIERNO DE LA INFORMATICA



Nueva Lente / Ingelek

125 ptas.

FUNCIONAMIENTO DEL MICROPROCESADOR

PARA aclarar el funcionamiento de un microprocesador vamos a definir en este artículo un sistema experimental en el que se estudiará la distribución de la información, tanto de los datos como de las instrucciones.

Sistema experimental basado en microprocesador

El sistema que vamos a describir constará de cuatro unidades básicas: un periférico de entrada de datos, el microprocesador, una unidad de memoria principal y un periférico de salida de datos. Para estudiar el método operativo las unidades de E/S son irrelevantes, por lo que simplemente consideraremos su existencia sin precisar sus características. En cambio, el microprocesador y la memoria principal funcionarán asociadas, de forma que aquél ejecute los programas y utilice los datos almacenados en ésta. Las características de estas dos unidades, en nuestro sistema experimental, serán las siguientes:

• El microprocesador estará compuesto por las tres zonas básicas ya conocidas: la unidad de control, la unidad aritmético-lógica y un acumulador. La unidad de control se encargará de gobernar y sincronizar todas las operaciones necesarias para la ejecución de los programas. Las instrucciones estarán formadas por un código de operación y la dirección de un operando, con el que se realizará el tratamiento indicado por el código de operación. Una vez interpretada la instrucción, la unidad aritmético-lógica se encargará de ejecutarla sobre el registro acumulador. En este sistema tan sólo consideraremos un pequeño subconjunto de posibles instrucciones:

1. Cargar en el acumulador un dato almacenado en memoria.

Código	Operando
CAR	dirección de memoria en la que se encuentra el dato

2. Almacenar el contenido del acumulador en una posición de memoria.

Código	Operando
ALM	dirección de memoria en que se almacenará el contenido del acumulador

3. Sumar al contenido del acumulador un dato almacenado en memoria.

Código	Operando
SUM	dirección de memoria en donde se encuentra el número a sumar

4. Restar del contenido del acumulador un dato almacenado en memoria.

Código	Operando
RES	dirección de memoria en donde se encuentra el número a restar

5. Multiplicar el contenido del acumulador por un dato almacenado en memoria.

Código	Operando
MUL	dirección de memoria en que se encuentra el número por el que se multiplicará el contenido del acumulador

6. Dividir el contenido del acumulador entre un dato almacenado en memoria.

Código	Operando
DIV	dirección de memoria en que se encuentra el número por el que será dividido el contenido del acumulador

7. Elevar el contenido del acumulador a un dato almacenado en memoria.

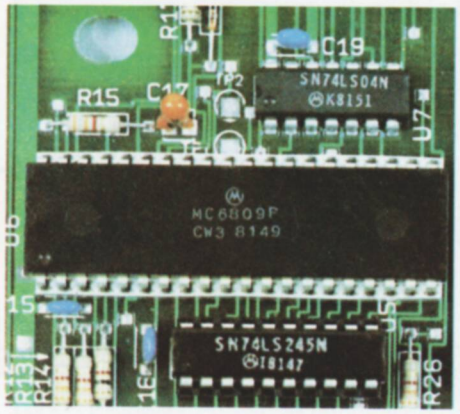
Código	Operando
ELE	dirección de memoria en que se encuentra el exponente al que será elevado el contenido del acumulador

Esta instrucción sirve también para la extracción de raíces del contenido del acumulador, ya que:

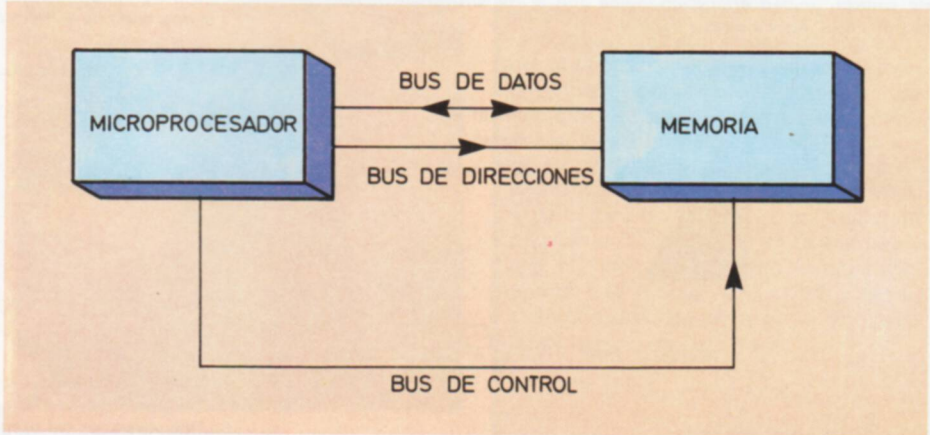
$$\sqrt[n]{\text{ACUMULADOR}} = \text{ACUMULADOR}^{1/n}$$

8. Salto incondicional al emplazamiento de una nueva instrucción.

Código	Operando
SAL	dirección de memoria en donde se encuentra la siguiente instrucción



El microprocesador, en este caso un 6809 de la firma Motorola que desempeña la función de CPU en un THOMSON TO-7, está compuesto por tres zonas básicas: la unidad de control, la unidad aritmético-lógica y el acumulador.



Radiografía del sistema experimental basado en microprocesador que se utiliza en este artículo para describir el funcionamiento de un microprocesador convencional.

FUNCIONAMIENTO DEL MICROPROCESADOR

9. Salto condicional, si el contenido del acumulador es distinto de cero no se ejecutará la próxima instrucción y seguirá la secuencia a partir de la siguiente; si el contenido del acumulador es cero el programa se ejecutará normalmente.

Código	Operando
CON	—

10. Lectura de un dato a través del periférico de entrada y carga en memoria.

Código	Operando
LEE	dirección de memoria en que se almacenará el dato leído

11. Escritura de un dato a través del periférico de salida.

Código	Operando
ESC	dirección de memoria en que se encuentra almacenado el dato a escribir

12. Fin del programa.

Código	Operando
FIN	—

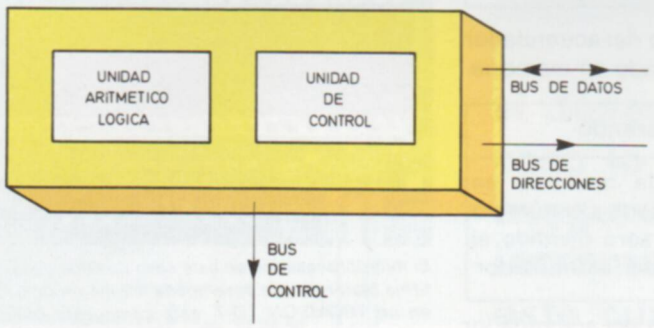
Todas las operaciones aritméticas se realizan «sobre» el acumulador, es decir, el contenido inicial del acumulador es el primer operando y el contenido final es el resultado de la operación.

- La memoria de nuestro sistema será de tipo RAM y se encargará de almacenar programas y datos. La comunicación se realizará a través de los buses de direcciones, de datos y de control ya estudiados.

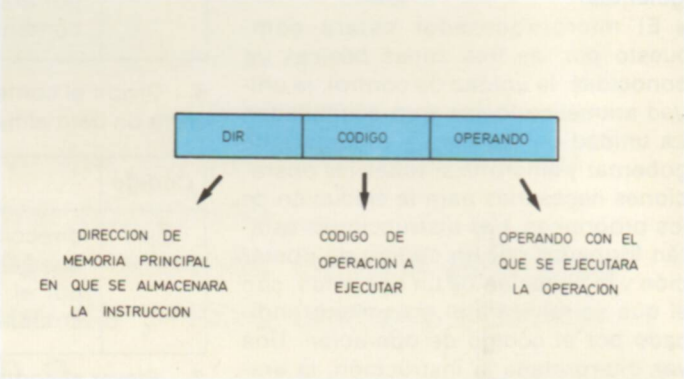
Es importante destacar que este sistema experimental no corresponde totalmente a la realidad, si bien es bastante parecido en cuanto al juego de instrucciones y al método de funcionamiento. También debe quedar claro que todas las características del microprocesador, que ya vimos en el capítulo anterior, jugarán un importante papel en el rendimiento global del sistema, pero no alterarán la forma de operar del mismo.

Funcionamiento del sistema

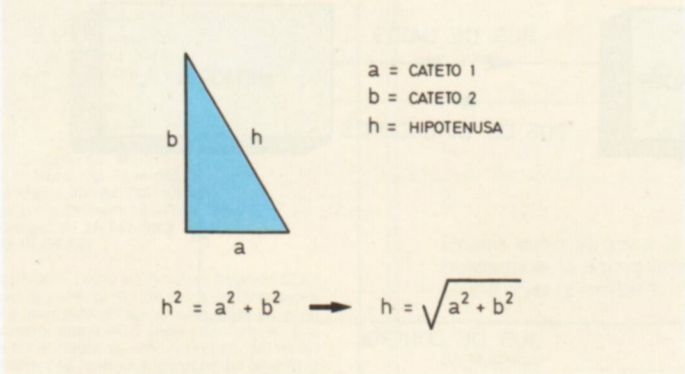
Realizaremos una demostración del funcionamiento del sistema mediante la ejecución de un programa que calcula hipotenusas de triángulos rectángulos en función de los catetos.
El lenguaje que estamos utilizando en esta simulación es de tipo máquina, con la salvedad de que hemos sustituido los códigos de operación por símbolos nemotécnicos y trabajamos en decimal. Un programa que realice el cálculo de la hipotenusa de un triángulo rectángulo a partir de los dos catetos podría ser el siguiente:



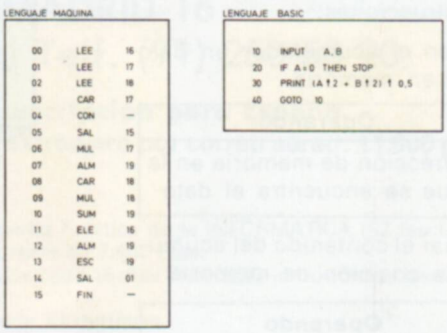
El gráfico muestra la estructura interna de nuestro microprocesador experimental. Consta de: unidad de control, unidad aritmético-lógica con el registro acumulador, bus de datos, direcciones y control.



Formato general de las instrucciones del microprocesador experimental. La casilla de la izquierda refleja la dirección de la memoria principal, la central el código de operación y la tercera el operando de la instrucción.



El programa ejemplo aplica el teorema de Pitágoras para encontrar el valor de la hipotenusa de un triángulo rectángulo, conocidos los valores de los catetos.



Dos versiones del programa ejemplo: escrito en lenguaje máquina (izquierda) y en lenguaje BASIC (derecha). La comodidad que supone para el programador el empleo de un lenguaje de alto nivel se hace patente al observar ambos listados.

Dirección de la instrucción	Código de operación	Operando
00	LEE	16
01	LEE	17
02	LEE	18
03	CAR	17
04	CON	—
05	SAL	15
06	MUL	17
07	ALM	19
08	CAR	18
09	MUL	18
10	SUM	19
11	ELE	16
12	ALM	19
13	ESC	19
14	SAL	01
15	FIN	—

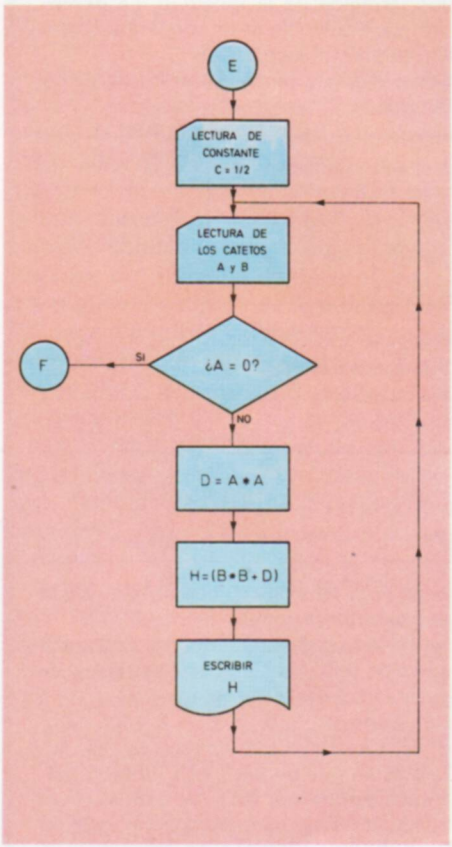
El número de instrucciones necesarias para programar el tratamiento adecuado de los datos es 16, por tanto, se ocuparán 16 palabras (desde la 0 hasta la 15) de la memoria principal para almacenar el programa. Se han elegido las posiciones 16 a 19 de memoria para los datos utilizados por el programa, aunque se podía haber tomado otra zona de memoria no consecutiva al programa.

El funcionamiento del microprocesador, al ejecutar el anterior programa será el siguiente:

— La primera instrucción tratada será la | 00 | LEE | 16 | , con lo que la unidad de control leerá un dato a través del periférico y lo cargará en la posición 16 de la memoria principal. Este dato debe ser una constante de valor 0,5 y será utilizada para extraer las raíces cuadradas.

— A continuación, siguiendo la secuencia del programa, se ejecuta | 01 | LEE | 17 | y en ese momento la unidad de control admitirá por el periférico de entrada el valor del primer cateto y lo cargará en la posición de memoria 17.

— Por medio de la instrucción | 02 | LEE | 18 | se realiza la misma



La figura representa el algoritmo utilizado para resolver el problema planteado: cálculo de la hipotenusa de un triángulo rectángulo.

Aunque resulte sorprendente, los modernos y potentes microordenadores no son más que una aplicación directa del microprocesador.

Glosario

¿Existe alguna diferencia en la forma de representar una instrucción y un dato?

En principio no. Cualquier byte de la memoria principal puede representar indistintamente una instrucción o un dato; será el programador, a través del programa, el que determine su distribución.

¿Qué parte del microprocesador se encarga de ejecutar las instrucciones?

La unidad de control examina el código de operación de la instrucción a ejecutar y según sea éste, realiza ella misma la operación deseada o transmite la ejecución a la unidad aritmético-lógica.

En la instrucción de bifurcación o salto condicional ¿cuándo se ejecuta la siguiente instrucción y cuándo se salta?

Según el valor que contenga el acumulador se saltará o no la siguiente instrucción. En el juego de instrucciones definido para el sistema experimental, la próxima instrucción se ejecuta cuando el contenido del acumulador es distinto de cero y se salta en caso contrario. No obstante, en algunos sistemas el proceso es al contrario.

¿De qué forma controla el programador los datos cargados en la memoria principal?

Al emplear lenguajes de máquina, el programador tiene que saber en todo momento la dirección de memoria en la que se almacena el dato; sin embargo, al utilizar lenguajes simbólicos o de alto nivel, se limita a darles un nombre nemotécnico y el propio sistema decide la dirección. Por ejemplo, en vez de escribir la instrucción | 02 | LEE | 18 | podrá hacerlo, por tanto, en la forma: | 02 | LEE | CATETO 2 | ; en cualquier otra parte del programa en la que necesite el valor del segundo cateto, sólo tendrá que escribir CATETO 2 y el microprocesador buscará en la dirección apropiada.

FUNCIONAMIENTO DEL MICROPROCESADOR

operación con el segundo cateto, que quedará cargado en la posición de memoria 18.

— La instrucción | 03 | CAR | 17 | se limita a cargar el contenido de la posición 17 de la memoria (primer cateto) en el acumulador.

— Una vez realizada la carga anterior el programa ejecutará la instrucción de salto condicional | 04 | CON | - | , si el contenido del acumulador es cero se ejecutará la siguiente instrucción | 05 | SAL | 15 | y si es distinto de cero no será ejecutada continuando directamente en la siguiente instrucción | 06 | MUL | 17 | .

Como el contenido del acumulador, al ejecutar esta instrucción, coincide con el primer cateto, el programa sólo «saltará» a la instrucción | 15 | FIN | — | cuando éste sea cero, es decir, el programa calculará tantas hipotenusas como pares de catetos se le suministren y sólo terminará cuando se le proporcione una pareja de catetos de valor cero.

— Si la anterior instrucción no ha provocado la conclusión del programa, al ejecutar | 06 | MUL | 17 | la unidad aritmético-lógica multiplica el contenido del acumulador (primer cateto) por el contenido de la posición de memoria 17 (primer cateto), almacenando el resultado (primer cateto al cuadro) en el acumulador.

— La siguiente instrucción | 07 | ALM | 19 | llevará el contenido del acumulador a la posición de memoria 19. El hecho de almacenar el primer cateto al cuadrado en una posición intermedia de memoria es debido a que será utilizado posteriormente.

— A continuación se ejecuta la instrucción | 08 | CAR | 18 | con la que se cargará la posición de memoria 18 (segundo cateto) en el acumulador. Si en la instrucción anterior no se hubiera almacenado el valor del primer cateto al cuadrado en una posición intermedia, ahora se habría perdido este dato.

— Mediante la instrucción | 09 | MUL | 18 | la unidad aritmético-lógica multiplicará el contenido de la posición de memoria 18 (segundo cateto) por el contenido del acumulador (segundo cateto) y almacenará el resultado (segundo cateto al cuadrado) en el acumulador.

— La instrucción | 10 | SUM | 19 | obtendrá la suma de la posición 19 de la memoria principal (primer cateto al cuadrado) y el contenido del acumulador (segundo cateto al cuadrado) y almacenará el resultado (suma de los cuadrados de los catetos) en el acumulador.

— Por último, la instrucción | 11 | ELE | 16 | eleva el contenido del acumulador al número contenido en la posición 16 de la memoria principal. Como dicho número es 1/2 = 0,5 (cargado mediante la instrucción | 00 | LEE | 16 |) el resultado obtenido es la raíz cuadrada de la suma de los cuadrados de los catetos, con lo que ya tenemos el resultado definitivo cargado en el acumulador.

— Para poder escribir el valor de la hipotenusa a través del periférico de salida, primero es necesario almacenar dicho valor en una posición de memoria, esto se consigue con la instrucción | 12 | ALM | 19 | que almacenará el contenido del acumulador en la posición 19 de la memoria principal.

— A continuación se ejecuta la instrucción | 13 | ESC | 19 | con lo que se obtendrá el listado (si el periférico de salida es una impresora) del valor de la hipotenusa.

— La instrucción | 14 | SAL | 01 | origina un salto incondicional a la instrucción | 01 | LEE | 17 | , con lo que se repetirá todo el proceso anteriormente descrito.

— Si en la instrucción | 05 | SAL | 15 | se produjo el salto a la instrucción | 15 | FIN | — | , al ser ejecutada ésta se dará por terminado el trabajo, con lo que el microprocesador quedará libre.

Conceptos básicos

Nociones de álgebra de Boole (II)

Propiedades de un sistema booleano y funciones booleanas

Las propiedades fundamentales de los sistemas booleanos son las siguientes:

1. Ley idempotente

$\forall A \in S : A \vee A = A$ y $A \wedge A = A$

2. Ley de involución

$\forall A \in S : \bar{\bar{A}} = A$ (esto es: el complementario del complementario de una variable lógica es la propia variable).

3. Ley conmutativa.

$\forall A, B \in S : A \vee B = B \vee A$

y $A \wedge B = B \wedge A$

4. Ley asociativa.

$\forall A, B, C \in S : A \vee (B \vee C) = (A \vee B) \vee C = A \vee B \vee C$ y

$A \wedge (B \wedge C) = (A \wedge B) \wedge C = A \wedge B \wedge C$

5. Ley distributiva.

$\forall A,B,C \in S : A \wedge (B \vee C) = (A \wedge B) \vee (A \wedge C)$

y $A \vee (B \wedge C) = (A \vee B) \wedge (A \vee C)$

6. Ley de absorción

$\forall A,B \in S : A \wedge (B \vee A) = A$

y $A \vee (B \wedge A) = A$

7. Ley de Morgan.

$\forall A,B \in S : \overline{A \vee B} = \bar{A} \wedge \bar{B}$

y $\overline{A \wedge B} = \bar{A} \vee \bar{B}$

El sistema utilizado para demostrar las anteriores proposiciones, y en general cualquier expresión booleana, es siempre el mismo. Se construye una tabla con todas las combinaciones de los posibles valores de las variables que intervengan en la expresión; en cada combinación se calcula el valor correspondiente para los dos términos de la igualdad. La expresión será correcta si en todos los casos los valores coinciden.

Como ejemplo demostraremos la ley de Morgan.

A	B	$\bar{A}$	$\bar{B}$	$A \vee B$	$\overline{A \vee B}$		$\bar{A} \wedge \bar{B}$
0	0	1	1	0	1	=	1
0	1	1	0	1	0	=	0
1	0	0	1	1	0	=	0
1	1	0	0	1	0	=	0

Luego $\overline{A \vee B} = \bar{A} \wedge \bar{B}$

y análogamente:

A	B	$\bar{A}$	$\bar{B}$	$A \wedge B$	$\overline{A \wedge B}$		$\bar{A} \vee \bar{B}$
0	0	1	1	0	1	=	1
0	1	1	0	0	1	=	1
1	0	0	1	0	1	=	1
1	1	0	0	1	0	=	0

Luego $\overline{A \wedge B} = \bar{A} \vee \bar{B}$ con lo que queda demostrada la ley de Morgan.

Se llama función booleana a toda aquella que esté definida a partir de variables lógicas sometidas a las operaciones $\vee$, $\wedge$ y $\sim$.

Por ejemplo: $f(\bar{X}, Y, Z) = X \wedge (Y \vee \bar{Z})$.

Si asignamos valores concretos a las variables en que se basa una función, podemos calcular el valor de ésta sin más que sustituir las variables por su valor en la expresión. Si, en el ejemplo anterior, X = 1 y Z = 0, tendremos que:

$f(1,1,0) = \bar{1} \wedge (1 \vee \bar{0}) = 0 \wedge (1 \vee 1) = 0 \wedge 1 = 0$.

EPSON QX-10

EL resultado de la unión del sistema operativo MultiFonts CP/M y el lenguaje MultiFonts BASIC, además de un hardware cuidadosamente diseñado, es un ordenador personal de notables características: el Epson QX-10.
El sistema operativo MultiFonts CP/M (desarrollado por Microsoft y EPSON), es una variante mejorada del CP/M 2.2 de Digital Research. El resultado es que se pueden representar, mediante la introducción de unos comandos específicos, caracteres en 8 idiomas diferentes (incluido el español), además de otras mejoras sustanciales.

Unidad central

La unidad central de proceso está constituida por el microprocesador de 8 bits Z-80A, trabajando a una frecuencia de 4 MHz. Posee 7 canales de DMA (acceso directo a memoria) y 15 niveles de interrupción. Dispone de una memoria EPROM encargada de almacenar el IPL y de 32 Kbytes de RAM de video.
La memoria RAM de usuario —en la versión básica— es de 192 Kbytes. Esta puede ser ampliada hasta 256 Kbytes mediante la colocación de 8 circuitos integrados en los zócalos dispuestos al efecto en el interior de la unidad central. Esta peculiar característica reduce considerablemente el coste de la ampliación de memoria RAM.
En el panel posterior de la unidad central, se encuentran las puertas de salida para la conexión del lápiz óptico (a través de un conector DIN), una impresora de tipo paralelo (conector de 36 patillas) y el monitor. Los interfaces de que dispone la versión básica son los siguientes: Centronics, RS/232C, interface para lápiz óptico e interface para teclado.
Además de la ampliación de memoria RAM ya comentada, el sistema dispone de cinco conectores de expansión en el interior de la caja de la unidad central, que permiten la adición de otro interface RS 232C, un interface para monitor a color, un adaptador para interface IEEE 488 y una tarjeta ROM de generación de caracteres.
El interface RS 232C, puede trabajar en modo síncrono lo que hace posible que el QX-10 se comporte como un terminal inteligente asociado a otros sistemas ordenadores. También puede operar en modo asíncrono, ocupándose de la transmisión de datos a distintos periféricos tales como: impresoras serie, acopladores acústicos, etc. El interface RS 232C de la versión básica puede operar con una velocidad de transmisión máxima de 9.600 baudíes.
Especial atención merece también la tarjeta ROM de generación de caracteres, que permite una alta definición de los mismos al representar los caracteres a partir de una matriz de 14 × 17 puntos. Esta tarjeta está controlada por un chip 8039 asociado a una memoria EPROM del tipo 2716. Debe mencionarse también que algunas de las características del MF-BASIC (MultiFonts BASIC), no se pueden utilizar sin este módulo.

Teclado

El teclado (tipo QUERTY), es independiente del mueble de la unidad central y tiene 102 teclas. Está disponible en 8 idiomas, entre ellos el español, que incluye símbolos propios tales como: la apertura de interrogación, los dos puntos de diéresis o la letra ñ.
Incorpora keypad numérico y 10 teclas de función programables por el usuario, además de otras 4 para el movimiento del cursor en todos los sentidos.

Ordenador: **EPSON QX-10.**
Fabricante: **Epson Corporation.**
Nacionalidad: **Japón.**
Distribuidor en España: **S. A., Tradetek Internacional.**

CARACTERISTICAS BASICAS

UNIDAD CENTRAL	MEMORIAS DE MASA
CPU: Microprocesador Z80-A. *RAM versión básica:* 192 Kbytes. *ROM versión básica:* 2 Kbytes. *Máxima RAM (con ampliación):* 256 Kbytes. *Accesos periféricos básicos:* Interface. Centronics para impresora, RS/232C serie, interface para lápiz óptico.	*Discos flexibles:* doble unidad de disco flexible de 5 1/4", doble cara/doble densidad, de 320 Kbytes por disco.
TECLADO	**SISTEMAS OPERATIVOS**
Versión estándar: tipo QWERTY con 102 teclas e independiente de la unidad central. Incorpora Keypad numérico, 10 teclas programables de función y teclas para el movimiento del cursor en todos los sentidos.	*Estándar:* MultiFonts CP/M. *Opcionales:* CP/M 2.2.
PANTALLA	**LENGUAJES**
Versión estándar: monocromática de fósforo verde. *Resolución gráfica:* 640 × 400 puntos. *Opciones:* monitor de ocho colores.	*Versión básica:* MF-BASIC. *Opcionales:* M/BASIC, FORTRAN, COBOL, PASCAL.

EPSON QX-10

En la parte superior derecha se encuentran 4 teclas cuya función es la selección del «estilo» o forma de impresión. Existen 16 formas distintas de representación de caracteres; algunas son: escritura clásica alemana, helvética media, americana media, comercial, clásica inglesa, etc.
El teclado se conecta a la unidad central por medio de un cable de tipo telefónico que permite un fácil y cómodo manejo del mismo.

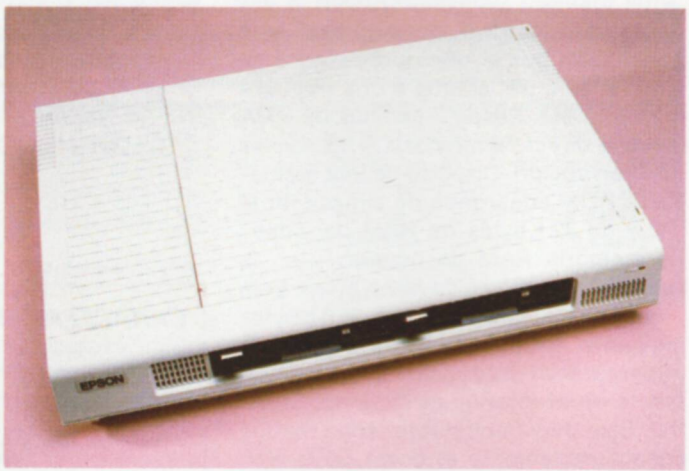
Pantalla

La versión básica del QX-10 incorpora un monitor monocromático (caracteres en verde sobre fondo negro) de 12", con una resolución horizontal de 640 puntos y vertical de 400.
Cuando el sistema trabaja en modo «no MF-BASIC», el máximo número de caracteres de un byte que pueden ser representados es 2.000 (25 líneas × 80 columnas); cada carácter se define sobre una matriz de 7 × 13 puntos.
Trabajando en modo «MF-BASIC», se pueden representar caracteres de uno o dos bytes, variando la resolución del monitor en cada caso. Así, para caracteres de un byte la resolución es de 20 líneas × 80 columnas (1.600 caracteres en total); mientras que en el segundo caso, la representación se realiza en 20 líneas de 40 columnas (800 caracteres).
El monitor, dispone de un potenciómetro situado en la zona posterior, con el que se puede ajustar el brillo.
Acoplando un monitor de color, a través del adaptador de interface correspondiente, se pueden representar gráficos en 8 colores con la misma resolución que para el monitor de B/N (640 × 400 puntos), monitor este último que también permite posibilidades gráficas.

Memorias de masa

El QX-10 está equipado con una doble unidad para discos flexibles de 5 1/4 pulgadas. Cada disco admite el alma-

El EPSON QX-10 incorpora el sistema operativo MultiFonts CP/M y el lenguaje MultiFonts BASIC, que lo convierten en un ordenador personal potente y con múltiples prestaciones.



El mueble de la unidad central sirve de soporte para la pantalla, dado su perfil bajo y la resistencia de su estructura.

El teclado que incorpora la configuración básica es del tipo QWERTY, de línea ergonómica e independiente de la unidad central.

La pantalla del EPSON QX-10 es de 12", con una resolución de 640 puntos en horizontal y 400 en vertical. El sistema está preparado para soportar un monitor en color para la presentación de gráficos.

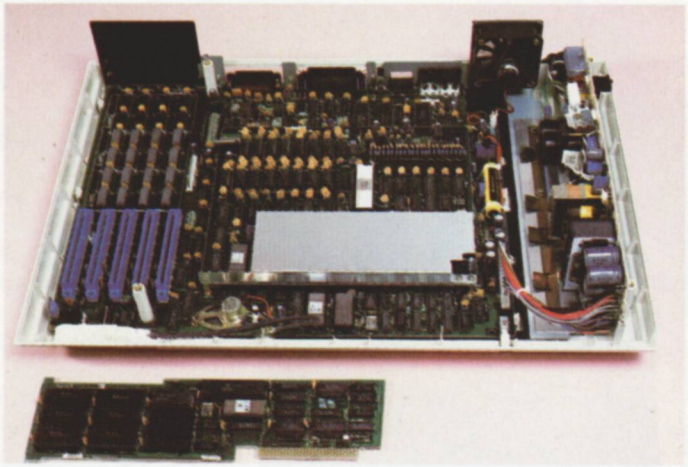
cenamiento de 320 Kbytes al ser de doble cara y doble densidad.
El formato de los discos es de 40 pistas por cara y 16 sectores por pista; cada sector contiene a su vez 256 bytes.
De los 320 Kbytes que pueden almacenarse en cada disco, 32 Kbytes se reservan para información del sistema, 2 Kbytes para el directorio y 8 Kbytes reservados como memoria alternativa; por tanto, la capacidad real de cada disco para el usuario, disminuye hasta los 278 Kbytes.
Uno de los discos contiene el sistema operativo que ocupa un total de 63 KB. Este hay que cargarlo desde la unidad de disco a la memoria central.

Periféricos

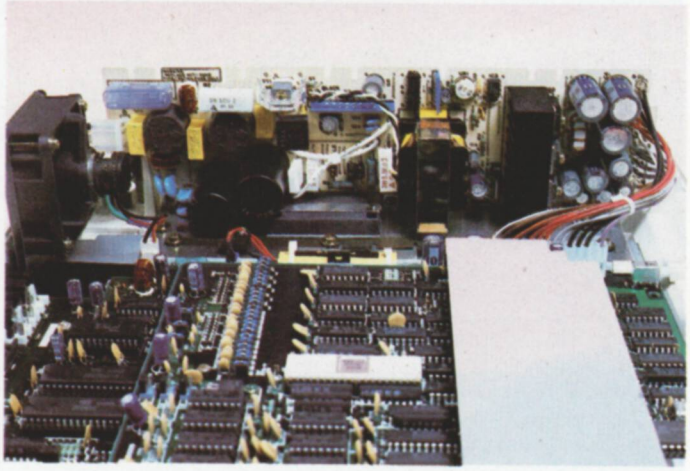
La mayoría de las impresoras de EPSON son compatibles con el QX-10, debido al interface paralelo Centronics con el que viene equipado el ordenador.
Entre las que soporta, podemos citar el modelo MX-80 III, de 80 caracteres por línea en escritura normal y con una velocidad de 80 caracteres por segundo. Admite también representaciones gráficas, con una resolución de 480 × 8 puntos por línea.
Al disponer de 5 slots de expansión, el QX-10, puede soportar otros tipos de periféricos tales como: acopladores acústicos (para transmisión de datos), interfaces para transmisión por fibra óptica o adaptadores para la interface de aplicación generales IEEE 488.

Sistemas operativos y lenguajes

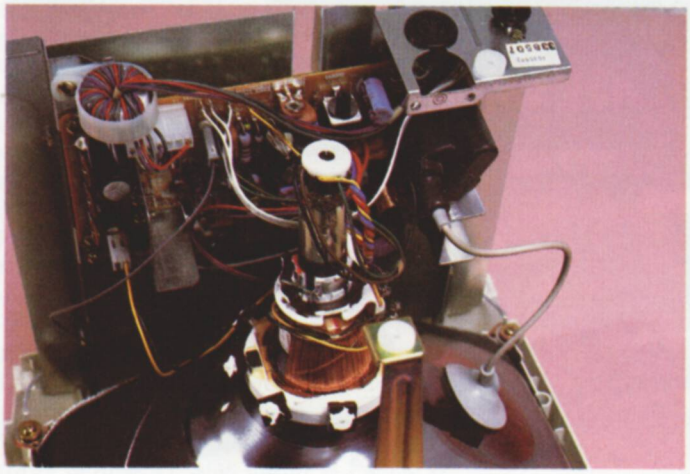
El sistema operativo estándar es una versión mejorada del CP/M 2.2, denominada MultiFonts CP/M (MF-CP/M), que ofrece la posibilidad al usuario de emplear uno u otro en función de la aplicación particular. Está compuesto por tres módulos de programas básicos denominados: CCP (Console Command Processor), BIOS (Basic I/O System) y BDOS (Basic Disk Operating System).



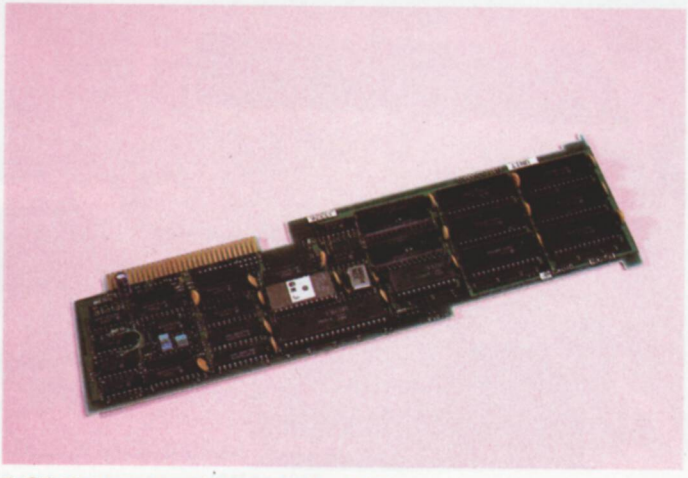
La unidad central del QX-10, que incorpora el popular microprocesador Z-80A de la firma Zilog, puede soportar una ampliación de RAM de hasta 256 Kbytes.



La zona posterior del mueble de la unidad central dispone de un conjunto de conectores para la adaptación de un lápiz óptico, una impresora, el monitor y diversos adaptadores de interface (Centronics y RS/232C).



El monitor del QX-10 dispone en su parte posterior de un mando que permite la regulación de la intensidad luminosa de la visualización en pantalla.



El QX-10 dispone, en el interior de la unidad central, de cinco conectores de expansión que permiten la adición, entre otros módulos, de la tarjeta ROM de generación de caracteres.

EPSON QX-10

Operando en MF-CP/M se puede seleccionar la comunicación con el alfabeto de hasta 8 idiomas. Para seleccionar el alfabeto español, se pulsa el comando ESC seguido de las teclas S y C.
En cuanto a lenguajes de programación, el QX-10 soporta, entre otros, el FORTRAN-80 y el CIS-COBOL, además del MF-BASIC.

Software de aplicación

La biblioteca de programas es muy extensa, ya que es posible ejecutar todas las aplicaciones desarrolladas para CP/M. Por lo demás, EPSON está desarrollando aplicaciones concretas que aprovechan todas las posibilidades del MF-CP/M.
Entre los paquetes actualmente disponibles, cabe destacar los siguientes:
— WORD STAR (Tratamiento de textos).
— DATA STAR (Tratamiento de datos).
— Dbase II (Control de bases de datos).

Soporte y distribución

El fabricante proporciona una excelente documentación entre la que destaca el manual de operación, el manual de MF-BASIC y un sumario de comandos, aunque todos ellos redactados en inglés.

El distribuidor oficial para España es Tradetek Inter Internacional, S. A.

Configuración básica: Unidad central con 192 Kbytes de RAM y EPROM para IPL, teclado independiente, monitor monocromático en fósforo verde de 12" y una doble unidad para discos flexibles de 5 1/4" y 320 Kbytes de capacidad por disco.

Configuración máxima: Unidad central con 256 Kbytes de RAM, EPROM para IPL, módulo ROM generador de caracteres, teclado independiente en español, monitor a color de 12", impresora bidireccional FX-100 de 160 c.p.s., lápiz óptico y doble unidad para discos flexibles de 5 1/4" (320 Kbytes por disco).

La unidad central del EPSON QX-10 puede soportar, aparte de la pantalla y el teclado, una impresora y una unidad doble de disquettes de 5 1/4" con 320 Kbytes de capacidad por disco.

EL LENGUAJE FORTRAN

DESDE la aparición del ordenador electrónico, las universidades y centros de investigación técnica y científica comprendieron la gran ayuda que éste les iba a proporcionar. Pero existía la barrera de los lenguajes de programación. Los lenguajes de máquina y de ensamble estaban bastante lejos de la forma de expresión técnica (módelos matemáticos, expresiones aritméticas, ecuaciones, etc.). De ahí surgió la idea de un lenguaje para la resolución de problemas científicos mediante técnicas de cálculos numéricos. El primero de estos lenguajes fue el SHORT CODE creado en 1949 por el Dr. Mandy para UNIVAC. Unos años más tarde, en 1953, aparece el SPEED-CODING de Backus para IBM.

Treinta años de FORTRAN

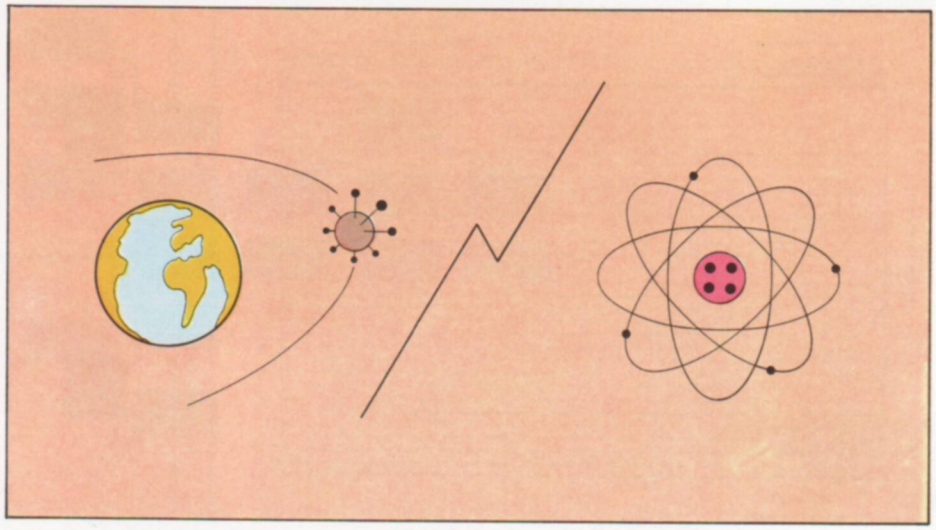
El mismo Backus inició el desarrollo del lenguaje FORTRAN en noviembre de 1954, publicando IBM el primer manual en 1956.
El FORTRAN II aparece en junio de 1958, aportando importantes ampliaciones fundamentalmente en el campo de las subrutinas, y el FORTRAN IV lo hace en 1962. A partir de entonces fueron varias las firmas que contribuyeron a su expansión, de tal forma que existen en la actualidad más de cincuenta compiladores.
Para equipos pequeños existen diversas versiones, siendo de destacar el FORTRAN-80 de Microsoft (firma especializada en software para ordenadores personales y micros). Esta contiene 28 tipos de instrucciones diferentes, entre ellas todas las normalizadas por ANSI.
Las principales ventajas del FORTRAN son su capacidad para manejar expresiones matemáticas y su velocidad de cálculo, por lo que es apropiado para las aplicaciones científicas en las que lo importante es el volumen de cálculo y no la entrada/salida.
Trabaja sólo con compilador, realizando la compilación en un solo paso.
Debido a su antigüedad, y a pesar del gran número de revisiones que ha sufrido, hoy día es bastante inferior a otros lenguajes tales como el Pascal o versiones avanzadas del BASIC. De todas formas, los programas escritos en FORTRAN constituyen todavía un elevado porcentaje de las bibliotecas de programas técnicos y científicos.

Las hojas de programación

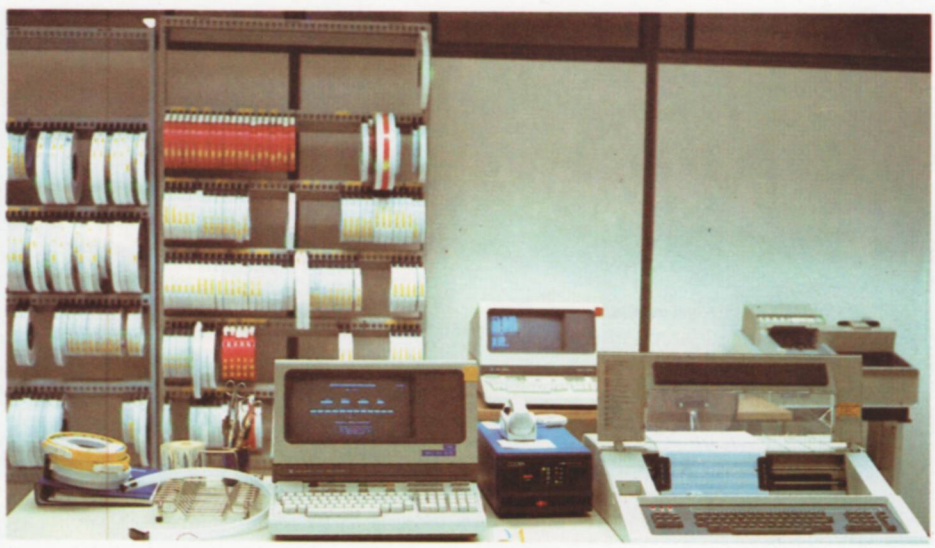
El FORTRAN utiliza los caracteres alfabéticos de la A a la Z, los dígitos del 0 al 9 y los caracteres especiales siguientes:

(espacio) = + – * / ( ),.$ '

Con ellos se escriben los programas en una hoja de diseño especial que admite 80 caracteres por línea.
Si en la columna 1 aparece una C, indica que es un comentario que no será traducido por el compilador.
Una cadena de 1 a 5 dígitos, ajustados a la derecha, sirve de etiqueta para referenciar una sentencia. Por consiguiente sólo se etiquetarán aquellas instrucciones a las que tengamos que saltar o identificar.
La sentencia o instrucción se escribe desde la columna 7 a la 72. Algunas versiones permiten continuar la sentencia en otra línea, poniendo en la línea siguiente un carácter distinto



El lenguaje FORTRAN está orientado especialmente a aplicaciones de tipo científico y técnico, desde la determinación de las órbitas de los satélites espaciales, a la definición de distancias entre elementos atómicos.



El primer manual de FORTRAN fue publicado por IBM en 1956, año en que comenzaron a redactarse numerosas versiones o dialectos, hasta llegar a nuestros días en los que el FORTRAN ya ha sido superado en eficacia por otros lenguajes.

EL LENGUAJE FORTRAN

de 0 o un espacio en la columna 6. Las posiciones 73 a 80 no son tenidas en cuenta por el compilador; se usan para identificar el programa y el número de secuencia dentro del mismo.

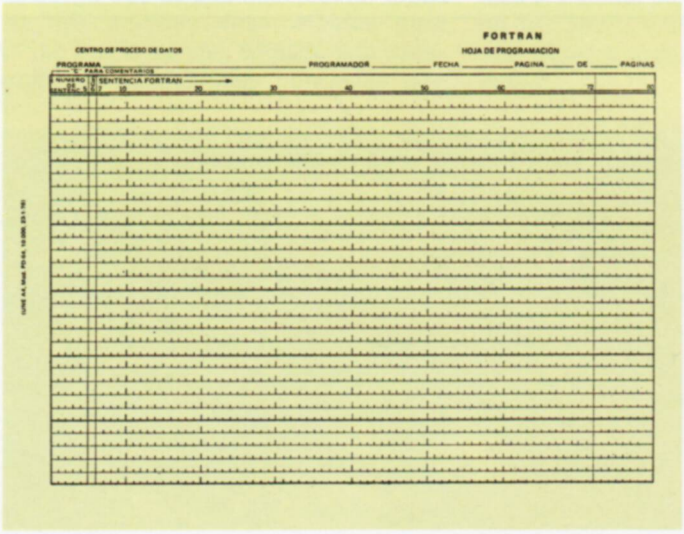
Datos y operaciones

El FORTRAN admite constantes y variables enteras, reales y de doble precisión. En general las de doble precisión, tan necesarias en el cálculo técnico-científico, permiten hasta 16 cifras decimales.

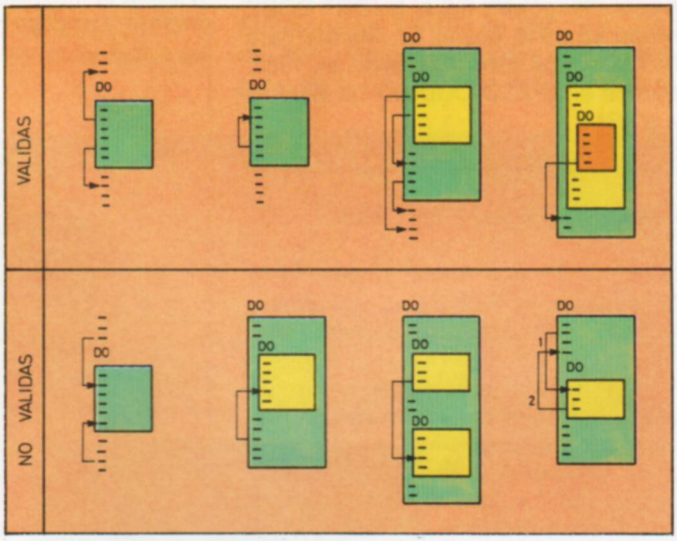
También se admiten constantes Hollerith, es decir, cadenas de caracteres que se escriben entre apóstrofos (por ejemplo, 'NOTA'). Las variables emplean nombres simbólicos que tienen una letra como primer carácter. El resto hasta 8 pueden ser letras o números. Si no se usan sentencias de especificación, el compilador entiende que toda variable cuya primera letra es I, J, K, L, M o N es una variable entera; mientras que las que empiezan por letras desde la A a la H y desde la O a la Z son variables reales.

Las variables matriciales se representan con el nombre seguido por los subíndices colocados entre paréntesis y separados por comas. Así, A(3,I,4) representa un elemento de una matriz tridimensional.

Las operaciones permitidas son: exponenciación (**), multiplicación (*), división (/), suma (+) y sustracción (–). Las normas de prioridad de operaciones y uso de paréntesis vistos en el BASIC



La figura muestra una hoja de programación típica empleada para la redacción de programas en lenguaje FORTRAN. Dispone de 80 columnas. Las ocho últimas se usan para identificar el programa y el número de secuencia dentro del mismo.



Las instrucciones de tipo «DO» para el control de bucles pueden anidarse, aunque no deben interferirse. El cuadro muestra las estructuras válidas y erróneas de este tipo de instrucción.

```
ASSIGN s TO i
BACKSPACE u
BACKSPACE (alist)
BLOCK DATA [sub]
CALL sub [([a [,a]...])]
CHARACTER [*len[,]] nam [,nam]...
CLOSE (clist)
COMMON [/[cb]/]nlist[[,]/[cb]/nlist]...
COMPLEX v [,v]...
CONTINUE
DATA nlist/clist/ [[,]nlist/clist/]...
DIMENSION a (d) [,a(d)]...
DO s [,] i = e1,e2[,e3]
DOUBLE PRECISION v [,v]...
ELSE
ELSE IF (e) THEN
END
END IF
ENDFILE u
ENDFILE (alist)
ENTRY en [([d [,d]...])]
EQUIVALENCE (nlist) [,(nlist)]...
EXTERNAL proc [,proc]...
FORMAT fs
fun ([d [,d]...]) = e
[typ] FUNCTION fun ([d [,d]...])
GO TO i [[,](s [,s]...)]
GO TO s
GO TO (s [,s]...)[,] i
IF (e) st
IF (e) s1, s2, s3
IF (e) THEN
IMPLICIT typ (a [,a]...)
  [,typ (a [,a]...)]...
INQUIRE (iflist)
INQUIRE (iulist)
INTEGER v [,v]...
INTRINSIC fun [,fun]...
LOGICAL v [,v]...
OPEN (olist)
PARAMETER (p=e [,p=e]...)
PAUSE [n]
PRINT f [,iolist]
PROGRAM pgm
READ (cilist) [iolist]
READ f [,iolist]
REAL v [,v]...
RETURN [e]
REWIND u
REWIND (alist)
SAVE [a [,a]...]
STOP [n]
SUBROUTINE sub [([d [,d]...])]
v = e
WRITE (cilist) [iolist]
```

Resumen de las sentencias del lenguaje FORTRAN.

son también de aplicación en el FORTRAN.
Existen numerosas funciones que permiten cálculos trigonométricos, logarítmicos, etc. El programador puede crear sus propias funciones.

Instrucciones

Veremos a continuación algunas de las instrucciones más características del FORTRAN.
Instrucciones de asignación. Tiene el formato

variable = expresión

y sirven para dar a una variable el valor obtenido en el cálculo de una expresión aritmética.

Ejemplo: Z(J) =
= (–B–SQRT(B ⋆B–4 ⋆A ⋆C))/(2 ⋆A)
que atribuye al elemento J de la serie Z el valor de una raíz de la ecuación $AX^2 + BX + C = 0$.

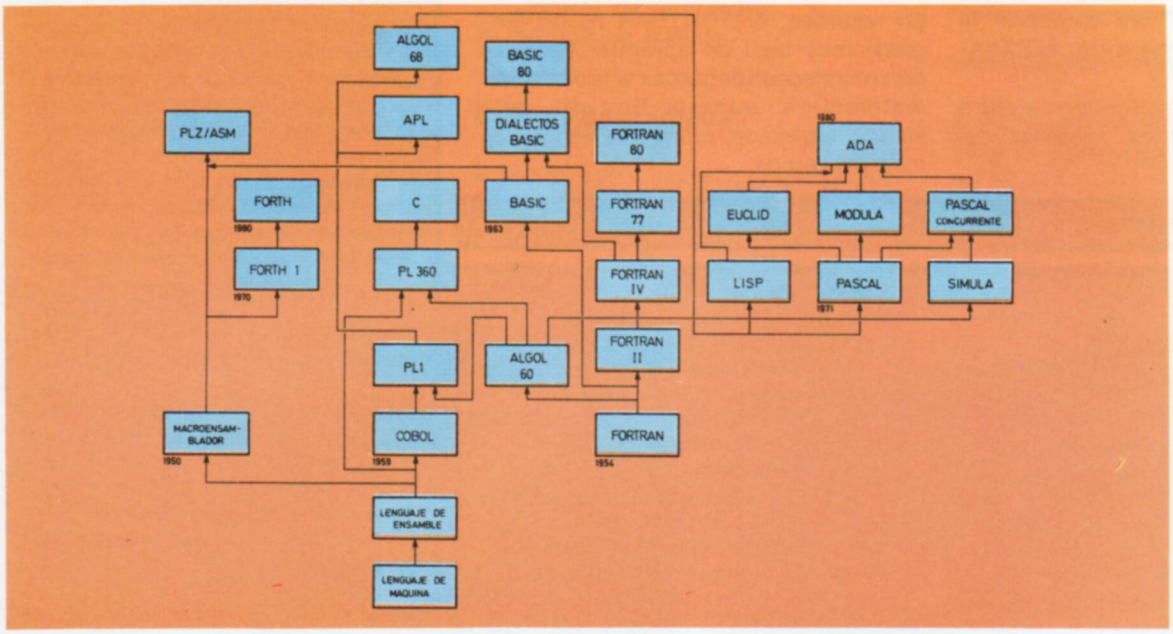
Instrucciones de control. Alteran la secuencia normal.

Salto incondicional: GO TO n (siendo n una etiqueta).

Salto calculado: GO TO $(n_1,n_2,...n_n)$, expresión aritmética (continua en n_1, n_2, etc. según que el valor de la expresión sea 1, 2,...).

Salto condicional: IF(exp) n_1, n_2, n_3 (va a n_1, n_2, ó n_3 según que el valor de la expresión sea negativo, cero o positivo).

Bucle: DO n_1 ind = exp_1, exp_2, exp_3 (repite todas las instrucciones que siguen hasta la n_1 variando el valor del índice desde exp_1 a exp_2 en incrementos de exp_3 en exp_3).



El gráfico muestra la evolución histórica de los lenguajes de programación. Puede observarse cómo el BASIC tomó en un principio conceptos y sentencias del FORTRAN.

Ejemplo de aplicación escrita en FORTRAN. Este programa emplea los caracteres alfabéticos de la A a la Z, los dígitos del 0 al 9, además de otros símbolos especiales.

EL LENGUAJE FORTRAN

Ejemplo: S = 0

```
DO 7    I = 2,10,2
7 S = S + A(I)
```

(calcula la suma de los elementos pares de la serie A).
Fin de bucle: CONTINUE (se usa como última instrucción de un DO, en lugar de la que correspondería en caso de transferencia, lo que no está permitido).
Otras: PAUSE, STOP, END.
Instrucciones de declaración. Para reservar zonas de memoria. Dimensionado de matrices: DIMENSION nombre matriz (i_1, i_2 ... i_n). Para compartir la misma posición de memoria: EQUIVALENCE (V_1, V_2 ... V_n).
Para establecer correspondencia entre variables de diferentes unidades del programa = COMMON V_1, V_2 ... V_n

Ejemplo:
Programa principal:
DIMENSION a (10)
COMMON A, B (50, 10)

Subprograma:
COMMON D (10),
E (50, 10)

Otras: INTEGER, REAL, DATAS
Instrucciones de entrada/salida. Todas las entradas se realizan con la instrucción READ, mientras que las salidas se ejecutan mediante la instrucción WRITE. Ambas llevan asociadas una instrucción FORMAT que define el formato de lectura o escritura.
En resumen, el FORTRAN es un lenguaje muy fácil de aprender para las personas acostumbradas a la notación matemática, aunque hoy día está siendo superado por otros lenguajes más modernos.

Conceptos básicos

Evolución de los lenguajes de programación

El primer lenguaje que se utilizó en la programación fue el lenguaje de *máquina*, codificado en binario o hexadecimal y que sólo era comprensible para la máquina. Debido a su complejidad para el programador se desarrolló el lenguaje de *Ensamble*, que aunque seguía estando próximo a la máquina, sustituía el código máquina por códigos nemotécnicos y simbólicos. A mediados de los años 50 aparecen los lenguajes *Macroensambladores*, con potentes instrucciones para sustituir a los procesos de codificación nemotécnicos largos e incómodos. Siguiendo los principios de los *Macroensambladores*, a principios de los 70 se desarrolló el FORTH 1 que dio lugar en los 80 al FORTH utilizado en microprocesadores, así como al PLZ/ASM.
De forma paralela a los lenguajes *Macroensambladores* se desarrollaron lenguajes que se alejaban de la máquina y se aproximaban mucho más al problema. Los dos lenguajes históricamente más importantes son el FORTRAN y el COBOL, el primero dedicado al campo científico y el segundo ligado al campo comercial y de gestión.
El FORTRAN fue presentado por IBM en 1954, su desarrollo dio lugar en 1970 al FORTRAN IV y en 1977 al FORTRAN 77.
En 1965 nace un lenguaje derivado del FORTRAN y que las universidades americanas empezaron a utilizar como lenguaje científico, el BASIC, que se ha desarrollado actualmente de tal manera que hoy es el lenguaje tradicional en los microordenadores. Otro lenguaje científico interesante que nació en el año 58 fue el ALGOL, cuya versión ALGOL 60 es la más representativa.
El COBOL se desarrolló en 1959, teniendo versiones mejoradas en los años 74 y 80, llamadas COBOL 74 y COBOL 80.
A la vez que se desarrollaban estos tipos de lenguajes, al principio de los años 60 se comienzan a diseñar lenguajes polivalentes, es decir, lenguajes que sirven para solucionar tanto problemas científicos como de gestión. Nace el PL/1, derivado del COBOL y del ALGOL 60; del PL/1 se deriva el APL, en los años 70, que se utiliza en trabajos interactivos y en la enseñanza asistida por ordenador. No hay que pasar por alto que actualmente el BASIC es también un lenguaje polivalente.
Siguiendo con la idea de llegar a un lenguaje universal, se desarrolló en los años 70 un lenguaje derivado del ALGOL 60 y del ALGOL 68, llamado PASCAL, cuyas versiones (PASCAL UCSD) y dialectos afines son utilizados ampliamente.
En la década actual los lenguajes PASCAL RECURRENTE y LIPS (derivado del ALGOL 68) han conducido al lenguaje ADA, que en principio tiene un amplio futuro. Por ejemplo, el Departamento de Defensa de los EE. UU. ha decidido que a partir de 1985 y con el fin de estandarizar sus aplicaciones, no aceptará ningún trabajo que no esté programado en ADA.

Glosario

¿Qué significa FORTRAN?

La palabra FORTRAN está formada por las sílabas iniciales de FORmula TRANslation, cuyo significado es traducción de fórmula, con lo que indica su utilidad para los problemas técnicos y científicos.

¿Por qué es necesario el cálculo numérico?

El lenguaje FORTRAN no admite más operadores que los aritméticos, mientras que los problemas técnicos precisan trabajar con integrales, derivadas, ecuaciones diferenciales, interpolaciones, ajustes de curvas, etc. El cálculo numérico es una colección de técnicas que permiten sustituir estos cálculos por conjuntos de operaciones aritméticas que dan una solución lo suficientemente aproximada.

¿Son suficientes dieciséis decimales para los problemas científicos?

Muchas veces no, ya que al sustituir los cálculos reales por métodos aproximados de cálculo numérico pueden ser precisas más cifras. El programador debe conocer la teoría de errores y tenerlo en cuenta a la hora de programar.

¿Por qué una línea FORTRAN sólo admite 80 caracteres?

Es una reminiscencia de las tarjetas perforadas, ya que cuando se desarrolló el FORTRAN éstas constituían el medio de entrada más ampliamente utilizado.

TERMINAL FACIT 4420

EL terminal FACIT 4420 está basado en el microprocesador de 8 bits F-8 de la firma Fairchild. Dispone, por lo demás, de una memoria ROM interna de 6 Kbytes y de una zona de RAM de 2 K × 12 bits para el almacenamiento de los datos de pantalla. La distribución de los 12 bits es la siguiente:

- 7 bits correspondientes al código ASCII del carácter.
- 1 bit de marca que indica si el cursor está en esa posición o no.
- 4 bits correspondientes a atributos de video:
  - — Subrayado o doble subrayado.
  - — Parpadeo en pantalla.
  - — Video inverso.
  - — Intensidad de brillo normal o reducida.

El FACIT 4420 tiene el teclado separado de la pantalla. Esta es monocroma de fósforo verde con dos intensidades de iluminación y un tamaño de 12" de diagonal. El mueble que contiene la pantalla puede inclinarse más o menos para conseguir el mejor ángulo de visión.

Los comandos de control pueden ser introducidos a través del teclado o bien pueden recibirse codificados desde el ordenador. Los diversos grupos de comandos de control son:

- — Control de pantalla.
- — Control del cursor.
- — Control de edición de textos.
- — Control de impresora.
- — Modos de trabajo.

Control de pantalla

- *Subrayado o doble subrayado de caracteres.* Cada carácter ocupa un área de pantalla distribuida en una matriz de 7 × 10 puntos. Las dos últimas filas se reservan para el subrayado del carácter que puede ser simple o doble.
- *Parpadeo.* Los caracteres pueden aparecer en pantalla fijos o parpadeantes. La frecuencia de parpadeo es seleccionable mediante microinterruptores entre 1, 2 ó 4 Hz.
- *Video normal o inverso.* Se selecciona la aparición de los caracteres iluminados sobre fondo oscuro u oscuros sobre fondo iluminado.
- *Brillo de pantalla.* La pantalla tiene dos niveles de intensidad de brillo.

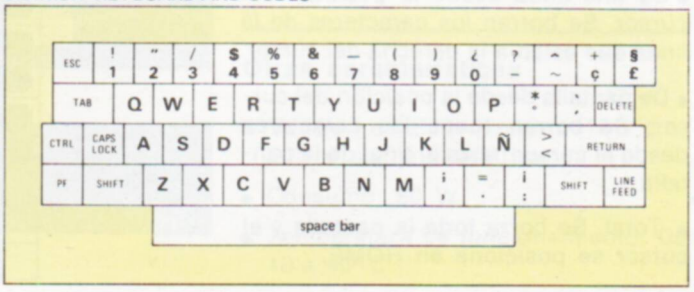
Control del cursor

- *Direccionado absoluto.* La pantalla tiene 24 líneas de 80 caracteres. El cursor se puede posicionar mediante la indicación de las correspondientes coordenadas.

El terminal FACIT 4420, de diseño sobrio y funcional, está controlado internamente por un microprocesador de 8 bits del tipo F-8 de la firma Fairchild.

El teclado del FACIT 4420 puede incorporar los caracteres alfanuméricos de siete idiomas, entre ellos el español. Está separado de la pantalla y es de perfil bajo.

APPENDIX 6
SPANISH KEYBOARD AND CODES



El dibujo muestra la disposición de las teclas en el teclado español, que incluye la letra «Ñ», así como otros símbolos de uso común en nuestro idioma.

TERMINAL FACIT 4420

• *Movimientos unitarios*. Se puede mover el cursor posición a posición a través de la pantalla: arriba, abajo, izquierda o derecha.

• *Retroceso de carro*. El cursor pasa a la primera posición de la siguiente línea.

• *HOME*. El cursor pasa a la primera posición de la parte superior izquierda de la pantalla.

• *LINE FEED*. El cursor se mueve hacia abajo una línea en la misma columna. Si el cursor está en la línea inferior de la pantalla permanece en el mismo sitio, pero todo el texto visualizado se desplaza una línea hacia arriba.

• *LINE FEED inverso*. El cursor se mueve hacia arriba una línea en la misma columna. Si el cursor está en la línea superior, permanece en el mismo sitio, pero el texto se desplaza una línea hacia abajo apareciendo la línea superior libre.

• *Movimientos de tabulación*. El cursor se mueve con desplazamientos correspondientes a las tabulaciones definidas.

Control de edición de textos

Borrados:

• De un carácter. Se borra el carácter marcado por el cursor, recuperando espacio los caracteres situados a la derecha.

• De una línea. Se borra línea en la que se encuentra el cursor, desplazándose las restantes a una posición superior.

• De una línea desde la posición del cursor. Se borran los caracteres de la línea que están a la derecha del cursor.

• De pantalla desde la posición del cursor: Se borran todos los caracteres desde el cursor hasta el final de la pantalla.

• Total. Se borra toda la pantalla y el cursor se posiciona en HOME.

Inserciones:

• De un carácter. En la posición indicada por el cursor.

• De una línea. Se deja espacio para una nueva línea.

Control de impresora

El terminal admite la conexión de una impresora local, para lo cual dispone de un conector al efecto en la zona posterior. Los comandos destinados al control de la impresora permiten las siguientes funciones:

• Impresión automática de una línea después de escribirla en la pantalla.

• Impresión de la línea en la que está situado el cursor.

• Impresión de la página completa.

Es posible controlar que los datos recibidos en el terminal, desde el teclado o desde el ordenador, aparezcan sólo en pantalla, sólo en la impresora o en los dos elementos a la vez.

Modos de trabajo

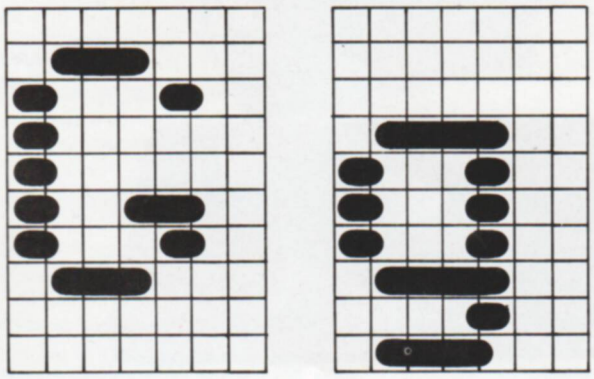
El terminal admite los siguientes modos de trabajo:

• *Modo de transparencia*. Sólo es posible cuando el terminal está en LOCAL. Los caracteres de control aparecen en la pantalla pero no son interpretados como comandos.

• *Modo gráfico:* Existe un juego de caracteres semigráficos codificados desde 5F hasta 7F.

• *Modo de mantenimiento de pantalla (HOLD SCREEN)*. En el modo normal, cuando se llena la pantalla de caracteres recibidos, las líneas se van desplazando hacia arriba, desapareciendo por la parte superior de la pantalla. En este modo, cuando se llena la pantalla, se mantiene su contenido y se manda una señal al ordenador para que suspenda la transmisión. La recepción posterior

En la pantalla, de fósforo verde, de 12" de diagonal y un área de visualización de 8" × 6", pueden escribirse hasta 24 líneas de 80 caracteres, con una resolución por carácter de 7 × 10 puntos.



El gráfico muestra la forma en la que el FACIT 4420 representa los caracteres en pantalla a partir de una matriz de 7 × 10 puntos.

de nuevos caracteres puede hacerse línea a línea con mantenimiento, o bien por pantallas completas.

• *Modo de formato.* En este modo se establecen espacios de pantalla protegidos por los que no puede pasar el cursor.

Existe además en este terminal un modo de autochequeo (SELF TEST). Hay programas almacenados de autochequeo a los cuales se puede acceder a través del teclado. En cualquier caso, al conectar el terminal se efectúa un chequeo automático de pantalla y teclado. Los resultados de esta prueba aparecen en la pantalla y en los diodos luminosos del teclado.

Características generales

El teclado es alfanumérico e incorpora un teclado numérico y otro de activación de funciones y de control del cursor.

El teclado es de la característica 2 Key roll-over (sobrepulsación de dos teclas) e incorpora las siguientes posibilidades:

• Click de realimentación fisiológica al pulsar una tecla.

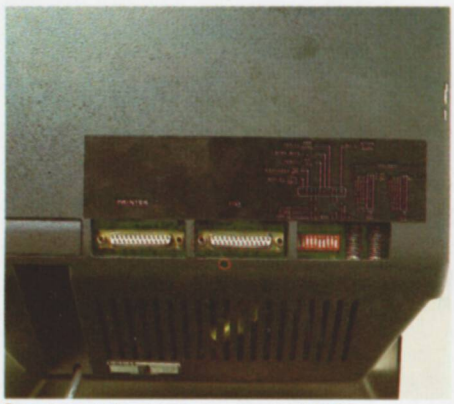
• Repetición automática si la tecla está pulsada más de 0,5 seg.

La velocidad de repetición se selecciona mediante microinterruptores entre 7,5, 15 y 30 caracteres por segundo. gundo.

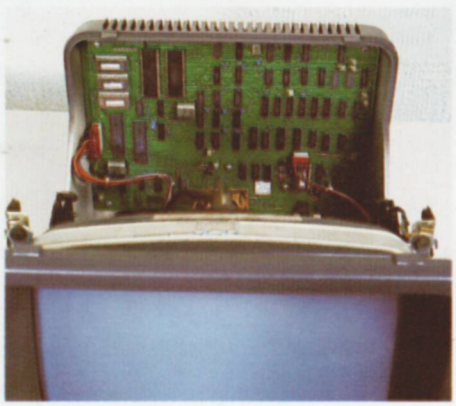
El teclado alfanumérico puede incorporar los caracteres correspondientes a los siguientes idiomas: sueco, alemán, danés, inglés, español, francés y noruego.

El teclado de funciones incorpora: control de cursor, HOME, Clear, Delete, inserción de caracteres, subrayado, parpadeo y video inverso.

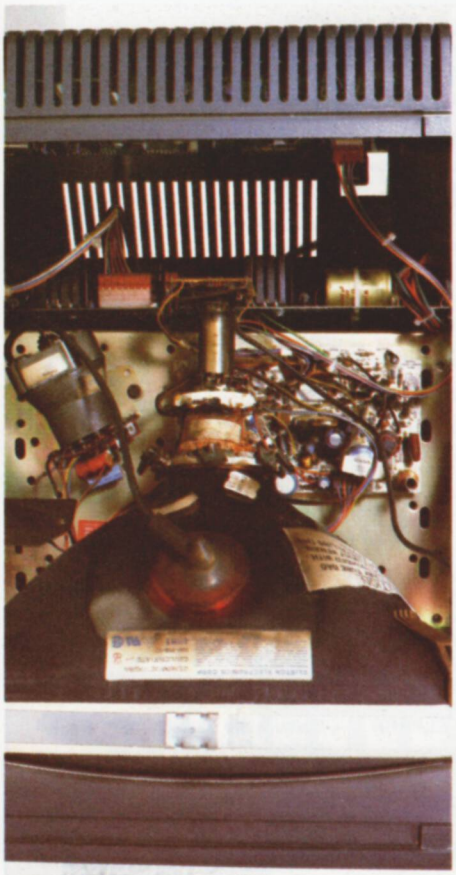
Por otra parte, la pantalla es de color verde con una diagonal de 12″ con área de visualización de 8″ × 6″. En ella se escriben hasta 24 líneas de 80 caracteres formados por una matriz de alta definición de 7 × 10 puntos, en total 1.920 caracteres.



Panel posterior del terminal FACIT 4420, donde pueden observarse los distintos conectores de comunicación del sistema y la red de microconmutadores de selección.



El terminal FACIT 4420 dispone de una memoria ROM de 6 Kbytes y una RAM de 2 K × 12 bits para la gestión de pantalla.



Los 12 bits de la RAM del 4420 se distribuyen en: 7 para el código ASCII de cada carácter, 1 bit de marca para la situación del cursor y otros 4 para las atribuciones de video.

Comunicación con el ordenador

Las comunicaciones de datos con el ordenador se efectuán en formato serie, gestionadas por una UART y a través de un interface estándar RS/232, siendo opcional el bucle de 20 mA.

Existe una memoria FIFO que actúa como buffer de 32 caracteres.La velocidad de transmisión es seleccionable mediante microinterruptores entre 300, 600, 1.200, 2.400, 4.800, 9.600 y 19.200 baudios.

Son posibles tres modos de comunicación con el ordenador:

• En línea, ful duplex: los datos se transmiten desde el teclado al ordenador y luego se devuelven al terminal.

• En línea, semiduplex: los datos. se transmiten desde el teclado al ordenador y al terminal al mismo tiempo.

• Local: los datos se transmiten al terminal y de éste al ordenador.

Para los citados modos de comunicación son posibles dos protocolos:

• X-ON, X-OFF: La indicación de que el terminal está preparado o no para recibir datos se da mediante la transmisión de códigos.

• READY/BUSY: La indicación anterior se evidencia con el nivel de tensión de la línea READY/BUSY.

Otras características

• Tensión de alimentación: 115 V c.a., 220 V c.a. ó 240 V c.a.

• Consumo: 60 W.

• Temperatura de funcionamiento: De 10 a 40° C.

• Humedad de funcionamiento: De 20 a 80 por 100.

• Peso (con teclado): 20 kg.

ANALISIS NUMERICO «OLINUM»

LA biblioteca de subrutinas científicas OLINUM, permite a los usuarios desarrollar programas, basándose en un paquete de rutinas matemáticas que pueden ser fácilmente incorporadas a los mismos, formando una unidad de programación.

Las subrutinas están escritas en BASIC y carecen de instrucciones de entrada y salida, lo que permite el uso encadenado de varias de ellas según las necesidades de programación.

Cada una está contenida en un fichero cuyo nombre se forma con los caracteres SL seguidos por un nemónico relativo al cálculo que ejecuta.

Empleo de las subrutinas

El usuario debe escribir su propio programa teniendo en cuenta los siguientes puntos:

- El programa debe contener las instrucciones de entrada y salida correspondientes a las variables que emplea la subrutina.
- El programa debe contener los comandos DIM y DCL necesarios para las tablas.
- La subrutina OLINUM debe invocarse con la instrucción GOSUB seguida del número de línea correspondiente.
- Si se incorporan varias subrutinas al mismo programa, cada una de ellas debe ser llamada por su instrucción GOSUB.
- El programa no debe contener ningún número de línea empleado por las subrutinas.

Comienzo de la aplicación

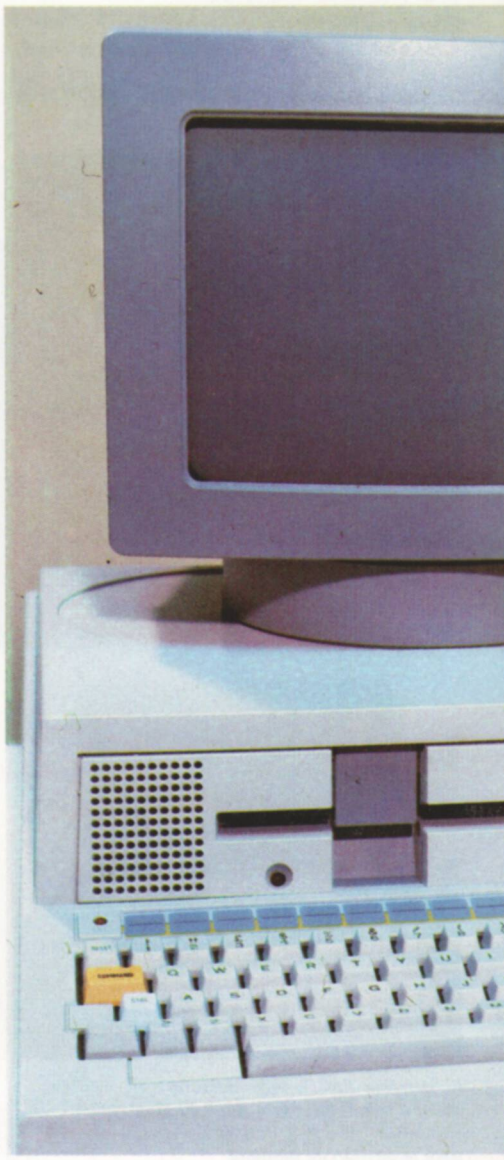
Después de cargar el Sistema Operativo se carga en memoria el programa principal y, seguidamente, se ejecuta un comando MERGE con el nombre del fichero que contiene cada una de las subrutinas a emplear. El programa puede ser ahora ejecutado, almacenado o listado en la forma convencional. Con cada subrutina se suministra un programa de TEST que permite trabajar di-

Aplicación: **Análisis numérico OLINUM**
Ordenador: **OLIVETTI M-20**
Configuración: **Unidad central, pantalla, doble unidad de disco e impresora.**
Sistema operativo: **PCOS**
Lenguaje: **BASIC**
Soporte: **Disco flexible de 5 y 1/4 pulgadas**
Documentación: **Manual de 154 páginas, en inglés**
Copyright: **OLIVETTI**
Distribuidor: **Hispano Olivetti, S. A.**

SUBRUTINAS DEL PAQUETE «OLINUM» *

SICONN	Cambio de base.	SLCATN	Arcotangente de un número complejo.
SLPRIM	Generación de números primos.		
SLEUCL	Máximo común denominador y mínimo común múltiplo.	SLCLN	Logaritmo natural de un número complejo.
SLRFCO	Conversión de fracción racional a continua.	SLCEXP	Exponencial de un número complejo.
SLSUCO	Conversión de irracionales cuadráticas a continuas.	SLCRZ	Recíproco de un número complejo.
SLCFCO	Convergentes de una fracción continua.	SLCZMZ	Multiplicación de dos números complejos.
SLFACT	Factorial y factorial logarítmica.	SLCZDZ	División de dos números complejos.
SLBINO	Coeficientes binomiales.		
SLMULT	Coeficientes multinomiales.	SLCSQR	Raíz cuadrada de un número complejo.
SLDUPL	Probabilidad de duplicación en un universo dado.	SLCZN	Potenciación de un número complejo.
SLATN2	Arcotangente de una razón.		
SLCONV	Reducción de un ángulo al primer cuadrante.	SLCZA	Potencia real de un número complejo.
SLRPCC	Conversión de coordenadas rectangulares a polares.	SLPLRC	Evaluación de polinomios reales (argumentos complejos).
SLPRCC	Conversión de coordenadas polares a rectangulares.	SLPLRR	Evaluación de polinomios reales (argumentos reales).
SLCSIN	Seno de un número complejo.	SLPRRR	Cálculo de coeficientes de un polinomio desde raíces.
SLCCOS	Coseno de un número complejo.		
SLCTAN	Tangente de un número complejo.	SLPLYM	Multiplicación de dos polinomios reales.
SLCASN	Arcoseno de un número complejo.	SLPLYD	División de dos polinomios reales.
SLCACS	Arcocoseno de un número complejo.	SLPTRA	Traslación de coeficientes de un polinomio real.

(*) De análisis combinatorio, funciones elementales y polinómicas.



La aplicación OLINUM, distribuida por Hispano Olivetti, S. A., se introduce en el M-20 a partir del soporte magnético en el que reside: un disco flexible de 5 1/4".

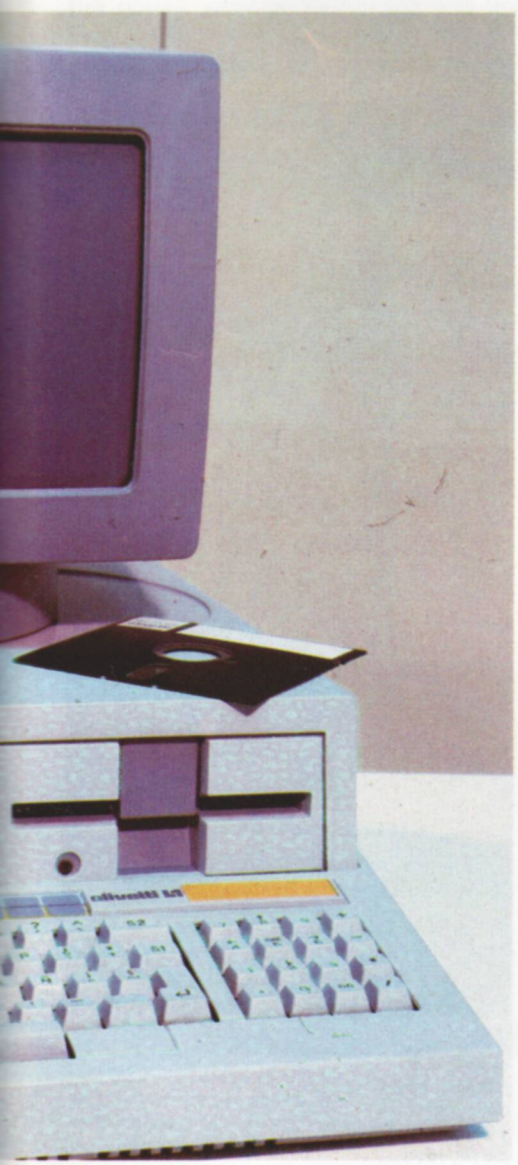
rectamente con la misma y obtener resultados. La estructura de estos programas puede servir de base a otros más específicos del usuario.

División de las subrutinas

Las subrutinas se encuentran encuadradas en 10 grandes grupos. Estos son: Análisis Combinatorio, Funciones Elementales, Funciones Elementales (complejas), Funciones Polinómicas, Funciones de Alta Matemática, Solución de Ecuaciones, Algebra Lineal, Aproximación de Curvas e Interpolación, Integración y Diferenciación y Ecuaciones Diferenciales Ordinarias.
Dada la complejidad de los cálculos, el manual de la Aplicación explica cuidadosamente los métodos a emplear para la obtención de resultados satisfactorios.

Cálculos que se realizan

Análisis combinatorio: Cambios de base, factoriales, tabla de números primos, máximo común denominador y mínimo común múltiplo, coeficientes binomiales y multinomiales, probabilidades de duplicación en un universo dado, etc.
Funciones elementales: Arcotangente de una razón, reducción de un ángulo al primer cuadrante, conversión de coordenadas cartesianas a polares y viceversa, etc.
Funciones elementales (complejas): Seno, coseno, tangente, arcoseno, arcocoseno, arcotangente, logaritmo natural, exponencial, recíproco, multiplicación, división, raíz cuadrada, potenciación, etc.
Funciones polinómicas: Evaluación de polinomios reales (argumentos complejos reales), cálculo de coeficientes de un polinomio desde sus raíces, multiplicación y división de dos polinomios reales y traslación de coeficientes, etcétera.
La posibilidad de obtener resultados en «doble precisión» junto con el detalle de la forma concreta en que se realizan los cálculos, permite una rigurosidad y fiabilidad en los resultados equiparable a la de los lenguajes científicos.



Subrutinas de análisis combinatorio, funciones elementales y polinómicas del paquete OLINUM.

```
Load "1: SLCONN
O.K.
MERGE "1: SLCONN.
O.K.
RUN.

 10 OPTION BASE 1 : DEFDBL A-Z : PI=3.14159265359#
 20 INPUT "ENTER BASE1,BASE2  ",B1,B2
 30 INPUT "ENTER INTEGER PART  ",N
 40 A$="Base "+STR$(B1)
 50 PRINT
 60 PRINT USING "\       \Integer Part    ##.##############^^^^",A$,N
 70 INPUT "ENTER FRACTIONAL PART  ",F
 80 PRINT USING "         Fractional Part ##.##############^^^^",F
 90 PRINT
100 PCONN1=N:PCONN2=F:PCONN3=B1:PCONN4=B2:PCONN5=C1
110 GOSUB 30401:FUNCONN=FUN.
120 IF ((FUNCONN+1)(.5) OR ((FUNCONN+1)) 6 ) GOTO 140
130 ON FUNCONN+1 GOTO 140,200,220,240,260,280
140 A$="Base "+STR$(B2)
150 PRINT USING "\       \Integer Part    ##.##############^^^^",A$,R
160 PRINT USING "         Fractional Part ##.##############^^^^",B2
170 PRINT
180 PRINT
190 GOTO 20
200 PRINT "INTEGER PART AN INTEGER )=0 ONLY"
210 GOTO 30
220 PRINT "0(= FRACTIONAL PART(1 ONLY"
230 GOTO 70
240 PRINT "BASE MUST BE AN INTEGER )1"
250 GOTO 20
260 PRINT "ERROR IN INTEGER DIGITS"
270 GOTO 30
280 PRINT "ERROR IN FRACTIONAL DIGITS"
290 GOTO 70
30401 PI=3.14159265359#
30402 REM SUBROUTINE
```

Ejemplo de carga de una subrutina y explotación de la misma dentro de un programa de aplicación elaborado por el usuario.

A través del teclado del M-20, el usuario de OLINUM puede realizar cálculos numéricos de: análisis combinatorios, funciones matemáticas elementales, funciones polinómicas, etc.

PROGRAMA

Nombre: ***Salto de dama***
Ordenador: ***Oric-1***
Memoria requerida: ***16 Kbytes***
Lenguaje: ***BASIC***

EL objetivo de este juego es dejar en el tablero una sola pieza (símbolo«*»). En un principio parece fácil, pero a medida que se avanza, la misión se torna más y más complicada. Si se carece de experiencia en el juego, se pueden poner las cosas algo más fáciles estableciendo como meta dejar sólo 5 ó 10 piezas. Esto se puede hacer alterando las líneas 450, 600 y 620.

Para retirar una pieza del juego, habrá que saltar con otro asterisco por encima de ésta, de modo similar a como se hace en el juego de damas, pero en este caso sólo en dirección horizontal o vertical. Así, una vez elegida la pieza que se quiere eliminar, habrá que encontrar otra en posición contigua que saltando en horizontal vertical, pueda ir a parar a una casilla vacía (no se consideran como tales los guiones que marcan el fondo del tablero).

El ordenador pide primero la coordenada de origen, indicada en forma fila-columna (F4, por ejemplo) y luego, la de la posición en blanco a la que se quiere saltar (D4, por ejemplo). En el caso de que se haya pretendido hacer el salto en diagonal, saltar más de una pieza partir de un cuadro en el que no hay asterisco o ir a una posición en la que no hay cuadro vacío, la máquina rechazará el movimiento y lo hará saber por medio del mensaje «PARAMETROS ERRONEOS». De no ser así, se efectuará automáticamente el movimiento y se escuchará un sonido de campanilla para indicar que se puede realizar otra jugada.

Si en algún momento las piezas adoptan una configuración tan aislada que se hace imposible continùar, pulsando la tecla «?» el ordenador informará sobre el número de jugadas efectuadas y la cantidad de piezas que quedaron sobre el tablero; de todas formas, no hay que desesperarse, ya que aunque matemáticamente es posible dejar tan sólo una pieza, la tarea no es precisamente fácil.

CUADRO DE VARIABLES

Variable	Descripción	Variable	Descripción
I-J	Variables FOR de diversa utilidad.	CO	Columna de la posición de origen.
M	Número de movimientos efectuados.	FD	Fila de la posición de destino.
P	Piezas sobre el tablero.	FM	Fila de la pieza a desaparecer.
T	Variable para la tabulación en la impresión del dato introducido.	FO	Fila de la posición de origen.
CD	Columna de la posición de destino.	C$	Cadena que contiene la columna de origen o destino en la toma de datos.
CM	Columna de la pieza a desaparecer.	F$	Similar a la anterior, con referencia a la fila.

```
10 REM Lopez Martinez
15 PRINTCHR$(17)
20 CLS:PAPER3:INK0
25 P=32:M=0
30 FORI=11TO27STEP2
40 FORJ=4TO20STEP2
50 PLOTI,J,"-"
60 NEXT
70 NEXT
80 FORI=17TO21STEP2
90 FORJ=6TO18STEP2
100 PLOTI,J,"*"
110 NEXT
120 NEXT
130 FORI=13TO25STEP2
140 FORJ=10TO14STEP2
150 PLOTI,J,"*"
160 NEXT
170 NEXT
180 FORI=1TO7
190 PLOT11+2*I,3,CHR$(48+I)
200 PLOT10,4+2*I,CHR$(64+I)
210 NEXT
220 PLOT19,12," "
230 FORI=1TO26:PRINT:NEXT
240 PLOT1,25,"Movimiento desde
250 T=0
260 GOSUB1000
270 FO=(ASC(F$)-64)*2+4
280 CO=VAL(C$)*2+11
290 IFSCRN(CO,FO)<>42THEN500
300 T=5
310 GOSUB1000
320 FD=(ASC(F$)-64)*2+4
330 CD=VAL(C$)*2+11
340 IFSCRN(CD,FD)<>32THEN500
350 IFCO<>CDANDFO<>FDTHEN500
360 IFCO<>CDTHEN390
370 IFABS(FO-FD)<>4THEN500
375 FM=FO+2*SGN(FD-FO)
376 CM=CO
380 GOTO400
390 IFABS(CO-CD)<>4THEN500
395 CM=CO+2*SGN(CD-CO)
396 FM=FO
400 IFSCRN(CM,FM)<>42THEN500
405 PING:M=M+1
410 PLOTCO,FO," "
420 PLOTCD,FD,"*"
430 PLOTCM,FM," "
440 P=P-1
450 IFP>1THEN240
460 GOTO550
500 PLOT1,25,"PARAMETROS ERRONEOS
510 WAIT100
520 GOTO240
550 CLS
555 FORI=0TO9:PRINT:NEXT
560 PRINTSPC(7)"FELICIDADES, ";
570 PRINT"LO CONSIGUIO.":PRINT
600 IFP<>0THENFORI=0TO11:PRINT:NEXT
605 PRINTSPC(11)"Empleo ";M;"jugadas."
610 PRINT
620 IFP<>0THENPRINTSPC(11)"Quedaron ";P;"piezas."
630 PRINTCHR$(17):END
1000 GETF$
1005 IFF$="?"THENCLS:GOTO600
1010 IFF$<"A"ORF$>"I"THEN1000
1020 PLOT18+T,25,F$
1030 GETC$
1035 IFC$="?"THENCLS:GOTO600
1040 IFC$<"1"ORC$>"9"THEN1030
1050 PLOT19+T,25,C$
1060 RETURN
```

El objetivo de este juego es dejar la pantalla en blanco con unas reglas parecidas a las del juego de damas, eliminando cada uno de los símbolos que aparecen en la misma.

La fotografía muestra la forma en la que el ordenador solicita del jugador las coordenadas de origen, posteriormente solicitará las de la posición en blanco a la que se quiere saltar.

Al final, el éxito o el fracaso dependerá de la paciencia y habilidad del jugador.

EL GOBIERNO DE LA INFORMATICA

INICIAMOS esta semana una serie de comentarios acerca de las repercusiones que las modernas tecnologías de la información están provocando en el ámbito social de los países industrializados. Este artículo intenta ofrecer una visión general de cómo las nuevas tecnologías, especialmente las que tienen su base en el empleo de microprocesadores inciden en la configuración de las funciones, necesidades y posibilidades de desarrollo de un Estado moderno.

El Estado informático

La informática, así como otras tecnologías denominadas de «última generación» (redes de comunicación por fibra óptica, satélites dedicados a la investigación y a las comunicaciones, sistemas de control de procesos industriales, etc.), ha irrumpido en los organigramas funcionales clásicos de los Estados desde tres vertientes distintas. Estas se imbrican y forman el conjunto de la problemática con que los actuales responsables de la política y la Administración nacionales tienen que enfrentarse.

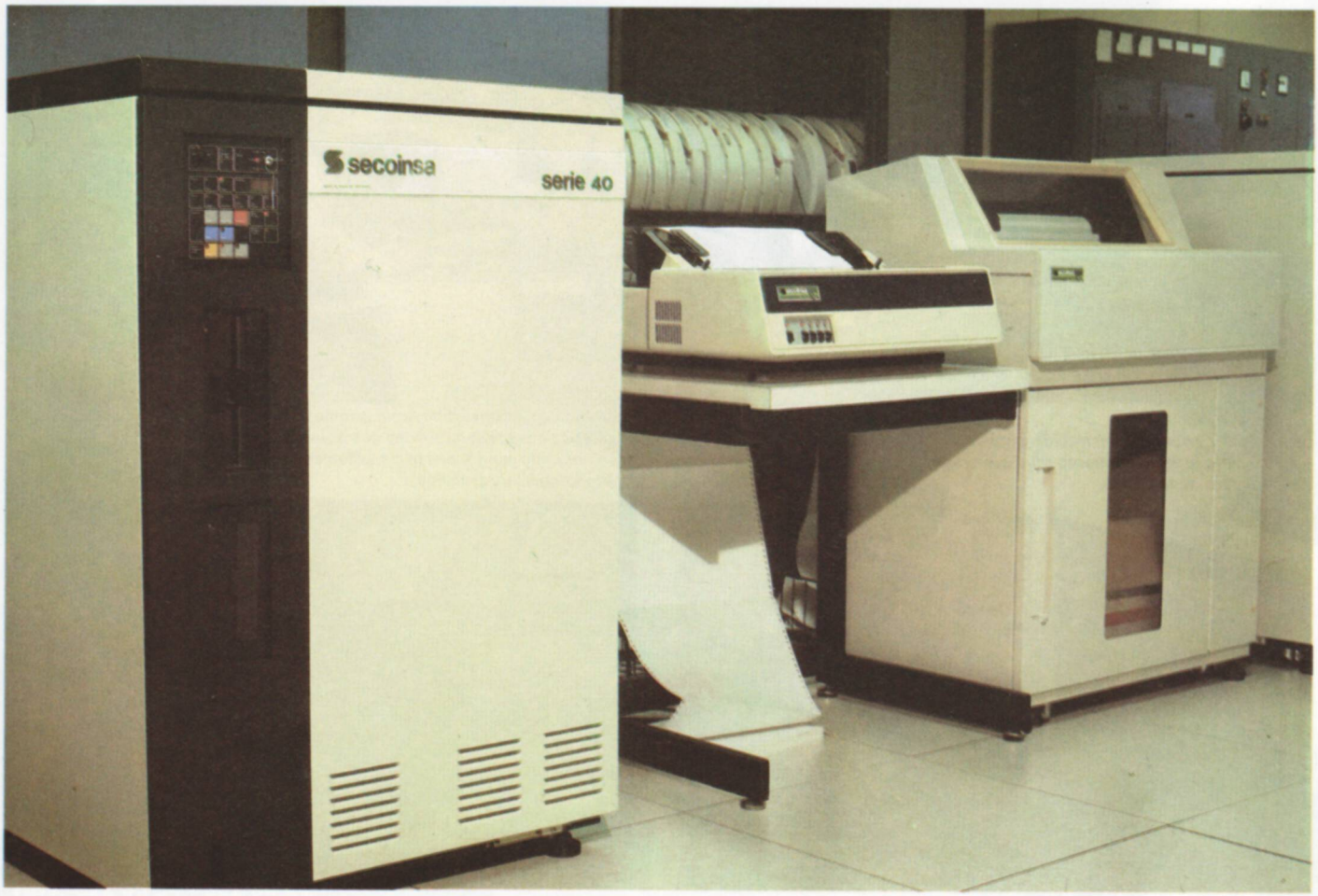
La primera implicación de la informática en el Estado se encuentra en estos momentos en un estadio muy avanzado. Se trata de la automatización propiamente dicha de la gestión de las Administraciones Públicas.

El estudio de las nuevas tecnologías

La informática, como objeto de estudio y elaboración de estrategias de desarrollo industrial, de investigación, difusión y enseñanza, es la segunda de las grandes áreas en las que el Estado moderno se ve envuelto.

El estudio de las repercusiones que en la sociedad occidental está provocando el uso de los ordenadores, así como las tecnologías de última generación, es una labor interdisciplinar por excelencia. Su realización es encomendada, por regla general, a organismos creados al efecto, en los que suelen tener participación delegaciones de los Ministerios o departamentos del Gobierno, (Trabajo, Industria, Educación, Comunicaciones, etc.).

Las comisiones de estudio de los Gobiernos de los países industrializados se enfrentan en este momento con el problema de fomentar, desde las perspectivas de las nuevas tecnologías, el desarrollo económico, social y cultural de sus respectivos países. La elaboración de normas de apoyo y canalización



Uno de los aspectos de la intervención de la informática en el Estado es la mecanización de los procesos de gestión de la Administración pública.

EL GOBIERNO DE LA INFORMATICA

del uso de ordenadores, sistemas telemáticos, etc., además de la salvaguardia de los derechos a la libertad e intimidad de los ciudadanos, pueden considerarse como los problemas a solucionar con más urgencia por las mencionadas comisiones de estudio, dentro, claro está, de unos límites más que recortados de tiempo.

Como resumen de los problemas a los que el Estado moderno debe enfrentarse, como consecuencia del desarrollo de nuevos procedimientos industriales y de gestión de la información, puede afirmarse que los Gobiernos tienen la responsabilidad de evitar el retraso de sus respectivos pueblos, a medio y largo plazo, en el paso de una sociedad basada en la producción industrial a otra centrada en el tratamiento de la información, así como en la toma de decisiones que otros países ejecutarán. Todo ello conforme a la futura división internacional del Trabajo.

El apoyo gubernamental a la informática

Los Gobiernos del mundo desarrollado se ven, cada vez con mayor urgencia, impulsados a promulgar medidas concretas orientadas al apoyo de la informatización de sus países. En este sentido cabe señalar, entre las acciones que ya se están llevando a cabo en algunas naciones de la OCDE, las siguientes: implantación de una política crediticia, fiscal y de planificación que prime el uso de métodos o tecnologías innovadoras por la empresa privada, especialmente la pequeña y mediana; desarrollo de la industria nacional de producción de tecnologías punteras; normalización y homologación de equipos y sistemas; orientación de la producción y políticas de investigación de empresas privadas, por medio del diseño de una estrategia coherente de compras del sector público, etc.

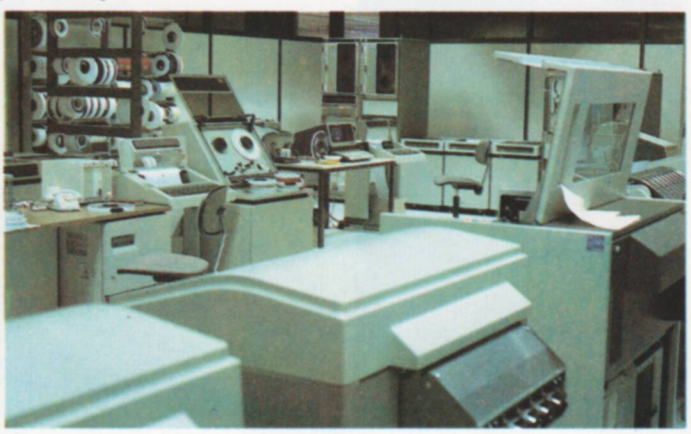
Por último, cabe señalar otra función del Estado dentro del esfuerzo por adecuar sus estructuras tradicionales a las nuevas exigencias del desarrollo técnico: la puesta en marcha de programas de educación y difusión de las nuevas tecnologías. Es este nuevo papel del Estado el que mayor trascendencia tendrá, sin duda alguna, en el desarrollo de las futuras generaciones.

En este sentido, puede señalarse que una política educativa adecuada a las nuevas exigencias del mercado laboral y, al mismo tiempo, a las necesidades futuras de los ciudadanos, señalarán el triunfo o el fracaso de países que, como España, no pueden permitirse el lujo de perder el tren de la Historia.

El uso de sistemas informáticos en la Administración pública se encuentra en este momento muy avanzado en los países de la OCDE.

El Estado se ha convertido en el principal estudioso de las repercusiones que sobre la sociedad en su conjunto tendrá el uso de sistemas y metodologías informáticas.

La puesta en marcha de programas de educación y formación informática es uno de los puntos clave en este campo de la política actual de los países desarrollados.



La búsqueda de soluciones eficaces, así como el análisis de los problemas derivados del uso de la informática en la Administración pública y en la sociedad pasa, necesariamente, por el uso de ordenadores.