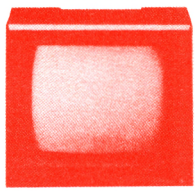
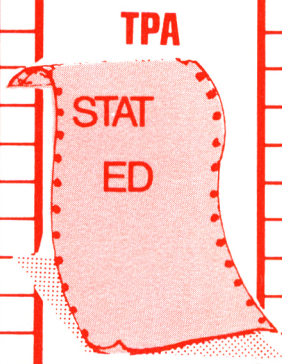
WEILER - SCHIEB

CP/M
EL LIBRO DE EJERCICIOS PARA CPC

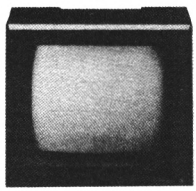


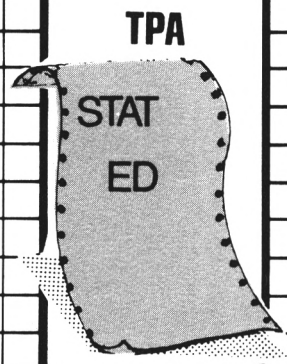


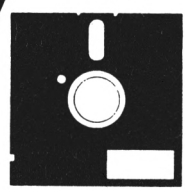
UN LIBRO DATA BECKER
EDITADO POR FERRE MORET, S.A.

WEILER - SCHIEB

CP/M
EL LIBRO DE EJERCICIOS PARA CPC







UN LIBRO DATA BECKER
EDITADO POR FERRE MORET, S.A.

Los programas y utilidades que se encuentran en este libro, pueden ser adquiridos en cassette o Disquette, solicitándolos directamente a FERRE MORET, S.A., o en sus distribuidores

Este libro ha sido traducido por Dn. Christoph Küstler, traductor técnico y amplio conocedor del CPC.

Imprime: APSSA, ROCA UMBERT, 26 - L'HOSPITALET DE LL. (Barcelona)

Depósito legal B-44.244/85

ISBN 84-86437-38-5

Advertencia importante

Los circuitos, procedimientos y programas reproducidos en este libro, son divulgados sin tener en cuenta el estado de las patentes. Están destinados exclusivamnte al uso amateur o docente, y no pueden ser utilizados para fines comerciales.

Todos los circuitos, datos técnicos y programas de este libro, han sido elaborados o recopilados con el mayor cuidado por el autor y reproducidos utilizando medidas de control eficaces. No obstante esposible que exista algún error. FERRE MORET,S.A. se ve por tanto obligada a advertirles, que no puede asumir ninguna garantía, ni responsabilidad jurídica, ni calquier otra responsabilidad sobre las consecuencias atribuibles a datos erróneos. El autor les agradecerá en todo momento la comunicación de posibles fallos.

INDICE

I. El computador

Para ser sinceros: Yo le envidio a Vd. porque Vd. ya sabe lo que no sabe. Sin embargo yo sólo puedo tratar de imaginarme hasta que punto ha proliferado su nivel de conocimiento sobre los computadores y acerca de todo cuanto les rodea. Como que desgraciadamente no puedo preguntar a cada uno de Vds. por separado, en el presente capítulo comenzaré por así decirlo desde "Adán y Eva", para que todos tengamos una misma base de partida, y logremos entendernos mejor de ahora en adelante. Este tipo de procedimientos se utiliza en todas partes. ¡Fíjese por ejemplo en las normas DIN, o intente escribir un programa para su computador! Las personas siempre recurren a convenios establecidos para lograr entenderse mejor.

De modo que vamos a empezar. En primer lugar preguntamos: ¿qué es realmente un computador? Siendo fieles a un método altamente científico, aclararemos lo que es un computador.

A este efecto, primero simularemos ser completamente ignorantes, diciendo: "un computador es una caja en la que sucede algo". Por suerte el nombre inglés "computer" ya nos revela algo acerca de lo que sucede. "Compute" puede traducirse por "calcular", y por lo tanto el computador es una máquina qu calcula. Es fácil de imaginar que un mago de los números de es categoría, pueda emplearse excelentemente para procesar grandes cantidades de números, por ejemplo en el terreno de la contabilidad, o en la interpretación de mediciones en serie. Los escolares y estudiantes seguramente serán capaces de encontrar muchas más aplicaciones prácticas para un esclavo electrónico de este tipo, que además sepa calcular.

I. El teclado

Sin embargo - hay un problema. Simplemente tenemos ante nosotros una caja que sabe calcular. ¿Pero de qué forma le decimos, qué es lo que tiene que calcular? Una posibilidad consistiría en decírselo al oido. Pero esta forma de comunicación con computadores actualmente todavía forma parte del terreno de la ciencia ficción, y por lo tanto no nos sirve. Es por ello que nos hemos acordado del bueno y viejo teclado, conectando dicho "dispositivo de entrada de letras v caracteres" a la susodicha caja.

Ya que un computador ofrece bastante más que una simple máquina de escribir, el teclado también presenta un aspecto ligeramente diferente. Entre 60 y 100 teclas es lo que ofrece el teclado de un computador, según sea su precio y su nivel de prestaciones. Además del pulcro órden de letras, números y caracteres, a menudo existe un teclado numérico por separado que posibilita la introducción rápida de ristras de números, y si además se trata de un ordenador muy versátil, también suele incorporar un teclado adicional, con unas teclas llamadas "de funciones".

Puede comprobar fácilmente con una ojeada, si el teclado de su computador es alemán o americano. Si en la fila de letras superior empezando por la izquierda se lee la palabra QWERTZ, Vd. es el flamante propietario de un dispositivo de entrada que concuerda con el gusto alemán. Si por lo contrario en dicho lugar Vd. encuentra la palabra QWERTY, seguramente tendrá más facilidades con muchos programas, ya que la mayoria de éstos actualmente proceden de América. En tal caso tendrá que prescindir de algunas de las letras típicamente alemanas. No puede tenerse todo.

Ya sea en alemán o en inglés, existen algunas teclas que son muy importantes para el trabajo con el computador, y que a menudo llevan una denominación idéntica o similar en ambos tipos de teclado. Dado que éstas excepciones son importantes a lo largo de éste libro y para todos los programas, ahora verá de cuales se trata:

RETORNO DE CARRO

Vd. conoce ésta tecla de la máquina de escribir, aunque con el ordenador normalmente se usa en un sentido completamente

distinto. Las abreviaciones que lleva dicha tecla, normalmente son éstas:

RETURN ("retorno" en inglés)

ENTER (final de una entrada)

CR ("carriage return" = retorno de carro en inglés)

<--- (se explica por sí mismo)

RETROCESO

BS (viene de "back step" = paso atrás en inglés)

TECLA DE BORRADO O DE CORRECCION

DELETE ("borrar" en inglés)

DEL (abreviación de DELETE)

RUB OUT ("erradicar" en inglés)

CONVERSION

SHIFT (pasa a mayúsculas)

FIJADOR

SHIFT LOCK (fija permanentemente el modo mayúsculas)

La función exacta de ésta última tecla depende del tipo de teclado. En parte, mediante ésta tecla no se conmuta todo el teclado, sino realmente sólo las letras mayúsculas. Ello puede representar un alivio considerable para el trabajo. En algunos teclados también existen teclas especiales para la conversión de letras y que vienen marcadas con

ALPHA LOCK o

CAPS LOCK.

Si tiene mucha suerte, el estado actual del teclado se indica adicionalmente por medio de un diodo luminoso.

Para el trabajo con el computador es sumamente importante la tecla CONTROL, que puede encontrar en el extremo derecho de su teclado Amstrad, y que debe accionarse para muchos comandos de control conjuntamente con alguna tecla alfanumérica o numérica. Por regla general lleva la abreviación:

CTRL o (en el Amstrad)

CTL.

I.2 La pantalla

Ahora que disponemos de la posibilidad de introducir datos en la caja, también necesitamos un camino, por el cual los datos puedan regresar hacia nosotros. No nos serviría de nada, que el ordenador efectuara aún los cálculos más increibles, sin que dispusiera de alguna posibilidad para comunicarnos su resultado. La emisión de los datos se realiza en forma bastante ecológica y poco contaminante por medio de una pantalla. En primer lugar se trata de un método bastante rápido, que en segundo lugar, no produce ningún ruido. Los datos y los mensajes de error del computador se visibilizan inmediatamente, y en caso de que además pretenda apuntar sus errores, siempre puede conectarle una impresora.

Una pantalla en el fondo no es otra cosa que un televisor "modificado", y funciona según el mismo principio. Solamente que el "televisor del computador", que a menudo también se denomina monitor generalmente no tiene que representar imágenes en movimiento, sino letras, números, y otros tipos de caracteres. Es por ello que las prestaciones que se esperan de un monitor son diferentes, a las que se esperan de un televisor. La resolución debería ser claramente superior, para que después de una sesión de varias horas no se produzca una "niebla" dentro de la cabeza del usuario y para que tampoco se abuse de su retina hasta el límite.

I.3 La impresora

En la época primaria de los computadores gigantes con prestaciones enanas, las máquinas de telex servían como medio de salida de datos en forma permanente. De modo que el computador enviaba sus datos a la impresora, donde por regla general consumía una respetable cantidad de papel. Otro inconveniente al lado de la insaciable necesidad de papel, era la forma ruidosa, en que las impresoras solían plasmar sus caracteres sobre el papel. Pero - todavía hay mucha gente que insiste en disponer de las informaciones por escrito, por lo que no se puede eludir del todo la necesidad de disponer de una impresora. Como las hay de todos tipos y de todas las categorias, a continuación se presenta un breve resúmen:

los precios y las prestaciones difieren de tal modo, que incluso los especialistas en la materia, apenas ya son capaces de abarcar toda la gama que actualmente ofrece el mercado. El precio de una impresora puede oscilar entre 25.000 Pts. y 600.000 Pts. y en consecuencia las impresoras pueden ofrecer velocidades desde 15 hasta 800 caracteres por segundo.

I.3.1 Impresoras matriciales

Las así denominadas impresoras matriciales, han dado un resultado especialmente bueno, como colaboradoras de los micros. Son las de mayor importancia en el mercado, y de entre sus filas proceden los ejemplares más económicos y rápidos. Todo empezó con unas impresoras matriciales, que imprimían pulcramente punto por punto, produciendo así una escritura, que lo era todo, menos pulcra. Hay que buscar la causa de ello, en el principio de la impresión de los años "tempranos" (es decir, alrededor de 1980). Unas púas individualmente dirigibles, ejercían presión sobre una cinta impregnada de tinta, que de ésta manera entraba en contacto con el papel, imprimiendo un pequeño punto. Mediante siete puntos ya era posible, reconocer más o menos una letra, o un número. Sin embargo la lectura de textos mas largos ya se dificultaba considerablemente.

Pero - éstos tiempos han pasado. A excepción de aparatos muy baratos, las impresoras matriciales de hoy en dia, ofrecen una longitud de línea de 80 caracteres y una velocidad del órden de 100 caracteres por segundo, además de una impresión, que aunque no sea apta para la correspondencia en general, sí lo es para el intercambio interno de información. La gran ventaja de las impresoras matriciales: por regla ofrecen una gran variedad de tipos de escritura, además de juegos de caracteres para cargar o programar a voluntad, y a los que puede accederse mediante una sencilla conmutación. De este modo también se simplifica la correspondencia con Grecia y con El Japón.

Las impresoras de alto rendimiento llegan a alcanzar velocidades de hasta 800 caracteres por segundo, imprimiendo 132 caracteres por línea, y sus precios oscilan entre 400.000 Pts. y 600.000 Pts. Estas "nuevas" impresoras todavía son relativamente recientes.

Generalmente disponen de 24 púas impresoras, con las que a velocidad reducida llegan a alcanzar la calidad de una máquina de escribir. Aquí los valores corrientes son del órden de 70 caracteres por segundo para la correspondencia, y de 140 a 150 caracteres en modo de escritura rápida.

I.3.2 Impresoras margarita

Las impresoras margarita pertenecen a la segunda categoría, que también incluye las impresoras del tipo de "cesta" impresora, y del tipo de cabeza impresora esférica. Dichas impresoras generalmente están basadas en anteriores conceptos de máquinas de escribir, y por regla necesitan un mantenimiento elevado. Asimismo su fabricación resulta bastante costosa. Las impresoras del tipo margarita, reemplazan la cesta, o la cabeza esférica de tipos por una rueda, sobre la que se fijan los distintos caracteres de forma elástica. Un martillo diminuto golpea sobre los caracteres, imprimendo éstos en el papel, de modo que no se aprecia ninguna diferencia respecto a una máquina de escribir. De modo que si la correspondencia debe tener un aspecto "como de imprenta", una impresora de tipo margarita, es la opción más apropiada.

Velocidad: en modelos baratos generalmente apenas 20 caracteres por segundo, en los modelos más caros, hasta 80 caracteres.

I.3.3 Impresoras a chorro

Son bastante más baratas las nuevas impresoras a chorro. De manera similar a la impresora matricial, las letras y los caracteres se forman a partir de puntos. Sin embargo no se produce ningún contacto mecánico entre el papel y la cabeza impresora, sino que se disparan diminutas gotas de tinta sobre el papel. La calidad de la impresión es comparable a la de las sencillas impresoras matriciales. Las ventajas más destacadas de las impresoras a chorro: a un nivel acústico muy bajo de unos 45 dbA (decibelios A), se logran hasta 150 caracteres por segundo. Inconveniente: sólo es posible obtener copias mediante impresión reiterada.

I.3.4 Impresoras térmicas

Las así denominadas impresoras térmicas poseen el mismo inconveniente. Los caracteres se generan mediante calor, el cual altera directamente la superficie del papel. Naturalmente es necesario un papel especial, que que con el tiempo amarillece bajo la luz. Por otro lado, éstas impresoras que también trabajan con el método de los puntos, son extremadamente económicas, siendo todavía perfectamente acceptables para funciones protocolarias. Las calculadoras de bolsillo por debajo de las 6.000 Pts. hoy día ya incorporan sencillas impresoras térmicas.

La solución óptima para una impresión rápida y limpia con el ordenador personal, son las impresoras láser. La impresión de éstas máquinas no puede distinguirse del trabajo de una imprenta profesional.
Por cierto, se me ocurre una cosa. Nuestra caja, que en lo que a su descripción se refiere, entretanto ya se parece bastante a un computador, naturalmente también debería disponer de un interruptor de corriente. Con la buena intención de dificultar un encendido o apagado accidental, muchos fabricantes de ordenadores llegaron a la genial conclusión, de que dicho interruptor debía esconderse en alguna parte. Si Vd. todavía no ha encontrado el suyo en su computador, búsquelo en donde nadie lo esperaría. Seguro que lo encuentra allí.

I.4 Memoria de datos

De momento ya podemos hacer bstantes cosas con nuestro computador teórico. Disponemos del computador, de un teclado, y de una pantalla, e incluso ya tenemos una impresora. Lo que todavía nos falta, es un lugar en el que podamos almacenar los datos. Por una parte dentro del computador y - para más duración - también en su exterior. Básicamente puede decirse que la forma de almacenar datos en cualquier medio del ordenador es siempre la misma. Ya sea en la memoria de trabajo, que es la que guarda éste texto que estoy escribiendo aquí, ya sea en un disco, o en un disco duro. Ahora Vd. verá, porqué, y cómo, el computador es capaz de almacenar algo.

Dado que este asunto es de naturaleza un poco teórica, le sugiero que se sirva una taza de café, tome un buen sorbo, y a continuación siga leyendo atentamente. Le esperaré entretanto...

I.5 Uno o cero

Vd. no pondrá en duda que un computador es un aparato eléctrico. Por lo tanto las informaciones dentro del computador también tendrán que procesarse de alguna forma eléctrica. ¿Pero cómo? ¿Cómo le decimos al ordenador, lo que queremos? Para ello hace falta un tipo de representación, que el computador entienda. Lo hay: se trata de los dos estados: presencia de corriente - ausencia de corriente. Una bombilla funciona según el mismo principio. Cuando la bombilla se percata de que hay corriente, toma nota de ello, encendiéndose. Y si se da cuenta, que no hay corriente, no se le ocurre otra cosa, que apagarse. Lo que sabe hacer una bombilla tan poco inteligente, seguramente será pan comido para nuestro ordenador. Los constructores han plasmado en su cerebro electrónico, el hecho de que el estado "presencia de corriente" ha de interpretarse como el valor "uno". De la misma manera, para nosotros el "1" también representa un valor determinado.

Por otro lado, la ausencia de corriente significa para el computador: "0". Ahora nuestro computador ya sabe distinguir dos estados, cosa que nos lleva a unas alturas insospechadas.

I.6 Contar en binario

Para el uso cotidiano, la manera binaria de contar, es decir con unos y con ceros, no es demasiado útil. Si Vd. recita de esta forma su fecha de nacimiento, acabará con la boca seca. Por suerte el computador no tiene boca, de manera que aunque maneje los dos números como un desesperado, no se le secará nada.

Examinemos básicamente el trabajo con números. Existen varios sistemas numéricos: el binario, el decimal, el hexadecimal y algunos más. Todos nosotros dominamos el sistema decimal a la perfección. Apenas podemos pensar en otros valores, que no sean valores decimales. ¿Pero, qué es lo que hacemos realmente? Nos imaginamos un número, por ejemplo el cero. Si la cantidad a representar es mayor, tomamos el uno, y así consecutivamente. Cuando hemos alcanzado el "9", componemos el siguiente número de los dos números más bajos, lo que da por resultado "10".

Podemos proceder de la misma forma, cuando sólo tenemos dos números. El primer valor es el cero. El valor siguiente es el uno. Aquí no podemos seguir, pues ya no nos quedan más números. Por lo tanto componemos el próximo número de los dos anteriores, con lo que para el valor dos, obtenemos la cifra 10. El tres tendrá éste aspecto: 11. Para el cuatro ya volvemos a necesitar otro número más, lo que nos da 100.

Para que nos entendamos mejor, aquí tiene una tabla con los números binarios:

cero = 0
uno = 1
dos = 10
tres = 11
cuatro = 100
cinco = 101
seis = 110
siete = 111
ocho = 1000

y así sucesivamente.

Si sólo tenemos en cuenta las posiciones de este sistema numérico, mediante una posición es posible representar dos valores, con dos posiciones cuatro, con tres posiciones ocho, con 4 posiciones 16, y así indefinidamente. De posición a posición las posibilidades siempre se duplican. También conocemos el valor de las posiciones en nuestro sistema decimal. Solamente que aquí tratamos con potencias de diez. Una posición puede representar 10 valores, dos posiciones cien, tres posiciones mil...

El tingldo de los dos números, se denomina sistema binario (de dos números), y les ha caido tan bién a aquellos que nos depararon las hamburguesas y los computadores, que inmediatamente se han puesto a inventar algunas expresiones técnicas.

La posición de un número, en americano se denomina "digit", y en lugar de binario la supernación dice "binary". De modo que las dos cosas juntas adquieren el nombre de "binary digit", lo que sin más preámbulos se ha abreviado a "bit".

Sin embargo una posición como lugar de almacenamiento, es bién poca cosa, y los computadores modernos tienen una buena cantidad de lugares para almacenar. Si nos quedamos en un bit, las cifras para designar los lugares de almacenamiento, se desorbitarían. Por esta razón se han creado unidades, "palabras", que con una palabra, o con un número, describen todo una serie de bits. Por lo tanto con una palabra de ocho bits, pueden describirse 256 bits, llegando a 65536 bits con una palabra de 16 bits.

Muy a menudo aparece la unidad de 256 posibilidades, por lo que a dicha unidad se le ha asignado el nombre "byte". Es sin embargo un rumor, que por ésta razón la revista americana, que lleva por nombre "Byte", deba publicarse sólo con un máximo de 256 páginas. En realidad contiene más del doble.

I.7 Almacenar valores

Para poder trabajar correctamente, el computador tiene que estar capacitado para almacenar los números. Como hemos visto, para el computador un número consiste en un estado eléctrico: estado de carga, o estado de descarga. Si agrupamos ocho de dichas posiciones de almacenamiento, obtendremos una palabra de ocho bits, con la que puede representarse un total de 256 valores. "Casualmente" ello equivale exactamente a un byte. Dado que un byte tampoco es mucho, también existe la denominación Kilobyte. Sin embargo, un Kilo no equivale exactamente a 1000 bits, sino que, siguiendo la forma binaria de contar, obtenemos 1024 bits. Para unidades muy grandes, todavía existe el Megabyte.

Con ayuda de circuitos electrónicos, ahora es posible leer los valores de un byte, procesarlos, y también volver a escribir en él, los valores nuevos. Esto es todo el secreto de un computador. Su ventaja es, que puede realizar dichas operaciones en brevísimo tiempo, por lo que realmente se le puede considerar un genuino y útil procesador de datos.
Dado que cada posición de memoria tiene que existir realmente, la memoria de cada computador sólo dispone de un espacio limitado para almacenar datos. Los tamaños usuales para la memoria de trabajo de un ordenador personal, con una unidad central de 8 bits, que es propiamente el cerebro, suelen ser de 64 Kilobytes, abreviado 64 KB. Los nuevos ordenadores de 16 bits, ofrecen entre 128 KB y alrededor de ocho Megabytes.

Todavía no hemos aclarado, como por ejemplo es posible, almacenar texto en el computador, si éste sólo puede operar con números.

En ésta cuestión, los computadores se comportan de forma muy humana. Igual que nosotros, cuando traducimos un idioma extranjero, ellos también utilizan un diccionario. En el interior del computador hay una tabla, en la que se asignan

valores numéricos a cada una de las letras, y a los caracteres restantes. Cuando se introduce una letra en el computador, éste busca en la tabla el valor binario correspondiente. Con dicho valor, el computador puede trabajar sin problemas. En la salida, el asunto funciona exactamente al revés.

Para que ésto no sólo funcione razonablemente en el computador de un fabricante determinado, hay que llegar a un acuerdo, sobre la "tabla de traducción". Desgraciadamente - según el punto de vista europeo - dicho acuerdo se realizó en América, por lo que también sólo tiene en cuenta los caracteres que allí se usan. Se denomina ASCII (American Standard Code for Information Interchange = código estándar americano para intercambio de inforación) y precisamente no incorpora los signos especiales españoles o alemanes. El código ASCII contiene 128 caracteres, de los cuales cada uno posee su propio valor. En el apéndice encontrará una tabla con dichos valores.

Ahora bién: ¿qué puede hacer el computador con una memoria llena de valores? Nada. No es tan inteligente como para poder hacer algo con estos datos. Para que algo suceda, primero Vd. tiene que darle la instrucción correspondiente. Puede ordenarle por ejemplo, que dé al texto almacenado en su memoria, un margen derecho uniforme, o que simplemente encienda el alumbrado. Sólo cuando se le suministran comandos sensatos, el computador puede hacer algo. Al nivel más bajo, ésto ocurre así: ¡toma el bit del registro "a", súmalo al bit del registro "b", y deposita el resultado en el registro "c". En la Unidad Central de Proceso (UCP), que es el cerebro calculador del computador, se almacena toda una serie de estos comandos. Combinando un gran número de dichas operaciones, pueden llegarse a realizar procesos complicados. Ello es precisamente la tarea de la programación en código máquina, o en ensamblador. Se trata de lenguajes de programación, que controlan directamente las funciones internas de la unidad cental de proceso. Basic, Pascal o Fortran sólo controlan la UCP, despúes de la traducción interna del lenguaje correspondiente al código máquina.

Para resolver una tarea especifica, hay que indicarle al computador cada paso con todo detalle. Por regla, ésto se realiza mediante un programa, que no es otra cosa que una secuencia de instrucciones que se ejecutan una tras otra. El

programa como tal, naturalmente también tiene que guardarse en la memoria para que pueda ser ejecutado, y en consecuencia también ocupa espacio.

Según las distintas necesidades, un programa puede ocupar entre diez y 120 Kbyte. En ocasiones, el programa ocupa tanto espacio en la memoria, que ya no sobra sitio ni para ejecutarlo, ni para los datos del usuario. Ello significa, que el programa es demasiado grande para este computador, y por lo tanto no puede ejecutarse. Pero estos casos más bién son raros. Ya es más frecuente, que los datos del usuario no quepan todos en la memoria de trabajo del ordenador.

I.8 Almacenamiento masivo

No es muy frecuente la necesidad de tener todo un programa completo a la vez en la memoria. Normalmente es suficiente que la memoria de trabajo contenga las partes importantes del mismo, de modo que el resto sólo se cargue en caso de necesidad. Por ésta razón, los computadores, además de su memoria interna de trabajo, también suelen disponer de medios de almacenamiento externo, que no están sujetos a limitaciones de espacio tan importantes, y que encima son más baratos. Estos dispositivos de almacenamiento masivo pueden ser: una unidad de cassette, una estación de disco Floppy, o una placa fija. Los tres tienen sus ventajas e inconvenientes, pero todos ofrecen abundante espacio para los datos, que además pueden intercambiar con el computador en un tiempo admisible.

I.8.1 Floppy

El Floppy, también llamado disco Floppy o simplemente disco, es una especie de disco, hecho del mismo material que las cintas magnéticas. Se trata de un disco fino, con una superficie magnetizable, y que viene dentro de una funda, que le protege de la grasa de los dedos del usuario, o de las partículas de polvo y de la suciedad. Cuando el Floppy se inserta en el computador, una cabeza de lectura/escritura recorre la superficie del disco, magnetizando, o no, determinados lugares sobre la misma.

De nuevo nos encontramos con los dos estados: estado magnético, y estado no magnético. Ello concuerda con el método de almacenamiento del computador, y ésto con toda la intención del mundo. Los datos del computador, que allí se almacenan en forma eléctrica, pueden de esta manera transferirse al Floppy. Cuando un bit se encuentra en estado de carga, la cabeza de escritura del Floppy, magnetiza un pequeño punto sobre la superficie del disco, mediante una descarga eléctrica. Si el siguiente bit no contiene ninguna carga, tampoco se enviará carga alguna al Floppy. A partir de la secuencia de puntos magnetizados y no magnetizados del Floppy, ahora puede leerse un programa, o una serie de datos, los cuales nuevamente pueden transferirse al ordenador. De este modo hemos creado la posibilidad de almacenar datos en

el exterior.

El disco gira con 200 a 300 rotaciones por minuto, con lo que dejaría atrás un terrible caos de datos, si éstos no fuesen escritos sobre él, siguiendo algún órden o esquema concreto. De modo que el disco se divide en un determinado número de pistas adyacentes, comparables a los carriles de una autopista.

Cuando deseamos almacenar datos, la cabeza de escritura se coloca sobre la pista correspondiente, y comienza a escribir. Sin embargo una pista todavía es bastante larga, y por lo tanto habría que esperar un buen rato, hasta que el computador recuperase todos los datos de dicha pista.

Para acortar este plazo, el disco se vuelve a dividir en sectores (según el fabricante entre 128 bytes y 1024 bytes, en el Amstrad 512 bytes), lo que le da el aspecto de una tarta cortada en pedazos.

Este método de almacenamiento es bastante práctico, pero entre otras cosas impide que, para leer datos, simplemente podamos introducir el disco de un computador en la unidad de disco de otro computador. Cada fabricante va a la suya, e inventa su propio método, para fastidiarnos a nosotros los usuarios. Las empresas Apple y Victor son especialistas en este terreno.

Cuando Vd. compra discos nuevos vírgenes, generalmente primero tendrá que formatearlos. A este efecto, cada computador ofrece un programa especial, que escribe sobre el disco todas aquellas pistas y sectores de los que Vd. necesita disponer.

Si quiere rabiar un poco, formatee por ejemplo el disco que contenga sus datos más valiosos. Se admirará de lo poco que queda de ellos. Por lo tanto tenga cuidado, y aplique una protección sobre la ranura lateral del disco. Los diminutos adhesivos que vienen con cada caja de discos, pueden salvarle la vida. En los discos de 3 pulgadas, como los utiliza el computador Amstrad, puede desplazarse un pequeño corredor de plástico, lo que imposibilita al computador la escritura sobre este disco.

También es una buena idea, hacer una copia de seguridad de todos los datos importantes, y guardarla en un lugar

inaccesible a niños, perros, gatos, fuego, ladrones, tormentas de nieve, terremotos y demás desgracias. Aunque no guarde sus diamantes en la caja fuerte - ¡hágalo por lo menos con sus copias de seguridad! En el próximo capítulo hallará la forma de confeccionar copias de seguridad y otras copias, con ayuda de CP/M.

Actualmente el mercado ofrece los más diversos formatos de disco. En parte, todavía puede encontrarse el viejo y respetable disco de 8 pulgadas, que en su momento inició la marcha triunfal del procesamiento electrónico de datos. Gozan de especial difusión los discos de 5 1/4 pulgadas, que suelen utilizarse en ordenadores personales. El formato es moderadamente manejable, y entretanto un disco de este tipo ya casi admite dos megabytes.

Especialmente desde El Japón nos vienen también discos de 3 y de 3 1/2 pulgadas, que a su vez prometen una manejabilidad todavía mejor, ya que la apertura de escritura/lectura está protegida por una lengüeta de metal, siempre que el disco esté fuera de la unidad. Además los discos de 3 pulgadas disponen de una funda rígida de plástico, de manera que sólo es posible doblarlos a la fuerza.

I.8.2 Disco duro

En los computadores de uso profesional todavía pueden encontrarse los discos duros, o memorias de placa fija. Estos objetos tienen las mismas dimensiones que un disco de 5 1/4 pulgadas y ocupan el mismo espacio en la carcasa de un ordenador (como que la tendencia hacia la miniaturización no para ante nada, ya existen también unidades para disco duro de 3 1/2 pulgadas). En su interior, éstas placas de metal revestidas giran a una velocidad de unas 3000 revoluciones por minuto, y las cabezas de escritura/lectura se desiplazan por la superficie a vuelo bajo. Una partícula de polvo, o de humo sobre la superficie, tendría el mismo efecto sobre la cabeza, que el Montblanc sobre un Jumbo volando a insuficiente altura.

Es por ello que los discos duros se suministran dentro de una carcasa herméticamente sellada, de la cual no deberían sacarse para nada. La capacidad estándar, desde principios de los ochenta, se sitúa en el órden de los 10 megabytes. Pero los precios bajan, y la placa fija de 20 megabytes poco a poco hace su entrada en el mundo de los ordenadores.

A parte de su capacidad elevada, la placa fija ofrece otra ventaja escencial: una velocidad de acceso hasta 20 veces superior a la de un disco.

Bién, si no me falla la cuenta, ya hemos hablado sobre todas las partes que componen un computador de verdad, y Vd. ahora debería tener las nociones básicas sobre ésta útil, y algunas veces divertida máquina. El próximo capítulo le revelará algo acerca del sistema operativo, y acerca de la misión de tan extraño dispositivo.

I.9 Lo que ahora ya sabe

* Ahora conoce los componentes de un computador, desde el teclado hasta la pantalla

* Conoce los diferentes tipos de impresoras, sus ventajas e inconvenientes.

* También sabe lo que es un medio de almacenamiento

* Conoce los números uno y cero, es capaz de contar en binario, y sabe como se almacenan dichos valores.

II. El sistema operativo

El capítulo anterior le ha proporcionado algunas nociones sobre lo que es el hardware, es decir, sobre los elementos fijos de un computador. Si toma solamente el valor material de un computador, este aparato no vale gran cosa. La razón: prácticamente no se puede hacer nada con él. El computador sólo se convierte en una máquina útil, si puede ejecutar en ella programas que para Vd. sean importantes. Para realizar ésto, Vd. necesita lo que se llama software. Sin embargo también aquí tenemos que distinguir entre conceptos diferentes.

Para no perder claridad, nos ocuparemos de dos tipos de software: el sistema operativo, y los programas. Debería tener presente, que naturalmente el sistema operativo también es un programa. La diferencia a la que yo me refiero, se deriva de la función de los diversos software, y con gusto se la voy a explicar.

A menudo escucho a personas, sin experiencias con ordenadores, que preguntan: "¿Con éste también puedo procesar textos?" o algo similar. Estas personas todavía no han entendido la diferencia que hay entre un programa, y un sistema operativo, por lo que ahora sigue una explicación, por si a Vd. también se le haya ocurrido alguna vez, preguntar algo semejante.

Imagine que no estamos tratando de computadores, sino de la vida misma. En esta vida, un bocadillo es una cosa totalmente normal. Un bocadillo suele componerse de pan, de una base (tomate, mantequilla etc..), y del contenido, que se le ponga dentro, y que además debería ser algo bien sabroso. Este bocadillo nos viene que ni pintado, para esclarecer la diferencia que hay entre el sistema operativo y un programa.

Lo que siempre hace falta, es el pan. Volviendo a los ordenadores, el pan es el ordenador. Para el bocadillo también es absolutamente indispensable el tomate, o la mantequilla (para que no haga tanto ruido cuando se lo coma). Volviendo al computador: si Vd. tiene un computador con el que quiera hacer algo, Vd. necesita un sistema operativo. Lo que Vd. ponga dentro de su bocadillo, es cuestión de su gusto o de su apetito. Lo mismo pasa con el computador. El programa que Vd. utilice para jugar en la pantalla, o para procesar textos, o para hacer cualquier otra cosa, es una cuestión que depende exclusivamente de sus necesidades y de sus gustos personales.

Ahora ya se percatará de que la pregunta anterior simplemente estaba mal formulada. Porque: si tenemos un computador, funcionará cualquier programa que haya sido escrito para este tipo de ordenador, y bajo este sistema operativo.

II.1 ¿Qué es un programa?

Básicamente cada programa es una secuencia de instrucciones escritas, que se ejecutan por el ordenador una tras otra. El computador necesita recibir las instrucciones en un lenguaje que pueda entender, y procesar. Dado que él mismo es una máquina, dicho lenguaje se denomina "lenguaje máquina" o "código máquina". Los programas escritos en código máquina resultan incomprensibles para personas "normales". Sin embargo, el computador se alegra. Dado que este tipo de lenguaje es el que mejor entiende, por así decir, su idioma materno, también ejecutará las instrucciones con especial rapidez.

Para facilitar el trabajo a las personas "normales" existen los así llamados lenguajes de alto nivel o lenguajes de programación. Seguramente conocerá algunos. Son especialmente conocidos el BASIC, o también el COBOL, o FORTRAN, o PASCAL, o también el nuevo MODULA 2. En estos lenguajes de alto nivel, los comandos son palabras compensibles para los humanos, que dentro del computador se traducen a código máquina, ejecutándose a continuación.

Un programa no es más que una secuencia de comandos que le dicen al computador exactamente, y paso a paso, qué es lo que tiene que hacer, y cómo lo tiene que hacer. La longitud de un programa puede abarcar desde dos líneas, hasta casi un número infinito de líneas. Esto simplemente depende de la capacidad de almacenamiento de la memoria principal del computador, y de las unidades de almacenamiento externo.

Sin embargo no es suficiente, que un programa satisfaga las necesidades del usuario, posibilitando por ejemplo, el tratamiento de textos. También tiene que ocuparse internamente de toda una serie de tareas secundarias, para que todo funcione. Debe determinar por ejemplo qué tecla ha sido pulsada, qué carácter hay que representar en la pantalla o enviar a la impresora, si hay datos necesarios en el dispositivo de almacenamiento externo, y dónde se encuentran, o cargar datos del disco, a la memoria de trabajo del computador.

Posteriormente hay que asegurar, que los datos leídos del disco, vayan a parar en el lugar correcto de la memoria de trabajo, y que en el disco haya suficiente sitio para una nueva grabación. Además hay que confeccionar un directorio del disco, etc., etc.

II.2 Subrutinas

Como Vd. puede ver, hay un montón de trabajo a parte. Cada vez, y para cada programa, hay que programar este "trabajo aparte" nuevamente. En cada parte del programa hay que determinar exactamente, qué es lo que ha de suceder en el ordenador. Lo que además dificulta el trabajo, es que estas tareas han de solucionarse de forma diferente para cada tipo de computador, por mucho que éstos se parezcan. Siempre hay una pequeña diferencia entre cada computador. Si por lo tanto se escribiera un programa largo de una tirada, habría que modificarlo para cada tipo de computador, para que se ejecute correctamente.

Es posible ahorrar este trabajo innecesario, si se divide el programa en un programa principal, y varias subrutinas. Las subrutinas sólo se cargan en la memoria de trabajo, cuando realmente se necesitan.
De modo que para adaptar un programa a otro tipo de computador, solamente hay que eliminar el pequeño bloque que contenga la subrutina, reemplazándolo por otra subrutina nueva, y el programa entero ya puede ejecutarse en el nuevo computador, sin ningunas limitaciones. Otra ventaja de la técnica de subrutinas es, que el programa principal ya ni siquiera sabe como se soluciona una determinada tarea secundaria. Simplemente envía una órden, como por ejemplo "imprimir texto", y la subrutina se hace cargo de la ejecución. El hecho de que el texto se imprima en la pantalla, o a través de una impresora, o por un telex, le es completamente indiferente al programa principal. Solamente con el envío de la órden, el trabajo se ejecuta.

Este método funciona, si la transferencia de datos hacia la subrutina se estandariza. Puede imaginárselo como una carrera de estafetas, en donde el corredor que releva al corredor anterior siempre espera en un lugar previamente establecido, que es donde recibe la estafeta. En la jerga informática, dicho lugar de transferencia normalizado y exactamente definido, se denomina interface. En dicho interface, una subrutina prácticamente puede cortarse y reemplazarse por otra nueva, sin que por ello se resienta la funcion primordial del programa principal, e incluso sin que el programa principal tan siquiera note el cambio. Todo vuelve a concordar perfectamente, y funciona sin problemas.

Si lo pensamos bién, no tiene sentido que cada programador tenga que volver a incluir cada vez las operaciones internas en su programa. Ello supone mucho trabajo, tiempo y dinero. Esto es lo que también pensaron los señores de DIGITAL RESEARCH, por lo que se pusieron a desarrollar un programa estándar para microcomputadores. Reunieron las subrutinas que más frecuentemente se necesitan para el control interno, definieron los puntos de enlace, y la forma de transferencia de datos entre programa principal y programa operativo, y así crearon un sistema operativo común para los más diversos tipos de ordenador.

Naturalmente que dicho sistema sólo funciona en computadores con unidades centrales similares o idénticas, dado que las instrucciones en código máquina del programa tienen que entenderse, para que puedan ser ejecutadas. El sistema operativo, que por primera vez pone a disposición todas estas numerosas operaciones internas de forma estandarizada, se denomina CP/M. Dicha abreviación figura en lugar de "Control Program for Micro-processors", lo que en español significa, programa de control para microprocesadores. Este sistema operativo trabaja con microprocesadores del tipo 8080, 8085 o Z80.

II.3 Las tareas de CP/M

Principalmente, CP/M se hace cargo de las siguientes tareas básicas: introducción de caracteres, expedición de caracteres, administración de la memoria disponible sobre el dispositivo de almacenamiento masivo, lectura de grabaciones en disco, y escritura de nuevas grabaciones sobre disco, u otro dispositivo de almacenamiento masivo. Estas rutinas de servicio pueden utilizarse en todos los programas de aplicaciones, de forma unificada y estandarizada.

Sin embargo, para que todo vaya bién, CP/M tiene que ejecutar dos tipos diferentes de trabajo, por lo que se divide en dos grandes bloques: la primera parte, el BDOS (Basic Disk Operating System = sistema operativo básico para disco) se encarga de todas las tareas que independientemente del sistema de computadores, siempre se tengan que resolver de la misma manera. La segunda parte del sistema operativo, el BIOS (Basic I/O-System = sistema básico para entrada y salida) contiene todas aquellas partes del programa, que hayan sido adaptadas de acuerdo a las necesidades de un determinado sistema computerizado.
Lo que sucede es, que cada modelo de ordenador tiene sus propios puntos de vista y métodos, acerca de cómo solucionar determinados problemas, de modo que no se puede tomar el sistema operativo de una máquina y ejecutarlo sin más en otra. Antes de que funcione un sistema operativo adoptado de esta manera, hay que acondicionar el BIOS para la nueva máquina. Normalmente no debería tener dificultades con ello en su computador, ya que cada computador se suministra con un sistema operativo acondicionado, y a su medida (esperémos).

II.4 CP/M y las diferentes versiones

Todo programa que sobreviva por algún tiempo en el mercado, también está sujeto a mejoras, volviendo a lanzarse al mercado en una nueva versión. Por mucho que se le dé vueltas, no hay programa perfecto y exento de fallos, y alguno de estos fallos sólo se descubren cuando el programa ya lleva algún tiempo utilizándose, habiendo agotado realmente todas las posibilidades que ofrece.
Si el fabricante de dicho programa recibe suficientes quejas sobre determinados fallos, algún dia decidirá corregirlos, y lanzará al mercado una nueva versión del mismo. El programa CP/M en la versión 2.2 puede considerarse casi como carente de fallos, y prácticamente es el sistema operativo estándar para todos los ordenadores de 8 bits. Por tanto, la versión más reciente CP/M 3.0, no es una versión nueva que elimine viejos errores, sino que es una versión que ha experimentado claras mejoras en sus rendimientos, y que incluye además algunos comandos nuevos. Además CP/M ha tenido que adaptarse a los cada vez mas sofisticados computadores, y a sus respectivas tecnologías.

II.5 El prompt de CP/M

Bién, creo que con ésto es suficiente. Dejemos la teoría a un lado, y hablemos de la vida real, es decir, del ordenador, y de su sistema operativo. Ahora lo que necesitará, es un computador en condiciones de funcionar, una unidad de disco Floppy, y un disco con el sistema operativo CP/M. Si lo tiene todo, entonces puede activar su computador, introducir el disco que contenga el sistema operativo CP/M en la unidad "A" o "1", y cerrarla. Este es el procedimiento, que debería tener siempre presente, ya que si Vd. activa o desactiva el computador, mientras se encuentre un disco en la unidad, puede suceder que a causa del golpe inicial de corriente, se produzcan movimientos indeseados de la cabeza de lectura/escritura, con lo que su valioso disco con toda seguridad quede inservible.

Bién - ahora Vd. ha conectado su ordenador, ha introducido el disco del sistema CP/M, y ha puesto en marcha el proceso de inicilización, según las descripciones del manual. En el Amstrad CPC ello se hace, conectando (o desconectando y

volviendo a conectar) la máquina, e introduciendo el comando:

ICPM

La bonita raya vertical se consigue, pulsando las teclas SHIFT y la tecla de la "algarroba" conjuntamente.

A continuación la unidad de disco empieza a funcionar, emitiendo unos ruidos muy interesantes, a la par que se enciende la lamparilla de funcionamiento, y al poco rato aparece un mensaje en la pantalla. Este mensaje de presentación, o inicial es un poco diferente en cada computador. La razón es que este mensaje es emitido por la parte BIOS del sistema operativo, es decir de aquella parte, que tiene que ser acondicionada especialmente para cada ordenador. Cada fabricante de ordenadores, adapta esta parte a su máquina, elaborando un mensaje de presentación según sus propias ideas. Un mensaje de presentación usual en CP/M es éste:

CPM 2.2 - AMSTRAD Consumer Electronics plc.
A>

En el CPC 6128 este mensaje también es algo diferente, ya que aquí se trata del mensaje de presentación de CP/M 3.0.

CP/M Plus Consumer Electronics plc.
V 1.0, 61K TPA, 1 (2) Disc drive

El mensaje de presentación significa para Vd., que el sistema operativo CP/M ha sido cargado correctamente en la memoria de trabajo del ordenador. Acto seguido, CP/M se presenta con un mensaje, que indica que está preparado para recibir órdenes. Este mensaje se denomina 'prompt' del sistema operativo. Dicho prompt se presenta de esta forma:

A>

La "A" mayúscula indica al usuario, que trabaja con la unidad de disco "A", o "1" de su sistema, y el signo ">" es el propio mensaje de disponibilidad de CP/M. En la línea detrás del mensaje CP/M, se espera la entrada de un comando. Bién. No dejemos que CP/M se impaciente, y escribamos:

A>ABCDEFGH

Ahora el cursor se encuentra detrás de la última letra, y a parte de ésto, no ha pasado nada, nada se ha movido. La razón: CP/M todavía no sabe si hemos terminado con la introducción del comando, o si quizás queremos añadir algo más. Por lo tanto, para finalizar una introducción, tenemos que comunicar a CP/M de alguna manera, que ya hemos terminado.

La introducción de una línea en el computador, igual que en una máquina de escribir, se finaliza con un retorno de carro. En los computadores, esta tecla muy a menudo lleva la denominación RETURN o ENTER, lo que equivale a una abreviación de CARRIAGE RETURN o de "entrada". Cuando Vd. haya pulsado esta tecla, el computador sabe, que Vd. ha terminado con la introducción de su instrucción, y comienza con la ejecución. Ahora pulse la tecla 'retorno de carro', o 'RETURN' y verá:

A>ABCDEFGH
ABFG?
A>

Bueno, ¿qué es lo que ha pasado? CP/M ha leído la línea, ha entendido nada, y así nos lo comunica. El sistema operativo nos avisa cuando no comprende algo, repitiendo lo que no ha entendido, y acompañándolo con un interrrogante. El interrogante significa algo así como: "¿Esto es una tomadura de pelo, o qué? Esta instrucción no me dice nada."
Así que el primer intento ha fracasado. Probemos, si quizás es debido a las letras mayúsculas. Introduzca la misma "órden" de antes, sólo que en letras minúsculas:

A>abcdefgh

La respuesta es:

A>abcdefgh
ABCDEFGH?
A>

El sistema operativo por tanto tampoco ha comprendido la instrucción escrita en minúsculas, pero ha hecho otra cosa. ¿Vd. se ha dado cuenta que la instrucción se ha repetido en mayúsculas, a pesar de haberla introducido en minúsculas? Esto es una particularidad de CP/M que en algunos casos incluso tenemos que tener en cuenta intencionadamente. CP/M primero transforma todas las letras que Vd. introduzca, en mayúsculas, para poder tratarlas a continuación.

Para que vea, que CP/M realmente trabaja, ahora introduciremos una instrucción, que provoque algo de verdad. Anteriormente Vd. ha leído, que una de las tareas del sistema operativo es, llevar un directorio de los ficheros que contenga el disco. A tal efecto CP/M crea un directorio (directory en inglés), que puede mostrar a petición.

Si queremos ver este directorio, tenemos que comunicarle al sistema operativo de alguna manera, que deseamos ver el directorio, y que nos lo muestre en la pantalla. El comando se compone de las tres letras DIR, lo que es una abreviación de directory. Ahora introduzca dicho comando, seguido de ENTER:

A>DIR

Ahora Vd. verá en la pantalla, el directorio de su disco del sistema operativo. ¡Obsérvelo con atención! En el extremo izquierdo se indica la unidad de disco, que en este caso es "A:". A su lado figuran nombres de ficheros, y después de algunos espacios, la categoria del fichero. A menudo también podrá ver el sufijo COM, por ejemplo en DDT.COM, PIP.COM, ASM.COM. Este código informa sobre el contenido del fichero. Para nosotros, de momento son especialmente interesantes los ficheros COM, ya que contienen programas que pueden ejecutarse inmediatamente.

COM es la abreviación de la palabra inglesa "command", que significa comando o instrucción. Los ficheros así denominados contienen comandos, que pueden ejecutarse instantáneamente. Si Vd. teclea el nombre de un programa de este tipo, sin el sufijo COM detrás del prompt del sistema operativo, dicho programa se carga directamente en la memoria de trabajo del computador, y se ejecutará sólo. En consecuencia, basta teclear PIP, seguido de un RETURN, para que el programa se cargue y se autoejecute.

II.6 Más vale prevenir

Si Vd. todavía está trabajando con su disco original, tenemos que ocuparnos rápidamente en confeccionar una copia de seguridad. Debería adquirir la sana costumbre de no trabajar nunca con los originales de un programa, sino confeccionar una copia del programa, y guardar el original en un sitio seguro. Como cabe de esperar de un sistema operativo que pretenda facilitar el trabajo al usuario, CP/M posibilita la confección de copias de seguridad.

II.7 Lo que ahora ya sabe

* Conoce una explicación sencilla de las tareas, que ha de cumplir un sistema operativo

* Sabe lo que es un programa, y para lo que se necesitan subrutinas

* También conoce las tareas de CP/M

* Vd. sabe lo que es el prompt de CP/M, y como ver el directorio de sus discos.

* Vd. se ha propuesto firmemente confeccionar, como mínimo una copia de seguridad de cada disco importante

III. Trabajar con CP/M

III.1 El disco del sistema

Le he recomendado hacer copias de sus valiosos discos originales, para que en un caso dado, su original no sufra ningún daño. Ahora le voy a enseñar como se copia un disco completo, utilizando comandos CP/M, que se explicarán más tarde. Necesitará dos de los programas del disco de CP/M, para poder hacer la copia completa de un disco. Si vuelve a introducir DIR y examina el directorio, entre otros encontrará dos programas de nombre:

SYSGEN.COM y **PIP.COM** en CP/M y

COPYSYS.COM y **PIP.COM** en la versión 3.0

Sin embargo con la empresa Amstrad la cosa es un poco diferente en lo que respecta a CP/M 3.0. El fichero COPYSYS.COM no se encuentra en ninguno de las tres caras de los discos. El programa para copiar el sistema COPYSYS, está integrado en el paquete de programas DISCKIT3. Aunque este paquete de programas sea muy confortable, con permiso voy a decir, que seguramente no habría sido tan difícil incluir también el fichero COPYSYS.COM. Además falta el fichero COM para formatear un disco. En el texto que sigue, describiré el procedimiento "normal" con CP/M, es decir, como si los dos ficheros COM estuvieran incluidos en los discos del sistema. En el capítulo 7, bajo FORMAT y COPYSYS hallará la forma de aplicar estos comandos con el Amstrad.

Con estos dos programas, Vd. hace cosas diferentes. Con el programa SYSGEN (COPYSYS), Vd. copia el sistema CP/M en las primeras dos pistas de cada disco. Por ello, estas dos pistas también se denominan pistas del sistema. Estas son las que se leen primero al activar el ordenador, cargándose dicho programa en la memoria de trabajo. Como usuario, Vd. no tiene acceso directo a las pistas del sistema, por lo que tampoco encontrará ningún CP/M.COM o algo por el estilo, ni en el disco, ni en el directorio. Sólo debe saber que las pistas del sistema siempre están reservadas a CP/M, y que contienen todas las partes necesarias del programa, para un buen funcionamiento del ordenador.

Si de lo contrario, Vd. desea copiar un disco completo, es decir con todos sus programas, incluyendo el sistema operativo CP/M, entonces sí que necesitará una posibilidad de acceder a estas dos pistas del sistema. Esta posibilidad la ofrece el programa SYSGEN (CP/M 2.2), o el programa COPYSYS, cuando se trata de CP/M 3.0. Ambos son programas especiales, que posibilitan la transferencia del contenido de las pistas del sistema, de un disco a otro.

Antes de empezar con el trabajo, debería conseguir un disco formateado y vacío (un disco se formatea mediante el comando FORMAT que ofrece CP/M, introduciendo dicho comando detrás del prompt de CP/M, y siguiendo las instrucciones que aparezcan en la pantalla).

III.2 Copiar con una unidad

Cuando tenga todo preparado, introduzca su disco original CP/M en la unidad A, cerrándola a continuación. Si sólo trabaja con una unidad, tendrá un poco más de trabajo, que con dos. Pero también puede hacerse, si se siguen los pasos siguientes. Para copiar su sistema operativo, debe introducir el disco original en la unidad y teclear:

A>SYSGEN (ENTER)

A continuación aparece el mensaje:

Please insert SOURCE disc into drive A
then press any key:

en donde SOURCE se refiere a su disco original, o aquel del que se deba tomar el sistema operativo. Después de introducir dicho disco en la unidad y pulsar ENTER, aparece otro mensaje:

Please insert DESTINATION disc into drive A
then press any key:

Ahora debe introducir el nuevo disco en la unidad, la cual efectuará su trabajo, después de que haya pulsado ENTER. Esto no tardará mucho y pronto verá:

Do you wish to reconfigure another disc (Y/N)?

Si responde afirmativamente mediante Y, podrá repetir el proceso para otro disco más. Si teclea N, en representación de 'no', retornará al prompt del sistema.

III.3 Copiar con dos unidades y CP/M 3.0

Si trabaja con dos unidades, el proceso es el mismo, sólo que Vd. estará copiando de una unidad a otra. Coloque el disco vacio en la unidad B y cierre la portezuela. Después de activar el sistema operativo, teclee COPYSYS (SYSGEN en CP/M 2.2) detrás del prompt de CP/M, y luego pulse ENTER.

A continuación los dos programas se presentarán con su nombre, y su número de versión, pidiendo el nombre de la unidad, de la que se debe copiar. Dado que su disco CP/M se encuentra en la unidad A, introduzca una "A", a lo que se le preguntará inmediatamente, si el disco que hay en la unidad A, es el correcto. Si es así, pulse ENTER, y el cuestionario continuará.

Lo próximo que debe hacer, es indicar la unidad en la que se encuentra su disco virgen, pero ya formateado. Tampoco aquí necesitará confirmar mediante ENTER, ya que el programa enseguida querrá saber si el disco realmente está allí, pidiendo esta vez expresamente un ENTER para la confirmación. Cuando haya seguido todas las instrucciones, y haya introducido el último ENTER, la unidad B se pondrá en marcha, y el sistema operativo CP/M se grabará sobre el nuevo disco. Acto seguido se formulará la pregunta, si también desea copiar CPM3.SYS. Se trata de importantes programas auxiliares, que debería incluir en cada copia de su disco del sistema. Cuando la unidad B se haya parado, el trabajo habrá concluido, y Vd. podrá introducir el próximo disco para la transferencia del sistema operativo. Pero como pretendemos hacer algunas cosas más con nuestro disco del sistema, pulse simplemente ENTER, para finalizar el programa copiador del sistema.

III.4 Mostrar directorio

Vd. acaba de copiar algo de A a B, y está curioso por averiguar lo que ahora contiene el disco B. Para que no reviente de curiosidad, introduzca detrás del prompt de CP/M:

DIR B

Si no se ha olvidado del espacio después de DIR, verá como la unidad B se pone en marcha, enviando un mensaje a la pantalla. Mediante DIR B ha dado instrucciones a CP/M de mostrar el directorio del disco de la unidad B. Dado que hasta ahora Vd. sólo ha copiado el sistema operativo, pero todavía no ha copiado ningún ningún fichero, aparecerá el mensaje NO FILE. FILE en español se traduce por fichero, y por lo tanto NO FILE significa que no hay ningún fichero grabado en el disco. Como recordará, los programas de las primeras dos pistas del disco, siempre quedan ocultos. Vd. no tiene acceso directo a dichas pistas, y por lo tanto tampoco verá nada en el directorio.

III.5 Copiar con PIP y dos unidades

Dado que deseamos copiar íntegramente el disco CP/M, y no sólo el sistema operativo, ahora utilizaremos otro programa. Este segundo programa tiene el larguísimo nombre de 'procesor para intercambio de datos entre unidades periféricas', lo que en inglés se llama Peripheral Interchange Processor, y abreviado se convierte en un gracioso PIP. Lo que hace el programa, sin embargo es algo menos gracioso, y además tremendamente útil.

Si todavía tiene los dos discos en las unidades, teclee PIP y pulse ENTER. Ahora el computador carga PIP en su memoria de trabajo, lo que se indica mediante un PROMPT, de esta forma:

```
A>PIP
*
```

El asterisco le indica, que PIP está preparado para aceptar instrucciones. Este programa es extremadamente útil y potente. Conocerá las numerosas posibilidades de PIP con más detalle en el capítulo 6.

Antes de dar un comado concreto a PIP, pensemos primero en lo que queremos que haga. Los ficheros del disco de la unidad A, deben transferirse al disco de la unidad B. También podría decirse, que "B" recibe todos los ficheros grabados en "A".

Debería mirar bién esta última frase, y acordarse de ella. Es decisiva para el trabajo con PIP. Cuando todos los ficheros hayan sido transportados de A a B, el contenido de los dos discos es idéntico. Por ello, en las instrucciones de PIP se utiliza el signo 'igual a'. Solamente queda la dificultad de cómo decirle a PIP que debe copiar todos los ficheros, y no sólo un fichero cualquiera.

Este caso está previsto. Un asterisco en lugar de un nombre de un fichero, para PIP significa "todos los ficheros". El asterisco puede concebirse como un signo de sustitución. Dado que los ficheros se distinguen por un lado mediante su nombre, y por otro mediante su código, que son las tres letras que siguen al punto, para copiar todos los ficheros hay que introducir un asterisco/punto/asterisco. En la

pantalla, ello cobra el siguiente aspecto:

B:=A:*.*

Como queremos estar seguros, de que todos los ficheros se copien correctamente, aprovecharemos para verificar cada uno de ellos. Podemos delegar este trabajo a PIP, añadiendo una "V" entre corchetes a nuestra instrucción. "V" representa "Verify", lo que significa comprobar, verificar. Sin embargo, para que PIP entienda el comando correctamente, éste tiene que ponerse entre corchetes.
Si ha introducido el comando completo,

A>PIP
***B:=A:*.COM**

verá el siguiente directorio de CP/M 3.0 en la pantalla:

```
A:   CCP  COM : DATE  COM : DEVICE  COM   :   DIR  COM    :
DUMP  COM
A: ED COM : ERASE COM : GENCOM COM : GENCPM COM : GETCOM
A: LIB COM : LINK COM : MAC COM : PATCH COM : INITDIR COM
A: PUT COM : RENAME COM : RMAC COM : SAVE COM : SET COM
A:  HEXCOM COM : SETDEF COM : SHOW COM : SID COM  :  SUBMIT
COM
A:  TYPE COM : XREF COM : COPYSYS COM : FORMAT COM  :  HELP
COM
A: PIP COM
```

Como puede ver, durante el copiado, PIP le indica el nombre del fichero que está copiando. Una vez concluido el trabajo, PIP vuelve a presentarse con el asterisco. Si no tiene más trabajo para PIP, y creo que de momento no tenemos nada más para él, pulse ENTER, y volverá a presentarse el sistema operativo con su A-prompt.

Si no confía demasiado en la técnica, puede controlar mediante DIR B:, si realmente se han enviado todos los ficheros a la unidad B. Si todo está en órden, saque su disco original de la unidad A, métalo en la funda, y guárdelo en un lugar bién protegido. En un futuro sólo trabajará con la copia, que acaba de confeccionar, y así se ahorrará muchos disgustos en caso de que algo vaya mal, porque un disco para copiar un sistema nuevo puede costarle alrededor de 400 Pts. (los discos de 3 pulgadas para el Amstrad son un poco más caros), pero un nuevo sistema operativo CP/M, ya sube a varias decenas de miles de Pts.

III.6 Reglas para nombres de fichero

Hasta ahora ya ha oido hablar un par de veces sobre ficheros, y también ha visto algunos nombres de ficheros del directorio en su pantalla, pero todavía no sabe lo que es, y para que sirve un fichero.

Como Vd. sabe, cada fichero para CP/M posee un nombre. Por una parte, ésto es importante para Vd., de modo que sepa lo que contiene, y en segundo lugar es importante para CP/M, para que pueda encontrar el fichero que Vd. le pida. Desgraciadamente los constructores de CP/M han incorporado una limitación, que por ejemplo a mí no me agrada nada.

El nombre de un fichero en CP/M puede componerse de un máximo de ocho caracteres, de un punto, y de otros tres caracteres para el código. Ello significa que Vd. tiene que encontrar nombres para sus ficheros, que no excedan de ocho caracteres, y que además permitan una conclusión sobre su contenido.
Los caracteres admitidos para un nombre de fichero son las 26 letras del alfabeto, así como el signo de sumar, el guión, la barra, el signo de porcentaje, y el signo del dólar, en cualquier posición dentro del nombre, o dentro del código. Los signos "menor que", "mayor que", "igual a", punto, coma, dos puntos, punto y coma, asterisco, interrogante, los dos corchetes, y el signo de exclamación, no se deberían usar en el nombre de un fichero, ni para su código, ya que dichos signos tienen un significado especial para CP/M, y por lo tanto pueden dar lugar a errores.

A causa de mi larga experiencia con CP/M y otros programas, sólo puedo aconsejarle, que dé a sus ficheros nombres que tengan sentido, y que contengan alguna referencia sobre su contenido. Especialmente si tiene muchos ficheros en su disco, un nombre como XK2512AC.ABC ya no le dirá nada. A parte de ésto, no tiene que tener en cuenta nada más en la elección de un nombre, ya que en un disco no se admiten dos ficheros con el mismo nombre, y con el mismo código. Sin embargo sí que se admiten dos ficheros con el mismo nombre, pero de código diferente.
ssw

III.7 Codificación de los ficheros

Puede crear libremente el código de un fichero, lo que quiere

decir que puede darle un código cualquiera, pero debería tener en cuenta las abreviaciones típicas de CP/M. Al final del capítulo hallará una tabla de las abreviaciones comunes de CP/M y de su significado. Allí por ejemplo encontrará programas con el código COM. Esto es la abreviación de COMMAND y significa, que estos programas contienen comandos, que pueden ejecutarse inmediatamente.
Recordará que PIP y SYSGEN o COPYSYS son programas COM. Además no debería usar nunca un código como BAK o $$$. BAK figura por "back up", lo que significa una copia de seguridad, y se usa por el editor CP/M para designar copias de seguridad de ficheros. El fichero con los tres signos de dólar al final, se entiende como fichero temporal, que ha sido creado por un programa, y que se borrará rigurosamente al finalizar el mismo. Por ejemplo PIP crea este tipo de ficheros temporales, y los elimina completamente del disco, cuando termina su trabajo. Por tanto tenga cuidado.

También es poco aconsejable emplear los mismos nombres para los ficheros, y servirse del código para numerarlos. Ejemplo: Vd. tiene un texto y le da el nombre TEXTO.1 al primer fichero, TEXTO.2 al segundo, y TEXTO.3 al tercero. Lo que de momento parece bastante lógico, después con toda seguridad le causará un disgusto. Porque cuando haya repasado el primer fichero, éste fichero original se convertirá en TEXTO.BAK, y la versión repasada se llamará TEXTO.1. Ahora Vd. repasa el fichero TEXTO.2, y también aquí el fichero original se convierte en TEXTO.BAK, siendo la versión repasada TEXTO.2.

Dado que CP/M no acepta dos ficheros con el mismo nombre y el mismo código en un disco, el sistema operativo, ni corto ni perezoso borra su primer fichero BAK, sustituyéndolo por el segundo. De este modo ya no tiene ninguna posibilidad de volver a ver el texto de su fichero original TEXTO.1.

Para evitar este problema, puede darle al primer fichero el nombre de TEXTO1.TXT, al segundo TEXTO2.TXT, y al tercero TEXTO3.TXT

Si ahora quiere repasar los ficheros, cada uno obtendrá su propio fichero back up, con lo que siempre tendrá la posibilidad de volver a acceder a su texto original.

III.8 Reencontrar un fichero

Si Vd. trabaja mucho con su ordenador creando muchos ficheros, posiblemente pronto tendrá otro problema. No logrará encontrar en el directorio al primer intento, los ficheros que busca. Sin embargo puede servirle de mucho, averiguar cuantos ficheros con el nombre TEXTO.TXT se encuentran ya sobre el disco.

Tiene Vd. dos posibilidades, para buscar estos datos. En las dos versiones del sistema operativo, nos ayudan el asterisco, y el interrogante. Ya conoce al asterisco como signo de sustitución en nuestra operación de copiado con PIP, y el interrogante trabaja de forma similar. Mientras que el asterisco sustituye a todos los caracteres del nombre del fichero o de su código, el interrogante actúa como sustituto de un sólo carácter, en una posición determinada.

La introducción "texto?", por ejemplo busca y encuentra todos los ficheros que lleven por nombre "texto", o "texto" y otro carácter adicional. De esta manera también se encontrarían los tres ficheros "TEXTO1", "TEXTO2", y "TEXTO3". Si figurara otro fichero adicional "TEXTO" en nuestro disco, éste también sería encontrado y también se mostraría. La razón es, que CP/M siempre reserva ocho caracteres para el nombre de un fichero. Si el nombre que Vd. ha introducido es más corto, CP/M rellena con espacios los caracteres que faltan. Los espacios además son igualmente válidos, que las letras y los números.

III.9 Búsqueda con interrogante (?)

Dado que el interrogante puede sustituir a cualquier carácter, no sólo se encontrarán los ficheros "TEXTO1", "TEXTO2", y "TEXTO3", sino también el fichero "TEXTO". Sin embargo el empleo del interrogante no se limita sólo a un carácter. De la misma manera puede poner hasta ocho interrogantes para el nombre del fichero, y hasta tres, para el código. Pero ésto es un trabajo molesto, que Vd. se puede ahorrar, usando un asterisco para el nombre del fichero, y otro para su código. Al sistema operativo le es perfectamente indiferente, que Vd. introduzca ocho interrogantes, o un asterisco. Internamente, de todos modos el asterisco primero se convierte en ocho interrogantes, antes de empezar con la búsqueda.

También puede usar el asterisco para una búsqueda más precisa, si le antepone la, o las letras iniciales del fichero, que está buscando. Si de lo contrario, primero pone el asterisco, y luego las letras, la búsqueda no funcionará. Este método, la introducción de nombres de fichero mediante interrogantes o asteriscos, puede usarse conjuntamente con comandos como DIR, TYPE, STAT, FILECOPY y PIP.

III.10 Lo que ahora ya sabe

* Sabe copiar un disco del sistema, y además guardará muy bién el original

* Sabe operar con el asterisco y el interrogante en sustitución de las letras que falten, facilitándose así su labor.

* Sabe que siempre deben darse nombres razonables a los ficheros, para poder localizarlos más tarde.

IV. Comandos incorporados

En el capítulo anterior hemos conocido algunas funciones básicas del sistema operativo CP/M. Pero con ésto no hemos llegado al final, ni mucho menos. En este capítulo nos ocuparemos más detenidamente de los comandos incorporados de CP/M, y que son los siguientes: USER, DIR DIRSYS(CP/M 3.0), REN, RENAME(CP/M 3.0) y TYPE.

A parte de ello, más tarde nos ocuparemos de los así llamados programas transitorios (que son programas de CP/M, que Vd. puede encontrar en el directorio, y que están almacenados en forma de ficheros COM). Adicionalmente, sus conocimientos sobre los comandos DIR y PIP, que ya ha conocido en el capítulo anterior, se verán ampliados.

IV.1 Comandos, parámetros y opciones

Antes de explicarle otros comandos de CP/M, tiene que conocer la diferencia básica entre un comando, un parámetro, y una opción. Un comando, o una instrucción es una sentencia, que pone a CP/M, en condiciones de ejecutar una acción determinada. Un parámetro, por regla general es el nombre de un fichero, que indica, a cuál de los ficheros se refiere la instrucción. Una opción debe ir entre corchetes, y como ya indica su nombre, puede omitirse. Si de lo contrario se introduce la opción, la función del comando que le precede, se verá modificada de una manera determinada.

Ya ha visto un ejemplo de lo que es una opción, cuando hemos copiado su disco del sistema, mediante PIP. La opción era "V", y daba por resultado, que PIP verificaba todos los ficheros copiados.

Según el comando del que se trate, deben introducirse parámetros. Si en algún lugar faltara un parámetro, que Vd, hubiera omitido, CP/M expresamente pide la introducción de dicho parámetro. Por lo tanto, no tenga temor, que algo no funcione.

Cuando introduzca comandos y parámetros, no debe olvidar de poner como mínimo un espacio entre el comando, y el parámetro. Y éste también es el único lugar, en el que puede figurar un espacio. Ni en el comando, ni en el parámetro, ni en la opción, se toleran espacios.

Normalmente, después de un comando sólo introducirán uno, o dos nombres de fichero como parámetro. Sin embargo, Vd. no está obligado a limitarse a tan pocas indicaciones. Si se le ocurren suficientes cosas, o si así lo requiere su aplicación, también puede rellenar toda una línea, empezar otra nueva mediante control E (^E), y seguir escribiendo allí. CP/M interpretará estas dos líneas, como si fuese una sóla.

IV.2 Los comandos incorporados

Cuando haya cargado CP/M en la memoria de trabajo de su computador, tiene a disposición dos clases de comandos. Los comandos incorporados se encuentran en la memoria de trabajo de su computador, y pueden llamarse con mucha rápidez, para cumplir con las tareas que se les encomiende. Los comandos restantes, figuran en su disco, en forma de fichero, pudiendo listarse, mediante el comando DIR. Estos programas pueden copiarse, o borrarse, como un fichero normal y corriente, y si Vd. dispone de los conocimientos necesarios, incluso los puede modificar.
Se ha realizado esta distribución de CP/M, porque las dos pistas del sistema que contiene cada disco, y que están reservadas exclusivamente a CP/M, no disponen de suficiente espacio, como para albergar todos los programas auxiliares. Para su trabajo con el ordenador, sin embargo da lo mismo, que Vd. se sirva de un comando incorporado, o de un comando transitorio. Si por ejemplo Vd. pide determinadas acciones, mediante el comando PIP, CP/M automáticamente carga el fichero PIP.COM en su memoria de trabajo, y ejecutará el comando.

El único inconveniente de ésto, es la pérdida de tiempo, que se produce a causa de la lectura del fichero. En total, el sistema operativo CP/M 2.2 ofrece cinco, y la versión 3.0 entretanto seis comandos incorporados. Cuatro de estos comandos experimentan una ampliación en forma de un fichero COM, que lleva el mismo nombre. Si en CP/M 3.0 Vd. desea acceder a algunos de estos comandos, no tiene que teclear cada vez el nombre completo. Puede abreviarlo, tal y como se muestra en la tabla de abajo:

comando	abreviación	función
DIR	DIR	muestra el directorio
DIRSYS	DIRS	muestra dir. de ficheros del SISTEMA
ERASE	ERA	borra ficheros
RENAME	REN	modificar nombre de fichero
TYPE	TYP	muestra fichero de texto
USER	USE	modifica zona del usuario

En la versión 2.2 no le servirá el nombre completo de los comandos, ya que el sistema operativo no los entiende. Aquí sólo se puede trabajar con abreviaciones.

Si lo mira bién, existe otro comando incorporado en CP/M, que sin embargo no posee ningún nombre. Puede usar dicho comando para pasar de la unidad A:, a la unidad B:, o a cualquier otra unidad. Si prefiere trabajar con la unidad B, en lugar de con la unidad A, teclee simplemente:

B: (RETURN)

y CP/M convierte a su unidad B, en la unidad en curso.

Si ahora está trabajando con la unidad B, mientras que su disco del sistema se encuentra en la unidad A, a pesar de ello puede acceder a los programas CP/M del disco A. Para ello, simplemente indique el nombre de la unidad, seguido de dos puntos, antes de la instrucción:

B>A:DIR

De la misma manera puede indicar la unidad, delante de un parámetro

B:DIR A:FICHERO.XXX

IV.3 USER

Este programa incorporado le ofrece la posibilidad de dividir un medio de almacenamiento en 16 zonas. La clasificación de las zonas va de 0 a 15. Dicha distribución puede ser muy útil, cuando un computador se comparte con varios usuarios, cuyos ficheros no deben mezclarse. Otra posibilidad consiste, en guardar los correspondientes programas con sus ficheros, en distintas zonas del usuario.

Por ejemplo, podría reservar una zona para sus ficheros de texto, otro para sus ficheros BASIC, otro para cartas comerciales, etc., etc. Sin embargo, sólo es con el empleo del disco duro, que cobra realmente sentido la división del medio de almacenamiento, ya que aquí se dispone de una gran cantidad de espacio, que dificulta considerablemente la administración de todos los ficheros.

Mediante la división en parcelas, también es posible que figuren dos ficheros con el mismo nombre y con el mismo código, sobre un medio de almacenamiento. Siempre y cuando cada fichero figure en una zona distinta.

IV.3.1 Zonas del usuario en CP/M 2.2

La posibilidad de contar con varios sectores para el usuario, se ha adoptado del sistema operativo multiusuario MP/M, para la versión CP/M 2.2. Cada zona opera como un disco individual. Esto significa: los ficheros se crean por separado en cada zona del usuario, y todos los programas requeridos deben figurar en dicha zona del usuario. El cambio entre zonas del usuario se efectúa mediante el comando:

USER n

La "n" representa un número entre el 1 y el 15. Puede trabajar plenamente en el interior de una zona, sin tocar para nada los ficheros de las demás zonas. Por ejemplo, el comando:

ERA*.*

borra todos los ficheros, pero sólo dentro de una zona del usuario. Si trabaja con dos unidades de disco, copiando ficheros de una unidad a otra, normalmente dichos ficheros aparecerán en la misma zona del usuario, de la unidad de destino. Si por lo tanto Vd. copia datos mediante PIP, desde la zona 3A a B:, los ficheros irán a parar a la zona 3B.

Después de cada arranque en frío de CP/M, el sistema operativo comienza en la zona cero. Si desea trabajar en otra zona, primero tendrá que indicarlo así. Lamentablemente, el prompt de CP/M no le revela nada acerca de la zona, en la que Vd. se encuentra. Sólo puede averiguarlo mediante el comando STAT.

IV.3.2 Zonas del usuario en CP/M 3.0

La zona del usuario cero(0) ocupa un lugar especial. Los programas y ficheros, que figuran en dicho lugar, y que han sido declarados como ficheros del sistema, están a disposición de todos las zonas del usuario. Hallará mas detalles sobre este particular, en un apartado ulterior. Si Vd. se encuentra en una determinada zona del usuario, y confecciona un fichero nuevo, CP/M automáticamente recuerda la zona, en la que se ha creado dicho fichero.

Cada vez que Vd. active su ordenador y cargue CP/M, Vd. se encuentra en la zona 0. Si quiere cambiar a otra zona, puede introducir el comando USER. Supongamos que quiera pasar a la zona número 3. Entonces teclee:

USER 3

No se olvide de poner un espacio detrás de USER, y detrás del número correspondiente. Cuando CP/M haya pasado a la nueva zona, podrá ver, que algo ha cambiado en la pantalla. El prompt de CP/M ya no es simplemente una "A", sino que delante figura el número de la zona del usuario, que en nuestro caso es un "3". Ahora seguramente también deseará trabajar con la segunda unidad, cambiando las zonas correspondientes. Si por ejemplo desea acceder a la zona 3, de la unidad B, entonces teclee:

USER 3B:

y verá que el prompt de CP/M se presenta así:

3B>

Si ahora introduce el comando DIR, para ver el directorio de su disco, aparecerá el mensaje "NO FILE". A condición, claro está, de que todavía no haya confeccionado ningún fichero en esta zona. Vd. puede acceder a todas los comandos incorporadas de CP/M, desde cualquier zona del usuario. Para volver a entrar en la zona 0, introduzca:

USER 0

con lo que pasará a la zona de usuario inferior. Puede preparar los programas de esta zona, para que estén a disposición de las demás zonas del usuario, mediante el comando SET. Mas tarde le enseñaré, como hacerlo. El cambio entre las zonas del usuario, todavía puede hacerse de forma más sencilla, introduciendo lo siguiente:

B1:

Con las zonas del usuario también debería tener en cuenta una cosa: si copia un fichero mediante PIP desde "A" a "B", o viceversa, la copia sólo se efectúa en la misma zona de usuario.

IV.4 DIR

Ya conoc el comando DIR del capítulo anterior, cuando mirábamos el directorio de su disco. El comando DIR es el mismo en las dos versiones de CP/M, a excepción de que en la versión 3.0, dicho comando se compone de dos programas. Por una parte está el comando DIR, que se asemeja al de CP/M 2.2, y por otra parte el programa DIR.COM, que ofrece unas funciones auxiliares, considerablemente ampliadas.

En primer lugar nos ocuparemos del comando DIR, y del modo en que se usa en ambas versiones.

Si introduce el comando DIR a secas, aparecerán todos los ficheros, que contenga su disco, en la zona correspondiente. Si en consecuencia Vd. ordena DIR estando en la zona 3, sólo se mostrarán los ficheros que contenga la zona 3. A veces, en CP/M 3.0, una zona del usuario contiene tantos ficheros, que no caben todos en una pantalla.

En tal caso, la impresión del directorio por pantalla se detiene, hasta que Vd. haya podido examinarlo todo satisfactoriamente, y prosiga mediante la pulsación de una tecla cualquiera. Si quiere parar la impresión del directorio, pulse simplemente control C (^C).

Si está trabajando con la unidad A, pero desea ver, si la unidad B contiene un determinado fichero, existen dos posibilidades para averiguarlo. Teclee:

```
B:
DIR
```

y verá el directorio de la unidad B. El segundo método es más rápido. Con la instrucción:

DIR B:

también verá el directorio de la unidad B en su pantalla. Una ventaja adicional de esta segunda versión: Vd. permanece en la unidad A. Si prefiere imprimir el directorio de su disco con la impresora, introduzca:

DIR B:

y luego control P (^P), y ENTER. Ahora el directorio completo se imprime por la impresora. Después de finalizar la impresión, vuelva a pulsar control P, con lo que la impresora finaliza con su trabajo.

IV.4.1 DIR con parámetros

Las funciones simples de DIR, en el fondo no son demasiado espectaculares, cosa que tampoco se espera de un sistema operativo normal. Sin embargo DIR se convierte en algo realmente útil, cuando Vd. utiliza las numerosas posibilidades adicionales que ofrece. Supongamos que tiene una gran cantidad de nombres en su directorio, y quiere encontrar un fichero concreto. En lugar de examinar todos los ficheros, uno por uno, para encontrar el suyo, simplemente introduzca:

DIR B:TEXT.TXT

y se le mostrará dicho nombre, si realmente existe en la unidad B. Si el fichero no se encuentra, obtendrá un mensaje de error de CP/M.

Si desea utilizar las funciones ampliadas de DIR, tendrá que trabajar con el programa DIR.COM en la versión 3.0. Ello significa: si está trabajando con la unidad B, y sin embargo DIR.COM se encuentra en la unidad A, no debe olvidar introducir también la denominación de la unidad, delante de DIR. Por ejemplo así:

A:DIR[Opciones]

Seguramente recordará, que el asterisco nos ahorra un montón de trabajo en la búsqueda. Mediante el comando DIR, en combinación con el asterisco, puede listar determinado tipo de ficheros. Ello funciona de la siguiente manera:

DIR*.COM

Si introduce el comando de este modo, en su pantalla aparecerán todos los ficheros, con el código COM.

IV.4.2 DIR ampliado en CP/M 3.0

En la nueva versión de CP/M Vd. puede, de la misma manera buscar varios tipos diferentes de ficheros. El comando correspondiente tiene este aspecto:

DIR$.COM$.SUB

Para que funcione realmente, tiene que solicitar a CP/M que utilice DIR.COM.

Puede conseguirlo, introduciendo el nombre de la unidad, como pueda ser A: o B:, antes de DIR, o bién escribiendo una opción entre corchetes detrás de los nombres de los ficheros. Las dos opciones posibles en este caso son FULL o DIR.

IV.4.3 DIR y sus opciones

En total, mediante DIR puede indicar hasta 18 opciones distintas. De esta manera, DIR tiene una potencia similar a PIP, y pertenece a los programas CP/M de especial utilidad. Normalmente no ocurrirá muy a menudo, que tenga que introducir más que una o dos opciones. Las opciones para DIR siempre deben ponerse entre corchetes.

Si se indican varias opciones, éstas deben separarse mediante una coma, o mediante un espacio. Si el nombre de la opción no se presta a confusiones, también puede abreviarlo con dos letras. Para mayor simplicidad también puede omitir el último corchete, si se trata del último carácter de una línea.

IV.4.4 DIRSYS

Si Vd. examina un directorio en CP/M 3.0, seguramente ya habrá notado, que en la parte inferior de la pantalla aparece un mensaje, que dice así:

SYSTEM FILE (S) EXIST

Mediante este mensaje, CP/M le indica, que a parte de los ficheros listados, el disco también contiene los así llamados ficheros del sistema. Los ficheros del sistema son los ficheros, que se almacenan enla zona del usuario 0, siendo accesibles y utilizables desde cualquiera de las zonas restantes. Si quiere ver sus ficheros del sistema, introduzca el comando DIRSYS, o simplemente DIRS. Entonces debajo de los ficheros listados obtendrá el mensaje, de que existen "ficheros que no son del sistema".

Naturalmente el comando DIRSYS le ofrece las mismas posibilidades que el comando DIR, por lo que también puede usar el asterisco y el interrogante, de la misma forma que se ha explicado anteriormente.

IV.5 ERASE

El espacio, así como el número de ficheros que acepta el directorio de su disco, tienen un límite. Llegará un momento en el que ya no habrá más sitio en el disco, y se verá obligado a borrar un fichero, o varios. El sistema operativo CP/M ofrece la posibilidad de borrar ficheros mediante el comando ERA.

Introduzca el comando de la manera siguiente:

ERASE D:NOMBRE

en donde D representa el nombre de la unidad de disco, y NOMBRE es el nombre de su fichero. Si por ejemplo está trabajando con la unidad A, y desea borrar un fichero allí, no es necesario introducir el nombre de la unidad.
Para borrar varios ficheros a la vez, naturalmente puede volver a utilizar el asterisco y los interrogantes. Sin embargo debe estar muy al tanto de lo que hace, cuando utilice estos comandos tan poderosos. No hay nada más fácil, que borrar un fichero sin intención, y entonces ya puede estallar de rabia.

Si por ejemplo desea borrar todos los ficheros, que posean el código BAK, introduzca:

ERA*.BAK

y en CP/M 2.2 habrá perdido todos los ficheros con el código correspondiente. Con ficheros backup, como en nuestro ejemplo, un error seguramente no sería tan fatal. Pero, por favor, imagínese que hubiera introducido este comando:

ERA*.*

y luego, sin pensarlo, lo hubiera rematado con ENTER. Bueno, pues ésto es todo. Ya se puede despedir de sus datos.

Pero como este comando es tan poderoso, y puede llegar a causar tanto daño, los inventores de CP/M han incluido una pequeña protección. Tan pronto como Vd. quiera borrar algo con dos asteriscos (*.*), por razones de seguridad se le pregunta, si realmente lo dice en serio:

```
ALL(Y/N)
```

Si realmente desea borrarlo todo, entonces teclee una Y, en representación de 'sí', seguida de un ENTER. De no ser así, teclee una N, en representación de 'no' y salvará la situación.

IV.5.1 Borrar con ERA en CP/M 3.0

Seguramente el jefe de DIGITAL RESEARCH algun dia también borraría inintencionadamente todos los ficheros con sus cuentas corrientes secretas. Lo digo, porque en la nueva versión de CP/M, después de introducir el asterisco de esta forma:

```
ERA*.BAS
```

el programa se detiene para preguntarle, si realmente debe borrarse todo lo que le pide:

ERASE*.BAS(Y/N)?

Antes de responder con "Y", en representación de 'sí', realmente debería recapacitar sobre lo que está haciendo. ¿Está totalmente seguro, que la instrucción ha sido formulada correctamente, y que realmente quiere borrar este tipo de ficheros? Una vez que haya tecleado "Y", ya es tarde para volverse atrás. Los ficheros desaparecerán. Vigile sobre todo también los errores sintácticos. Por ejemplo, es muy fácil, escribir PAS en lugar de BAS. Y si envía este comando, todos los ficheros PASCAL que contenga su disco, desaparecerán sin dejar huella...

Para evitar estas sorpresas tan desagradables, existen dos posibilidades. Por un lado, puede hacer que sus ficheros más valiosos queden "protegidos de escritura". Más tarde verá la forma de hacerlo. En segundo lugar puede añadir la opción CONFIRM al comando ERASE. CONFIRM quiere decir confirmar, y puede abreviar la opción con una "C". Si introduce el comando de tal forma:

ERASE*.BAK[C

se borrarán todos los ficheros "back-up", con la diferencia de que CP/M, antes de borrar cada uno de ellos preguntará, si realmente debe borrar el fichero en cuestión.

IV.6 Modificar nombres de ficheros mediante REN(AME)

Hasta ahora ha visto que nos referimos a los ficheros con su nombre. Sin embargo es posible que posteriormente ya no le agrade el nombre de un fichero determinado, o que pretenda copiar el fichero sobre otro disco, en el que ya figure un fichero del mismo nombre. En todos estos casos, Vd. puede utilizar el comando incorporado REN o RENAME en 3.0 (en español: renombrar), para dar otro nombre a sus ficheros. La síntaxis de este comando es:

RENAME nuevo=viejo

Por lo tanto, primero figura el nombre nuevo del fichero, y a continuación el viejo. Los dos nombres se separan mediante un signo de igualdad. Si el cambio de nombre se efectúa con la unidad de disco en curso, no es necesario indicar el nombre de la unidad delante de los nombres del fichero. Si el nombre que piensa dar al fichero ya existe, CP/M 2.2 se lo avisará:

File exists

El nuevo CP/M incluso le ofrece más ayuda, preguntando si el antiguo fichero debe ser borrado.

Un cambio de nombre sencillo de un fichero puede formularse así:

REN nuevo.bas=viejo.bas

También en RENAME 3.0 puede utilizar el asterisco y los interrogantes. Un ejemplo simple puede ser:

RENAME*.TXT=*.BAK

Mediante este comando, todos los ficheros back-up de este disco, se transforman en ficheros con el código TXT, sin importar su contenido. El comando RENAME no se preocupa en absoluto de lo que contengan los ficheros. Sólo tiene en cuenta sus nombres.

IV.7 TYPE

Esta instrucción le muestra ficheros en la pantalla. Sin embargo, ésto sólo funciona con ficheros que contengan texto, ya que éstos se forman a partir del juego de caracteres ASCII. Otros ficheros, con el código COM, REL o algo similar, contienen caracteres de control, que confundirían el proceso de impresión en la pantalla, y que provocarían que el ordenador "se cuelge". El comando TYPE, se introduce de esta manera:

TYPE A:FICHERO.XXX

Si quiere ver el fichero de otra unidad de disco, anteponga el nombre de la unidad correspondiente.

En CP/M 3.0 hay una opción para este comando: NO PAGE. De esta manera el texto que se muestra desfilará continuamente a través de la pantalla. Normalmente la impresión en pantalla se detiene después de cada 23 líneas, siguiendo sólo si Vd. pulsa ENTER.

Si de lo contrario, Vd. ha introducido NO PAGE, puede detener la impresión mediante control S (^S), y continuar mediante control Q (^Q). Si además antes ha introducido un control P (^P), la impresión se efectuará por la impresora.

IV.8 Lo que ahora ya sabe

* CP/M dispone de comandos incorporados, y de comandos, a los que sólo se puede tener acceso, cargando un determinado programa del disco

* Los comandos pueden acompañarse de parámetros y opciones

* Vd. sabe, como se denominan todas los comandos incorporados, cómo actúan, y qué tipo de opciones sólo son posibles con comandos transitorios

V. Comandos transitorios

Hasta ahora ya ha leído bastante sobre los comandos incorporados. El verdadero poder del sistema operativo, sin embargo reside en los comados transitorios, que pueden encontrarse en el directorio, en forma de ficheros COM. En la primera parte de este capítulo, nos ocuparemos de los comandos de CP/M 2.2, y muy especialmente del comando STAT.

En la segunda parte, que es bastante más extensa, conocerá las numerosas posibilidades de CP/M 3.0, que con toda seguridad le harán la boca agua.

V.I Indicaciones de estado mediante STAT

El comando STAT, por un lado sirve para indicar el estado del sistema, y por otro lado se utiliza en CP/M, para modificar la asignación de los dispositivos. La forma más sencilla, y más usada de este comando es:

```
STAT
```

y ENTER. De esta manera se le muestra el espacio todavía disponible en el disco:

```
A:R/W,SPACE:89K
```

Aquí se le indica, que todavía quedan 98 Kilobytes libres en el disco, para lectura o para escritura.

Esto último se deduce de las letras R/W, en representación de Read/Write, que significa lectura/escritura. Por otra parte también existe el estado R/O (Read Only; sólo lectura), si el disco lleva la protección contra escritura. En tal caso no podrá escribir nada sobre el disco, sólo podrá leer su contenido.

Puede ser igualmente interesante, averiguar no sólo, cuanto espacio libre queda en un disco, sino también la extensión de un fichero, o de un determinado tipo de ficheros. Puede ver el tamaño de un fichero en cualquier unidad, si introduce el nombre de dicha unidad:

STAT B:DATOS.CMD

Obtendrá la siguiente respuesta en la pantalla:

```
RECS BYTES EXT Acc
--------------------------
  12 2K 1 R/W B:DATOS.CMD

Bytes Remaing on A: xK
```

Si en esta consulta utiliza el asterisco, por ejemplo de esta manera:

STAT B:*.CMD

obtendrá todas las indicaciones correspondientes en el formato de arriba.

El campo RECS indica, cuantos bloques ocupa el fichero en cuestión. BYTES indica la extensión de un fichero en unidades de 1K, mientras que EXT informa sobre el número de bloques correlacionados.

Si desea proteger todo el disco de la escritura, puede añadir el atributo R/O mediante STAT. R/O es lo contrario del antributo R/W, y sólo permite que se lean los ficheros. Para realizar dicha protección contra escritura, teclee:

STAT B:$R/O

Puede proceder de la misma manera, cuando solamente desee proteger uno o varios ficheros, en lugar de todo el disco:

STAT FICHERO.BAS $R/O

Dado que la distribución de las distintas zonas del usuario en CP/M 2.2 todavía no se indica con el prompt, tiene que servirse del comando STAT, para averiguar la zona en la que se encuentra. En recompensa a sus esfuerzos, incluso se le indicarán las demás zonas, que contengan ficheros, y las zonas, en que éstos figuren. Por tanto introduzca:

STAT USR:

y obtendrá las siguientes indicaciones:

Active User: 0

Active Files:0 2 3 5 11 7

De ello puede deducir, en qué zonas existen ficheros, ya que si en una zona no existe como mínimo un fichero, no se indicará nada.

V.2 Comandos transitorios en CP/M 3.0

Ahora nos ocuparemos de comandos importantes, que se usan muy a menudo en CP/M 3.0. Se trata de comandos transitorios, que encontrará en su directorio, y que van provistos del código COM. Vd. ya sabe cómo acceder a un programa: introduciendo el nombre del programa, sin su código, y concluyendo con ENTER. Tiene tres posibilidades de acceder a los programas transitorios. Una, cuando se encuentra en la zona 0, y también pretenda trabajar allí. Dos, cuando mediante el comando SET haya puesto los programas a disposición de todas las zonas. Y trés, también puede utilizar el comando SETDEF. Sobre este último comando le hablaré más tarde.

V.3 SET

El comando SET tiene varias funciones en CP/M 3.0. Principalmente se utiliza para poner diversos atributos a los ficheros. Se trata de un tipo de información adicional, unida al fichero, y que tiene un efecto determinado. Por ejemplo Vd. puede utilizar el comando SET, para hacer que sus ficheros CP/M, que se hallen en la zona 0, sean accesibles desde todos las demás zonas del usuario. También puede proteger sus ficheros, diciendo: "este fichero sólo debe poder leerse", o bién, "este fichero está protegido por una palabra clave".

El comado SET también incorpora opciones, que influyen sobre el directorio. Por ejemplo, Vd. puede confeccionarse un directorio, en el cual cada fichero vaya provisto de un "sello de fecha y hora", que informa sobre el momento en que fue creado, y sobre el momento, en que se ha utilizado por última vez.

Antes de que pueda instalar un directorio con los "timestamps", los "sellos de fecha y hora", tiene que ejecutar el programa INITDIR.COM. No crea, que lo de las fechas es sólo un juego. Más tarde podrá utilizar esta instalación, para guardar automáticamente todos los ficheros que hayan sido modificados, en otro disco.

Para que conozca mejor las numerosas posibilidades de SET, aquí tiene una tabla:

OPCION	SIGNIFICADO
DIR	vuelve a visualizar un fichero del sistema en el directorio normal

SYS convierte un fichero a un fichero del sistema

RO provoca que un fichero quede abierto sólo a la lectura

RW provoca que un fichero pueda ser leído y modificado

ARCHIV=OFF desactiva el atributo de ARCHIV. Ello significa, que este fichero todavía no ha sido asegurado (archivado). El programa PIP, con su opción AAU puede copiar ficheros, que lleven el atributo ARCHIV=OFF. El comando PIP se introduce con un asterisco, en representación de los nombres de ficheros, con lo que PIP copia todos aquellos ficheros, que hayan sido modificados desde la última vez que se hayan copiado mediante PIP, y la opción AAU. Después que PIP haya efectuado la copia, los atributos del fichero se fijan en ARCHIV=ON.

ARCHIV=ON activa el atributo de ARCHIV. Ello significa que este fichero ha sido asegurado. Mediante la opción AAU, normalmente PIP modifica el atributo, después de asegurar los ficheros. Vd. mismo puede modificar el atributo, utilizando el comando SET.

F1=ON/OFF activa o desactiva el atributo de fichero F1, definido por el usuario.

F2=ON/OFF activa o desactiva el atributo de fichero F2, definido por el usuario.

F3=ON/OFF activa o desactiva el atributo de fichero F3, definido por el usuario.

F4=ON/OFF activa o desactiva el atributo de fichero F4, definido por el usuario.

De momento ha visto lo que hay, y ahora aprenderá a aplicarlo. Vd. dispone de algunos programas valiosos, por ejemplo de su programa de textos, o del banco de datos, y desea poder trabajar con ellos, desde todas las zonas del usuario. Normalmente estos programas, al igual que los programas CP/M, se guardan en la zona 0. Para que Vd. siempre pueda acceder a estos programas, y tambien a otros ficheros, debe utilizar el comando SET, con una o varias opciones, convirtiendo el fichero correspondiente, en un fichero del sistema. Ello se hace así:

SET PIP.COM[SYS]

Si quiere estar seguro, de que no se escriba nada en el fichero PIP.COM (o en cualquier otro), debe añadir una segunda opción adicional:

SET PIP.COM[SYS RO]

Con ello PIP.COM se convierte en un fichero del sistema, que además queda protegido de la escritura. Si quiere suspender el atributo SYSTEM, para volver a tener un acceso libre, teclee:

SET PIP.COM[DIR RW]

No importa, en qué órden introduzca las opciones.

V.4 Atributos de la unidad de disco

También tiene la posibilidad de definir unidades de discos, de tal modo, que sólo se pueda leer, pero no escribir en ellas. Si Vd. ha fijado una unidad en "RO", en representación de "read only" (sólo leer), ningún fichero puede ser borrado mediante ERASE. RENAME no funciona tampoco, y PIP no puede transferir ningún fichero a esta unidad.

Si Vd. introduce un C control (pulsando la tecla de control, conjuntamente con la "C"), el estado RO de la unidad, vuelve a suspenderse. Si Vd. introduce:

SET A:[RO]

la unidad A: queda protegida contra escritura. Si en lugar de RO, introduce RW, (significa read/write = leer/escribir) la unidad puede volver a utilizarse plenamente.

V.5 Etiquetas

Especialmente si Vd. posee muchos discos, o cuando varios usuarios trabajan con un computador, es muy útil la posibilidad de poder dar nombres a los discos. Para ello se utiliza de nuevo el comando SET, pero con la opción "NAME=". Los nombres de los discos están sujetos a las mismas limitaciones que los nombres de un fichero. Sólo pueden emplearse ocho caracteres como máximo, para el propio nombre, además de otros tres para el código.

Para darle nombre a un disco, introduzca:

SET B:[NAME=COFI]

Esto podría ser por ejemplo, el nombre de un disco, que contuviese su contabilidad casera. Si ya está trabajando con la unidad B, puede omitir el nombre de la unidad. El nombre de un disco no puede verse en el directorio, cuando se solicita mediante DIR. Debe utilizar el comando SHOW, que explicaremos más tarde, conjuntamente con la opción "LABEL". Puede abreviar dicha opción con una simple "L", por lo que la instrucción completa adquiere este aspecto:

SHOW B:[L]

donde puede omitir el nombre de la unidad, si ya está trabajando con B:.

V.6 PASSWORD

Para que nadie le estropee el precioso órden de sus discos, CP/M ofrece la posibiliad de establecer una palabra clave. De este modo Vd. protege el nombre de su disco, con una palabra que sólo Vd. conoce.

Una vez haya introducido una clave para un disco, o para un fichero, CP/M siempre se la pedirá, antes de ejecutar cualquier instrucción. De modo que es mejor confeccionar también un fichero, que contenga todas las palabras claves, para que a la larga, se eviten confusiónes, dado que una vez que haya olvidado una clave determinada, ya no tendrá ninguna posibilidad de acceder a sus datos. De modo que - cuidado.

Para proveer su etiqueta de una clave, debe utilizar el comando SET, con la opción "PASSWORD=clave", o en caso contrario "PASSWORD=<cr>" (<cr> representa RETURN). Introduzca la clave en esta forma:

SET[PASSWORD=SECRETO]

o bién para eliminar una clave:

SET[PASSWORD=<cr> (<cr> = ENTER)

Repito: no se olvide de guardar sus palabras clave en algún lugar. Si olvidara alguna de ellas, ya no tendría acceso a los datos del disco en cuestión.

V.7 PROTECT

No sólo puede aplicarle una palabra clave a la etiqueta de su disco, sino que también puede hacerlo con ficheros individuales, e incluso con comandos de CP/M. De esta manera realmente asegurará sus datos de manipulaciones indeseadas, y de la curiosidad de personas no autorizadas.

Pero antes de poder proteger un fichero o un comando, debe activar el modo PROTECT. Ello se hace tecleando simplemente:

```
SET[PROTECT=ON]
```

o viceversa:

```
SET[PROTECT=OFF]
```

Cuando haya introducido ésto, estará en condiciones de proteger sus ficheros.

V.8 PASSWORD de fichero

Para poner un fichero bajo su protección personal, en principio debe proceder de la misma forma, que cuando protege un disco.

También puede proteger varios ficheros simultáneamente con una palabra clave, si utiliza el asterisco (*). Entonces la clave tendrá validez para todos los ficheros, que cumplan la condición. El comando se introduce de esta manera:

SET TEXTO.TXT[PASSWORD=clave]

o si emplea "*":

SET*.TXT[PASSWORD=clave]

En el primer ejemplo, el fichero TEXTO.TXT queda protegido por "clave", y en el segundo ejemplo, todos los ficheros TXT se protegen con la palabra "clave".
Adicionalmente puede determinarse la forma, en que los ficheros deban protegerse. Existen las siguientes posibilidades:

READ Se pedirá la clave para leer, copiar, escribir, borrar o modificar el nombre del fichero en cuestión

WRITE Se pedirá la clave para escribir, borrar o modificar el nombre del fichero en cuestión

DELETE Sólo se pedirá la clave para borrar el fichero en cuestión

NONE No hay clave. Si ya existiera alguna clave, ésta puede ser borrada mediante dicha opción.

Esto se hace así:

SET TEXTO.TXT[PROTECT=DELETE]

De esta manera el fichero TEXTO.TXT queda protegido de un borrado accidental.

V.9 TIME STAMP

Con la instalación del sello de fecha y hora, Vd. puede llevar el control sobre la frecuencia con que utiliza determinado fichero, sobre cuando lo ha creado, cuando lo ha modificado, y admás administrar sus ficheros de seguridad. Un sello de fecha y hora, viene a ser como el reloj marcador de una empresa, que almacena todos los datos referentes al horario de trabajo.

Para poder guardar la hora y fecha de cada fichero, primero tiene que ejecutar el programa INITDIR. La administración de fechas y horas sólo es posible, si el directorio se organiza de manera especial. Ello se consigue mediante INITDIR.

Tiene a disposición tres opciones diferentes

CREATE=ON Activa la archivación de la hora y de la fecha para los ficheros de una determinada unidad.

ACCESS=ON Activa la archivación del último acceso a un fichero. Sin embargo sólo puede activar uno de los dos comandos CREATE o ACCESS. Si se desea ACCESS en una unidad para la que previamente se ha seleccionado CREATE, CREATE automáticamente se desactivará.

UPDATE=ON Se ocupa de marcar los ficheros, cada vez que hayan sido modificados. El hecho de mostrar un fichero en el directorio, no altera la marca.

Si elige como opciones UPDATE y CREATE, debe tener en cuenta, que en cualquier trabajo que realice con los ficheros, ambas marcas siempre serán actualizadas.
Esto es debido, a que cuando se trabaja con un fichero, dicho fichero se inicia de nuevo, mientras que el antiguo se convierte en un fichero BAK (copia de seguridad).

Puede poner en marcha su reloj computerizado mediante:

SET[ACCESS=ON]

Para ver el resultado de esta operación, teclee el comando:

DIR[FULL]

y obtendrá:

```
Directory for Drive B:

Name Bytes Recs Attributes Prot Update Access
---- ----- ---- ---------- ---- ------ ------
TEXT.TXT 5K 38 DIR RW NONE 04/01/86 17:31
COFI 20K 152 SYS RO NONE 04/01/86 09:10
```

El campo ACCESS le indica el fichero utilizado, y cuando se ha utilizado por última vez. Para que se efectúen dos anotaciones en el directorio, tiene que introducir el comando siguiente:

SET[CREATE=ON,UPDATE=ON]

El directorio entonces muestra esta estructura:

```
Directory for drive B:

Name Bytes Recs Attribute Prot Update Create
---- ----- ---- --------- ---- ------ ------
```

```
TEXTO.TXT 5K 38 DIR RW NONE 04/17/86 10:00 01/01/86 09:00
COFI 20K  152 SYS RO NONE 04/17/86 16:43 01/01/82 19:21
```

V.10 SETDEF

Este comando le ofrece la posibilidad de buscar programas en su disco. Vd. por ejemplo está trabajando con la unidad B:, y sin embargo guarda todos los ficheros de CP/M, o de otros programas en la unidad A:. Mediante SETDEF, CP/M ahora dispone de la posibilidad, de cargar todos los programas deseados, sin que por ello Vd. tenga que indicarle en cada caso el nombre de la unidad.

Si simplemente introduce:

SETDEF

obtendrá información sobre el aspecto actual de dicha posibilidad, sobre la unidad que se utiliza para ficheros temporales, y sobre el tipo de ficheros que se buscan. Si introduce:

SETDEF A:

CP/M buscará siempre los ficheros deseados en la unidad A, aunque esté trabajando con la unidad B. Si Vd. amplía el comando de esta manera:

SETDEF A:,*

entonces CP/M primero buscará los ficheros en la unidad A:, y sólo a continuación en la unidad en curso. El asterisco representa la unidad en curso.

Si Vd. desea que los ficheros temporales, es decir los ficheros intermedios, tal y como los confecciona PIP, se almacenen a través de una unidad determinada, introduzca:

SETDEF[TEMPORARY=C]

con lo que los ficheros intermedios se escribirán en una tercera unidad, o también en el disco RAM.

Sólo pueden buscarse ficheros, que dispongan de un código COM o SUB. Normalmente CP/M 3.0 busca ficheros COM, pero ésto puede modificarse con:

SETDEF[ORDER=(SUB,COM)]

Los ficheros SUB son aquellos, que puedan ser llamados por medio del comando SUBMIT, y que contengan instrucciones, que puedan ejecutarse en serie. Ahora mismo nos ocuparemos de ello.

V.11 SHOW

Ahora trataremos de un comando, que puede proporcionarle muchas informaciones acerca del espacio en sus discos, acerca de su nombre, y sobre el número de ficheros, a disposición de cada usuario. Si teclea:

SHOW

verá los atributos de todas las unidades, así como el espacio libre en cada unidad.

A:RW,Space: 101K

B:RW,Space: 5,934K

Si introduce SHOW con el nombre de una unidad, sólo se le proporcionarán las informaciones estadísticas de dicha unidad.

Como hemos dicho antes, mediante SHOW, también puede acceder a las etiquetas de sus discos:

SHOW A:[LABEL]

Además se prmite abreviar LABEL con una simple "L". Entonces sobre la pantalla verá:

```
Label for drive A:

Directory Passwds Stamp Stamp
Label Reqd Create Update Label Created Label Updated
-------------------------------------------------------
COFI.COM off off off 04/17/86 11:41 04/17/86 11:41
```

Una opción adicional al comando SHOW le muestra, cuales son las zonas del usuario que se utilizan, y cuantos ficheros pertenecen a cada una de ellas. Además se indica el número de anotaciones libres en el directorio:

```
A:Active user: 0
A:Active files: 0 2 11 12
A:# of files: 85 6 1 1

A:Number of free directory entries: 24
```

Si simplemente teclea:

```
SHOW A:[DIR]
```

sólo le será indicado el número de anotaciones libres.

V.12 SUBMIT

Hasta ahora, Vd. ha introducido todas los comandos directamente por el teclado. Esto parece perfectamente lógico, pero si tiene que repetir siempre los mismos comandos para obtener siempre los mismos resultados, a la larga la cosa se hace pesada. CP/M es capaz de quitarle este peso de encima.

Mediante el comando SUBMIT, Vd. puede conseguir que se ejecuten toda una serie de instrucciones contenidas en un fichero, y que se traten, como si fuesen comandos introducidos directamente por el teclado. Un fichero que contenga dicha serie de instruciones, debe ir provisto del código SUB, que lo acondiciona para ser leído y ejecutado por el programa SUBMIT.

Si por ejemplo su computador no dispone de una batería, que se encargue de mantener continuamente en marcha el reloj incorporado, cada vez que active el computador, tendrá que volver a introducir fecha y hora. Si Vd. quiere obligarse a efectuar siempre dicha introducción, para que los sellos de fecha y hora se apliquen continua- y correctamente, puede utilizar el fichero PROFILE.SUB.

Este fichero tiene una particularidad especial: cada vez que CP/M 3.0 se ponga en marcha, CP/M 3.0 lee y ejecuta los comandos que contiene dicho fichero. Si por lo tanto Vd. aquí pone la instrucción de introducir fecha y hora, no podrá eludir dicha introducción. El fichero se escribe:

B:DATE SET

con su procesador de textos, bautizándose PROFILE.SUB. Naturalmente también puede utilizar otra unidad de disco, que la arriba indicada. Esto depende únicamente de la unidad, en la que se encuentre el fichero DATE.COM. Si Vd. se olvida del código SUB, el comando SUBMIT no hará nada. Un fichero SUB puede contener comandos de CP/M, comandos SUBMIT, y datos de entrada para un programa, o para un comando de CP/M. Si piensa un poco, se dará cuenta, de que con ello puede hacer una buena cantidad de cosas.

En un fichero SUB, Vd. también puede trabajar con

instrucciones generales, que luego se utilizarán según las entradas que Vd.efectúe. Estas instrucciones generales se denominan parámetros, y se representan mediante el signo del dólar ($). Puede emplear parámetros desde $1 hasta $9.

Supongamos, que Vd. escriba el siguiente fichero:

```
ERA $1.BAK
DIR*.$2
```

y que, pongamos por caso, le dé el nombre DIR.SUB. Pues bién, con esta secuencia de instrucciones en el fichero, primero se borrarán todos los ficheros de un determinado nombre, y que lleven el código BAK. Acto seguido, se le mostrarán todos los ficheros que lleven un código determinado. Para conseguirlo, teclee:

```
SUBMIT TEXTO COM
```

Ahora, primero se borrarán todos los ficheros con el nombre TEXTO.BAK, y luego se le mostrarán todos los ficheros con el código COM. El fichero SUBMIT traducido, se leería así:

```
ERA TEXT.BAK
DIR*.COM
```

CP/M ejecutará estos comandos, del mismo modo que si se hubieran introducido por el teclado.

Si introduce menos parámetros de los previstos en el fichero SUBMIT, los restantes parámetros se ignorarán. Del mismo modo, también se ignorarán los parámetros que pasen del número máximo permitido. Si quiere, que en una instrucción de un fichero SUBMIT figure un signo dólar, simplemente teclee dos signos dólar seguidos ($$), y obtendrá lo que desea.

Un fichero SUBMIT, sin embargo no sólo puede contener comandos de una sola línea, sino que también acepta instrucciones para programas. Un ejemplo:

```
PIP
<B:=A:*.COM
<
DIR*.COM
```

Aquí tiene un corto fichero SUBMIT, que sin embargo contiene algo nuevo. En la primera línea se hace una llamada al programa PIP.COM, y en la segunda, se da el comando para copiar todos los ficheros COM a la unidad B:. A continuación se abandona el programa PIP, mostrándose el directorio. Todas los comandos dirigidos directamente a un programa, deben marcarse con el signo "menor que" (<). La omisión de dicho signo, como en la tercera línea de nuestro ejemplo, supone un RETURN, lo que en este caso, termina con PIP.

Sin embargo, el comando SUBMIT no sólo es un jueguecito entretenido, sino que puede emplearse de forma muy útil. Por ejemplo puede imprimir cuantos ficheros desee, y mientras tanto ir a comer algo. El fichero de instrucciones para estas acciones, se escribe así:

```
PIP LST:=FICHERO1
PIP LST:=FICHERO2
PIP LST:=FICHERO3
PIP LST:=FICHERO4
etc.
```

o también puede hacerlo, incluso con menos esfuerzo:

```
PIP LST:=FICHERO?
```

Entonces Vd. bautiza el fichero con IMPRESCO.SUB, y antes de marcharse introduce:

SUBMIT IMPRESCO (RETURN)

Naturalmente los ficheros pueden encontrarse en unidades distintas. Sólo tiene que escribir la letra de la unidad delante del nombre del fichero, y SUBMIT se encarga de buscar el resto.

También podría escribir un fichero de instrucciones, que se encargue de listar todos los ficheros de una placa, y a través de todas las zonas del usuario, mediante DIR, pero que convierta el listado en un nuevo fichero con el nombre índice. Entonces en este fichero, Vd. podría buscar otro fichero determinado, mediante el comando de búsqueda de su programa de textos, o podría imprimir la lista de los ficheros.

V.13 El comando HELP

Con sus numerosos comandos, CP/M 3.0 ya no es tan fácil de aprender, y de dominar, como su antecesor. Especialmente tardará un poco en retener y dominar todas las opciones. Afortunadamente el sistema operativo nos ofrece su ayuda "con sólo apretar un botón". Vd. tiene que gritar "socorro" en inglés, y enseguida obtendrá lo que desea. Bueno, quizás sea mejor que en lugar de gritar, escriba:

```
HELP
```

Se le presentará un menú con las ayudas posibles. Puede escoger una de las opciones, y se le informará con mayor detalle.

```
HELP UTILITY V1.1

At "HELP>" enter topic [subtopic]...

EXAMPLE: HELP> DIR EXAMPLES

Topics available:
```

```
COMMANDS CNTRLCHARS COPYSYS DATE DEVICE DIR

DUMP ED ERASE FILESPEC GENCOM GET

HELP HEXCOM INITDIRLIB LINK MAC

PATCH PIP (COPY) PUT RENAME RMAC SAVE

SET SETDEF SHOW SID SUBMIT TYPE

USER XREF


HELP>
```

También puede acceder directamente a las opciones, introduciendo por ejemplo:

HELP SETDEF

con ello se le dará directamente la información de ayuda. Pero desgraciadamente no es oro todo lo que reluce, y el inglés tampoco es patrimonio de cualquiera. Naturalmente, nuestros amigos, los programadores americanos, suponen que todo el mundo habla inglés. Después de mucho ir y venir, y de mucho mascar chicle, los inventores de CP/M decidieron finalmente crear una posibilidad, para que los que no fuesen americanos, también pudiesen acceder a las informaciones de ayuda, en otros idiomas.

El programa de ayuda se compone de los ficheros HELP.COM y HELP.HLP. Para poder modificar algo aquí, llame el fichero HELP.COM, e introduzca la opción EXTRACT:

HELP[EXTRACT]

También puede abreviar EXTRACT con una "E". Entonces el programa HELP crea un nuevo fichero de nombre HELP.DAT a partir del fichero HELP.HLP, que Vd. puede modificar según sus ideas y su idioma, mediante su procesador de textos.

Para introducir nuevos textos de ayuda, tiene que tener en cuenta lo siguiente:

Cada concepto debe empezar con tres barras (///), y con un número. El número indica el nivel de ayuda del concepto. Por ejemplo:

///1DIR (ENTER)

///2OPCIONES (ENTER)

///3PARAMETROS (ENTER)

///4EJEMPLOS (ENTER)

Cuando haya modificado todo, o haya instalado una ayuda para un programa poco frecuentado, debe almacenar el fichero y volver a llamar el fichero HELP.COM, pero esta vez con la opción CREATE, abreviado "C". De esta manera se creará un nuevo fichero HELP.HLP, que ya contendrá sus modificaciones.

V.14 Lo que ahora ya sabe

*Ha conocido el importante comando STAT de CP/M 2.2, mediante el cual puede acceder a las diversas informaciones sobre su sistema, o modificarlas

*Ha conocido los comandos tnsitorios de CP/M 3.0

*Entretanto conoce un gran número de opciones, mediante las cuales puede modificar, o ampliar el efecto de los comandos transitorios

*Sabe que puede aplicar etiquetas a sus ficheros

*Ha visto, de que manera pueden protegerse un disco entero, un fichero, o determinados comandos de CP/M, contra el acceso no autorizado

*También está claro, como puede hacer, que se les aplique un sello de fecha y hora a sus ficheros

*Ha conocido las posibilidades de la búsqueda de programas

*Ahora sabe que hacer, para que se le muestren los datos del sistema de un disco, mediante SHOW

*Puede utilizar el fichero PROFILE.SUB, para automatizar la rutina de inicialización de su computador

VI. Todo sobre PIP

De momento ya ha escuchado algunas cosas sobre un ingenio llamado PIP. Para que no piense que estoy exagerando descaradamente, le voy a resumir rápidamente, todo lo que PIP sabe hacer

*Copiar un fichero de un disco a otro disco

*Copiar un grupo de ficheros de un disco a otro disco

*Copiar un fichero y darle otro nombre

*Formatear un texto para la impresión

*Acortar una línea demasiado larga

*Imprimir una colección de ficheros con un comando

*Convertir varios ficheros en uno solo

*Extraer una sección de un fichero de texto

*Convertir minúsculas en mayúsculas y viceversa

*Volver a poner en cero el octavo bit, también llamado bit de paridad

*Proveer un fichero con números de línea

*Mostrar un fichero en pantalla, mientras se está transfiriendo

*Transferir ficheros del sistema

*Copiar ficheros de una zona del usuario a otra

*Guardar automáticamente ficheros nuevos o modificados

¿Es suficiente? De todas formas es razón suficiente, como para analizar profundamente el programa. Es mejor que lea este capítulo y su manual tres veces, en lugar de dos, y que se forme una idea, de todo lo que PIP realmente es capaz de hacer por Vd.

VI.1 Copiar un disco

Prácticamente todos los manuales para cualquier programa comienzan con el consejo de copiar, antes que nada, el programa PIP en un disco nuevo formateado, y a continuación sólo trabajar con dicha copia. En un capítulo anterior, ya he mencionado ésto, y también le he enseñado como se hace. Dado que la forma de pensar de PIP es un poco diferente a la forma de pensar de los humanos, le vuelvo a repetir el ejemplo. La copia de un disco a otro, se efectúa de esta forma:

```
PIP
*B:TEXTO.TXT=A:TEXTO.TXT
```

Con ello consigue, que el fichero TEXTO.TXT del disco en la unidad A:, se copie bajo el mismo nombre a la unidad B:. Su original permanece inalterado, sólo que ahora tiene una copia en la otra unidad.

Para copiar un disco entero, introduzca:

```
PIP
*B:=A:*.*
```

De esta manera Vd. copia todos los ficheros de A: a B:, a excepción del sistema operativo. Este sólo puede copiarse mediante un comando especial en las pistas del sistema de cada disco. Dicho comando es:

SYSGEN en CP/M 2.2 y
COPYSYS en CP/M 3.0 (¡¡se suministra con DISCKIT3!!)

Si ha copiado ambas cosas, posee una copia completamente idéntica al original, apta para ser copiada a su vez. Puede introducir la copia en la unidad de disco, y poner en marcha el computador. Para simplificar y hacer más rápido el procedimiento arriba mencionado, existen dos posibilidades.

El comando PIP puede acompañarse de toda una serie de opciones, que dan los más diversos resultados. Una de las opciones es "V", en representación de "verify", lo que significa "verificar". Cuando PIP trabaja con esta opción, verifica exactamente después de cada copia, si el fichero nuevo, realmente es idéntico al fichero original. De nuevo, las opciones deben ser introducidas entre corchetes. Para copiar el fichero TEXTO.TXT, y verificarlo al mismo tiempo, introduzca:

```
PIP
*B:=A:TEXTO.TXT[V]
```

Esto es el primer paso. Sin embargo me molesta tener que llamar cada vez PIP, y sólo luego poder introducir el comando en la segunda línea. De hecho, desde el punto de vista de CP/M ésto tampoco es necesario. También puede abreviarse el comando descrito arriba de la siguiente manera

PIP B:=A:TEXTO.TXT[V]

En tal caso, PIP comienza a trabajar instantáneamente, volviendo luego directamente al prompt (A>). Seguramente habrá notado que no he introducido ningún nombre de fichero para la unidad B:. Vd. puede proceder de la misma forma, si no pretende que la copia del fichero tenga otro nombre. Si de lo contrario desea cambiar el nombre, teclee:

PIP B:TEXTO1.TXT=A:TEXTO.TXT

Así puede copiar el fichero, y a la vez, dar a la copia un nombre nuevo.

Antes de copiar sobre otro disco mediante PIP, sin embargo deberia asegurarse de que allí exista suficiente espacio para el nuevo fichero. Debe tener en cuenta, que PIP primero copia el fichero en el disco destinatario en forma de fichero temporal, con el mismo nombre, reconociéndolo solo por el código $$$. Sólo cuando queda asegurado, que la transferencia de datos ha tenido éxito, PIP reemplaza el nombre del fichero temporal, por el nombre correcto.

Supongamos que Vd. copia el fichero TEXTO.TXT al disco B:, pero allí todavía figura la versión antigua de este fichero bajo el mismo nombre. ¿Qué hace PIP? Si el disco ya no ofrece espacio suficiente, se producirá un mensaje de error. La causa es el método de trabajo de PIP. El programa primero intenta escribir el fichero en cuestión sobre el disco, a pesar de que allí todavía se encuentre también la antigua versión. De allí viene la necesidad de espacio suficiente. Si lo hay, el fichero nuevo se copia, codificándolo con $$$, borrando a continuación el fichero antiguo, y cambiando finalmente el nombre del fichero nuevo. Si por lo tanto alguna vez la operación de copiar no da resultado, posiblemente es que no hay espacio suficiente para el fichero nuevo, por lo que primero habrá que borrar el fichero antiguo mediante ERASE.

En cada copia efectuada con PIP sin opción especial, los atributos del fichero, tales como SYS,DIR,RO y RW, se copian también. De modo que si Vd. copia un fichero del sistema con PIP, también la copia será un fichero SYSTEM.

Si Vd. le pide a PIP que copie un fichero cuyo nombre ya exista en el disco destinatario, y que además esté protegido contra escritura (RO=Read Only), PIP le preguntará si está autorizado, para borrar dicho fichero.

Responda a esta pregunta con "Y" en caso afirmativo, y con "N" en caso negativo.

VI.2 Copiar entre zonas del usuario

Lo que ha leído hasta ahora, sólo posibilitaba la transferencia de ficheros, dentro de una zona del usuario. Si Vd. se encuentra en la zona 3 y está utilizando PIP para copiar uno o varios ficheros, las copias también irán a parar en la zona 3 del otro disco. Piense en ello, si alguna vez echara a faltar sus ficheros copiados.

Para copiar de un disco a otro, y a la vez de una zona a otra, utilice la opción "Gn", donde "n" representa el número de la zona del usuario. Para copiar el fichero TEXTO.TXT de la zona 0 a la unidad B:, y a la zona 2, introduzca:

PIP B:[G2]=A:TEXTO.TXT

Piense en que otros comandos de CP/M, como DIR,ERASE,TYPE etc. sólo ejecutan algo en aquella zona del usuario, que haya sido seleccionada.

VI.3 Ficheros de texto y ficheros de datos

Hasta ahora sólo ha utilizado PIP, para copiar ficheros de un disco a otro. Como ya he mencionado antes, también tiene la posibilidad de confeccionar un fichero colectivo con varios ficheros individuales, y todo ello en una sóla pasada.

Antes de decirle más sobre este aspecto interesante, tengo que explicarle rápidamente algo acerca de las diversas formas de ficheros. Básicamente CP/M distingue dos clases de ficheros: el fichero que contiene texto, y el fichero que no contiene texto. Vd. confecciona un fichero de texto mediante su procesador de textos, o con el editor incorporado en CP/M (poco aconsejable), utilizando todos los caracteres que le proporcione su teclado. Un fichero de datos se compone de números binarios, y normalmente lleva el código COM.

Estos dos tipos de ficheros tienen que diferenciarse, ya que CP/M reconoce el final de ficheros de texto, por un control Z (^Z). El texto que sigue a un ^Z, no se imprime, ni se tiene en cuenta. Contrariamente, la ^Z en un fichero de datos, es un carácter totalmente normal, como todos los demás, y CP/M no se preocupa por él.

Normalmente PIP siempre supone, que hay que copiar ficheros de datos, y en la mayoría de las veces tiene razón. Pero si Vd. desea convertir varios ficheros en uno sólo, PIP supone, que se trata de ficheros de texto.

VI.4 Unificar ficheros

La coma, para PIP es la señal de unificar varios ficheros. Por esta simple razón, no debería utilizar una coma en un nombre de fichero, porque desorientaría completamente a PIP.

Una unificación sencilla de varios ficheros podría realizarse así:

```
PIP TODO.TXT=PARTE1,TXT,PARTE2.TXT,PARTE3.TXT
```

siempre y cuando todos los ficheros se encuentren en la misma unidad de disco. Si desea que el fichero unificado se escriba en una zona del usuario diferente, con CP/M 3.0 introduzca:

```
PIP TODO.TXT[G3]=PARTE1.TXT,PARTE2.TXT,PARTE3.TXT
```

Si además insiste en una verificación después del copiado, tendrá que introducir una [V] detrás de cada fichero, que vaya a formar parte del nuevo. En tal caso, la instrucción ya es más complicada:

PIPTODO.TXT[G3]=PARTE1.TXT[V],PARTE2.TXT[V],PARTE3.TXT[V]

De todas formas es más sencillo, que copiar los tres ficheros "a mano".

Si quiere unificar ficheros de datos mediante PIP, pueden surgir problemas. Como Vd. ha leído, cada vez que surge una ^Z, PIP piensa que ha llegado al final del fichero. En ficheros de datos, sin embargo la ^Z puede surgir más a menudo, lo que imposibilitaría, o por lo menos dificultaría bastante el copiado.

Dado que los inventores de CP/M conocen el problema, y saben, que los ficheros que se copian más a menudo suelen ser los ficheros COM, y no los de texto, en un fichero COM, PIP hace la vista gorda, cuando encuentra una ^Z, y copia el fichero hasta su amargo final.

Si desea copiar ficheros que no son ni COM, ni de texto, tiene que añadir el parámetro "O". Como siempre, entre corchetes, directamente detrás del nombre del fichero.

Incluso tiene la posibilidad de incluir algo en los ficheros, durante el proceso de unificación. En tal caso, el comando tendría que formularse así:

PIP TODO.TXT=PARTE1.TXT,CON:,PARTE2.TXT,CON:,PARTE3.TXT

De esta manera, a causa del comando CON:, PIP comienza a copiar el primer fichero, y se detiene a continuación, esperando su entrada desde el teclado.

Sin embargo tiene que introducir el comando ENTER y "salto de línea" (^J) manualmente, para que PIP no sobreimpresione ninguna línea. Continúe, introduciendo (^Z). Sin embargo tiene que tener en cuenta, que no es posible corregir errores ortográficos.

VI.5 Enumeración de líneas

Una bonita instalación para programadores, periodistas, y todos los que quieran tener numeradas las líneas de sus textos, o de sus programas, es el parámetro "N" de PIP. Si Vd. copia un fichero con este parámetro, o con su hermano "N2", cada línea de su texto obtendrá un número propio.

Si Vd. copia así:

PIP TEXTONU.TXT=TEXTO.TXT[N]

las líneas de su texto, posteriormente aparecerán en esta forma:

1: Este es su texto

Si de lo contrario utiliza el parámetro "N2", el texto aparecerá así:

000001 Este es su texto

Los números simplemente se incrementan en unidades, lo que no puede cambiarse. Por lo tanto la secuencia es: 1,2,3,4,5...

VI.6 Conversión de letras

Si en lugar de números de línea, lo que desea es tener todo su texto escrito en mayúsculas, entonces es Vd. un candidato para el parámetro "U". La "U" sustituye "Uppercase", lo que significa mayúsculas. Obtendrá lo contrario, si introduce el parámetro "L", por "Lowercase". En tal caso su texto sólo consistirá de minúsculas.

VI.7 Buscar una cadena

A veces ocurre, que la impresora entra en huelga, o que simplemente se agote el papel durante la impresión. Entonces puede tener el problema, de que una parte de su impresión todavía figure sobre el disco, pero no sobre el papel. Pues bién, antes de volver a imprimir las 200 páginas, es mejor que utilice otro parámetro de PIP, para conseguir el resto de la impresión del texto.

Con PIP Vd. puede copiar un bloque de texto, indicando la cadena de caracteres, a partir de la cual tenga que comenzar, e indicando la cadena con la que haya que finalizar. La cadena inicial se marca con una "S", en representación de "Start", y la cadena final, con una "Q",en representación de "Quit", lo que significa acabar. Escriba una (^Z) detrás de cada cadena, y PIP se encargará del resto. Comenzará a buscar en su texto la primera cadena indicada, copiando a partir de aquella, y terminando en la segunda cadena, que Vd. le haya indicado. Un comando de este tipo, se formula así:

```
PIP
*BLOQUE=TEXTO.TXT[Qclave^Z]
```

Si lo escribe de esta manera, PIP comienza a copiar desde el principio del fichero, deteniéndose cuando encuentre la palabra "clave". Por favor observe que primero PIP se ha llamado sin niguna opción, y que sólo en la segunda línea se ha introducido el resto. De esta manera, PIP busca "clave", tal y como Vd. la ha introducido, es decir, en minúsculas.

Si el comado de arriba se escribe en una sola línea, PIP convertirá "clave" en mayúsculas, iniciando la búsqueda a continuación. Si su "clave" no figura en el texto en forma de "CLAVE", PIP no encontrará nada, enviando un mensaje de error.

Un bloque determinado dentro de un texto, se copia mediante el comando siguiente:

```
PIP
*BLOQUE=TEXTO.TXT[Sprincipio^ZQfinal^Z]
```

El programa empezará a copiar desde la palabra "principio" y terminará cuandoaparezca por primerA vez la palabra "final".

VI.8 Impresión de varios ficheros en serie

Hasta ahora Vd. conoce el método de imprimir un fichero, introduciendo ^P, desviando la salida hacia la impresora mediante el comando TYPE. Ello es un tanto complicado, y puede solucionarse mejor con la ayuda de PIP. Cada dispositivo periférico tiene su propio nombre para PIP. El dispositivo de salida se denomina LST:. Los dos puntos son importantes, para que PIP pueda distinguir este nombre del dispositivo, de un fichero. Si quiere sacar su fichero de texto por la impresora, introduzca:

```
PIP LST:=TEXTO.TXT
```

Naturalmente en esta operación también es posible utilizar todas las demás opciones que normalmente influyen en PIP. Para imprimir por ejemplo varios ficheros en serie, introduzca:

```
PIP LST:=TEXTO1.TXT,TEXTO2.TXT,TEXTO3.TXT
```

y los textos saldrán por la impresora, uno detrás de otro. Si quiere influir en el formato de su texto, utilice el nombre PRN:. Entonces obtendrá líneas numeradas, una tabulación de ocho caracteres, y cada 60 líneas se iniciará una nueva página.

Al final de este capítulo le mostraré algunos ejemplos de las cosas razonables, que pueden hacerse con PIP.

VI.9 Archivado automático de ficheros

No sólo es importante imprimir datos, sino que es igual de importante guardarlos, es decir, archivarlos. Ya se lo he dicho, y además hablo por amarga experiencia. En mis excursiones a través del manual de PIP, he encontrado una función que facilita considerablemente el archivado. Es la opción de "archivo", abreviada "A".

Especialmente si Vd. trabaja con un disco duro, pronto conocerá las ventajas de este parámetro. Cada fichero en CP/M 3.0, reserva en su cabecera un espacio para un bit, denominado "archive flag". Un flag es algo así como un indicador, que indica un determinado estado. Cada vez que Vd. abra o modifique un fichero, dicho flag cambia, por ejemplo a "1", cuando haya modificado un fichero, y a "0" después de que lo haya archivado.

De este modo PIP reconoce los ficheros, que desde el último almacenamiento hayan sido modificados, o los que hayan sido creados de nuevo, guardando sólo éstos en un disco. Con ello Vd. se ahorra tener que copiar siempre el disco duro entero, lo que supondría un despilfarro considerable de tiempo, así como de gastos en discos adicionales.

Después de una larga jornada de trabajo, Vd. puede archivar y

asegurar sus ficheros modificados, introduciendo:

PIP B:=A:*.TXT[A]

Si quiere ir sobre seguro, añada también la opción "V", con lo que se verificará cada fichero después de su transferencia.

PIP B:=A:*.TXT[AV]

Si ahora ya no toca los originales de los ficheros archivados en el disco duro, éstos se ignorarán, cuando archive ficheros la próxima vez.

En el directrio puede ver, cuales son los ficheros que se han archivado, y cuales son los que no se han archivado. Para ello debe utilizar el comando DIR con las opciones FULL o RW. En el directorio verá, que los ficheros archivados llevan la etiqueta "Arcv".

VI.10 Sobreimprimir sin confirmación

Todavía hay que tener en cuenta algunos pequeños detalles, cuando se trabaja con PIP. Normalmente PIP sobreimprime, es decir, borra sin dubitaciones un fichero en el disco destinatario, si tiene el mismo nombre. Si de lo contrario se trata de un fichero protegido por el atributo R/O (Read Only), PIP esperará su confirmación, antes de borrar el fichero en cuestión, haciendo la pregunta correspondiiente.

DESTINATION FILE IS R/O, DELETE (Y/N)?

Si introduce "Y" en representación de sí, el fichero del disco destinatario se sobreimprimirá, es decir, se borrará. Si insiste en lo contrario tecleando "N", PIP aborta la operacion de copiado. En este caso no tiene que apretar RETURN después de "Y" o "N".
Si desea, que en cualquier caso se escriban todos los ficheros sobre el disco destinatario, sin esperar a que se confirme el permiso de borrar, debe utilizar el parámetro "W".

Entonces el comando es el siguiente:

```
PIP B:=A:TEXTO1.TXT,TEXTO2.TXT,TEXTO3.TXT[W]
```

Sin embargo de este modo Vd. asume toda la responsabilidad sobre el copiado, y luego no podrá reprocharle nada a PIP.

VI.11 Copiar ficheros del sistema

Si Vd. intenta copiar un fichero que disponga del atributo SISTEMA, pondrá a PIP en un aprieto. No podrá hallarlo. Tiene que ayudarle un poco, incluyendo la opción "R". Naturalmente también puede combinar las dos opciones. Si desea escribir ficheros y ficheros del SISTEMA sobre el disco destinatario, sin que se tenga en cuenta que vayan o no, protegidos contra escritura, introduzca:

```
PIP B:=A:$.COM[RW]
```

De esta manera se copian todos los ficheros COM, borrando los que tengan el mismo nombre en el disco destinatario.

VI.12 "Limpiar" el octavo bit

El juego de caracteres americano (ASCII) se representa con siete bits. El octavo bit, que representa la longitud de palabra usada en el ordenador, queda libre para tareas especiales. WordStar por ejemplo, lo utiliza para marcar los espacios, cuando se escribe en modo de bloques. Por otra parte MBASIC exige, que el octavo bit quede libre. Si trabaja sobre un programa con WordStar, y equivocadamente ha seleccionado el modo de documento (D), su programa no podrá ejecutarse, porque el octavo bit no está libre. Este problema tiene rápida solución, si utiliza la opción "Z" de PIP. Simplemente vuelve a copiar el fichero con PIP, y todo marcha sobre ruedas. El comando correspondiente es:

```
PIP BLOQUE.BAS=BLOQUE.BAS[Z]
```

Si así lo desea, también puede añadir la opción "V". Pero no utilice el comando de esta forma, para copiar ficheros de WordStar, o perdería el formato del texto.

VI.13 Ejemplos prácticos

Ahora verá algunos ejemplos, como utilizar PIP de forma razonable. Ya ha conocido los diversos parámetros a lo largo de este capítulo, y también sabe, que pueden emplearse conjuntamente.

La instrucción siguiente da un mejor formato al texto impreso, que la utilización de ^P o de un simple LST:.

PIP LST:TEXTO.TXT[NT8P60]

De esta forma el fichero TEXTO.TXT se envía a la impresora, enumerándose todas las líneas, con tabuladores en cada octava columna y una longitud de página de 60 líneas. Si hojea hacia atrás comprobará, que se trata exactamente de los valores establecidos para PRN:. Ahora sabe lo que PRN: le ofrece: menos trabajo.

Si en el ejemplo de arriba adicionalmente desea que todas las letras aparezcan como minúsculas, introduzca:

PIP LST:TEXTO.TXT[NT8P60L]

Sólo añadiendo la opción "L", la cosa ya marcha.

Para racionalizar el trabajo de archivo, en CP/M 3.0 puede escribirse un fichero SUBMIT con el siguiente contenido:

PIP A:=B:*.*[WAR]

dándole el nombre ARCHIVO.SUB. Las opciones "WAR" provocan que se copien los ficheros del SISTEMA, que los ficheros protegidos contra la escritura se sobreimpriman sin confirmación y que sólo se copien aquellos ficheros, que todavía no hayan sido archivados. Si Vd. utiliza este comando regularmente, prácticamente no puede pasarle nada a sus datos. La llamada se efectúa simplemente mediante:

SUBMIT ARCHIV

y el resto se ejecuta de forma automática. Todavía puede ampliar el comando si así lo desea. Mediante la opción "V" se verifica la transferencia correcta de los ficheros, y mediante el parámetro "E" se le mostrarán todas las copias en la pantalla. Sin embargo sólo debería utilizar la "E" con ficheros de texto, ya que de otra manera tendría problemas.

Otra posibilidad es la de incluir los comados DIR y SHOW en el fichero SUBMIT. Entonces puede ver qué ficheros, y cuántos van a ser copiados, así como el espacio libre que queda en el disco destinatario.

VI.14 Lo que ahora ya sabe

*PIP es una de las subrutinas más potentes de CP/M

*Puede transferir ficheros individuales a otro disco

*Puede copiar discos enteros

*Mediante PIP convierte varios ficheros en uno solo

*Puede extraer un bloque de un fichero

*PIP enumera automáticamente para Vd., las líneas de un fichero de texto

*Mediante PIP puede archivar automáticamente los ficheros, que haya modificado últimamente

*Es capaz de convertir Letras mayúsculas en minúsculas, y viceversa

*Puede hacer copias entre diversas zonas del usuario

VII. CP/M interno

VII.1 Confección de un disco de seguridad de CP/M

Antes de volcarnos de lleno en CP/M, primero debe confeccionar sin falta una copia de seguridad de su disco CP/M. Es tan fácil que se borre involuntariamente un fichero del disco, o que incluso se destruya el disco entero de forma inexplicable. (Y créame, ello sucede con más frecuencia, de lo que a Vd. y a mí nos pueda agradar.)

Para evitar en tal caso, quedar cual interrogante en paisaje desolado, buscando un disco CP/M en condiciones de funcionar, debería confeccionarse una copia, llamada también backup, del disco del sistema CP/M.

Entretanto ya sabe como se formatea un disco, y como se copian ficheros individuales, o discos enteros. Si sólo dispone de una unidad de disco, utilice el comando

DISCCOPY

La rutina le formatea el disco destinatario, si es necesario, indicándole en la pantalla, si debe introducir el disco original (source) o el disco destinatario (destination). Tendrá que cambiar los discos varias veces durante el proceso de copiado. Si dispone de dos unidades, se recomienda el comando COPYDISC, que trabaja con más rapidez, dado que no es necesario cambiar los discos.

Después de confeccionar una o dos copias de seguridad del disco CP/M, la fiesta puede empezar.

VII.2 Formatos de disco del floppy de AMSTRAD

El floppy del AMSTRAD dispone de tres formatos diferentes de disco: el formato normalizado CPC, el formato de datos y el formato IBM. Los tres tienen algunas cosas en común:

-Se formatean 40 pistas (pista 0 a pista 39).
-La longitud de los sectores es de 512 bytes.
-El directorio permite un total de 64 anotaciones.

El formato normalizado CPC, y el formato de datos, se formatean con nueve pistas por sector. Los dos formatos se diferencian simplemente, en que en el formato de datos, las dos pistas superiores no son ocupadas por CP/M, quedando de esta forma libres para el almacenamiento de programas y de datos. El formato IBM se diferencia, en que sólo formatea ocho pistas por sector. Siendo usuario del sistema operativo CP/M, a Vd. no le sirve el formato de datos, dado que las importantes pistas 0 y 1 no se ocupan con CP/M.

Vamos a limitarnos al formato normalizado CPC, dado que el formato de datos no tiene nada que ver con CP/M, y que el formato IBM (y otros) se describirá más tarde. En dicho formato las dos pistas "superiores" se ocupan de la manera siguiente:

```
Pista 0, sector &41      : sector de boot.
Pista 0, sector &42      : sector de configuración.
Pista 0, sector &43 a &47: no se usa.
```

Pista 0, sector &48, &49 y
Pista 1, sector &41 a &49: CCP y BDOS.

(& precede a números hexadecimales)

CCP significa Console Command Processor, y BDOS figura en lugar de Basic Disc Operating System.
Vd. sabe que mediante el comando transitorio FORMAT puede formatear sus discos. En CPC sin embargo, Vd. solamente puede formatear los discos en la unidad A:. Dado que existen varios formatos, Vd. debe seleccionar el formato en que CP/M 2.2 deba formatear su disco. Existen las posibilidades siguientes:

FORMAT : formatea en formato CPC normalizado

FORMAT I : formatea en formato IBM

FORMAT V : formatea en el formato Vendor

FORMAT D : formatea en formato de datos.

Si lo intentara con otro parámetro que (S),I,V, o D, entonces FORMAT reclamaría dicho parámetro. (S) significa, que cuando formatea en el formato del sistema, también puede omitir la "S".

VII.3 Formatos de disco ajenos

Quizás sea interesante para Vd. poder leer y escribir también en otros formatos. Naturalmente sería fabuloso poder formatear estos formatos ajenos también en el CPC. Ahora mismo hemos aprendido, que en el Amstrad los sectores abarcan 512 bytes. Dado que el controlador del floppy es programable, este valor puede cambiarse fácilmente. Los tamaños para un sector pueden ser 128, 256, 512, 1024 etc. bytes. La forma de programar el controlador del floppy (PD 765A) se describe exhaustivamente en el libro del floppy para CPC.

Al controlador del floppy pueden conectarse dos unidades, que entretanto también pueden ser unidades de 5.25". De todas maneras, el hecho de poder leer otro formato sólo tiene sentido para aquellos lectores, que dispongan de una unidad de 5.25", ya que sólo ellos pueden cargar software CP/M y datos, de otros ordenadores. (No olvide que el Amstrad CPC es el primer ordenador con unidades de 3".) Pero también aquellos lectores que "sólo" dispongan de la unidad de 3", pueden dejarse impresionar de las posibilidades, que les ofrece su controlador floppy.
Antes que nada, hay que constatar que existen cientos de formatos para discos de 5.25" bajo CP/M 2.2. A continuación ponemos a su disposición una tabla confeccionada por la empresa DIGITAL RESEARCH. La tabla informa sobre los distintos parámetros de los diversos formatos CP/M, y nos pone en condición de realizar los demás formatos en el CPC. Aquí nos limitamos a los formatos para discos de 5.25" de una sóla cara, ya que los discos de 8" seguramente ya no se dan en cantidades tan importantes.

```
D    BLS   SPT  BLM  DSM    ALO/1   OFF PHM Sec1 Skew
  B/S   Cap.  BSH EXM    DRM     CKS  PSH VR  letz  Nr.
-------------------------------------------------------
S 128 1   83 18 3 7 0  82  31 80/0  8 3 0 0 0 1 18 5 <1>
S 128 1   83 18 3 7 0  82  63 C0/0 16 3 0 0 0 1 18 4 <2>

S 256 1   92 20 3 7 0  91  63 C0/0 16 3 1 1 0 1 10 1 <3>
S 256 2   92 20 4 F 1  45  63 80/0 16 3 1 1 0 1 10 2 <4>
D 128 1  123 30 3 7 0 122  63 C0/0 16 2 0 0 0 1 30 1 <5>
D 256 1  144 32 3 7 0 142  63 C0/0 16 4 1 1 0 1 16 1 <6>
D 256 1  148 32 3 7 0 147  63 C0/0 16 3 1 1 0 1 16 1 <7>
D 256 1  152 32 3 7 0 151  63 C0/0 16 2 1 1 0 1 16 1 <8>
D 256 1  152 32 3 7 0 151  63 C0/0 16 2 1 1 0 1 16 2 <9>
D 256 1  157 34 3 7 0 156  63 C0/0 16 3 1 1 0 1 17 1 <10>
D 256 2  171 36 4 F 1  84  63 80/0 16 2 1 1 0 1 18 2 <11>
D 256 2  171 36 4 F 1  84 127 C0/0 32 2 1 1 0 0 17 1 <12>

D 512 1  153 32 3 7 0 155  63 C0/0 16 1 2 3 0 1  8 1 <13>
D 512 1  152 32 3 7 0 151  63 C0/0 16 2 2 3 0 1  8 1 <14>
D 512 1  171 36 3 7 0 170  63 C0/0 16 2 2 3 0 1  9 2 <15>
D 512 1  190 40 3 7 0 189  63 C0/0 16 2 2 3 0 1 10 1 <16>
D 512 1  195 40 3 7 0 194  63 F0/0 16 1 2 3 0 0  9 1 <17>
D 512 2  166 36 4 F 1  82  63 80/0 16 3 2 3 0 1  9 1 <18>
D 512 2  190 40 4 F 0  94  63 80/0 16 2 2 3 0 1 10 2 <19>
D 512 2  190 40 4 F 1  94  63 80/0 16 2 2 3 0 1 10 1 <20>
D 512 2  190 40 4 F 1  94  63 80/0 16 2 2 3 0 1 10 2 <21>

D1024 1  185 40 3 7 0 184  63 C0/0 16 3 3 7 0 1  5 1 <22>
D1024 2  160 40 4 F 1  78  63 80/0 16 3 3 7 0 1  5 1 <23>
```

A primera vista seguramente la tabla parecerá bastante indescifrable; sin embargo contiene informaciones muy importantes, sin las que, como pronto veremos, no podemos realizar nuestro empeño.

Importante para los interesados en el formato <21>: después de su lectura, los datos primero tienen que complementarse; ello puede efectuarse mediante XOR con &FF.

Pero antes que nada, aquí tiene la lista de los fabricantes, para mejor identificación de los diversos formatos:

<01>: XEROX
<02>: Lifeboat R2, TRS-80, Mayon
<03>: Eurocom
<04>: Osborne SD
<05>: Superbrain
<06>: MC-CP/M Ecma-70, MC-CP/M+#7
<07>: Eurocom II formato 2
<08>: Alphatronics DD, NEC PC 8001
<09>: Olympia ETX-II, Philips P-2000
<10>: Spectravideo
<11>: TRS-M4
<12>: Bondwell-12
<13>: IBM CP/M-86
<14>: ADPS
<15>: Dec VT-180
<16>: Rentiki
<17>: Kaypro II
<18>: Olympia Boss A
<19>: SEL Delsy 2000
<20>: Newbrain, Mayon
<21>: Superbrain, HKM-ZDOS
<22>: Osborne DD
<23>: BASF-7120

Las abreviaciones de la primera y segunda línea tienen el siguiente significado:

D=Density En esta columna se indica la densidad del formato, es decir: **S=Single density** (densidad normal) y **Double density** (densidad doble). Sólo es cuestión de tiempo, hasta que también se encuentre un formato CP/M de densidad cuadruple; los discos correspondientes con la etiqueta Quad density, de hecho ya existen.

B/S significa **Bytes por Sector.** Ya hemos aludido al hecho de que el CPC almacena 512 bytes por sector. Sin embargo es posible realizar sectores de 128, 256, 512 o 1024 bytes.

BLS designa el **tamaño de bloque** (Block Size) en Kbytes. De modo que en un formato de 128 bytes por sector y de un BLS de 1 Kbyte, 8 sectores componen un bloque. En el CPC un bloque consiste de 2 sectores, es decir, de 1 Kbyte.

Cap., como Vd. imaginará, representa **Capacity=capacidad.** Por lo tanto, aquí se indica la capacidad de almacenamiento del formato en cuestión, expresada en Kbytes. El CPC tiene una capacidad de 169 Kbytes.

SPT significa **Sectors Per Track=sectores por pista.** Sin embargo no debe tomarse la palabra sector en su sentido físico, sino más bien en un sentido lógico. Dado que al principio sólo había sectores de 128 Kbytes, dichos 128 bytes formaban un así denominado record. Por lo tanto SPT indica, cuántos de dichos records, o sea unidades de 128 bytes, caben en una pista. El CPC dispone de 9 sectores de 512 bytes, lo que da por resultado 9*512=4608 bytes por pista. Si dividimos este número 4608 por 128, obtendremos nuestro factor SPT: 4608/128=36.

BSH y **BLM** representan **Block-Shift** y **Block-Mask.** Estas dos indicaciones determinan el tamaño del bloque. El controlador necesita dichas indicaciones.

EXM significa **Extent-Mask**, y determina el número de anotaciones en el directorio.

DSM determina el **máximo número de bloques.**

Bajo **DRM** encontrará el **máximo número de notaciones en el directorio (i-1!).**

AL0/1 representa el **tamaño del directorio**, codificado en forma binaria. En el CPC se reservan 2 bloques para el directorio.

CKS indica, de cuántos **sectores lógicos** (records) se compone el directorio. En el CPC son 16.

OFF representa **OFFset de pista.** Aquí se indica el número de pistas reservadas a CP/M.

PSH y **PHM** dan informaciones para el **blocking** y el **deblocking** de sectores físicos en sectores lógicos.

Sec1 describe el **número del primer sector**, y **letz** el número del **último sector** de una pista. En el CPC el primer sector es el sector número 1, teniendo el último el número 9.

Finalmente tenemos la indicación del factor **SKEW.** Sólo tiene interés para los verdaderos especialistas.

Después de disponer de todas estas indicaciones, vamos a ver cuales son las que podemos comunicar al CPC, o mejor dicho, a nuestro controlador PD 765A. Para ello vamos a ver la memoria del sistema del floppy en el CPC:

encontrará las indicaciones en la RAM del sistema en las direcciones &A890-&A8A5.

A890,A891	SPT	records por pista (36)
A892	BSH	block shift (3)
A893	BLM	block mask (7)
A894	EXM	extend mask (0)
A895,A896	DSM	número máximo de bloques
A897,A898	DRM	número máximo de anotaciones en el índice
A899,A89A	AL0/1	tamaño del directorio (C000h) codificado en binario, equivale a 2 bloques
A89B,A89C	CKS	número de las anotaciones a controlar en el directorio (&0010) 16 anotaciones
A89D,A89E	OFF	offset de pista (2) pistas del sistema ocupadas
A89F	FSC	primer sector de cada pista (041h)
A8A0	PST	sectores físicos por pista
A8A1	GPS	longitud de GAP3 en sector Read/Write (2Ah)
A8A2	GPT	longitud de GAP3 en formateo de pista (52)h)
A8A3	FLB	byte de relleno en formateo de pista (E5h)
A8A4	BPS	bytes/sector (2) equivale a 512 bytes
A8A5	RPS	número de records/sector (4)

Los valores entre paréntesis indican los valores de default, es decir, los valores prefijados.

El formato <13>, el formato IBM, es el único que el CPC pone a disposición automáticamente.

En el fondo la realización de otro formato, no supone ningún problema: Vd. ya ve que es posible indicarle realmente todo al controlador. La única dificultad que nos queda, es la elección de la longitud del GAP3. GAP3 es un hueco, que se utiliza en los discos para compensar pequeñas irregularidades en la velocidad del motor de la unidad de disco, dado que al escribir un sector, el motor con casi toda seguridad no tendrá nunca exactamente la misma velocidad, que cuando lo está formateando. Por lo tanto existe el peligro, que a una velocidad inferior se escriba sobre el sector (físico) adyacente. Por lo tanto los huecos GAP3 actúan como zonas tampón. Mientras más grande sea GAP3, menos sectores cabrán sobre una pista. Si de lo contrario se selecciona un GAP3 demasiado pequeño, se incrementa el peligro de escribir sobre

un sector adyacente. Puede imaginarse un GAP3 como una secuencia de códigos &4E, siendo exactamente igual de posible cualquier otro carácter.

Intentemos realizar sobre nuestro Amstrad el formato OSBORNE <22>.

Podemos deducir los siguientes factores importantes de nuestra tabla:

1024 bytes por sector, 5 sectores por pista, número de sector 1 hasta 5. Estas y otras informaciones por lo tanto han de escribirse en la memoria del sistema del ordenador. Es importante - y esto ha costado un buen tiempo en el desarrollo del programa en código máquina - que ya que se esperan valores de 16 bits Vd. realmente almacene 16 bits. De otra manera se desplazarían todas las informaciones subsiguientes en la memoria de sistema del controlador del floppy, y con toda seguridad no obtendría el resultado deseado.

Después de almacenar los valores correspondientes en las direcciones de la memoria del sistema &A890-&A8A5, se ha creado la base para otro formato. Como puede deducir del programa en código máquina de la página siguiente, los datos pueden copiarse en la memoria de forma fácil con una tabla y el comando LDIR. Puede seguir utilizando perfectamente las rutinas del sistema, sin percance alguno, tal y como muestra nuestro programa ejemplo. Primero se formatean las 40 pistas con 5 sectores cada una. A continuación, escribiremos algo en todos los sectores, a modo de prueba. Se trata de un test, que nos deberá mostrar, si realmente todos los sectores están abiertos al acceso. Podría suceder por ejemplo, que se hubiese seleccionado un GAP3 demasiado grande, o demasiado pequeño, o que cualquier otra cosa no hubiese funcionado como cabía esperar (naturalmente así sucedió durante el desarrollo). Si observa la rapidez con que el floppy formatea las 40 pistas, y también la rapidez, con que todas se vuelven a sobreimpresionar, entonces realmente reconocerá la potencia de la unidad de disco.

Pero, antes que nada, aquí tiene el programa ensamblador para el formateo en formato OSBORNE:

```
 10
 20 ;formatear en formato OSBORNE, JS 9/6/85
 30 ;
 40 ;
 50       ORG  #7000 ;direccion inicial
 60 ;
 70 ;inicializa FDC con valores nuevos:
 80 ;
 90       LD   BC,22      ;22 valores
100       LD   HL,FDC     ;tabla de nuevos valores
110       LD   DE,#A890   ;memoria sistema floppy
120       LDIR
130 ;
140       LD   E,0        ;unidad de disco
150       LD   D,0        ;pista
160       LD   B,40       ;40 pistas
170 L1:   PUSH DE         ;salva DE
180       PUSH BC
190       LD   HL,FTAB    ;tabla para formateo
200       LD   A,D        ;pista=>D
210       LD   B,5        ;5 sectores
```

```
220         LD    (HL),A     ;pista de la memoria
230         INC   HL
240         INC   HL
250         INC   HL
260         INC   HL         ;puntero a siguiente sector
270         DJNZ  L0         ;siguiente sector
280         LD    C,#1       ;primer sector
290         LD    HL,FTAB    ;tabla para formateo
300         RST   #18
310         DEFW  FORMAT     ;formatea pista
320         POP   BC
330         POP   DE
340         INC   D          ;siguiente pista
350         DJNZ  L1
360 ;
370         JP    TESTE      ;comprueba todos los sectores
380 ;
390 ;
400 FORMAT:     DEFW #C652      ;direccion #C652
410         DEFB  7          ;en la ROM del floppy
420 ;
430 FTAB:   EQU   $
440 ;
450         DEFB  0          ;pista
460         DEFB  0          ;dirección de cabecera de la
                              unidad
470         DEFB  1          ;sector
480         DEFB  3          ;3=1024 bytes
490         DEFB  0,0,3,3    ;sector 3
500         DEFB  0,0,5,3    ;sector 5
510         DEFB  0,0,2,3    ;sector 2
520         DEFB  0,0,4,3    ;sector 4
530 ;
540 ;
550 ;tabla de los valores del controlador
560 ;
570 FDC:    DEFB  400,3,7,0,184,0,63,0,#C0,0,16
```

```
580       DEFB 0,3,0,1,5,42,50,#E5,3,8
590 ;
600 ;escritura de prueba sobre todos los sectores
610 ;
620       ORG  #7100        ;¡¡nueva dirección de salto!!
630 ;
640 TESTE:CALL #BB06        ;espera pulsación de tecla
650       LD   E,0          ;unidad
660       LD   D,0          ;pista
670 LL1:  LD   C,5          ;sector
680       LD   B,5          ;contador
690       LD   HL,#5000     ;tampón que se almacena
700 LL2:  RST  #18
710       DEFW WRITE        ;escribe sobre un sector
720       JR   NC,ERROR     ;se ha producido error
730       DEC  C            ;siguiente sector
740       DJNZ LL2
750       INC  D            ;siguiente pista
760       LD   A,D
770       CP   40           ;¿¿todas las 40 pistas??
780       JR   NZ,LL1       ;todavía no todas
790       RET
800 ;
810 WRITE:DEFW #C54E
820       DEFB 7            ;direccion #C64E en la ROM del
                             floppy
830 ;
840 ERROR:      EQU  $      ;indicar error con pitido
850       LD   A,7          ;beep (pitido)
860       CALL #BB5A        ;emitir
870       RET               ;corta control
880 ;
890
900       ORG  #7200        ;nueva dirección de salto
910 ;
920       LD   E,0          ;unidad
930       LD   D,1          ;pista
```

```
940         LD   C,2          ;sector
950         LD   HL,#5000     ;tampón
960         RST  #18
970         DEFW READ         ;lee sector
980         RET               ;fin del proceso
990 ;
1000 READ:DEFW #C666
1010        DEFB 7            ;dirección #C666 en la ROM del
                               floppy
1020 ;
```

Por favor, comprenda que en este lugar no incluimos ningún cargador BASIC - no tendría sentido. Si por ejemplo quisiera realizar otro formato que el de OSBORNE, todo el cargador prácticamente no serviría para nada. Aquellos lectores entre Vds., que deseen cargar en su CPC otro formato, de todas formas necesitan disponer de algunos conocimientos de código máquina, dado que "su" CP/M no puede procesar el formato nuevo tan fácilmente. Tendrán que cargar los datos mediante código máquina, y con la ayuda de las rutinas BDOS. Un pequeño consejo: ¡el ensamblador Macro-80 (Z-80), al igual que el ensamblador ASM de su discco CP/M, confecciona, si lo desea ficheros COM!

Si se modifica la tabla FDC (líneas 570-580) de forma adecuada, cualquiera de los formatos impresos es posible. Cuide solamente de almacenar los valores según la secuencia de la memoria del sistema. Todos los valores que aparecen en la tabla, pero no en la memoria del sistema, pueden omitirse. Para hacerse una idea de la rapidez con que se cargan 1024 bytes, pruebe lo siguiente: dado que en todas las pistas se escribe algo, después del formateado, algúna zona de la memoria tiene que utilizarse para el almacenamiento. En el ejemplo siguiente se ha tomado la zona desde &5000-&5400. Introduzca las siguientes líneas BASIC antes de formatear:

```
DEFINT I
FOR I=&5000 TO &5400
 POKE I,I AND &FF
NEXT
CALL &7000 (para empezar a formatear)
```

Con ello se ha ocupado la memoria desde &5000-&5400 con datos. (Se cuenta de 0 a 255, y se vuelve a empezar desde 0.) El mismo registro se ha almacenado en el disco (200 veces) a causa de la llamada **CALL &7000.**

En la dirección &7200 todavía se encuentra una corta rutina, para leer un sector de 1024 bytes. Sin embargo, antes de llamar dicha rutina, debería borrar el registro, ya que de otra manera no podría controlar, si el registro realmente ha recibido los datos del floppy:

```
FOR I=&5000 TO &5400
 POKE I,0
NEXT
```

Después de haber borrado la memoria, puede cargar un bloque cualquiera - todo el disco contiene lo mismo:

```
CALL &7200
```

y en un abrir y cerrar de ojos, el sector se carga. Puede asegurarse de ello, dejándose mostrar la zona de la memoria:

```
FOR I=&5000 TO &5400
 PRINT PEEK(I);
NEXT I
```

Como resultado debería obtener:

1 2 3 4 5 6 7 8 9 10 11 12 13 ...

Ello no sólo sirve como demostración práctica de la velocidad con que se escriben los datos, sino que también es una prueba, para saber si realmente se han cargado los 1024 bytes.

Otro pequeño, pero importante consejo: dado que el CPC normalmente sólo lee sectores de 512 bytes, también existe sólo un tampón de 512 bytes para la lectura de los sectores. Pero como que cuando se leen 1024 bytes, dicho tampón prácticamente se amplía también de forma "automática" a 1024 bytes, en según qué caso puede suceder, que se borren algunos datos, o partes importantes del programa. Sin embargo ello no representa un problema muy grande, ya que Vd. puede trasladar el tampón de sectores a un lugar cualquiera. El vector para el tampón de sectores, se encuentra en la dirección:

&BE62, &BE63

Esto por ejemplo puede conseguirlo, añadiendo las siguientes líneas al programa ensamblador:

```
135    LD HL,SEKBUF
137    LD (#BE62),HL      ;trasladar zona tampón
1030 SEKBUF:  DS    1024 ;1024 bytes como zona tampón
```

VII.4 El disco CP/M

Pero volvamos al formato de disco que no tengamos que simular. Con el Amstrad CPC 464 y CPC 664 se suministra un disco, que contiene LOGO de Digital Research y también CP/M 2.2. Ahora primero observaremos más de cerca, lo que contiene el disco de CP/M - no sólo hay ficheros de comandos CP/M, aunque sí son los que más abundan.

```
A:  MOVCPM  COM  :  PIP     COM  :  SUBMIT  COM  :
XSUB    COM
A:  ED      COM  :  ASM     COM  :  DDT     COM  :
LOAD    COM
A:  STAT    COM  :  DUMP    COM  :  DUMP    ASM  :
AMSDOS  COM
A:  FILECOPY COM : SYSGEN   COM : BOOTGEN  COM  :  COPYDISC
COM
A:  CHKDISC  COM  :  DISCCOPY  COM  :  DISCCHK  COM  :
SETUP   COM
A:  FORMAT   COM  :  CSAVE   COM  :  CLOAD   COM  :
EX1     BAS
A: EX2      BAS : ROINTIME DEM
```

El disco por ejemplo contiene ficheros de demostración para BASIC bajo AMSDOS, a los que aquí no prestaremos mayor atención. Si nos limitamos a los comandos de CP/M, comprbaaremos que a parte de los comandos estándar de CP/M 2.2, hay otros ficheros COM en el disco, tales como CLOAD.COM y CSAVE.COM. Estos dos ficheros COM sirven para copiar ficheros CP/M de disco a cassette (CSAVE), y viceversa (CLOAD). Si por ejemplo desea almacenar el fichero FILECOPY.COM en un cassette, introduzca el siguiente comando:

CSAVE FILECOPY.COM (RETURN)

Esto es todo. En su pantalla aparecerá:

CSAVE V2.0

Press REC and PLAY then any key:
Saving FILECOPY.COM block x

Si ha presionado las dos teclas correspondientes (REC y PLAY) en su cassette, puede pulsar cualquier tecla del ordenador para que comience el proceso de copiar. El mensaje "block x" informa sobre el bloque que actualmente se está almacenando en el cassette. Después de que finalice el proceso, en la pantalla se verá:

CSAVE V2.0 finished

Puede ser muy práctico almacenar en cassette, ficheros determinados, que se puedan recuperar más tarde mediante el comando CLOAD, dado que los cassettes son un medio de almacenamiento muy económico, al contrario de los discos de 3", que todavía son muy caros. De este modo puede archivar determinados ficheros. ¿y porqué no? Con las grandes computadóras se hace lo mismo. Regularmente se copian las pilas de discos, en cintas magnéticas mucho más ecómicas.

VII.5 El ensamblador ASM

El disco CP/M, que acompaña a cada unidad de discos DDI-1, que a su vez naturalmente está incorporada en el CPC 664, así como en el CPC 6128, contiene un ensamblador. Podemos acceder a él mediante la llamada ASM. Pero, ¡alto! No se alegre demasiado pronto. Como Vd. quizás sepa, la UCP del Amstrad es un procesador del tipo Z-80. Quizás ya haya aprendido a programar en código máquina con paciencia y mucho esfuerzo. Pues bién, este ensamblador no es un ensamblador para el Z-80, sino para el 8080. Para aquellos entre Vds., que todavía no se aclaren del todo: el 8080 es más antiguo que el Z-80, podría decirse que es su padre. Aunque el Z-80 entienda los programas escritos para su padre, y entienda incluso mucho más, su código mnemónico es diferente. No se trata de una falta de ortografía. Un código mnemónico es el código, del que se sirven los programadores en lenguaje máquina, para codificar sus programas. Vd. seguramente sabe, que el código máquina, y en consecuencia los programas que se escriban en él, se componen exclusivamente de números. El Z-80 por ejemplo entiende el comando &41 y es capaz de ejecutarlo. Sin embargo, para una persona es bastante difícil, imaginar lo que ésto pueda significar. Por ello se han desarrollado unos mnemónicos, que en parte puedan reemplazar estos comandos en código máquina. En el Z-80, el mnemónico para el comando &41, por ejemplo es éste:

```
LD B,C
```

Esto sí que le dice algo al programador de código máquina, y el trabajo se simplifica considerablemente, si uno no se ve obligado constantemente a consultar cualquier tabla, para averiguar las consecuencias que pueda tener por ejemplo, el código &41.

El ensamblador se encarga de traducir estos mnemónicos, que el hombre entiende "perfectamente", en códigos descifrables para la máquina. El procesador por ejemplo, no entiende en absoluto el mnemónico LD B,C. Aquí, el ensamblador actúa como traductor o interface, traduciendo el lenguaje humano al lenguaje máquina. El ensamblador apoya al humano en su trabajo, prestándole algunas ayudas más, a las que aludiré brevemente más tarde.

De modo que ahora ya sabe, lo que es un código mnemónico, y que el antiguo procesador 8080 es el padre del Z-80. También sabe, que el Z-80 domina mayor cantidad de comandos, que el 8080. Sin embargo todos los programas para CP/M tienen que programarse en código 8080, porque también existen otros ordenadores CP/M, que llevan un 8080. Esto no significa otra cosa para un programador del Z-80, que simplemente debe limitarse a los comandos que pueda entender el 8080 (por lo que tiene que prescindir de algunos comandos realmente interesantes del Z-80). Desgraciadamente ésto no es todo, dado que también se han cambiado los mnemónicos al desarrollar el Z-80, a partir del 8080. Recordemos el comando &41. Los dos procesadores hacen lo mismo, cuando reconocen y ejecutan dicho comando: el contenido del registro C, se copia en el registro B. El mnemónico correspondiente para el Z-80 es éste:

LD B,C

La misma (!!) instrucción para el 8080 es:

MOV B,C

Seguramente reconocerá la problemática: sus conocimientos del Z-80 poco le servirán, cuando quiera programar un 8080, aunque se limite a las instrucciones que pueda entender el 8080. Resumiendo: el ensamblador ASM, sólo permite la introducción de instrucciones, que un ensamblador del 8080 pueda entender.

Sin embargo vamos a ocuparnos un poco más de este ensamblador, que nos permite programar ficheros COM propios.

VII.6 El manejo del ensamblador ASM

En la programación de código máquina, a menudo surge el problema de tener que referirse a direcciones determinadas, por ejemplo para cargar, o para almacenar registros. Frecuentemente también hay que efectuar un salto a una dirección determinada, etc. Imagine, que Vd. ha escrito un programa, que comienza en la dirección $5000, y que ahora desea desplazar hacia la dirección $4000. Tendría que cambiar todas las direcciones de salto, a excepción de los saltos relativos. O si ha introducido un comando en el programa, se desplazarían todos los saltos relativos, y en consecuencia también todos los saltos directos, que fuesen a continuación. Es a lo sumo entonces, cuando el programador empieza a subirse por las paredes. Para facilitar este trabajo, los ensambladores se han provisto de la posibilidad de definir marcas (labels), que se insertan en el programa durante el ensamblado (la traducción de los mnemónicos al código máquina). A partir de entonces, ya no es ningún problema, añadir un comando en cualquier parte del programa, o desplazar una rutina, e incluso todo el programa - el ensamblador ya le quita de encima todo el trabajo pesado.

¿Vd. quiere saber, qué aspecto tiene un programa ensamblador de este tipo? No es ningún problema. En su disco CP/M, se ha tenido la amabilidad de guardar un programa ensamblador, con el que también podemos ejercitar el manejo del ensamblador. Vuelva a mirar el directorio de su disco CP/M. Tenemos un fichero, con el nombre de DUMP.COM; se trata, como ya sabemos, de un fichero COM, que podemos ejecutar bajo CP/M, tecleando simplemente:

DUMP <filename>

sería la sintaxis en tal caso. Dump le suministra una impresión del fichero indicado, en formato HEX (dump hexadecimal). No tenga miedo de probarlo:

DUMP FILECOPY.COM

En la pantalla aparecerá el dump hexadecimal del fichero COM filecopy, del mismo programa, por así decir. Puede detener la impresión mediante <ctrl>/S, y continuarla mediante <ctrl>/Q. Mediante <ctrl>/C puede interrumpir la impresión.

DUMP.ASM es el programa en lenguaje máquina, tal y como ha sido codificado por el programador. Puede obtener la impresión de este fichero en la pantalla (o en la impresora). Introduzca:

TYPE DUMP.ASM

En la cabecera del programa reconocerá, que la empresa DIGITAL RESEARCH está en posesión de los derechos de este programa. Ello no es de extrañar, ya que ha sido exactamente esta empresa, la que ha desarrollado CP/M. También apreciará otra diferencia crucial entre ensamblador y código máquina: la posibilidad de incluir comentarios. Dichos comentarios deben precederse de un punto y de una coma, para que el ensamblador sepa, que la traducción finaliza en esta posición. De otra manera, el ensamblador lógicamente intentaría entender el comentario, es decir, que trataría de traducirlo, y entonces seguramente se tiraría de los pelos y exclamaría: ¡¡pero si aquí hay algo que no cuadra!! Aunque el programa le parezca muy largo, créame que es muy corto. Teniendo en cuenta, que los sistemas operativos también se codifican en ensamblador, existen sistemas operativos, cuya impresión ocupa 2000 paginas, y más. También los programas "menores", como por ejemplo el procesador de textos TEXTOMAT en su Amstrad, fácilmente abarcan de 400 a 500 páginas. Ya ve, que la programación en código máquina es algo sumamente aparatoso. Sin embargo, vamos a ensamblar el programa ensamblador, para que en un futuro pueda utilizar el ensamblador Vd. sólo. Tenga en cuenta, que ahora su disco no debe estar protegido contra la escritura, porque también tendremos que escribir sobre el disco, que alberga el código fuente. ¡¿Seguramente ya ha hecho una copia de seguridad de su disco de CP/M?! (De no ser así, ¡hágalo rápidamente!) Ahora vamos a ensamblar "nuestro" programa, en lenguaje máquina:

ASM DUMP

Aquí ya puede apreciar una particularidad llamativa del ensamblador: no necesita indicar ninguna extensión, es decir que no necesita indicar DUMP.ASM, ya que el ensamblador añade la extensión .ASM por si sólo. En su pantalla aparecerá:

```
CP/M ASSEMBLER - VER 2.0
0257
002H USE FACTOR
END OF ASSEMBLY

A>
```

Es rápido nuestro ensamblador, ¿verdad? Ahora se crean dos ficheros:

1) un protocolo bajo el nombre de DUMP.PRN
2) un dump hexadecimal bajo el nombre de DUMP.HEX

Observe primero el protocolo. Aquí se vuelve a listar el programa ensamblador, con todas las informaciones que añade el ensamblador. La primera columna siempre contiene la dirección, sobre la que se está codificando - a no ser que se defina un label (marca): en tal caso la primera columna contiene el valor definido del label. El programa comienza en la dirección $0100. Después de mirar bién el protocolo, o incluso de imprimirlo, pase a ver el fichero siguiente, que ha creado nuestro aplicado ensamblador. Para ello vuelva a introducir (entretanto ya lo sabrá):

TYPE DUMP.HEX

En la pantalla ahora verá una serie de números, que han sido creados por el mismo ensamblador; se trata de códigos del programa. Además todavía se dan algunas informaciones aparte, tales como la dirección, en la que más tarde figurará el código, etc. Este fichero ya es considerablemente más corto, que nuestro fichero DUMP.PRN, ¿verdad? Sólo que todavía no podemos ejecutar este programa en lenguaje máquina, ya que todavía no es ningún fichero COM. Pero existe otro programa, que se encarga de este trabajo. Dicho programa convierte un fichero HEX, en un fichero COM, es decir, en un fichero accesible y ejecutable para CP/M. El programa lleva el nombre **LOAD.** Llamémoslo:

LOAD DUMP

Ya ve, que tampoco aquí hace falta, ni se debe indicar ninguna extensión (código). LOAD añade automáticamente la extensión ".HEX". Sobre la pantalla aparece:

FIRST ADRESS 0100
LAST ADRESS 0212
BYTES READ 0113
RECORDS WRITTEN 03

Por fin lo ha logrado: ha confeccionado un fichero COM, el fichero DUMP.COM, al que podemos acceder con CP/M simplemente introduciendo:

Sin embargo no es del todo fácil programar ficheros COM, dado que en tal caso, uno tiene que estar muy bién familiarizado con el CCP y con el BDOS, porque hay que recurrir a dichas rutinas.

Pero todavía quiero deshacerme de algunas informaciones sobre el ensamblador, para que Vd. pueda manejarlo correctamente (si así lo desea).

Los programas fuente para el ensamblador, tienen que poseer la siguiente sintaxis:

<número de línea><marca>:<comando><argumento>;<comentario>

El <número de línea> solamente es opcional, es decir que no tiene que incluirse a la fuerza (como en nuestro fichero ejemplo). El ensamblador lo ignora de todos modos, y sólo está previsto, porque algunos editores incluyen estos números automáticamente, como por ejemplo el editor ED, en el disco de CP/M. La <marca> naturalmente tampoco es obligatoria, pero puede ponerse en este lugar. Lo que sí es obligatorio en una línea, que no consista exclusivamente de un comentario, es el <comando> y el <argumento>. A éstos eventualmente se les añade un comentario.

Antes de la codificación, el ensamblador convierte todas las letras minúsculas en mayúsculas, tal y como ya lo conoce de CP/M. Esto si que simplifica la programación de forma considerable. La excepción de la regla son las letras minúsculas, que figuren entre comillas, y que compongan un texto fijo, como por ejemplo la instrucción:

```
DB 'file dump version 1.4$'
```

Los diveros componentes de la línea en ensamblador, tienen que ir separados por un espacio como mínimo. Es usual empezar con la <marca> en el extremo izquierdo, y luego saltar al resto de los componentes mediante un tabulador del procesador de textos, con el que esté escribiendo los códigos fuente en ensamblador. Como ya se ha dicho, ésto no es absolutamente necesario, pero ayuda mucho a la claridad del programa, dado que los componentes de misma naturaleza, siempre figuran uno debajo de otro. un ejemplo:

```
DISKR:    ;READ DISK FILE RECORD
          PUSH H! PUSH D! PUSH B
          LXI  D,FCB
          MVI  C,READF
          CALL BDOS
          POP B! POP D! POP H
          RET
```

Encontrará este pasaje en nuestro programa DUMP. DISKR es una <marca>, que se identifica mediante dos puntos. Le sigue el <comando> y el <argumento>, siempre uno bajo el otro. En una línea también tenemos un comentario, identificado por un punto y una coma ";". Esta pequeña rutina también le revela otra peculiaridad del ensamblador ASM: los diversos comandos en ensamblador, pueden separarse mediante signos de exclamación.

En ASM pueden marcarse comentarios mediante un punto y una coma, o definirse toda una línea como comentario, si en la primera columna se pone un asterisco. Los desarrolladores de ASM lo han previsto así, para salvaguardar una cierta compatibilidad con los ensambladores ya presentes del 8080. De tal manera pueden enmarcarse muy bién las diversas rutinas en los programas ensambladores, como por ejemplo:

```
*****************************************

***     read disk file record       ***

*****************************************
```

Cuando utilice marcas, tenga cuidado de no utilizar palabras clave fijas. Son palabras clave por ejemplo los códigos mnemónicos del procesador 8080, es decir, palabras como MVI,MOV,STA etc.

Con el ensamblador ASM, al igual que con muchos otros, también puede determinar la posición momentánea del contador del programa PC durante el ensamblado. Ello se realiza automáticamente en las marcas del programa, tal y como se muestra en el siguiente ejemplo:

```
posit: EQU $
```

Los argumentos que aparezcan en el ASM, pueden concatenarse mediante diversos operandos aritméticos, por ejemplo:

```
MVI A,12+2*3
```

Los operandos aritméticos son:

+X	número positivo
-X	número negativo
X+Y	suma de dos valores de 16 bits
X-Y	resta de dos valores de 16 bits
X*Y	producto de X*Y
X/Y	división de los argumentos
X MOD Y	función restante de la división X/Y

También pueden utilizarse estos dos comandos de desplazamiento:

X SHL Y	desplaza a la izquierda el valor de 16 bits X por Y posiciones. Los bits desbordantes se pierden.
X SHR Y	desplaza a la derecha el valor de 16 bits X por Y posiciones. Los bits desbordantes se pierden.

Además están los operadores lógicos:

NOT X	negación lógica del argumento X
X AND Y	concatenación lógica "y" de los argumentos X e Y
X OR Y	concatenación lógica "o" de los argumentos X e Y
X XOR Y	concatenación lógica "o exclusivo" de los argumentos X e Y

Todas estas posibilidades, a primera vista quizás le parezcan superfluas. Pero no lo son, sino todo lo contrario: son muy útiles. Si por ejemplo quiere cargar el acumulador con el byte de mayor peso de una dirección, simplemente puede hacerlo, desplazando el valor lógicamente por 8 bits hacia la derecha:

MVI A,Label SHR 8

Si sólo desea tener los 8 bits inferiores, tiene que excluir expresamente los 8 bits superiores, ya que de otra forma cargaría el registro de 8 bits del acumulador, con un valor de 16 bits. La exclusión se efectúa de esta forma:

MVI A,Label AND OFFH

Se ha señalado esta aplicación, porque seguramente es la más frecuente, sin embargo puede pensarse en miles de aplicaciones.

También existen los pseudo-códigos operacionales

ORG,EQU,END,DS,DB,DW,SET,IF y ENDIF.

(Los pseudo-códigos operacionales son comandos del ensamblador, que éste lee y procesa durante el ensamblaje. Se les llama "pseudo", porque aunque aparezcan en el programa ensamblador al igual que los códigos normales, sin embargo no se ensamblan, ni pueden ser entendidos por la UCP.)

No vamos a entrar en detalles acerca de los diversos pseudo-códigos operacionales, ya que ésta no es la finalidad de un libro sobre CP/M, de modo que sólo relataremos brevemente su efecto y su funcionamiento.

ORG <dirección inicial>

Este comando define la dirección inicial, o de origen del programa a ensamblar. ORG siempre debería ser el primer comando de un programa.

EQU <valor>

EQU asigna un valor fijo a un label (una marca). El <valor> también puede ser una expresión aritmética.

Ejemplo: **Diskan: EQU DISKR+055H**

DS

Este comando reserva un espacio definido en el programa, para por ejemplo depositar datos en él. El contador del programa se incrementa de forma apropiada.

Ejemplo: **Diskm: DS 100**

En este ejemplo se reservan 100 bytes, siendo la dirección inicial Diskm, y la dirección final Diskm+99. El siguiente comando comenzará en Diskm+100.

DB

Aquí se almacenan en la memoria bytes individuales a definir. Los diversos bytes deben ir separados por una coma. También se admiten cadenas de caracteres.

Ejemplos:
```
Disktxt: DB "Introduzca disco"
Diskt2: DB 23,32,0,3,122
```

DW

DW define palabras, es decir valores de 16 bits. También aquí pueden separarse los diversos valores de 16 bits mediante una coma.

Ejemplo: **Diskadr: DW 03AB9H,marca1**

END

END define el final del programa a ensamblar. Para cometidos especiales, es posible introducir aquí también una dirección inicial, para la ejecución del programa ensamblado.

SET

Mediante el comando SET, también se asigna un valor a una marca, de forma similar a EQU. Hay una diferencia: mientras que la asignación mediante EQU permanece fija, las marcas definidas mediante SET, todavía pueden modificarse durante el ensamblado. Las marcas definidas por SET, tampoco aparecen en la columna izquierda durante el ensamblado.

Ejemplo:

```
marca1: SET alfa
marca2: SET marca1 + 32
```

Pero con ello todavía no hemos tocado los límites de ASM. Al igual que muchos otros ensambladores potentes, también ASM dispone de la posibilidad del ensamblado condicional, es decir, de decodificar y traducir al lenguaje máquina un determinado pasaje en ensamblador, solamente bajo ciertas condiciones. Ello se consigue utilizando los comandos IF y ENDIF. (Sin embargo la experiencia nos revela, que dichos comandos se utilizan muy poco.)

Supongamos que Vd. desee escribir un programa, que lea dos valores, para sumarlos o multiplicarlos a continuación. El algoritmo es el mismo para los dos programas. Ahora puede escribir un programa, que durante el ensamblado consulte un flag, que le indique al programa, si debe sumar o multiplicar.

```
;
multi  EQU  0 ;el programa debe sumar
ORG 0100H
;
;lee valores
CALL lectura
;
IF multi
  CALL multir ;llamar rutina de multiplicación
ENDIF
IF NOT multi
  CALL sumar  ;llamar rutina de suma
ENDIF
CALL emisión
END
```

Si ahora desea, que la rutina multiplique, debe poner en la primera línea el valor para multi en NOT 0, y ésto es todo.

Después de haber tratado el ensamblador tan ampliamente, volveremos a ocuparnos brevemente de la llamada a ASM:

Como Vd. recuerda, no necesita indicar la extensión ".ASM" cuando efectúe la llamada. Los dos ficheros <nombre>.PRN y <nombre>.HEX se crean en el disco. ¡Pero ésto puede impedirse! Además puede indicar la unidad, de la que deba cargarse el fichero a ensamblar, y en qué fichero deban depositarse los dos ficheros a crear. La síntaxis del comando ASM es ésta:

ASM <fichero>.<fichero-ASM><fichero-HEX><fichero-PRN>

A excepción del nombre de fichero <fichero>, todas las demás indicaciones son opcionales, pero tienen que ir por el órden arriba indicado, si van a ser incluídas. Ahora Vd. dispone de las siguientes opciones:

para fichero-ASM: A o B
A o B es la unidad en la que se encuentra el fichero original.

para fichero-HEX: A, B o Z.
También aquí, A y B representan la unidad, en la que hay que depositar el fichero, bajo el nombre de <fichero>.HEX. Asimismo existe la posibilidad, de decirle al ensamblador, que no genere ningún fichero-HEX. Esto tiene sentido, cuando por ejemplo Vd. solamente quiera hacer una prueba, para por ejemplo, controlar la síntaxis. Entonces debe introducir la opción 'Z' en la segunda posición.

para fichero-PRN: A, B, X o Z
A y B de nuevo indican la unidad, en la que debe crearse el fichero <fichero>.PRN. Sin embargo, dicho protocolo también puede suprimirse, poniendo una 'Z' (véase arriba). También puede hacerse mostrar el fichero protocolario en la pantalla, mediante la opción 'X'.

Supongamos que Vd. desea cargar el fichero original, de nombre Asterix, desde la unidad B, sin que por ello quiera

obtener el fichero protocolario, ni tampoco el fichero-HEX, para por ejemplo comprobar la síntaxis de su programa. Entonces teclee:

ASM Asterix.BZX

Ya ve que es muy sencillo. Pero concluyamos ya el capítulo sobre el ensamblador. Estoy seguro, que ha recibido bastantes informaciones sobre ASM.

VII.7 Operaciones con SUBMIT y XSUB

A menudo sucede, que hay que repetir y repetir determinadas secuencias de comandos. Imagine, que desea ensamblar un determinado fichero original, convirtiendo luego el fichero-HEX en un fichero-COM, y finalmente ejecutar este nuevo fichero-COM, para probarlo.

Es exactamente entonces, cuando se hace recomendable establecer un fichero SUBMIT, que contenga todas las instrucciones necesarias, y las ejecute automáticamente. SUBMIT no significa otra cosa que transferir: el programa SUBMIT le transfiere una cadena de comandos al CCP (Console Command Processor = procesador de los comandos de la consola), que es el responsable de las instrucciones que se efectúan por el teclado y de su ejecución. Esto sucede de la siguiente manera:

Si Vd. le dice a SUBMIT, cual fichero ha de transferirse en forma de fichero SUBMIT, SUBMIT crea un fichero temporal $$$.SUB. En dicho fichero temporal se guardan todos los datos del fichero SUBMIT, con algunos suplementos. Antes de recibir una instrucción del teclado, CCP controla si no existe el fichero temporal $$$.SUB en la unidad en curso. De ser así, se carga una línea de instrucciones en el tampón del CCP, borrándose a continuación dicha línea del fichero. Después de copiar la línea de instrucciones, las instrucciones se ejecutan. Ello se repite, hasta que el fichero $$$.SUB esté vacio. Sólo entonces vuelve a aceptarse un comando del teclado.

(Dado que en la prensa especializada, que en gran mayoria es la inglesa, nunca encontrará la expresión fichero, a partir de ahora, vamos a sustituir en lo posible esta expresión por file).

Como Vd. sabe, el comando DIR no le informa acerca del espacio libre que queda en el disco. Dicha información se suministra con el comando STAT. Ahora confeccionaremos un fichero SUBMIT, que reuna el comando DIR y el comando STAT, para que mediante un solo comando, podamos conseguir las informaciones sobre la capacidad y el contenido del disco. Para ello confeccionaremos el fichero SUBMIT, mediante un procesador de textos, por ejemplo con ED, y lo llamaremos DISC.SUB. Todos los ficheros SUBMIT deben llevar la abreviación ".SUB", ésto es ley suprema.

Pues bién, nuestro primer fichero SUBMIT tendrá este aspecto:

```
STAT
DIR
```

Hemos de admitir que es un fichero SUBMIT muy corto, pero ¡nadie nace sabiendo! Vamos a ejecutar este primer fichero SUBMIT mediante:

SUBMIT DISC

Ya ve, que no debemos indicar la extensión ".SUB". Ahora nuestra unidad de disco, da la impresión de estar muy atareada, pero nosotros también sabemos que están ocurriendo muchas cosas. (confección del fichero $$$.SUB, copiado del contenido de DISC.SUB, borrado de línea por línea de las instrucciones del fichero $$$.SUB etc.) Y funciona.

Pero recordemos, qué es lo que habíamos planeado. Queríamos ensamblar un file, convertirlo en un COM-file mediante el comando LOAD, y ejecutarlo a continuación. Esta secuencia de instrucciones, más o menos tendría este aspecto:

```
ASM <filename>.<options>
LOAD <filename>
<filename>
```

Las expresiones <filename> y <options> prácticamente representan variables - ¿pero como pueden rellenarse? Los que desarrollaron el programa SUBMIT, reconocieron la problemática, y se les ocurrió algo, que permite confeccionar los ficheros SUBMIT de forma más confortable. Al efectuar la llamada al fichero SUBMIT, Vd. puede transferirle dichas expresiones, si las separa como mínimo por un espacio. Las expresiones pueden ser simples números, o bién conceptos complejos, como por ejemplo, nombres de fichero. Las expresiones se numeran, empezando con el número uno. En el fichero SUBMIT, hay que marcar estas expresiones con un signo de dólar ($), seguido directamente del número de la expresión. Pueden repetirse a voluntad las expresiones individuales en el fichero SUBMIT. Cuando se crea el fichero $$$.SUB, las variables se sustituyen automáticamente por las expresiones en cuestión, de modo que el CCP puede procesarlas perféctamente. Nuestro fichero, en consecuencia cobra el siguiente aspecto:

```
ASM $1.$2
LOAD $1
$1 $3
```

Ya ve, que la primera introducción, es decir la primera expresión aparece tres veces en el fichero SUBMIT. Si por ejemplo ejecutamos este fichero SUBMIT con nuestro programa DUMP, sin desear un fichero de protocolo (tiene que confeccionarse un fichero HEX, dado que LOAD lo necesita), entonces la llamada podría ser ésta, si hemos almacenado el fichero SUBMIT, bajo el nombre de "ASSEM.SUB":

SUBMIT ASSEM DUMP AAZ DUMP.COM

La última indicación señala el file que debe ser ensamblado, en nuestro caso el mismo DUMP.COM. Si lo ha probado, y todo funciona, entonces ha dado el primer paso en la utilización de los ficheros SUBMIT. ¡El primer paso, porque SUBMIT sabe hacer más cosas!

Quizás ya se haya preguntado: ¿qué hacer, si aparece, o debe aparecer un signo de dólar dentro del fichero SUBMIT? Muy sencillo: se hace lo que en casos similares suele hacerse en informática, es decir que se escriben dos signos de dólar seguidos. Si Vd. por ejemplo quiere borrar todos los ficheros temporales, normalmente lo haría a través de la consola mediante:

ERA *.$$$

En un fichero SUBMIT, ésto cobraría un aspecto más inquietante:

ERA *.$$$$$$

pero significaría lo mismo. Sin embargo, dado que los signos de dólar no son muy frecuentes, el trabajo adicional se reduce a un mínimo.

Si se copia una línea de instrucciones del file $$$.SUB al registro del CCP, dicha línea primero se presenta en la pantalla. Acto seguido se consulta el teclado, comprobando si se ha pulsado una tecla. Si es así, o si la instrucción en el registro de instrucciones no se ejecuta, o no se ejecuta correctamente, SUBMIT interrumpe la ejecución, borrando el FILE $$$.SUB.

Si una línea en un fichero SUBMIT comienza con un carácter, que normalmente no se permite para el nombre de un fichero, entonces dicha línea se presentará en la pantalla, pero no se ejecutará. De este modo es fácil añadir comentarios o instrucciones para el usuario, que se presenten en la pantalla. Por ejemplo sería posible, incluir en nuestro

fichero SUBMIT 'ASSEM.SUB' lo siguiente:

```
:Se está ensamblando el fichero.
:Tenga paciencia por favor!
ASM $1.$2
LOAD $1
$1 $3
```

Y ya se ha conseguido avivar un poco el fichero SUBMIT, durante su ejecución. Los caracteres que provocan ésto, son:

< > ; : . , ? = * -

Sin embargo en SUBMIT, dichos términos se limitan al CCP, lo que quiere decir que no se pueden hacer prescripciones para una programa como PIP o ED. A veces, sin embargo ésto también sería deseable, para copiar determinados ficheros mediante PIP, o dar unas determinadas instrucciones a ED. Pues bién, también hay solución para ello. A este efecto existe el programa XSUB, que capta todas las entradas efectuadas a través de la consola, y que se encarga de las consultas del fichero $$$.SUB. Si por ejemplo queremos dar algunas indicaciones a PIP (lo que también puede hacerse directamente), ello podría tener este aspecto:

```
XSUB
PIP
b:=a:*.COM
```

En la primera línea debe pulsar primero RETURN, para abandonar la rutina transitoria PIP. Con PIP, esta aplicación todavía no tiene mucho sentido, dado que el comando podría darse de forma diferente, por ejemplo: **PIP b:=a:*.COM** para copiar todos los ficheros COM de la unidad A: a la unidad B:. Sin embargo en ED dicha aplicación se convierte en muy útil. Vd. vé, que basta con llamar XSUB una vez en la primera línea. Las instrucciones que normalmente se dan con el teclado, ahora provienen del fichero $$$.SUB.

Las antiguas versiones de CP/M 2.2 todavía ocultaban un error en el programa SUBMIT. Cuando se transferían datos mediante XSUB, no podían incluirse caracteres de control, a pesar de que ésto estuviera previsto en el manual. Ello es muy molesto, ya que no pocas veces se quiere o debe emitir el carácter de control Z, en PIP o en ED. Pero como este error ha sido suprimido en la versión de CP/M 2.2, no tiene que preocuparse por ello.

Espero que haya comprendido la aplicación, y el manejo de los ficheros SUBMIT suficientemente, como para poder crear sus propios ficheros SUBMIT, en caso necesario.

Hemos hablado sobre el ensamblador y sobre los ficheros SUBMIT. Todavía deberíamos mencionar brevemente un desensamblador, que también se halla sobre el disco. Se accede a dicho desensamblador mediante **DDT**. Si por ejemplo desea ver el fichero de instrucciones FILECOPY.COM desensamblado, teclee simplemente:

DDT filecopy.com

Antes de apretar <RETURN>, no debe olvidar introducir el comando que lo desvia todo hacia la impresora, es decir <ctrl>/P.

Desensamblar es lo contrario de ensamblar: el código máquina se traduce a mnemónicos, que son más aptos para los humanos.

VII.8 La distribución de la memoria

Hasta ahora hemos mencionado varias veces conceptos como BDOS, CCP, BIOS etc. No se le habrá ocultado el hecho de que dichos conceptos realmente son muy importantes, considerando que además la tapa del libro los muestra de forma tan prometedora. Seguramente algunos de los lectores ya se habrán preguntado, en que partes de la memoria se ocultan todas estas cosas. Repitamos antes que nada, lo que debemos entender bajo estos conceptos:

CCP

CCP representa Console Command Processor (procesador de comandos de la consola). Esta zona regula las entradas provenientes del teclado, e incluye los comandos incorporados, a saber: DIR, ERA, REN, SAVE, TYPE y USER.

BDOS

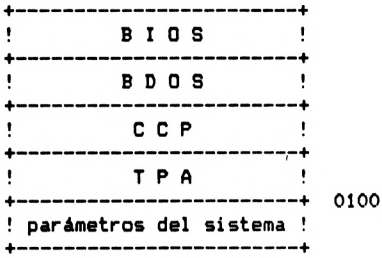
BDOS significa Basic Disk Output System (sistema Basic de salidas disco). Esta parte regula el intercambio de datos entre el ordenador, y las unidades de discos.

BIOS

Es la abreviación de Basic Input/Output System (sistema Basic de E/S). BIOS y BDOS trabajan siempre muy juntos, por lo que ambos a menudo también se mencionan con un nombre en común: FDOS.

TPA

La zona en la memoria del ordenador, que se reserva a la programación para el usuario, se denomina TPA. Los programas a ejecutar, y los datos que han de ser tratados por estos programas, se administran en este registro.



Esta es la distribución global de los registros de CP/M. La zona del usuario comienza en la dirección $0100.

Dado que el hecho de explicar, cómo programar en CP/M, reventaría el marco de este libro, sólo aludiremos brevemente a la programación bajo CP/M, con la ayuda de las rutinas BDOS y BIOS

El BDOS incorpora una colección de subrutinas, que básicamente posibilitan el trabajo con ficheros en disco, y otros dispositivos periféricos. Dado que CP/M ya está un poco "entrado en años", todavía existen por ejemplo algunas rutinas de entradas y salidas para tarjetas perforadas. Claro que también hoy día todavía se usan frecuentemente dichas rutinas, pero en nuestro CPC podemos olvidarnos tranquilamente de ellas.

Para hacer una llamada a una rutina BDOS, hay que seguir las siguientes normas estandarizadas:

En el registro C del procesador, se transfiere el código de la rutina deseada a BDOS. En el registro E, y si se trata de valores de 16 bits, en el par de registros DE, se comunican datos, en caso necesario a la subrutina. Cuando los registros

estén debidamente ocupados, se salta a BDOS a la dirección 5, mediante una llamada de subrutina (CALL). A continuación BDOS calcula la dirección de la rutina deseada, mediante el registro C. Ya vé, que de esta manera, los programas pueden ejecutarse realmente en todos los ordenadores, porque se siguen unas reglas fijas. Todo lo que dependa del ordenador, como puedan ser las entradas, y las salidas desde el teclado, y el control de la pantalla, se controla por rutinas BDOS especialmente desarrolladas para cada ordenador. De modo que durante el desarrollo de un nuevo ordenador, "solamente" hay que escribir estas rutinas, con lo que ya funcionan todos los programas de CP/M para dicho ordenador.

Claro que deben cumplirse dos condiciones. El ordenador debe disponer de un procesador 8080, o de uno compatible, y tiene que ser posible leer estos programas existentes. Si ésto último no fuera posible, por lo menos teóricamente los programas de CP/M también podrían funcionar. Sin embargo, según la experiencia, la transferencia de progamas ya existentes, es el problema menor. Ello puede efectuarse a) mediante la programación del controlador (véase arriba), o b) transfiriendo los programas existentes mediante un programa relativamente sencillo a través de un interface común a dos ordenadores, por ejemplo el RS 232. Pero aún cuando dicho interface común no existiera, siempre habrá algún manitas, que encontrará un camino; por ejemplo, el port de las palancas de mando, puede ser muy útil para la transferencia de un programa.

Un colega de la casa DATA BECKER incluso ha llegado a transferir datos del CBM 8296 al CPC, realizando una conexión mediante dos cables a través del port del usuario en un puente sobre la platina del CPC, en la que pueden "marcarse" los diferentes fabricantes del CPC. Según el punto de soldadura interrumpido (puente marcado), se indica un fabricante diferente durante el arranque en frío.

Pero volvamos al BDOS. Si BDOS devolviese valores de la rutina BDOS al programa principal, en el acumulador del procesador se depositarían resultados de 8 bits, y los valores de 16 bits para el retorno, se almacenarían en el par de registros HL. En retornos de 16 bits, además el bit de menor peso vuelve a encontrarse en el acumulador, y el bit de mayor peso en el registro B.

Si por ejemplo desea emitir un carácter a través de la

consola, debe utilzar para ello la rutina BDOS número 2. Vamos a emitir un interrogante (en mnemónicos del Z-80):

```
LD   C,2       ;salida por consola
LD   E,"?"     ;cargar interrogante
CALL 5         ;saltar a BDOS
```

Si desea dedicarse seriamente a la programación en código máquina bajo CP/M, existe literatura especializada, especialmente para los programadores con CP/M. Aquí sólo le damos una lista de las rutinas BDOS con parámetros de entrada y de salida, para que experimente. La programación seria, seguramente exige conocimientos bastante más profundos, sobre las diversas rutinas.

nombre de rutina	C	recibe	suministra
activar arranque en caliente	0		
entrada por consola	1		A:entrada
salida por consola	2	E:carácter	
leer cinta perforada	3		A:carácter
perforar cinta	4	E:carácter	
carácter a impresora	5	E:carácter	
entrada y salida directa por consola	6	E: 255 E:carácter	A:0 A:carácter
consultar byte de E/S	7		A:IOBYTE
poner byte de E/S	8	E:IOBYTE	
emitir cadena de caracteres	9	DE:=>cadena caract.	
entrada desde buffer	10	DE:=>buffer	A:carácter

nombre de rutina	C	recibe	suministra
estado de la consola	11		A:=0 (ninguna tecla)
versión CP/M	12		HL:versión
reset del sistema de disco	13		
evaluar unidad de referencia	14	E:LW-número	
abrir fichero	15	DE:informes	A:255(error)
cerrar fichero	16	DE:informes	A:255(error)
buscar primera anotación	17	DE:informes sino:	A:255(error) A:caracter
anotación siguiente	18		A:255/carac.
borrar fichero	19	DE:informes	A:255(error)
leer grabación	20	DE:informes	A:0 (OK)
escribir grabación	21	DE:informes	A:0 (OK)
crear fichero	22	DE:informes	A:255(error)
modificar nombre de fichero	23	DE:informes	A:255(error)
evaluar unidades activas	24		HL:vector LW
evaluar unidad de referencia	25		A:no. unidad
fijar buffer de datos	26	DE:buffer	
evaluar tabla de ocupaciones	27		HL:dirección

proteger unidad de referencia	28		
unidades protegidas	29		HL:vector LW
poner opciones de fichero	30	DE:informes	A:255(error)
evaluar parámetros de disco	31		HL:dirección
administrar número de usuario	32	E:255	A:número
		E:número	
leer grabación	33	DE:informes	A:0 (OK)
escribir grabación	34	DE:informes	A:0 (OK)
evaluar extensión del fichero	35	DE:informes	(en buffer)
poner descriptor	36	DE:informes	(en buffer)
restaurar unidad	37	DE:vector LW	A:0

Esta tabla solamente pretende mostrar, qué aspecto más o menos puede tener una programación con BDOS, y de qué posibilidades se dispone con BDOS. Como ya se ha mencionado, naturalmente no es suficiente para la programación, ya que ello requiere conocimientos internos sobre las diversas rutinas.

Si BDOS ya incorpora rutinas como "proteger unidad de referencia", las tareas de BIOS todavía son más específicas. BIOS, al igual que BDOS se compone de una lista de rutinas - que sin embargo sirven exclusivamente a la entrada y salida de datos. Parcialmente en BDOS se encuentran rutinas, que realizan las mismas tareas que algunas rutinas del BIOS, como por ejemplo la evaluación del estado de la consola.

BIOS participa de forma decisiva en el éxito de CP/M, ya que permite a los programas de CP/M, ejecutar mediante rutinas especiales, las entradas y salidas por consola, que dependen totalmente del ordenador.

BIOS se divide en cuatro grandes sectores:

- interface para BDOS y los programas CP/M (por ejemplo en los ficheros SUBMIT),
- interface para los medios de almacenamiento externos (floppys),
- entrada y salida a través de los periféricos controlados por BDOS,
- tampón para datos, que luego son puestos a disposición en el momento oportuno (ficheros SUBMIT).

Como Vd. sabe, BDOS y BIOS pueden desplazarse en la memoria. Las rutinas del BDOS se activan mediante la transferencia del número de rutina al registro C - el salto se efectúa a la dirección fija &0005. La llamada de rutinas de BIOS, es un tanto diferente: existe una tabla de saltos de órden preestablecido. Sin embargo, la dirección inicial de dicha tabla puede desplazarse. Pero veamos primero la tabla de saltos:

```
00+offset: JMP BOOT    ;arranque en frío
03+offset: JMP WBOOT   ;arranque en caliente
06+offset: JMP CONST   ;consulta estado consola
09+offset: JMP CONIN   ;entrada de consola
0C+offset: JMP CONOUT  ;salida por consola
0F+offset: JMP LIST    ;salida por impresora
12+offset: JMP PUNCH   ;perforador de cinta
15+offset: JMP READER  ;lector de cinta perforada
18+offset: JMP HOME    ;cabezal a pista 0
1B+offset: JMP SELDSK  ;seleccionar unidad
1E+offset: JMP SETTRK  ;fijar pista
21+offset: JMP SETSEC  ;seleccionar sector de grabación
24+offset: JMP SETDMA  ;seleccionar tampón de datos
27+offset: JMP READ    ;leer sector de grabación
2A+offset: JMP WRITE   ;escribir sector de grabación
```

```
2D+offset: JMP LISTS   ;consulta estado impresora
30+offset: JMP SECTRAN ;traducir número de la grabación
```

Offset representa el desplazamiento en la memoria, al que me referiré con más detalle un poco más tarde. Si no se necesita una de las funciones indicadas - en el caso del CPC, ésto serían las dos rutinas referentes a la cinta perforada -, las direcciones correspondientes simplemente se rellenan con

RET ! NOP ! NOP

para que la memoria detrás de dicha rutina no se desplace. Un salto a esta rutina provocaría un RETURN inmediato - por tanto no pasaría nada. Para poder evaluar la dirección inicial del BIOS debe saberse, que la dirección 0 de la página básica del sistema, contiene un salto al vector de arranque en caliente del BIOS (el segundo vector en la tabla). Por ello se hace fácil, calcular la dirección inicial. Vamos a hacerlo. Primero ponga CP/M en marcha, y luego cargue el desensamblador:

DDT

Para desensamblar la memoria a partir de la dirección 0, introducimos lo siguiente en el ordenador:

10000

y obtendremos una respuesta inmediata de DDT:

```
0000 JMP AD03
0003 ADD C
0004 NOP
0005 JMP
0008 JMP 8F00
0008 JMP B982
```

etc. Lo que nos interesa a nosotros, es la primera línea: se efectúa un salto a la dirección &AD03, y por lo tanto la tabla de saltos de BIOS comienza en &A300. Esto ya es todo lo que tenemos que saber, para nuestros programas. Si queremos saltar a un vector, obtendremos la dirección de salto mediante:

dirección:=número de salto * 3 + &A300

Igual que con las rutinas BDOS, también con las rutinas BIOS Vd. puede enviar y recibir parámetros. Si deben suministrarse valores a las rutinas de BIOS, se depositan valores de 8 bits en el registro C, y valores de 16 bits en el par BC. Aquí ya se aprecia una diferencia respecto a las rutinas de BDOS, que esperan los parámetros en el registro E, o en el par DE. Los parámetros que sean devueltos por las rutinas de BIOS al programa principal, se depositan paralelamente a las rutinas de BDOS en el acumulador, cuando se trata de valores de 8 bits, y en el par de registros HL, cuando se trata de 16 bits. Los manitas seguramente ya han descubierto una posibilidad muy atractiva: dado que se accede a las diversas rutinas de BIOS mediante vectores contenidos en la RAM, es posible cambiar dichos vectores, para hacer que apunten por ejemplo, a una rutina propia,la cual en caso dado puede volver a bifurcarse hacia el programa original.

Vamos a hablar sobre algunas propiedades más de la página básica (dirección &0000 a &0100) del sistema. Mediante la instrucción 10 hemos desensamblado la zona de &0000 a &0016. Hay algunos comandos JMP, que destacan: es especialmente importante el salto a la dirección 5 - ahora debería Vd. caer, porque ¡es a la dirección 5, a la que hemos dirigido nuestra llamada de subrutina, cuando queríamos acceder a una rutina de BDOS! Por lo tanto, en nuestro CPC, BDOS debe

comenzar en la dirección &8F00, dado que la instrucción de salto de la dirección &0005, apunta hacia allí. Entonces veamos más de cerca el contenido de esta dirección:

18F00

Para que no sea tan fácil, inmediatamente tenemos que seguir saltando, esta vez a la dirección &95A2. No vamos a renunciar ahora por lo que también efectuaremos este salto:

L95A2

Para ser breves: todavía se salta a la dirección &9F06, y acto seguido a &9F11. Sólo entonces comienza poco a poco, la decodificación del registro C, para evaluar la rutina a la que deba saltarse. Seguramente nos llevaría demasiado lejos, seguir este proceso. Con la ayuda de DDT, y durante una sesión de noche en su habitación, Vd. mismo puede hacerlo tranquilamente.

VII.9 El comando SETUP

El comando SETUP sólo existe en CP/M 2.2; sin embargo es interesante para los propietarios del CPC 6128, dado que en el 6128 también puede ejecutarse CP/M 2.2.

¿Quizás también a Vd. no le agrade el color de fondo, que normalmente ofrece CP/C? O quizás desee, que el CPC no le salude con el impersonal mensaje

CP/M 2.2 - Amstrad Consumer Electronics plc.

Entonces el comando SETUP está hecho a su medida. SETUP sirve para amoldar al CPC-CP/M a sus gustos personales. SETUP se controla mediante menu, y consulta los diversos puntos en la pantalla en forma de diálogo. De esta manera puede conseguir, que CP/M se comporte de forma mucho más amable con Vd., saludándole por ejemplo con un

Hola viejo - otra vez me necesitas?

Pero llamemos primero el comando SETUP. Así trataremos de los diversos puntos con mayor exactitud.

Después de introducir SETUP, en la pantalla aparece lo siguiente:

SETUP V2.0

**** Initial command buffer: empty**

Is this correct (Y/N):-

La pregunta **Is this correct (Y/N)** se formulará en cada componente del comando SETUP. Si responde a ella con Y en representación de sí, los parámetros que se indican serán incorporados, y si responde N por no, puede efectuar la corrección oportuna.

Ya ve, que nuestro "Initial command buffer" está vacio (empty). Si escribe algo en el buffer de comandos iniciales, dicho buffer se ejecutará, cuando se incorpore a CP/M. Si por ejemplo después de copiar CP/M, primero desea ver un directorio del disco del sistema para tener una relación de los comandos transitorios, puede hacerlo mediante este Initial command buffer. De modo que Vd. responde a la pregunta "Is this correct", con un rotundo "NO", para que pueda adaptar su Initial command buffer. Después de pulsar la tecla N, en la pantalla aparecerá:

Enter new initial command buffer:-

Deseamos llenar nuestro buffer con el comando DIR. Para que pueda ejecutarse una línea, ésta naturalmente ha de terminar con un retorno de carro - pero ¿como se puede introducir un retorno de carro? Para todos los caracteres ASCII menores que 32, deben utilizarse las secuencias de control de ^A a ^Z, así como ^(,^/,^),^> y ^-. Para poder introducir el retorno de carro, tiene que emplear el carácter ^M. (El retorno de carro posee el código ASCII 13, ¡y la M es la letra número 13 del alfabeto!). Por lo tanto, introduzca lo siguiente en el Initial command buffer:

Después de pulsar la tecla <ENTER>, vuelve a aparecer la pregunta **Is this correct (Y/N):-**, a lo que ahora puede responder con una Y por sí.

Lo siguiente que se presenta y se consulta, es el SIGN-ON-STRING. El SIGN-ON-STRING contiene toda una gama de caracteres de control, que regulan el modo de 80 columnas, así como los colores de fondo y del marco. También aquí, los caracteres ASCII menores que 32 deben introducirse mediante el <código> ^. Los caracteres ASCII correspondientes al color de fondo, modo de 80 columnas etc., se encuentran en la tabla de caracteres ASCII del manual del usuario. Aquí es donde Vd. puede crear su propio mensaje de presentación, como por ejemplo, nuestro "Hola viejo...".

Cuando haya definido el Initial command buffer, así como el SIGN-ON-STRING, puede cambiar la asignación del teclado, o incluir caracteres. Para ello existe la opción

Keyboard translations

Las instrucciones que ponga en esta opción, son idénticas al comando **KEY DEF** del BASIC. Las tablas de códigos correspondientes figuran en el manual. También la siguiente opción, **Keyboard expansions**, tiene los mismos efectos que el comando **KEY** en BASIC.

A continuación se le muestran las asignaciones estándar de los dispositivos de entrada/salida, que puede editar a su vez. Siguen algunas indicaciones técnicas del floppy (retardo del motor del floppy etc.), así como indicaciones para los canales de entrada/salida del procesador. Después de confirmar todas estas indicaciones mediante el teclado, se le formulará la pregunta:

Do you want to update your System disc (Y/N):-

Si responde afirmativamente con Y, todas sus indicaciones se archivarán en disco (si pulsa "N" todas las indicaciones se perderán). La última pregunta que CP/M le hace, es:

Do you want to restart CP/M (Y/N):-

Mediante esta posibilidad puede comprobar directamente las consecuencias de sus modificaciones en la pantalla, si responde afirmativamente con una "Y". Entonces CP/M se copia de nuevo, y si ha modificado el SIGN-ON-STRING, o el Initial command buffer, dicha modificación se ejecutará inmediatamente.

Seguramente ahora ya se habrá despertado su interés por CP/M. De hecho es bastante difícil, que CP/ se quede sin alicientes, especialmente cuando uno mismo pretende programar bajo CP/M. Esperamos sinceramente que este libro le haya servido, y le siga sirviendo de ayuda.

VIII. Corrección de PIP

VIII.1 Descripción del error

El comando PIP tiene un pequeño detalle, que es capaz de sacarle a uno de quicio. Cuando se trabaja con PIP para copiar ficheros, y se cambia el disco, sin realizar un arranque en caliente mediante CTRL-C, se obtiene el mensaje de error "BDOS ERROR R/O". Esto es muy fastidioso, dado que ahora hay que pulsar la tecla CTRL-C, y volver a introducir todo desde el principio.

Como no nos gusta fastidiarnos, y tampoco deseamos que Vd. se fastidie en el futuro, queremos sugerirle una solución, mediante la cual podemos hacer que el mensaje "BDOS ERROR R/O" quede desterrado para siempre al baúl de los recuerdos. Esto seguramente también será interesante para Vd., dado que obtendrá una visión directa, de la forma en que se utiliza DDT, y como pueden inspeccionarse y modificarse los ficheros, mediante CP/M.

VIII.2 Corrección del error

Vamos a hacer desaparecer el error arriba mencionado, corrigiendo un poco el programa PIP.COM. Lo mejor será, que de momento sólo efectúe la corrección en su copia de seguridad - ¡Puede imaginarse lo que pasaría, si cometiese un error, sin disponer de ninguna copia más, de CP/M!

Al principio vamos a llamar la función Disc-Reset, para que el disco, eventualmente nuevo, sea inicializado. Así que tendremos que hacer algunas chapuzas en el fichero PIP.COM, y la mejor forma de hacerlo, es empleando DDT.

Introduzca:

```
DDT PIP.COM
```

Al igual que cualquier otro fichero COM, el programa comienza en la dirección $0100. Puede desensamblar esta zona:

```
-I0100
 0100 JMP 04CE
 0103 RET
```

(sus introducciones en negrita)

Por lo tanto el programa PIP comienza en 04CE. Sin embargo nosotros queremos incluir und DISC-Reset, por lo que colgamos esta pequeña rutina al final del programa PIP, apuntando hacia él, al principio de PIP. El final del programa PIP se encuentra en $1DB1. Ahora introducimos la rutina siguiente:

```
-a1db2
 1DB2 MVI C,0D
 1DB4 CALL 0005
 1DB7 JMP 04CE
 1DBA (pulsar RETURN)
```

Esta rutina llama la rutina de DISC-Reset. Ahora tenemos que indicar al principio de PIP, que primero hay que saltar a dicha rutina. Lo hacemos de la forma siguiente:

```
-a0100
 0100 JMP 1DB2
 0103 (pulsar RETURN)
```

VIII.3 Almacenamiento del nuevo PIP

El programa ha sido corregido. Ahora debería ejecutar PIP (con todas las opciones posibles). Si no funcionara según se ha descrito, algo ha ido mal. Pero en tal caso, Vd. posee (¡gracias a Dios!) otra copia de seguridad de CP/M, que en primer lugar volverá a copiar, y con la que a continuación puede volver a intentarlo. (Ya vé lo razonable que es, tener guardada una copia de seguridad en el armario. Un disco no sólo puede despedirse de Vd. para siempre, a causa de algunas manipulaciones equivocadas. ¡Imagine, que su deliciosa hijita o hijito utilicen el disco como pala para sus castillos de arena!)

Ahora nos falta almacenar en disco nuestro programa corregido, dado que de momento sólo está almacenado en la memoria. Para ello primero tiene que abandonar DDT, pulsando la tecla CTRL-C.

Entonces el programa se graba mediante el comando SAVE. La instruccción es ésta:

```
SAVE 29 PIP.COM
```

Si el disco no está protegido contra escritura, ésto es todo. PIP queda definitivamente corregido, y el error ya no se producirá.

Como vé, incluso programas tan grandes y significantes como el sistema operativo CP/M, de una empresa tan importante como lo es Digital Research, todavía puede contener algún que otro error. Simplemente es imposible, eliminar todos los errores en un programa - y no hay ningún programa libre de ellos.

En este lugar pretendíamos dar cuenta de otro error en la versión corriente de CP/M 2.2, y sugerir una solución; pero, visto y no visto: dicho error ya ha sido eliminado en la versión de CP/M para el Amstrad. Como vé, las empresas se esfuerzan en eliminar inmediatamente todos los errores que vayan saliendo a la luz.

IX. Todos los comandos de CP/M

IX.1 ASM (CP/M 2.2)

ASM.COM

Confecciona un fichero destinatario en formato HEX, a partir de un fichero fuente con instrucciones del 8080 o Z 80.

Formato de entrada

ASM FICHERO
ASM FICHERO.BBZ

Descripción

Este programa confecciona un fichero hexadecimal, a partir de un fichero con instrucciones para el 8080 o el Z80. El así denominado fichero objeto, puede cargarse en el sistema, a modo de comando transistorio, mediante el programa LOAD.

Aplicación

El programa ASM siempre busca un fichero con el código ASM, de modo que no hay que introducir dicho código. Puede ir acompañado de un total de tres parámetros: shp. La S representa Source (fuente), y es la denominación de la unidad (A..P), en la que se encuentra el fichero de origen. H representa HEX, y se reserva a la denominación de la unidad, en la que debe crearse el fichero hexadecimal. Si desea suprimir la confección del fichero hexadecimal, introduzca una Z. P representa PRN, y aquí se puede indicar la unidad, en la que tenga que confeccionarse el fichero PRN. Para suprimirlo, vuelva a introducir una Z, o una X, si quiere que se le muestre en pantalla.

IX.2 COPYSYS

COPYSYS.COM

Copia las pistas del sistema y CP/M3.SYS en un disco

Formato de entrada:

COPYSYS (CP/M 3.0)

Descripción:

Para poder ejecutar CP/M 3.0 desde un disco, tienen que estar presentes ambas partes de CP/M: la primera, en las pistas del sistema, y la segunda en forma de fichero CPM3.SYS. Los discos que no se usen para copiar, no necesitan ninguna de las dos.

Aplicación:

Introduzca COPYSYS, seguido de ENTER. Se le preguntará, de qué unidad hay que copiar:

Source drive name (or return for default)

Introduzca la letra de la unidad, en la que se encuentre su disco con el sistema CP/M (unidad A: o 1, normalmente). Si el disco ya se encuentra dentro de la unidad, simplemente pulse ENTER. A continuación introduzca la letra de la unidad destinataria, seguida de ENTER. Si todavía no ha introducido el disco, se le pedirá, que lo haga. Si las pistas del sistema están ocupadas, aún puede copiar mediante CPM3.SYS. Se le preguntará:

Do you wish to copy CPM3.SYS?

Si responde afirmativamente con "Y", el fichero CP/M será copiado.

Nota

COPYSYS es el comando estándar para copiar las pistas del sistema bajo CP/M 3.0. Desgraciadamente, en niguno de los discos CPM que se suministran con el CPC, se encuentra dicho fichero COM. El paquete de programas DISCKIT3, sin embargo ofrece la posibilidad de formatear discos en cualquier formato, y de copiar el sistema.

DISCKIT3 es un paquete de programas muy manejable, que la empresa Amstrad ha desarrollado especialmente para el CPC 6128. Lamentablemente, a la hora de redactar este libro no se dispuso de ningún tipo de datos sobre dicho paquete de programas. Lo único seguro es, que mediante DISCKIT3 puede formatear discos, y copiar el sistema CP/M 3.0, sin ningún problema. Para más informaciones sobre DISCKIT3, debe dirigirse al manual del usuario.

IX.3 DATE (CP/M 3.0)

DATE.COM

Muestra fecha y hora, posibilitando su introducción

Formato de entrada:

```
DATE
DATE CONTINUOUS
DATE SET
DATE MM/DD/YY HH:MM:SS
```

Descripción:

Mediante este programa puede introducir una fecha y una hora en su sistema, o acceder a ambas informaciones.

Aplicación:

Si solamente introduce el comando DATE, se le muestra la fecha y la hora

```
Tue 05/06/86 23:49:17
```

esto es una indicación estática. Si desea comparar la hora con otro reloj, introduzca el comando DATE con la opción "C". Entonces verá el reloj en marcha, pero no podrá ejecutar ningún programa, ni introducir ningun comando durante este periodo. Pulse una tecla cualquiera para interrumpir el programa.

Puede introducir la fecha y la hora en dos formas distintas: si utiliza la opción SET, el programa pedirá los valores de seguido, y Vd. puede saltarse una introducción mediante ENTER. La segunda posibilidad es esta:

DATE 05/06/86 23:49:17

Introduzca una hora, que adelante por uno o dos minutos a la hora actual. Cuando llegue la hora exacta, que Vd. haya introducido, pulse la barra espaciadora, y el reloj se pondrá en marcha.

Si su computador no dispone de un reloj asegurado por una batería, que es el caso del Amstrad CPC 6128, tendrá que efectuar la introducción, cada vez que ponga en marcha el computador. Puede facilitárselo, escribiendo el comando DATE SET en el fichero PROFILE.SUB.

IX.4 DDT

DDT.COM

Programa desensamblador para cargar, ejecutar, modificar y comprobar programas

Formato de entrada:

DDT

DDT fichero.COM

Descripción:

El programa DDT sustituye al CCP, cargando el fichero indicado en la memoria de trabajo. Mediante DDT puede consultarse la dirección siguiente a la dirección final de un programa (NEXT), y el contenido del contador de programas (PC).

Aplicación:

El programa tiene que estar disponible sobre el disco, en forma de DDT.COM. Si sólo se introduce DDT, el programa espera la introducción de un nombre de fichero. Mediante la introducción Dn, se muestra una zona de la memoria, que comienza en la dirección n. Para finalizar el programa, lo mejor es utilizar el comando GO.

IX.5 DEVICE (CP/M 3.0)

DEVICE.COM

Muestra y modifica los caminos hacia los dispositivos periféricos

Formato de entrada:

```
DEVICE NAMES
DEVICE VALUES
DEVICE LST:=XXXXXX[NOXON,1200]
DEVICE CONOUT:=XXX,XXX
DEVICE CON:[PAGES]
DEVICE CON:[COLUMNS=nn.LINES=mm]
```

Descripción:

Este programa se utiliza para mostrar y modificar los canales de entrada y salida. Los tres canales lógicos son: CON:, LST: y AUX:. También puede desviarse un canal lógico hacia varios dispositivos periféricos. Por ejemplo pueden enviarse los datos del computador simultáneamente hacia la impresora y hacia la pantalla. Mediante DEVICE también puede determinar el número de líneas y columnas en la pantalla.

Aplicación:

Los nombres de los dispositivos de salida están preestablecidos por sus fabricantes. Si introduce DEVICE con la opción NAMES, obtendrá una lista con los nombres y las velocidades de transmisión de los diversos dispositivos. Si introduce DEVICE VALUES, obtendrá una indicación similar a:

```
Current Assignments:

CONIN: = CRT
CONOUT: = CRT
AUXIN: = LPT
AUXOUT: = LPT
LST: = CEN
```

CONIN: y CONOUT: significan CONsol INput, entrada por el teclado, y CONsol OUTput, salida por el teclado. La denominación CON: a secas se refiere a ambas cosas. LST: significa LIST DEVICE, y accede a la impresora.

Mediante la opción PAGE puede averiguar, cuantas líneas y columnas muestra la pantalla actual. Si introduce una determinada cantidad de baudios en la forma [nnn], la transmisión de datos se efectuará a dicha velocidad. El comando DEVICE, le muestra conjuntamente las indicaciones de DEVICE NAMES y de DEVICE VALUES.

Observe que en CP/M 2.0, el comando DEVICE se sustituye por el comando STAT DEV: (en realidad tendría que decirse, que en CP/M 2.2 se ha partido el comando STAT, porque es demasiado complejo.)

IX.6 DIR (CP/M 2.2 y CP/M 3.0)

DIR.COM y comando incorporado (CP/M 3.0)

Muestra determinados nombres de ficheros y el directorio

Formato de entrada:

Comando incorporado:

```
DIR
DIR A:
DIR FICHERO.XXX
DIR AB*.*
```

Comando transitorio:

```
DIR[OPCION]
DIR FICHERO.XXX[OPCION]
DIR AB*.*
```

Descripción:

DIR es un programa muy útil, para localizar ficheros en un disco, y sobre todo en discos duros. (Lamentablemente todavía no hay nadie, que ofrezca discos duros para un Amstrad) La versión incorporada funciona en cada unidad, y desde cualquier zona del usuario. Pueden utilizarse los caracteres de sustitución "*" y "?". Si no se introduce ninguna opción, se muestran todos los ficheros, que no sean del sistema, de la zona correspondiente.

El comando transistorio DIR, es capaz de hacer mucho más: mostrar ficheros con fecha, ordenar los ficheros por órden alfabético etc. A continuación damos una lista de las diversas funciones, que siempre deben ponerse entre corchetes. Si existen ficheros del sistema en el directorio, aparece el mensaje:

SYSTEM FILE(S) exist

OPCIONES:

opción función
ATT muestra atributos de fichero definidos por el usuario
DATE muestra ficheros con hora y fecha
DIR muestra ficheros que no sean del sistema
DRIVE=ALL muestra los ficheros de todas las unidades
DRIVE=A muestra los ficheros de la unidad A:
DRIVE=(A,B) muestra ficheros de las unidades indicadas
EXCLUDE muestra todos los ficheros excepto el indicado
FF adelantar página antes de mostrar el directorio
FULL muestra ficheros con descripción completa
LENGTH=n título a partir de n líneas
MESSAGE muestra continuamente unidades y zonas del usuario
NOPAGE muestra directorio sin detenerse
NOSORT muestra ficheros sin sortear
RO muestra ficheros de sólo lectura
RW muestra ficheros de lectura/escritura
SIZE indicar tamaño de cada fichero
SYS muestra sólo ficheros del sistema
USER=ALL muestra ficheros de todas las zonas del usuario
USER=5 muestra ficheros de la zona del usuario indicada
USER=(3,5) muestra ficheros de las zonas del usuario indicadas.

IX.7 DIRSYS (CP/M 3.0)

Comando incorporado

Muestra los ficheros, que han sido declarados ficheros del sistema.

Formato de entrada:

```
DIRSYS
DIRSYS B:
DIRSYS FICHERO.XXX
DIRSYS AB?*.*
```

Descripción:

Mediante DIRSYS se muestran todos los ficheros del sistema. Aunque lo mismo pueda conseguirse mediante DIR y sus opciones, DIRSYS es más rápido, porque es un comando incorporado. Puede abreviarlo:

```
DIRS
```

IX.8 DUMP (CP/M 2.2 y CP/M 3.0)

DUMP.COM

Muestra ficheros de datos en hexadecimal y ASCII

Formato de entrada:

DUMP
DUMP FICHERO.XXX

Descripción:

DUMP es especialmente interesante para los programadores, que trabajen con ensamblador. Los ficheros de texto se emiten por la pantalla o por la impresora mediante TYPE, y los ficheros de tipo COM, REL o OVR, mediante DUMP.

En el capítulo VII. encontrará descripciones detalladas de los ficheros COM ASM, DDT y DUMP.

IX.9 ERA(SE) (CP/M 2.2 y CP/M 3.0)

Comando incorporado y ERASE.COM

Borra uno, varios, o todos los ficheros de un disco

Formato de entrada:

Comando incorporado:

```
ERASE
ERASE FICHERO.XXX
ERASE AB?*.*
```

Comando transitorio:

```
ERASE FICHERO.XXX[CONFIRM]
```

Descripción:

Mediante ERASE se borra el fichero indicado, si no está en "Read Only" (protegido contra escritura). Sólo puede borrar dentro de una zona del usuario. Si utiliza "*" o "?", se repite la instrucción de borrar, con la pregunta, si realmente se debe borrar.

Aplicación:

Sin opciones, ERASE puede trabajar desde cada unidad. Si se utilizan opciones, se necesita el programa ERASE.COM en la unidad. ERASE puede abreviarse ERA. La abreviación de la opción "CONFIRM", es "C".

IX.10 FORMAT (CP/M 2.2 y CP/M 3.0)

FORMAT.COM

Formatea un disco

Formato de entrada:

FORMAT

Descripción:

Cada disco que utilice en su computador, tiene que ir debidamente formateado. El programa destruye todos los datos de un disco durante el formateo. FORMAT no es ningún programa estándar de CP/M. Por ésto es posible, que quizás no se encuentre en su disco del sistema. En tal caso, busque el programa COPY.COM, y mire si con él puede formatear un disco.

Aplicación:

FORMAT trabaja ininterrumpidamente, mientras que no lo detenga mediante control C (^C). De este modo puede formatear varios discos sin interupción.

Observación:

CP/M 3.0 para Amstrad CPC 6128 no ofrece FORMAT a modo de fichero COM. FORMAT se incluye como subrutina en el paquete DISCKIT3. Si a pesar de todo, Vd. desea disponer de FORMAT en forma de fichero COM, para por ejemplo poder integrarlo en ficheros SUBMIT, puede copiar el fichero COM del disco de CP/M 2.2.

IX.11 GENCOM (CP/M 3.0)

GENCOM.COM

Crea una versión especial de CP/M 3.0

Formato de entrada:

GENCOM

Descripción:

Los programadores lo necesitan para adaptar una versión de CP/M, o incorporar partes adicionales a CP/M

IX.12 GET (CP/M 3.0)

GET.COM

Lee los datos de un fichero, en lugar de del teclado

Formato de entrada:

```
GET CONSOLE INPUT FROM FILE FICHERO.XXX
GET CONSOLE INPUT FROM CONSOLE
GET CONSOLE INPUT FROM FILE FICHERO.XXX[SYSTEM]
GET FILE FICHERO[NOECHO]
```

Descripción:

Con GET, en lugar de teclear los comandos o instrucciones, puede hacer que éstas se lean de un fichero cualquiera.

Aplicación:

La primera línea arriba ordena a CP/M que procese la instrucción o el programa siguiente que se introduzca por el teclado. Siempre cuando se necesite una instrucción, CP/M busca ésta en el fichero FICHERO.XXX. Si el programa ha terminado, o si ya no queda ninguna instrucción en el fichero, CP/M regresa a las instrucciones por teclado.

El comando de la segunda línea instruye a CP/M, para que vuelva a aceptar comandos introducidos por el teclado. Dicho comando puede figurar en el fichero FICHERO.XXX, o ser introducido por el usuario.

La tercera forma del comando desvía inmediatamente la entrada del teclado hacia el fichero indicado, para que éste pueda leer y procesar las instrucciones del teclado. Si introduce la opción (NOECHO), no se mostrarán en la pantalla, las istrucciones que se estén ejecutando.

IX.13 HELP (CP/M 3.0)

HELP.COM y HELP.HLP

Presta ayuda a la utilización de los diversos comandos CP/M, mediante ejemplos.

Formato de entrada:

HELP
HELP concepto
HELP concepto subconcepto
HELP concepto[opción]
HELP .subconcepto
HELP opción

Descripción:

El programa HELP.COM le ofrece ayuda e informaciones para los diversos comandos y programas de CP/M. No puede acceder a HELP si está trabajando con otro programa.

Aplicación:

Cuando llame HELP, se le presentarán una serie de conceptos. Puede seleccionar el punto que le interese, e introducirlo. Está permitido abreviar el concepto a sólo dos caracteres. Si la información abarca más de 23 líneas, la impresión se detendrá, cuando haya cubierto toda la pantalla. Mediante la barra espaciadora se continúa con la impresión. Si antes de introducir HELP, se introduce control P (^P), todas las informaciones salen por la impresora.

Si desea modificar el texto del fichero HELP.HLP, tiene que introducir primero HELP con la opción EXTRACT, con lo que puede modificar el texto. A continuación, mediante la opción CREATE, se crea un fichero nuevo, legible para CP/M.

El comando HELP es especialmente útil para los principiantes, ya que de esta manera no es necesario tener siempre el manual a mano. También otros paquetes de software conocidos han imitado este ejemplo de Digital Research - cuantos programas existen ya, que integran las así llamadas HELP-SCREENS (pantallas de ayuda), con el fin de facilitar el trabajo a los usuarios.

Especialmente para llegar a conocer CP/M 3.0+, debería ocuparse más detenidamente de los textos que ofrece HELP. Aprenderá el manejo de CP/M 3.0+, y además conocerá la mayoría de los comandos disponibles.

IX.14 HEXCOM (CP/M 3.0)

HEXCOM. COM.

Crea un fichero directamente ejecutable con el código COM

Formato de entrada.

HEXCOM FICHERO
HEXCOM

Descripción:

Este programa se necesita, cuando se programa en ensamblador. Los ficheros creados con el programa MAC.COM en formato hexadecimal INTEL, son transformados mediante HEX.COM en ficheros ejecutables con código COM.

IX.15 INITDIR (CP/M 3.0)

INITDIR.COM

Prepara el directorio de un disco para recibir el sello de fecha y hora

Formato de entrada:

INITDIR B:

Descripción:

CP/M 3.0 ofrece la posibilidad de aplicar un sello de fecha y hora a los ficheros. Las informaciones correspondientes se almacenan en una parte separada del directorio. Es por ello, que cada disco debe prepararse de la manera adecuada, mediante el programa INITDIR. El modo de codiificación de los ficheros se selecciona con el programa SET.

Aplicación:

Lo mejor es que introduzca INITDIR en discos nuevos, dado que las anotaciones en el directorio ocupan espacio. Si va a utilizar INITDIR con un disco que ya contenga datos, es mejor que antes haga una copia de seguridad. La razón de ello es, que si el trabajo de INITDIR queda interrumpido, por ejemplo a causa de un apagón, Vd. perderá todas las anotaciones, y en consecuencia, también todo el contenido del disco.

Ya vé, que no sólo vale la pena hacer copias de seguridad de los discos del sistema CP/M 3.0.

IX.16 LIB (CP/M 3.0)

LIB.COM

Confecciona y modifica una "biblioteca" de subrutinas

Formato de entrada:

LIB.FICHERO.XXX[opciones]

Descripción:

Muchos compiladores y los ensambladores RMAC y MACR-80 utilizan el método de las subrutinas separadas, provistas en mayoría del código REL o IRL. Puede conseguir una recopilación de las subrutinas más importantes, a modo de "biblioteca", mediante LIB.

IX.17 LINK (CP/M 3.0)

LINK.COM

Crea un fichero ejecutable a partir de módulos de subrutinas

Formato de entrada:

LINK FICHERO, FICHERO2, FICHERO3,..[opciones]

Descripción:

Muchos compiladores y los ensambladores RMAC, así como MACRO-80, utilizan el método de las subrutinas separadas, marcadas a menudo con el código REL o IRL. Mediante LINK todos estos ficheros se pueden convertir en un único fichero COM.

IX.18 LOAD (CP/M 2.2 y CP/M 3.0)

LOAD.COM

Crea un fichero COM a partir de un fichero HEXA

Formato de entrada:

LOAD fichero (¡sin su código!)

Descripción:

El ensamblador ASM almacena su código creado en dos pasadas, en forma de un fichero hexadecimal. El aspecto de dicho fichero hexadecimal ha sido definido por la empresa INTEL. El programa LOAD modifica dicho fichero hexadecimal, convirtiendo los datos hexadecimales codificados en ASCII, en un fichero COM ejecutable. Hay que observar, que no debe introducirse el código del fichero ".HEX". Dicha información es obligatoria, dado que el ensamblador almacena el fichero hexadecimal bajo el nombre de fichero.HEX.

Aplicación:

LOAD debe ejecutarse seguramente después de cada ensamblado, dado que el código hexadecimal de un programa, de poco nos sirve. La aplicación exacta de éste, y de otros comandos relacionados con el proceso, se explica en el capítulo VII.

IX.19 MAC (CP/M 3.0)

MAC.COM

Un macroensamblador para programas escritos en ensamblador

Formato de entrada:

```
MAC FICHERO
MAC FICHERO $opciones
```

Descripción:

El macroensamblador MAC confecciona tres ficheros, con una macrobiblioteca. El programa fuente tiene el código ASM, y la "biblioteca", el código LIB. El código ensamblado del programa figura en el fichero de código HEX, la tabla de símbolos en el fichero SYM, y el listado imprimible en el fichero PRN. En lugar de corchetes se usa el signo del dólar ($), para señalar las opciones.

IX.20 MOVCPM (CP/M 2.2)

MOVCPM.COM

Desplazamiento del sistema en unidades de 256 bytes

Formato de entrada:

MOVCPM<factor>*

Descripción:

El comando MOVCPM sirve para desplazar (relocalizar) el sistema CP/M 2.2 en pasos de 256 bytes. El <factor> puede moverse entre 64 y 179. Si se indica 179 como <factor>, se crea un CP/M estándar, y si por ejemplo se indica MOVCPM 178*, se crea un CP/M 2.2 desplazado 256 bytes hacia abajo. El <factor> muestra la memoria disponible para los comandos de CP/M (transitorios) y para los programas del usuario, en unidades de 256 bytes.

Aplicación:

Si por alguna causa se hace necesario desplazar el sistema, por ejemplo a causa de un solapamiento de BIOS/BDOS con los programas del usuario, ello puede realizarse mediante MOVCPM. El sistema desplazado puede almacenarse por medio del comando SYSGEN (vea IX.33).

IX.21 PATCH (CP/M 3.0)

PATCH.COM

Instala modificaciones en CP/M 3.0, o en programas

Formato de entrada:

PATCH FICHERO
PATCH FICHERO n

Descripción:

Con el programa PATCH Vd. puede realizar cambios en el sistema CP/M, e introducirlos en CP/M o en alguno de los programas transitorios. Este proceso se llama patchear. Mediante PATCH, las modificaciones se introducen automáticamente en el programa correspondiente. Si existen varios patches, Vd. debe referirse al patch correspondiente, usando números de referencia. Los números de referencia comienzan en 1.

IX.22 PIP (CP/M 2.2 y CP/M 3.0)

PIP.COM

Copia ficheros y transporta ficheros entre periféricos

Formato de entrada:

```
PIP
PIP fichero destinatario=fichero original[opción]
PIP B:=fichero.XXX
PIP todo.txt=TEXTO1.TXT,TEXTO2.TXT...
PIP AB1?*.*=AB2?*.*[opción]
```

Descripción:

Seguramente PIP es el programa más potente de CP/M. Con él Vd. puede copiar programas de un disco a otro, y también entre diferentes zonas del usuario. Puede convertir varios ficheros en uno solo, archivar automáticamente los ficheros con los que haya trabajado últimamente, enumerar líneas, convertir todo en mayúsculas, y poner el octavo bit en cero. Para informaciones más detalladas, vea el capitulo VI.

Aplicación:

PIP puede utilizarse con, o sin opción. Cuando PIP esté preparado para aceptar comandos, así lo señala mediante un asterisco (*). PIP confecciona un fichero destinatario a modo de copia exacta del fichero original, a no ser que se hayan añadido determinadas opciones. Los corchetes deben ponerse inmediatamente detrás del nombre de fichero, sin ningún espacio. Si desconoce algunas de las letras del nombre de un fichero, o si desea copiar varios ficheros de una determinada categoría, también puede utilizar los signos "?" en sustitución de caracteres desconocidos dentro del nombre, o "*" en sustitución de todos los caracteres. Algunos ejemplos:

Su unidad en curso es A:

PIP B:=FICHERO.XXX[V]

Con esta instrucción, Vd. copia el fichero fichero.XXX de A: a B:, dejando a PIP que verifique la correctitud de la copia.

PIP B:=FICHERO?.*[V]

De esta manera Vd. copia una serie de ficheros, con el nombre FICHERO, de A: a B:, sin importar para nada el código que tengan.

PIP B:=*.*[V]

Aquí se copian todos los ficheros de una unidad a la otra, verificándose todas las copias.

PIP NUEVO.XXX=VIEJO.XXX

Con esta instrucción, Vd. copia el fichero VIEJO.XXX al nuevo fichero NUEVO.XXX en la misma unidad. A partir de allí, ambos ficheros existirán sobre este disco.

PIP TODO.TXT=TEXTO1.TXT,TEXTO2.TXT,...

De esta manera, Vd. reune varios ficheros en uno solo, que contiene todos los demás.

PIP B:[G] =TEXTO.TXT

El fichero TEXTO.TXT se copia en la unidad B: y en la zona del usuario "5".

Opciones

A Función de archivado. Sólo copia ficheros, que hayan sido creados o modificados desde el último archivado. Debe estar activada la función del sello de fecha y hora.

C Confirm. CP/M pregunta, antes de copiar un fichero, si está autorizado a copiarlo.

Dn Borra a partir de n columnas. PIP borra todos los caracteres, que se encuentren detrás de la columna n en un fichero.

E Muestra (echo) el texto en pantalla. El contenido de datos recién copiados se muestra en la pantalla. No debe utilizarse con el parámetro "N". Emplearlo solamente con ficheros de texto.

F Elimina el carácter de "página siguiente" (Form Feed). Algunas impresoras necesitan una (^L), para cambiar de página. Si su impresora no lo necesita, puede eliminar los caracteres de control correspondientes, mediante "F".

Gn Toma o transporta un fichero de, o hacia la zona del usuario "n". Esta instrucción debe ir directamente detrás del primer nombre de fichero, y es la única que puede figurar allí. Ejemplo:

```
PIP B:[G5]=TEXTO.TXT[G1]
```

Copia el fichero TEXTO.TXT de la zona del usuario "1" a la zona del usuario "5".

H Transferencia de datos hexadecimales. Esta opción siempre debe usarla, cuando transfiera ficheros HEX. Entonces PIP comprueba si los datos están escritos correctamente en formato INTEL.

I Ignora la señal de final del fichero en ficheros HEX. Introduzca esta opción para todos los ficheros, a excepción del último. Con la opción "I" se activa simultáneamente la opción "H" de PIP. Utilice el parámetro "H" para el último fichero:

```
PIP FICHERO.HEX=PROG1.HEX[I],PROG2.HEX[H]
```

K Suprime la muestra de los nombres de fichero durante el copiado.

L Convierte todas las letras mayúsculas en minúsculas. Utilice esta opción sólo con ficheros de texto y utilice adicionalmente el parámetro "Z" para los ficheros de WordStar.

N Enumera las líneas de un fichero.
Los números comienzan con 1 en la primera línea, incrementándose en uno, por cada línea. Los números consisten de seis cifras como máximo, y las posiciones, que no se necesitan, aparecen como espacios. Detrás de las cifras hay dos puntos, y un espacio. No utilice esta opción conjuntamente con las opciones "E" o "N". Para los ficheros del WordStar debe introducir también la opción "Z".

N2 Enumera las líneas de un programa en BASIC

O Transferencia de ficheros-objeto. Se utiliza para transferir ficheros, que no sean ni de texto, ni del tipo COM.

Pn Fija la longitud de página. Viene preajustado en 60 líneas por página. Debería introducir también la opción "F", para que se borren los caracteres de control (^L) de cambio de página ya existentes. Utilizar solamente con ficheros de texto.

Qxxxx^Z Final del copiado a partir de esta cadena de caracteres. PIP copiará un fichero hasta, e inclusive la cadena de caracteres introducida. Introduzca el comando en dos líneas de instrucciones, para que la cadena de caracteres no se convierta en mayúsculas:

```
PIP
CON:=TEXTO.TXT[Weiler]
```

Si escribiera todo en la primera línea detrás de PIP, la anotación aparecería en letras mayúsculas.

R Copia ficheros del SISTEMA. Estos ficheros no pueden mostrarse mediante DIR, y PIP normalmente no los copia. Mediante "R" puede suspenderse esta limitación.

Sxxxx^Z Comienza en esta cadena de caracteres. PIP empieza a copiar desde esta cadena de caracteres. Escriba el comando en dos líneas, o la cadena se convertirá en mayúsculas. Utilizar sólo con ficheros de texto.

Tn fijar amplitud de paso del tabulador. Normalmente CP/M trabaja con una amplitud de ocho caracteres. Sin embargo, cuando se imprime un fichero, ésto no funciona, por lo que hay que definir la amplitud de pasos. "n" indica el número de espacios para un paso del tabulador. Utilizar sólo con ficheros de texto.

U Convierte en mayúsculas. Convierte todas las letras minúsculas en mayúsculas. Utilizar sólo con ficheros de texto. Para los ficheros del WordStar, hay que introducir adicionalmente la opción "Z".

V Verificar. Este parámetro se ocupa, de que PIP compare con exactitud el fichero copiado, con el fichero original.

W Escribir sobre ficheros con el atributo Read Only (RO). Estos ficheros normalmente sólo pueden ser leídos, pero no se pueden modificar ni borrar. Mediante la opción "W" todos estos ficheros se borran sin esperar confirmación.

Z Borrar el bit de paridad (octavo bit). El juego de caracteres ASCII solamente utiliza siete bits. El octavo bit se emplea para diversas tareas, en diversos programas. La opción "Z" no es necesaria, al copiar a unidades ASCII, tales como COM: o LST:.

IX.23 PUT (CP/M 3.0)

PUT.COM

Escribe los datos para la impresora o pantalla en un fichero

Formato de entrada:

```
PUT CONSOLE OUTPUT TO FILE FICHERO.XXX[OPCION]
PUT PRINTER OUTPUT TO FILE FICHERO.XXX[OPCION]
PUT CONSOLE OUTPUT TO CONSOLE
PUT PRINTER OUTPUT TO PRINTER
```

Descripción:

Normalmente CP/M envía los datos para la pantalla a la pantalla, y los datos para impresora a la impresora. Mediante el comando PUT, los datos pueden escribirse además en un fichero.

Aplicación:

Las dos primeras líneas instruyen a CP/M, para que abra un fichero, con el nombre FICHERO.XXX, en el cual debe escribir todo lo que vaya destinado a la pantalla, o a la impresora. Cuando la opción termina, el programa vuelve al modo normal de CP/M. Las últimas dos líneas interumpen el programa PUT. Puede emplearlas, si por ejemplo Vd. tiene un fichero SUBMIT, que utilice el comando PUT, y que a continuación deba volver al modo normal.

Mediante la opción NO ECHO, Vd. impide que los datos transferidos se muestren en la pantalla. Con la opción FILTER, Vd. convierte cada carácter de control, en un carácter imprimible.

La opción SYSTEM provoca, que no sólo los datos, sino también las líneas de instrucciones pasen a formar parte del fichero.

IX.24 REN(AME) (CP/M 2.2 y CP/M 3.0)

RENAME como comando incorporado,o RENAME.COM

Modificar nombre de un fichero

Formato de entrada:

```
RENAME
RENAME FICHERO2.XXX=FICHERO1.XXX
RENAME AB?*2=AB?*1
```

Descripción:

Mediante RENAME se cambia el nombre de un fichero, - y solamente el nombre. El contenido permanece intacto.

Aplicación:

Si el fichero no figura en la unidad en curso, puede indicarse también el nombre de la unidad. Si introduce RENAME sin opciones, el programa preguntará por ellas.

IX.25 RMAC (CP/M 3.0)

RMAC.COM

Macroensamblador para la programación en ensamblador

Formato de entrada:

RMAC FICHERO
RMAC FICCHERO $OPCIONES

Descripción:

El macroensamblador RMAC confecciona tres ficheros, y una macrobiblioteca. El programa fuente tiene el código ASM, y la "biblioteca", el código LIB. El código del programa ensamblado, se encuentra en un fichero con código REL, la tabla de símbolos, en el fichero SYM, y el listado imprimible, en el fichero PRN. Mediante el programa LINK, se crea un programa ejecutable, a partir del fichero REL.

Aplicación:

En lugar de los corchetes, se utilizan signos de dólar ($) para las opciones. Antes del signo del dólar, hay que poner un espacio.

IX.26 SAVE (CP/M 2.2 y CP/M3.0)

SAVE.COM

Almacena el contenido de la memoria de trabajo en un fichero

Formato de entrada:

SAVE

Descripción:

El comando SAVE crea un fichero, que contiene una determinada zona de la memoria. Antes del almacenamiento, algún programa tiene que haber depositado algo en la memoria.

Aplicación:

Debe llamar SAVE, ANTES de cargar cualquier otro programa en la memoria de trabajo. SAVE se desplaza automáticamente al final superior de CP/M, devolviendo acto seguido el control a CP/M. A continuación Vd. puede ejecutar su programa. Tan pronto como éste finalice, SAVE automáticamente vuelve a hacerse cargo del control. Se le preguntará el nombre de fichero, la dirección inicial, así como la dirección final en la memoria.

IX.27 SET (CP/M 3.0)

SET.COM

Pone atributos de fichero, posibilita la introducción de una clave protectora, da nombres a los discos, y selecciona el modo de los sellos de fecha y hora

Formato de entrada:

SET FICHERO.XXX[opción]
SET AB?*.*[opción]
SET B:[opción]

Descripción:

El programa SET ofrece diversos servicios. Los más imoortantes son: activar el modo de archivo, y posibilidad de poner ficheros y unidades enteras, en estado de sólo lectura.

Puede darse un nombre individual a cada disco, amén de una palabra clave protectora, no sólo para cada disco, sino también para cada fichero. Los nombres de los discos pueden ser mostrados mediante SHOW.

La tercera posibilidad, es la de aplicar un sello de fecha y hora a los ficheros.

Para más informaciónes, vea el capítulo V.

IX.28 SETDEF (CP/M 3.0)

SETDEF.COM

Muestra o define el orden de búsqueda, y activa o desactiva las indicaciones en la pantalla

Formato de entrada:

SETDEF
SETDEF B:
SETDEF[opción]

Descripción:

Normalmente Vd. activa un programa transitorio, introduciendo su nombre y pulsando RETURN. Si desea llamar el programa desde otra unidad, Vd. además introduce el nombre de dicha unidad.

Si por ejemplo siempre trabaja con la unidad B:, pero sus ficheros de CP/M se encuentran en la unidad A:, puede indicar a CP/M donde debe buscar los ficheros en primer lugar. Ello se denomina definirir un orden de búsqueda. De esta manera puede decirle a CP/M, que busque primero en la unidad A:, luego en la unidad C:, y sólo entonces en la unidad en curso.

IX.29 SHOW (CP/M 3.0)

SHOW.COM

Muestra las opciones de un disco

Formato de entrada:

SHOW[opción]
SHOW B:[opción]

Descripción:

SHOW le muestra, por así decir, los "datos técnicos" de un disco. La información que puede obtener, es: el espacio libre en el disco, el número de anotaciones libres en el directorio, el número de zonas del usuario activas, así como el número de la zona seleccionada. También se dan informaciones referentes a la capacidad total del disco, tamaño de bloques, tamaño de sectores, y número de sectores por pista. También puede mostrarse el nombre de un disco.

Cuando trabaje con CP/M 2.2, tenga en cuenta, que este comando también se realiza mediante STAT.

IX.30 SID (CP/M 3.0)

SID.COM

Un programa desensamblador, para cargar, modificar y comprobar un programa ensamblador

Formato de entrada:

```
SID
SID FICHERO.XXX
SID FICHERO.XXX TEXTO.SYM
```

Descripción:

Mediante SID puede cargar ficheros COM o HEX en la memoria de trabajo, donde puede mirarlos y modificarlos. Si también carga un fichero SYM, puede ejecutar el programa bajo un control estricto. SID también puede traducir programas ejecutables, a mnemónicos INTEL 8080.

IX.31 SUBMIT (CP/M 2.2 y CP/M 3.0)

SUBMIT.COM

Posibilita la entrada de instruccionesnes desde un fichero, en lugar de desde el teclado

Formato de entrada:

SUBMIT
SUBMIT FICHERO.SUB
SUBMIT FICHERO FICHERO1 FICHERO2 FICHERO3...

Descripción:

En un fichero SUBMIT pueden escribirse comandos, que en otras ocasiones se introducirían directamente por el teclado. Dichos comandos se ejecutan automáticamente. Cuando el fichero ya no contiene más comandos, el control es devuelto a CP/M, y Vd. puede proseguir normalmente.

Cada vez que CP/M vuelve a iniciarse, el programa busca un fichero llamado PROFILE.SUB, y si lo encuentra, ejecuta los comandos que pueda contener.

Aplicación:

Un fichero SUBMIT, simplemente se confecciona con su procesador de textos, escribiendo cada uno de los comandos deseados en una línea. El fichero debe ir provisto del código SUB, ya que de otra forma, SUBMIT no lo aceptaría.

STAT (CP/M 2.2)

STAT.COM

Muestra la distribución en la memoria, así como el estado de protección contra escritura, y el estado del sistema de los ficheros indicados, en forma de lista. Asimismo, STAT devuelve una lista sobre el estado de todas las unidades actualmente activadas en el sistema.

Formato de entrada:

```
STAT
STAT FICHERO.XXX
STAT FICHE??.*
STAT DEV:
STAT DSK:
STAT USR:
STAT FICHERO.XXX $SYS
```

Descripción:

Mediante STAT, Vd. prácticamente puede ver las opciones de sus discos, y de los ficheros que contenga. El comando por tanto es similar al comando SHOW de CP/M 3.0. Puede averiguar el espacio libre de su disco, la longitud de cualquier fichero determinado, en records, o por ejemplo las zonas del usuario utilizadas. Por lo tanto, el comando STAT es, como si incluyera el comando DEVICE de CP/M 3.0. Mediante STAT DSK:, puede hacer que se le muestren todas las características, de un disco cualquiera. Además, mediante el comando STAT, Vd. puede poner ficheros en $R/O (Read Only), $R/W (Read and Write), $SYS (fichero del sistema), y $DIR (fichero de directorio). Dicho estado también se indica con el comando STAT. De modo que STAT, también incorpora el comando SET, de CP/M 3.0. Ya puede ver, que STAT es muy flexible y extenso.

STAT en concreto:

STAT: emite el estado de todas las unidades activadas en el sistema, en forma de lista, de esta manera:

```
<unidad>:<protección>, space: <n>k
```

STAT USR: muestra el número actual del usuario, así como todos las zonas del usuario ocupadas.

```
Active User : 5
Active Files: 0 1 3 5 7
```

STATDEV: equivale al comando DEVICE de CP/M 3.0, y emite una lista de los dispositivos de E/S. Es posible remodelar las asignaciones.

```
CON: is CRT:
RDR: is TTY:
PUN: is TTY:
LST: is LPT:
```

STAT <canal>=<dispositivo> asigna un nuevo dispositivo de E/S a un canal.

```
STAT CON:=CRT:, LST:=TTY:
```

STAT <unidad> $R/O, o STAT <unidad> $R/W, pone a la unidad indicada, en estado de sólo lectura, o en estado de lectura/escritura.

STAT <unidad>:, indica el espacio actual disponible en el disco indicado.

```
Bytes Remaining On A: 5k
```

STAT <fichero.XXX> puede utilizarse con los signos de sustitución (*,?), y muestra la distribución en la memoria, y la protección contra la escritura, o el estado correspondiente del, o de los ficheros indicados.

```
Recs Bytes EXT Acc
 144 18k 2 R/W A:(D2.BAS)
  14 2k 1 R/W A:DISC.BAS
```

El fichero se indica entre paréntesis, si se trata de un fichero del sistema, es decir, de un fichero provisto de la opción $SYS. No se muestra en el directorio "normal".

Con la opción $S (Size=tamaño) además puede averiguar el tamaño de un fichero, que sin embargo normalmente coincidirá con el número de los records.

STAT <fichero.XXX> $<opción> pone al, o a los ficheros (cuando se utilizan signos de sustitución) en estado <opción>. Las siguientes opciones son posibles:

```
$R/O - (Read Only) aplicar protección contra escritura
$R/W - suspender protección contra escritura
$SYS - (System) poner atributo del sistema
$DIR - (Directory) borrar atributo del sistema
```

STAT <unidad>:DSK: lista una tabla, indicada en una parte del BIOS, como sigue:

```
   A: drive characteristics
1368: 128 byte record capacity
 171: kilobyte drive capacity
  64: 32 bytes directory entries
  64: checked directory entries
 128: records/extend
```

```
   8: records/block
  36: sectors/track
   2: reserved tracks
```

STAT VAL: imprime una lista de los comandos, que pueden utilizarse con STAT. La lista tiene este aspecto:

```
Temp R/O disc: d:=R/O
Set indicator: d:filename.typ $R/O $R/W $SYS $DIR
Disc status : DSK: d:DSK
User status : USR:
IOByte assign:
CON: = TTY: CRT: BAT: UC1:
RDR: = TTY: PTR: UR1: UR2:
PUN: = TTY: PTP: UP1: UP2:
LST: = TTY: CRT: LPT: UL1:
```

Aplicación:

El comando STAT ofrece un poco más información, que el comando DIR, sobre el tamaño y la distribución de los ficheros. Además Vd. puede averiguar las asignaciones actuales de los periféricos de entrada y de salida (DEVices).

Dado que el comando STAT incorpora los comandos SET, DEVICE, SHOW de CP/M 3.0, y parcialmente DIR, con sus opciones, también puede averiguar su significado, estudiando en el manual, o en este libro, los diversos significados de dichos comandos.

SYSGEN (CP/M 2.2)

SYSGEN.COM

Copia el sistema CP/M 2.2 a un disco

Formato de entrada:

SYSGEN
SYSGEN fichero.XXX
SYSGEN *

Descripción:

SYSGEN escribe el resultado de un comando MOVCPM (vea abajo) sobre la pista del sistema de un disco. SYSGEN *, por lo contrario escribe sobre disco el sistema, que inmediatamente antes se ha confeccionado mediante el comando MOVCPM. SYSGEN <fichero> archiva un fichero del sistema, confeccionado mediante MOVCPM, en las pistas del sistema de un disco. Finalmente, SYSGEN sin parámetros, pregunta por el disco de origen, y por el disco destinatario, copiando el sistema a continuación.

Mediante MOVCPM es posible desplazar CP/M 2.2 en la memoria, algo que a veces se hace necesario. Vd. puede desplazar CP/M por unidades de 256 bytes. La posición en la memoria se indica con un valor entre 64 y 179. La sintaxis es la siguiente:

MOVCPM <tamaño>* ejemplo: MOVCPM 178*

MOVCPM 178* genera un CP/M desplazado 256 bytes más hacia "abajo", que el CP/M estándar. Ahora puede archivar dicho sistema desplazado en un disco, o almacenarlo directamente como sistema, mediante SYSGEN.

IX.34 TYPE (CP/M 2.2 y CP/M 3.0)

TYPE integrado y TYPE.COM

Muestra un fichero en la pantalla

Formato de entrada:

```
TYPE
TYPE FICHERO.XXX
TYPE AB?.*
TYPE FICHERO.XXX[NO PAGE]
TYPE AB?.*[NO PAGE]
```

Descripción:

El comando TYPE lista uno o varios ficheros en la pantalla. Cuando la pantalla se llena de texto, la impresión se detiene. Mediante la barra espaciadora o mediante RETURN, se prosigue con la impresión. Si introduce la opción "NO PAGE", la impresión no se detendrá.

Aplicación:

La versión incorporada de TYPE puede llamarse desde cualquier unidad, y desde cualquier zona del usuario. Si desea interrumpir el programa, introduzca control C (^C). Se obtiene una impresión del fichero por la impresora, si antes de pulsar RETURN, se introduce el comando control (^P).

IX.35 USER (CP/M 2.2 y CP/M 3.0)

USER comando integrado

Cambia la zna actual del usuario

Formato de entrada:

USER
USER n

Descripción:

Cada disco puede dividirse en 16 distintas zonas del usuario. De esta manera, para cada tipo de trabajo se dispone de una zona especifica, que sólo contenga los ficheros y programas necesarios para dicho trabajo. Los ficheros del sistema que figuren en la zona cero, son accesibles desde cualquiera de las demás zonas del usuario.

Aplicación:

La actual zona del usuario, se muestra en el prompt de CP/M (1B>), y puede modificarse mediante la introducción de USER. si no indica la nueva zona, el programa preguntará por ella. El programa SHOW, le muestra las zonas del usuario activas.

IX.36 XREF (CP/M 3.0)

XREF.COM

Confecciona una lista de referencia, para programas en ensamblador

Formato de entrada:

```
XREF FICHERO
XREF FICHERO $P
```

Descripción:

Los ensambladores MAC y RMAC, confeccionan una lista de todos los símbolos de un programa, con sus valores correspondientes. XREF aumenta el confort, indicando con cada símbolo, la línea del programa, en la que se encuentra dicho símbolo. El programa XREF necesita los ficheros SYM y PRN.

X. Diferencias entre micros CPC

La empresa Amstrad, que trabaja conjuntamente con la empresa alemana Schneider, y con la empresa Locomotive Software, ha logrado lanzar al mercado nada menos que 3 (!) ordenadores en poco menos de un año, que se supone que debían ser compatibles, pero que a pesar de todo presentan algunas variaciones importantes.

El primer ordenador de esta serie era el Amstrad CPC 464. Este ordenador dispone de un motor de cassette incorporado, y ha sido y sigue siendo muy apreciado por su BASIC rápido y confortable, así como por su precio bastante asequible. No duraría mucho hasta que la empresa anunciara una nueva máquina: el CPC 664.

Seguramente que la diferencia más relevante con respecto al CPC 464, es la unidad de disco incorporada. Los dos ordenadores disponen de una memoria RAM de 64 K, de los cuales sin embargo se reserva una buena cantidad a la memoria de pantalla (16 K), así como a las variables del sistema. En BASIC, el usuario puede disponer libremente de 42249 bytes. El CPC 664 posee un BASIC ligeramente modificado (V 1.1), que corrige algunos errores del BASIC del CPC 464, e incluye algunos comandos nuevos. Para ambos ordenadores, CPC 464 y CPC 664, se requiere CP/M 2.2. Esta es la versión usual de CP/M para ordenadores de 64 K.

Pero no sólo la vida interior ha sufrido algunos cambios, sino que también se aprecian modificaciones en el exterior: el CPC 664 es un poco más plano, y dispone de un teclado diferente, con un bloque de cursor elevado. El CPC se ha vuelto más profesional. Sin embargo, las empresas de software tienen sus problemas con estos dos "hermanos", porque lamentablemente no se ha llegado muy lejos con lo de la compatibilidad. Es lógico: si se eliminan errores del BASIC, y se incluyen comandos adicionales, es difícil que los dos ordenadores sean compatibles.

Los que realmente se ven afectados, son aquellos, que programen en código máquina, dado que aquí es, donde ha habido los mayores cambios. ¡Menos mal que existe CP/M! Aquí no ha cambiado nada para el usuario. Ya vé, que puede ser estupendo, disponer de una norma.

Dado que las empresas rivales han lanzado al mercado computadores de 128 kbytes y más, también la empresa Amstrad y asociados, se habrán visto obligados, a sacar un ordenador de 128 kbytes. De modo que Amstrad sorprendió a todos, con el Amstrad CPC 6128.

También el CPC 6128 dispone de una unidad floppy incorporada, que es seguramente la razón, por la que la primera cifra de su nombre sea un "6". De hecho, el CPC 6128 se parece más al CPC 664, que al CPC 464. El ordenador de 128 kbytes todavía es más plano, casi podría decirse que superplano. Dispone de un teclado mas profesional, que el CPC 664. Sobre todo las teclas del cursor, han tenido que pagar las consecuencias; se las ha desplazado a un sitio totalmente diferente. Es necesario acostumbrarse al teclado. La tecla COPY ya no se encuentra en la mitad del bloque del cursor, y las teclas shift, ya no figuran en la línea inferior del teclado, sino encima de las teclas "CONTROL" y "ENTER". Aquí ya es fácil cometer errores, por lo menos al principio.

El BASIC no ha cambiado en el CPC 6128. Se trata como siempre de la versión 1.1. El AMSDOS, es decir el sistema operativo incorporado para disco, por suerte tampoco ha cambiado, en ninguno de los tres ordenadores. A proposito, hay que decir, que las unidades de disco de 3 pulgadas son muy rápidas y fiables, y que alguna que otra empresa de unidades de disco, podría tomarse un ejemplo de ellas.

También se han modificado un poco las conexiones en la parte posterior del ordenador. Las conexiónes para la unidad de discos Floppy, y para la impresora (Centronics) han subido en calidad. Adicionalmente el CPC 6128 dispone de una conexión llamada Extension.

Al comprar La DDI-1 o el CPC 664, Vd. recibe un disco de sistema, que contiene CP/M 2.2 así como el lenguaje de programación LOGO. Con el CPC 6128 incluso se suministran dos discos, dado que no era posible almacenar todos los ficheros de CP/M 3.0 en un solo disco. Sobre todo los textos HELP, es decir aquellos textos, a los que puede acceder mediante el comando HELP de CP/M 3.0, son los que devoran una gran cantidad de espacio en el disco. Adicionalmente el comprador de un CPC 6128, también recibe un disco con la versión CP/M 2.2.

Sin embargo, la diferencia más importante del nuevo CPC con respecto a sus antecesores, es la memoria disponible. Prácticamente se ha duplicado. Pero no es el programador de BASIC quien disfruta de esta mejora, ya que éste sigue disponiendo solamente de los 42249 bytes. Sin trucos no hay nada que hacer. Sin embargo el registro disponible de 128 kbytes del CPC 6128, habilita al ordenador para pasar de CP/M 2.2 a CP/M 3.0+. Esta es la versión común de CP/M para los ordenadores de 128 kbytes. El sufijo + indica algunos cambios e inovaciones respecto a CP/M 3.0.

Las dos versiones de CP/M son muy agradables para trabajar. Claro que CP/M 3.0+ es un poco más confortable, pero ésto es una consecuencia lógica; también es más agradable conducir el nuevo Golf, que el viejo "escarabajo". Para ésto sirve el progreso precisamente, y la empresa Digital Research, desde luego que no dejará de mimar a su programa estelar.

Dado que este libro pretende servir como libro de entrenamiento, para todos los ordenadores CPC, se describen, tanto los comandos de CP/M 2.2, como los de CP/M 3.0. Casi todos los comandos de CP/M 2.2, pueden utilizarse también con CP/M 3.0. Los comandos que sólo pueden utilizarse con CP/M 3.0, están marcados de la forma correspondiente.

Apéndice 1

Tabla de conversiones

Tabla de conversiones

DECIMAL	HEXADECIMAL	BINARIO	ASCII
0	00	000 0000	NUL
1	01	000 0001	SOH
2	02	000 0010	STX
3	03	000 0011	ETX
4	04	000 0100	EOT
5	05	000 0101	ENQ
6	06	000 0110	ACU
7	07	000 0111	BEL
8	08	000 1000	ES
9	09	000 1001	HT
10	0A	000 1010	LF
11	0B	000 1011	VT
12	0C	000 1100	FF
13	0D	000 1101	CR
14	0E	000 1110	SO
15	0F	000 1111	SI
16	10	001 0000	DLE
17	11	001 0001	DC1
18	12	001 0010	DC2
19	13	001 0011	DC3
20	14	001 0100	DC4
21	15	001 0101	AAK
22	16	001 0110	SYU
23	17	001 0111	ETB
24	18	001 1000	CAN
25	19	001 1001	EM
26	1A	001 1010	SUB
27	1B	001 1011	ESC
28	1C	001 1100	FS
29	1D	001 1101	GS
30	1E	001 1110	RS

DECIMAL	HEXADECIMAL	BINARIO	ASCII
31	1F	001 1111	VS
32	20	010 0000	SP
33	21	010 0001	!
34	22	010 0010	"
35	23	010 0011	#
36	24	010 0100	$
37	25	010 0101	%
38	26	010 0110	&
39	27	010 0111	'
40	28	010 1000	(
41	29	010 1001	)
42	2A	010 1010	*
43	2B	010 1011	+
44	2C	010 1100	,
45	2D	010 1101	-
46	2E	010 1110	.
47	2F	010 1111	/
48	30	011 0000	0
49	31	011 0001	1
50	32	011 0010	2
51	33	011 0011	3
52	34	011 0100	4
53	35	011 0101	5
54	36	011 0110	6
55	37	011 0111	7
56	38	011 1000	8
57	39	011 1001	9
58	3A	011 1010	:
59	3B	011 1011	;
60	3C	011 1100	<
61	3D	011 1101	=
62	3E	011 1110	>
63	3F	011 1111	?

DECIMAL	HEXADECIMAL	BINARIO	ASCII
64	40	100 0000	@
65	41	100 0001	A
66	42	100 0010	B
67	43	100 0011	C
68	44	100 0100	D
69	45	100 0101	E
70	46	100 0110	F
71	47	100 0111	G
72	48	100 1000	H
73	49	100 1001	I
74	4A	100 1010	J
75	4B	100 1011	K
76	4C	100 1100	L
77	4D	100 1101	M
78	4E	100 1110	N
79	4F	100 1111	O
80	50	101 0000	P
81	51	101 0001	Q
82	52	101 0010	R
83	53	101 0011	S
84	54	101 0100	T
85	55	101 0101	U
86	56	101 0110	V
87	57	101 0111	W
88	58	101 1000	X
89	59	101 1001	Y
90	5A	101 1010	Z
91	5B	101 1011	Ä
92	5C	101 1100	Ö
93	5D	101 1101	Ü
94	5E	101 1110	^
95	5F	101 1111	_
96	60	110 0000	`

DECIMAL	HEXADECIMAL	BINARIO	ASCII
97	61	110 0001	a
98	62	110 0010	b
99	63	110 0011	c
100	64	110 0100	d
101	65	110 1101	e
102	66	110 0110	f
103	67	110 0111	g
104	68	110 1000	h
105	69	110 1001	i
106	6A	110 1010	j
107	6B	110 1011	k
108	6C	110 1100	l
109	6D	110 1101	m
110	6E	110 1110	n
111	6F	110 1111	o
112	70	111 0000	p
113	71	111 0001	q
114	72	111 0010	r
115	73	111 0011	s
116	74	111 0100	t
117	75	111 0101	u
118	76	111 0110	v
119	77	111 0111	w
120	78	111 1000	x
121	79	111 1001	y
122	7A	111 1010	z
123	7B	111 1011	ä
124	7C	111 1100	ö
125	7D	111 1101	ü
126	7E	111 1110	~
127	7F	111 1111	DEL

Apéndice 2

Caracteres de control de CP/M

Caracteres de control de CP/M

comando	efecto
^A	cursor izquierda*
^B	cursor a comienzo de línea, o a final si cursor ya está en el comienzo
^C	abortar CP/M; ejecutar RESET
^E	cursor a línea siguiente
^F	cursor derecha*
^G	borrar carácter debajo del cursor
^H	borrar carácter a la izquierda del cursor
^I	cursor a siguiente posición de TAB
^J	comando de ejecución (conmutación de líneas)
^K	borrar desde cursor hasta final de línea
^M	igual que RETURN
^P	activar/desactivar impresora
^Q	activar "scroll" después de ^S
^R	repetir línea de instrucciones
^S	detener impresión en pantalla

^U	borrar todos los caracteres de una línea
^W	volver a mostrar línea de instrucciones anterior*
^X	borrar carácter a la izquierda del cursor
^Z	final de cadena en PIP y ED

* significa: sólo disponible en CP/M con "bankswitching"

Apéndice 3

Todos los parámetros de PIP

Parámetros de PIP

A — Función de archivo. Sólo copia ficheros creados, o modificados desde el último archivado.

C — Confirm. CP/M espera confirmación, antes de borrar un fichero.

Dn — Borra a partir de n columnas. PIP borra todos los caracteres de un fichero, que se encuentren a partir de la columna n. Utilizar sólo con ficheros de texto.

E — Muestra (echo) texto en la pantalla. El contenido de datos, que se acaban de copiar,es mostrado en la pantalla. Utilizar sólo con ficheros de texto.

F — Suprime el carácter para "página siguiente" (Form feed). Algunas impresoras necesitan una (^L), para cambiar de página. Si su impresora no la necesita, puede eliminar los caracteres de control correspondientes mediante "F".

Gn — Toma o envía un fichero de o a la zona del usuario "n". Esta indicación debe figurar directamente detrás del nombre de fichero, y es la única que debe figurar allí. Ejemplo:

```
PIP B:[G5]=TEXTO.TXT[G1]
```

Copia el fichero TEXTO.TXT de la zona "1", a la zona "5".

H — Transferencia de datos hexadecimales. Siempre debería utilizar este parámetro, cuando quiera transferir ficheros HEX. PIP comprueba, si el contenido está escrito en el formato correcto de INTEL.

I Ignora el carácter de final de fichero, en ficheros HEX. Introduzca este parámetro para cada fichero, excepto para el último. Con la opción "I", simultáneamente se activa la opción "H" de PIP. Utilice el parámetro "H" para el último fichero:

```
PIP
fichero.HEX=PROG1.HEX[I],PROG2.HEX[I],PROG3.HEX[H]
```

K Impide que se muesten los nombres de fichero durante el copiado.

L Convierte todas las letras mayúsculas en minúsculas. Utilice este parámetro solamente con ficheros de texto, e introduzca además el parámetro "Z", si se trata de ficheros del WordStar.

N Enumera las líneas de un fichero. Las líneas comienzan con 1 en la primera línea, y se incrementan en uno, por cada línea. El mayor número posible es de seis cifras, sustituyéndose las que no se necesiten, por espacios. Detrás de las cifras hay dos puntos y un espacio. No lo utilice conjuntamente con los parámetros "E" o "N". Añadir el parámetro "Z" para ficheros del WordStar.

N2 Enumera las líneas para un programa BASIC.

O Transferencia de ficheros objeto. Se utiliza para transferir ficheros, que no sean, ni de texto, ni COM. No lo utiice con ficheros de texto.

Pn Fija la longitud de una página. El preajuste es de 60 líneas por página. Debería introducir también el parámetro "F", para borrar ocasionales caracteres de control, que indiquen final de página (^L). Utilizar sólo con ficheros de texto.

Qxxxx^Z Final de copiado después de esta cadena de caracteres. PIP copia un fichero, hasta, e inclusive, la cadena de caracteres introducida. Utilice el comando en dos líneas de instrucciones, para que la cadena no se convierta en mayúsculas:

```
PIP
CON:=TEXTO.TXT[Weiler]
```

Si escribiese todo en la primera línea detrás de PIP, dicha anotación se convertiría en letras mayúsculas.

R Copia ficheros del sistema. Estos ficheros no se muestran mediante DIR, y normalmente tampoco son copiados por PIP. Mediante "R" puede suspenderse dicha limitación.

Sxxxx^Z Comienza a partir de esta cadena de caracteres. PIP comienza a copiar a partir de esta cadena de caracteres. Escriba este comando en dos líneas, o la cadena se convertirá en letras mayúsculas. Utilizar sólo con ficheros de texto.

Tn Fijar amplitud de paso del tabulador. Normalmente CP/M trabaja con una amplitud de paso de ocho caracteres. Sin embargo esto no funciona, cuando se imprime un fichero, por lo que tiene que definirse de nuevo. "n" indica el número de espacios, que componen un paso del tabulador. Utilizar sólo con ficheros de texto.

U Convierte en letras mayúsculas. Todas las letras minúsculas son convertidas en mayúsculas. Utilizar sólo con ficheros de texto. Para los ficheros del WordStar, añadir el parámetro "Z".

V Verificar. Este parámetro hace que PIP compare la copia de un fichero, exactamente con el original.

W Escribir sobre ficheros con el atributo Read Only (RO). Normalmente estos ficheros sólo pueden leerse, pero no pueden modificarse, ni borrarse. Mediante el parámetro "W", todos estos ficheros se borran, sin esperar ninguna confirmación.

Z Borrar el bit de paridad (octavo bit). El juego de caracteres ASCII sólo emplea siete bits. El octavo bit se utiliza para varias tareas, en diversos progamas. El parámetro "Z" no es necesario, cuando se copia según unidades ASCII, como CON: o LST:.

Apéndice 4

Los parámetros de SET

OPCION	SIGNIFICADO
DIR	Vuelve a visualizar un fichero del SISTEMA en el directorio normal
SYS	Convierte un fichero en fichero del SISTEMA
RO	Determina que un fichero solo puede leerse
RW	Determina que un fichero puede leerse y modificarse
ARCHIV=OFF	Desactiva el atributo ARCHIV. Ello significa, que dicho fichero todavía no ha sido archivado. El programa PIP puede, mediante la opción [A], copiar ficheros con el atributo ARCHIV=OFF. El comando de PIP se introduce con los asteriscos, en sustitución de los nombres de los ficheros, y PIP copia todos aquellos ficheros, que hayan sido modificados mediante la opción AAU, desde la última vez, que hayan sido copiados con PIP. Después de que PIP haya copiado los ficheros, PIP fija sus atributos en ARCHIV=ON.
ARCHIV=ON	Activa el atributo ARCHIV. Ello significa que el fichero en cuestión ha sido archivado. Normalmente PIP modifica el atributo mediante la opción [A], después de archivar ficheros. Vd. también puede modificar el atributo, si utiliza el comando SET.

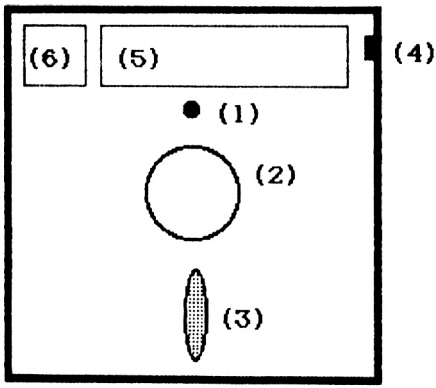
F1=ON/OFF Activa o desactiva el atributo F1 de fichero, definido por el usuario.

F2=ON/OFF Activa o desactiva el atributo F2 de fichero, definido por el usuario.

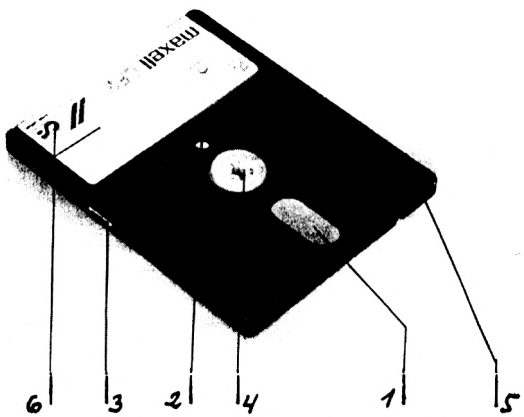
F3=ON/OFF Activa o desactiva el atributo F3 de fichero, definido por el usuario.

F4=ON/OFF Activa o desactiva el atributo F4 de fichero, definido por el usuario.

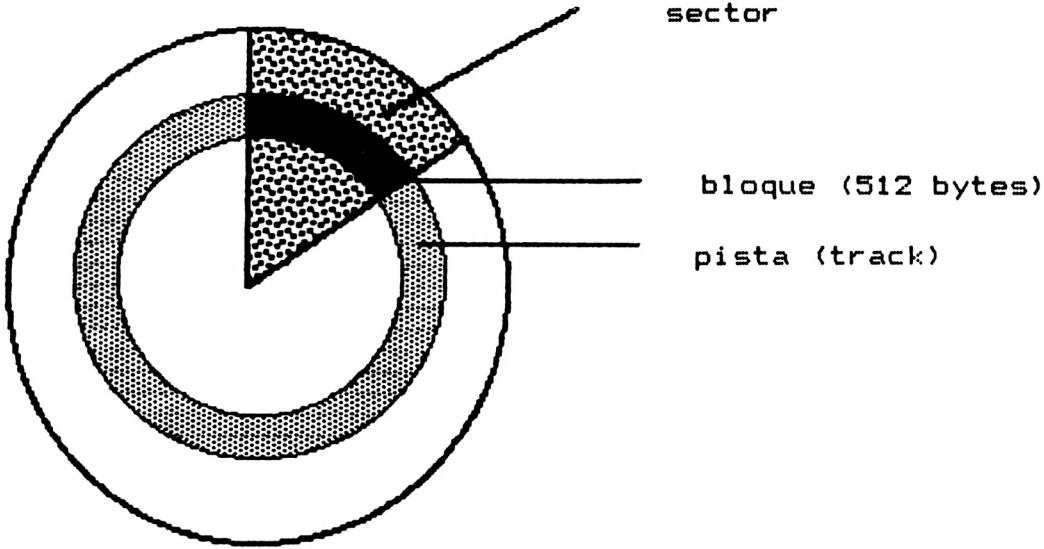
Discos de 5-1/4 pulgadas



(1) Agujero de alineación
(2) Agujero para eje de transmisión
(3) Ranura de acceso
(4) Ranura de protección
(5) Etiqueta (temporal)
(6) Etiqueta (permanente)



<1>CABEZA LECTURA/ESCRITURA
<2>AGUJERO DE ALINEACION
<3>CORREDOR DE PLASTICO
<4>EJE DE TRANSMISION
<5>PROTECCION CONTRA ESCRITURA
<6>ETIQUETA

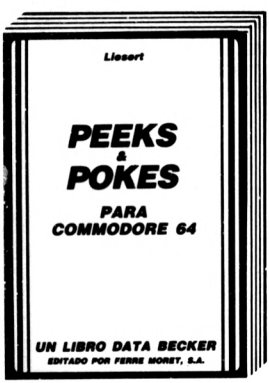

sector
bloque (512 bytes)
pista (track)

ROBOTICA PARA SU COMMODORE 64, 230 pág.
P.V.P. 2.800,- ptas.

En el libro de los robots se muestran las asombrosas posibilidades que ofrece el CBM 64, para el control y la programación, presentadas con numerosas ilustraciones e intuitivos ejemplos. El punto principal: Cómo puede construirse uno mismo un robot sin grandes gastos. Además, un resumen del desarrollo histórico del robot y una amplia introducción a los fundamentos cibernéticos.
Gobierno del motor, el modelo de simulación, interruptor de pantalla, el Port-Usuario cómodo del modelo de simulación, Sensor de infrarrojos, concepto básico de un robot, realimentación unidad cibernética, Brazo prensor, Oír y ver.

MANUAL ESCOLAR PARA SU COMMODORE 64, 351 pág.
P.V.P. 2.800,- ptas.

Este libro, escrito especialmente para escolares de grado medio y superior, contiene muchos interesantes programas de aprendizaje para solucionar problemas, descritos detalladamente y de manera fácilmente comprensible. Facilitan un aprendizaje intensivo y ameno, con, entre otros, los siguientes temas: Teorema de pitágoras, progresiones geométricas, palanca mecánica, crecimiento exponencial, verbos irregulares, ecuaciones de segundo grado, movimientos de péndulo, formación de moléculas, aprendizaje de vocablos, cálculo de interés y su capitalización. Una corta repetición de los elementos BASIC más importantes y una introducción a los rasgos esenciales del análisis de problemas, entre otros, completan el conjunto.

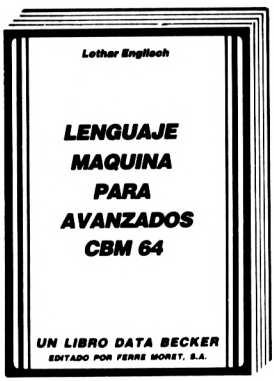


PEEKS y POKES, 177 pág.
P.V.P. 1.600,- ptas.

Con importantes comandos PEEK y POKE se pueden hacer también desde el Basic muchas cosas, para las que se necesitarían normalmente complejas rutinas en lenguaje máquina. Este libro explica de manera sencilla el manejo de PEEKs y POKEs. Con una enorme cantidad de POKEs importantes y su posible aplicación. Para ello se explica perfectamente la estructura del Commodore 64: Sistema operativo, interpretador, página cero, apuntadores y stacks, generador de caracteres, registros de sprites, programación de interfaces, desactivación del interrupt. Además una introducción al lenguaje máquina. Muchos programas ejemplo.

LENGUAJE MAQUINA PARA
COMMODORE 64, 1984, 201 pág.
P.V.P. 2.200,- ptas.

¡Por fin una introducción al código máquina fácilmente comprensible! Estructura y funcionamiento del procesador 6510, introducción y ejecución de programas en lenguaje máquina, manejo del ensamblador, y un atractivo muy especial: ¡un simulador de paso a paso escrito en BASIC!



LENGUAJE MAQUINA PARA AVANZADOS
CBM 64, 1984, 206 pág.
P.V.P. 2.200 ptas.

¿Ud. ha logrado iniciarse en código máquina? Entonces el «nuevo English» le enseñará cómo convertirse en un profesional. Naturalmente con muchos programas ejemplo, rutinas completas en código máquina e importantes consejos y trucos para la programación en lenguaje máquina y para el trabajo con el sistema operativo.

EL ENSAMBLADOR

Este libro ofrece al programador interesado una introducción fácilmente comprensible para los tan extendidos Assembler PROFI-ASS, SM MAE y T.EX.ASS. con consejos y trucos de gran utilidad, indicaciones y programas adicionales. Al mismo tiempo sirve de manual orientado a la práctica, con aclaraciones de conceptos importantes e instrucciones.
250 páginas. 2.200,- ptas.

TODO SOBRE EL FLOPPY 1541,
P.V.P. 3.200,- ptas.

La obra Standard del floppy 1541, todo sobre la programación en disquettes desde los principiantes a los profesionales, además de las informaciones fundamentales para el DOS, los comandos de sistema y mensajes de error, hay varios capítulos para la administración práctica de ficheros con el FLOPPY, amplio y documentado Listado del Dos. Además un filón de los más diversos programas y rutinas auxiliares, que hacen del libro una lectura obligada para los usuarios del Floppy.

MANTENIMIENTO Y REPARACION DEL FLOPPY 1541,
200 pág.
P.V.P. 2.800,- ptas.

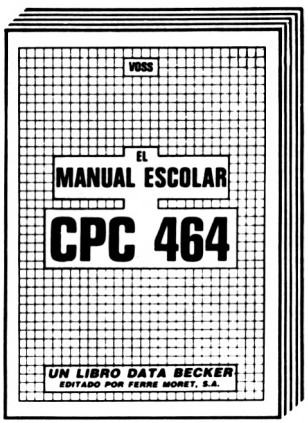
Saberse apañar uno mismo, ahorra tiempo, molestias y dinero, precisamente problemas como el ajuste del floppy o reparaciones de la platina se pueden arreglar a menudo con medios sencillos. Instrucciones para eliminar la mayoría de perturbaciones, listas de piezas de recambio y una introducción a la mecánica y a la electrónica de la unidad de disco, hay también indicaciones exactas sobre herramientas y material de trabajo. Este libro hay que considerarlo en todos sus aspectos como efectivo y barato.

EL MANUAL DEL CASSETTE, 190 pág.
P.V.P. 1.600,- ptas.

Un excelente libro, que le mostrará todas las posibilidades que le ofrece su grabadora de casettes. Describe detalladamente, y de forma comprensible, todo sobre el Datassette y la grabación en cassette. Con verdaderos programas fuera de serie: Autostart, Catálogo (¡busca y carga automáticamente!), backup de y a disco, SAVE de áreas de memoria, y lo más sorprendente: un nuevo sistema operativo de cassette con el 10-20 veces más rápido FastTape. Además otras indicaciones y programas de utilidad (ajuste de cabezales, altavoz de control).

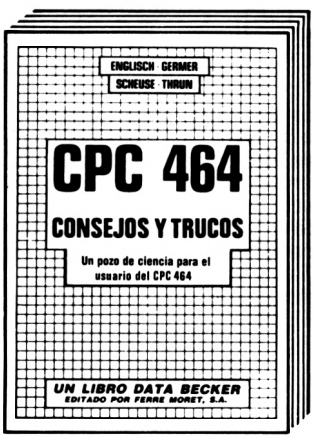
64 EN EL CAMPO DE LA TECNICA Y LA CIENCIA, 296 pág.
P.V.P. 2.800,- ptas.

Ofrece un campo fascinante y amplio de problemáticas científicas. Para esto el libro contiene muchos listados interesantes: Análisis de Fornier y síntesis, análisis de redes, exactitud de cálculo, formateado de números, cálculo del valor PH, sistemas de ecuaciones diferenciales, modelo ladrón presa, cálculo de probabilidad, medición de tiempo, integración, etc.



CPC-464 EL LIBRO DEL COLEGIO
P.V.P. 2 800,- ptas.

Escrito para alumnos de los últimos cursos de EGB y de BUP, este libro contiene muchos programas para resolver problemas y de aprendizaje, descritos de una forma muy compleja y fácil de comprender. Teorema de Pitágoras, progresiones geométricas, escritura cifrada, crecimiento exponencial, verbos irregulares, igualdades cuadráticas, movimiento pendular, estructura de moléculas, cálculo de interés y muchas cosas más.



CPC-464 CONSEJOS Y TRUCOS
P.V.P. 2.200,- ptas.

Ofrece una colección muy interesante de sugerencias, ideas y soluciones para la programación y utilización de su CPC-464: Desde la estructura del hardware, sistema de funcionamiento - Tokens Basic, dibujos con el joystick, aplicaciones de ventanas en pantalla y otros muchos interesantes programas como el procesamiento de datos, editor de sonidos, generador de caracteres, monitor de código máquina hasta listados de interesantes juegos.

MSX EL MANUAL ESCOLAR
P.V.P. 2.800,- ptas.

Escrito para alumnos de los últimos cursos de EGB y de BUP, este libro contiene muchos programas para resolver problemas y de aprendizaje, descritos de una forma muy completa y fácil de comprender. Teorema de Pitágoras, progresiones geométricas, escritura cifrada, crecimiento exponencial, verbos irregulares, igualdades cuadráticas, movimiento pendular, estructura de moléculas, cálculo de interés y muchas cosas más.

MSX GRAFICOS Y SONIDOS, 250 pág.
P.V.P. 2.800,- ptas.

Las computadoras MSX no sólo ofrecen una relación precio/rendimiento sobresaliente, sino que también poseen unas cualidades gráficas y de sonido excepcionales. Este libro expone las posibilidades de los MSX de forma completa y fácil. El texto se completa con numerosos y útiles programas ejemplo.

MSX PROGRAMAS Y UTILIDADES, 1985, 194 pág.
P.V.P. 2.200,- ptas.

El libro contiene una amplia colección de importantes programas que abarcan, desde un desensamblador hasta un programa de clasificaciones deportivas. Juegos superemocionantes y aplicaciones completas. Los programas muestran además importantes consejos y trucos para la programación. Estos programas funcionan en todos los ordenadores MSX, así como en el SPECTROVIDEO 318 328. ESTRACTO DEL CONTENIDO: Volcado memoria hexadecimal. Editor gráficos. Editor de sonido. Escritura de ordenador. Lista referencia de variables. Calendario. Desensamblador. ADMINISTRACION de una colección de discos L.P. HOLLOW - JUEGO DE LAS CEREZAS. DIAGRAMAS DE BARRAS. TABLAS DEPORTIVAS.

ZX SPECTRUM EL MANUAL ESCOLAR
P.V.P. 2.200,- ptas.

Escrito para alumnos de los últimos cursos de EGB y de BUP, este libro contiene muchos programas para resolver problemas y de aprendizaje, descritos de una forma muy completa y fácil de comprender. Teorema de Pitágoras, progresiones geométricas, escritura cifrada, crecimiento exponencial, verbos irregulares, igualdades cudráticas, movimiento pendular, estructura de moléculas, cálculo de interés y muchas cosas más.

ZX SPECTRUM CONSEJOS Y TRUCOS, 211 pág.
P.V.P. 2.200,- ptas.

Una interesante colección se sugestivas ideas y soluciones para la programación y utilización de su ZX ESPECTRUM. Aparte de muchos peeks, pokes y USRs hay también capítulos completos para, entre otros, entrada de datos asegurado sin bloqueo de ordenador, posibilidades de conexión y utilización de microdrives y lápices ópticos programas para la representación de diagramas de barra y de tarta, el modo de utilizar óptimamente ROM y RAM.

METODOLOGIA DE LA PROGRAMACION
P.V.P. 2.200,- ptas.

El primer libro recomendado para escuelas de enseñanza de informática y para aquellas personas que quieren aprender la programación. Cubre las especificaciones del Ministerio de Educación y Ciencia para Estudios de Informática. Realizado por un alto mando del ejército Español, un Dr. Ingeniero y Diplomado en Informática y profesor de la UNED y por un oficial técnico especialista en informática de gestión. Utilizado en todos los institutos politécnicos del ejército español. Es un seguro candidado a ediciones en lengua inglesa, alemana y francesa. Es el primer libro que introduce a la lógica del ordenador. Es un elemento de base que sirve como introducción para la programación en cualquier otro lenguaje. No se requieren conocimientos de programación ni siquiera de informática. Abarca desde los métodos de programación clásicos a los más modernos.

TODO SOBRE EL NUEVO COMMODORE 128

P.V.P. 2.200,- ptas.

El Libro de Primicias del Commodore 128 no ofrece solamente un resumen completo de todas las características y rendimientos del sucesor del C-64 y con ello una importante ayuda para su adquisición. Muestra, además, todas las posibilidades del nuevo equipo en función de sus tres modos de operación.

Entre otros se describen el hardware, los modos de operación: modo 64, modo 128 y modo CP/M, las configuraciones de memoria, la disposición de la página cero, trabajos con dos pantallas, modo de 80 caracteres, Basic V 7.0: comandos de gráficos y sonidos, comandos de control, periféricos rápidos (1571) etc.

RESPUESTA
COMERCIAL

F.D. Autorización 6975
(B.O. de Correos N.º 80 de 26-7-85)

HOJA PEDIDO
DE LIBRERIA

NO NECESITA
SELLOS

A franquear
en destino

FERRE MORET, S.A.

Apartado N.º 551. F.D.
08080 BARCELONA

RESPUESTA
COMERCIAL

F.D. Autorización 6975
(B.O. de Correos N.º 80 de 26-7-85)

HOJA PEDIDO
DE LIBRERIA

NO NECESITA
SELLOS

A franquear
en destino

FERRE MORET, S.A.

Apartado N.º 551. F.D.
08080 BARCELONA

Puesta al día de datos

EDITORIAL FERRER MORET, S.A. mantiene vivo y amplía el contenido informativo de sus libros y programas, mediante el envío de un servicio de puesta al día, junto con una síntesis noticiosa de la actualidad y perspectivas de la realidad informática española.
Agradecemos cualquier sugerencia o crítica que desee formular y que nos ayude a mejorar las ediciones. Muchas gracias.

¿Qué añadiría?
.....

.....

¿Qué suprimiría?
.....

.....

Observaciones
.....

.....

Título del libro
.....

Nombre
.....

Dirección Tfno.
.....

Código Postal y Población Provincia
.....

UN SERVICIO GRATUITO

Información y Pedidos

FERRE MORET, S.A. cuenta con un amplio fondo de libros y Software y mantiene un servicio de información por correo sobre las novedades que edita.
Agradecemos nos indique los temas que representan para Vd. mayor interes.

Libros

☐ ATARI	☐ MSX	☐ AMSTRAD
☐ COMMODORE	☐ LENGUAJES	☐ APPLE
☐ SINCLAIR	☐ IBM	☐ SOFTWARE

Si está interesado en recibir alguno de estos servicios, rellene y envíe la tarjeta correspondiente; no necesita franqueo. Muchas gracias.

DESEO RECIBIR EL LIBRO .....

EL PROGRAMA .....

☐ Adjunto cheque ☐ Contra reembolso
.....

Nombre
.....

Dirección Tfno.
.....

Código Postal y Población Provincia
.....

UN SERVICIO GRATUITO

EL CONTENIDO:

¡Dominar CP/M por fin! Desde explicaciones básicas para almacenar números, la protección contra la escritura, o ASCII, hasta la aplicación de programas auxiliares de CP/M, así como «CP/M interno» para avanzados, cada usuario del CPC rápidamente encontrará las ayudas e informaciones necesarias, para el trabajo con CP/M. Este libro tiene en cuenta las versiones CP/M 2.2, así como CP/M Plus (3.0), para el AMSTRAD CPC 464, CPC 664 y CPC 6128.

Extracto del contenido:

- La finalidad de CP/M
- El disco del sistema
- Reglas para nombres de fichero
- Ordenes incorporadas
  USER, DIR, ERASE
- Comandos transitorios
  SET, PROTECT, SHOW, SUBMIT
- Todo sobre PIP
- Impresión de varios ficheros en serie
- Leer formatos de discos ajenos
- Diferencias entre los ordenadores CPC
  y mucho más

ESTE LIBRO HA SIDO ESCRITO POR:

Elmar A. Weiler, periodista libre y especialista en representar materias complicadas de forma comprensible. Después de sus libros sobre WordStar y dBase, éste ya és su tercer libro claramente estructurado y hecho a la medida del usuario.
Joerg Schieb, experto programador de programas de ficheros, y autor del libro del Floppy para el CPC, conoce a la perfección la programación en código máquina del CPC.

ISBN 84-86437-38-5

AMSTRAD CPC

MÉMOIRE ÉCRITE
MEMORY ENGRAVED
MEMORIA ESCRITA

https://acpc.me/