

## ASPECTOS PARTICULARES DE DISCOS RIGIDOS Y UNIDADES DE DISCO RIGIDO

Temas anteriores ponían de manifiesto características comunes de disquetes y discos rígidos. En lo que sigue se tratan aspectos propios de los rígidos.

Los discos magnéticos rígidos o duros difieren de los disquetes por su gran capacidad de almacenamiento, por la mayor rapidez con que se accede a los datos, y por la mayor velocidad con que se los transfiere desde o hacia la memoria. Ello es fruto de su mayor densidad superficial (más bits por pista y más pistas por centímetro radial), de su mayor velocidad de rotación, de un sistema más veloz de posicionamiento del cabezal, y de una controladora más inteligente.

Por ser de material duro, un rígido no presenta las deformaciones de un disquete y permite una mayor precisión en el acceso a cada pista. Al respecto, un servomecanismo permite ubicar y seguir cada pista, lo cual permite una mayor confiabilidad, dada la proximidad entre las pistas contiguas.

El término "duro" ("hard disk") se refiere a que está constituido por platos rígidos de aluminio, o de cristal con implante cerámico en el presente. Existen discos rígidos fijos como los que están en una caja hermética en el interior del gabinete de una PC, y también los hay *removibles*, los cuales son transportables. Las unidades de disco, según el tipo que sean, pueden contener uno o más discos (figura 2.6).

Típicamente en una PC de escritorio son uno o dos platos, de un diámetro de 3 ½ pulgadas.

Por fabricarse los platos bajo normas estrictas, y variar muy poco de tamaño con la temperatura, el material magnético que los recubre permite 3000 tpi o más, a la par que 50.000 o más bytes por pista (o sea 100 ó más sectores por pista). También ha influido en esto la aplicación de magnetización perpendicular a la superficie de la capa magnetizable (figura 2.23), en lugar de la polarización de superficie (figura 2.5). Resulta así una elevada capacidad de almacenaje (500 MB - 1 TB o más) en uno o más platos pequeños, y unidades compactas.

Además, por la gran velocidad de giro<sup>2</sup>, y por tener el cabezal movimiento rápido en discos de pequeño radio, se tiene comparativamente cortos tiempos de acceso<sup>3</sup>.

Más sectores por cilindro posibilitan que un archivo entre en un solo cilindro, para que el cabezal en lo posible no deba cambiar a otro cilindro, resultando más rápida la escritura y posteriores lecturas; a la par que reduce la fragmentación de archivos en varios cilindros, con la pérdida de tiempo que ello ocasiona.

Las cabezas "magneto-resistivas" (MR) basadas en una resistencia variable con el campo magnético del disco, no usan bobina, y permiten mayor densidad de grabación. Pueden leer densidades 4 veces mayores que las cabezas convencionales (hasta 3 Gbits/inch<sup>2</sup>), y más separadas del plato. Se basan en el hecho de que si los electrones se mueven a través de una aleación de níquel-hierro, su movilidad disminuye (resistencia) cuando lo hacen paralelamente a la orientación magnética del material.

En cada cabeza debe existir un elemento con bobina para escribir y otro MR para leer.

Las cabezas Giant MR (GMR) varían más su resistencia, son más sensibles al magnetismo, que las MR (la nota al pie sigue en la página siguiente)<sup>4</sup>

<sup>1</sup> A veces denominadas "Winchester 30-30" –como el rifle– por presentar un modelo de disco 30 MB fijos y 30 MB removibles.

<sup>2</sup> De 3600 hasta 7200 r.p.m., frente a 300 r.p.m. de un disquetel

<sup>3</sup> El tiempo de acceso en un disco rígido es menor que 10 mseg., siendo el de un disquete de 170 mseg.

<sup>4</sup> Esta tecnología opera con el spin de los electrones, que como la masa y la carga es una propiedad de los mismos. Cada uno tiene un giro (spin) alrededor de un eje que pasa por él. El magnetismo de un átomo de hierro se debe a que tiene 4 electrones girando en un sentido (spin 0) y otro 4 girando en sentido opuesto (spin 1). En una corriente eléctrica ordinaria los spins se orientan al azar, y no influyen en el valor de la resistencia del conductor por donde se da la circulación. En los átomos ferromagnéticos (hierro, cobalto), los spins de determinados electrones de átomos vecinos tienden a alinearse. Cuando una corriente atraviesa este material éste impide el paso de electrones que tienen un cierto tipo de spin (0 ó 1), resultando una corriente formada por electrones de un solo tipo de spin, según sea. Un material ferromagnético puede afectar el flujo de corriente en un metal no magnético cercano. En las cabezas GMR una capa de metal no magnético está entre dos capas ferromagnéticas, una de las cuales presenta una magnetización fija. Cuando la cabeza se desplaza por una pista del disco sus débiles campos magnéticos bastan para cambiar la magnetización de la segunda capa ferromagnética, de forma de acompañar o ser opuesta a la magnetización fija de la primera capa citada. Si acompaña, los electrones que por su spin pueden desplazarse por el conductor, podrán hacerlo.

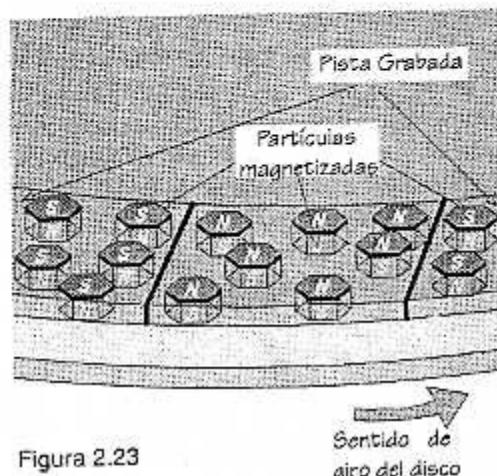


Figura 2.23

Así hoy día es factible leer discos con densidades de más de 10 Gbit/inch<sup>2</sup> con elementos GMR de 0,02 micrones.

Esto permite usar discos y cabezales más pequeños y livianos, lo cual redundaría en unidades más pequeñas, de menor peso, con mayores velocidades de giro (15.000 r.p.m) y menores tiempos de acceso.

Desde 1997 cada año se duplica los Gbit/inch<sup>2</sup>, pero a medida que ello ocurre, los dominios magnéticos resultan muy pequeños ("superparamagnetismo"). Ello atenta contra el tiempo que el material retiene los datos. Para evitar esto nuevos discos tendrán en cada cara dos capas magnetizables, con una capa de tan solo 3 átomos del metal ruthenio, lo cual fuerza a que capas adyacentes estén magnetizadas en forma opuesta. Esta tecnología se conoce como "antiferromagnetically-coupled (AFC)" y permite 100 Gbit/inch<sup>2</sup>.

Los discos rígidos de la década del 80 al presente han aumentado su capacidad de 10 MB a 2 Terabytes o más<sup>1</sup>; y su velocidad de transferencia de 100 KB a más de 50 MB/seg. Han disminuido su tiempo de acceso, de casi 100 msec. a menos que 10 msec. Su costo por MB almacenado pasó de U\$S 150 a centavos de dólar.

La estructura en cilindros, pistas y sectores, así como la escritura o lectura de las mismas es similar a la de los discos duros, y de hecho se han tratado al describir los discos duros. Pero en los discos duros cada cabeza se sitúa a unas pocas millonésimas de milímetro (menos que el grosor de un cabello) por sobre la pista que recorre, sin rozarla. Así se evita el desgaste de la superficie del disco debido a la fricción de la cabeza.

Cada cabeza flota como un navío catamarán en un colchón de aire producido por la gran velocidad de giro de los platos. Se reservan pistas de un cierto cilindro para estacionar las cabezas cuando el motor se detiene. Actualmente existen discos con cabezas de semicontacto, o de proximidad, que están en contacto con la superficie de la cara durante cortos tiempos, para sentir mejor variaciones de campos magnéticos.

En los discos, el "tiempo medio entre fallas (MTBF)" es una estimación estadística de cuánto en promedio durará antes de que falle. Por ejemplo, si MTBF = 87.600 horas implica que podría llegar a funcionar 10 años sin parar, libre de fallas que impidan su funcionamiento, aunque la garantía de devolución por este tipo de fallas, es típicamente de un año; siendo además que un disco se puede volver obsoleto en 2 ó 3 años.

El hecho de que un rígido esté contenido en una caja sobrepresurizada con filtro de aire, evita en gran medida que queden partículas abrasivas de suciedad entre una cabeza y una cara, que reducen su vida útil. Otra diferencia con las disqueteras, es que los platos de un rígido deben girar sin parar mientras el disco está en uso<sup>2</sup>, aunque no se lean o escriban archivos. Puesto que las cabezas no tocan las caras, no hay problemas de desgaste, y tampoco se pierde tiempo hasta que los platos alcancen la velocidad de rotación requerida.

En un disco con varios platos, la forma de numerar los cilindros y caras es similar a la descripta antes para un disquete. Igualmente como en éste, las cabezas de escritura/lectura se mueven al unísono, y sólo se puede escribir o leer una pista de una cara por vez, seleccionando la cabeza correspondiente a esa cara.<sup>3</sup>

También existen diferencias entre rígidos y disqueteras en relación con ciertos campos de las pistas y sectores, aunque en esencia la organización de cilindros, pistas y sectores se conserva.

En el presente, los siguientes parámetros sirven para comparar y decidir el tipo de disco a usar:

- Capacidad de almacenamiento
- Fabricante
- Tipo de unidad (IDE, SCSI)
- Tiempo promedio de posicionamiento de una pista a otra al azar
- Velocidad de transferencia
- Revoluciones por minuto (la latencia a 7200 r.p.m. dura la mitad que a 3600 r.p.m.)
- Tamaño y performance del caché para disco incorporado a la unidad
- Costo por MB almacenado

### ¿Cómo es en detalle una pista y un sector de un disco rígido?

La estructura de una pista y sector de un disco rígido (figura 2.24), es muy semejante a la de un floppy (figura 2.11). Por no presentar un rígido el agujero "index hole", no requiere el campo "GAP 80" al

En caso de tenerse magnetizaciones opuestas los electrones no podrán desplazarse. Así se controla la resistencia de la cabeza al paso de una corriente, en correspondencia con los campos magnéticos existentes las pistas

<sup>1</sup> Actualmente se consiguen densidades de más de 150 megabytes por pulgada cuadrada, con 3000 o más tpi

<sup>2</sup> Salvo que se quiera ahorrar energía, lo cual requiere una espera en cada nuevo arranque.

<sup>3</sup> Existen discos rígidos en los que se pueden activar varias cabezas simultáneamente, para transferir más datos

comienzo de cada pista, siendo que en consonancia con la lectura del campo IAM se activa un cable que llega a la electrónica ATA-IDE o SCSI. Los campos "GAP" y otros requieren menos bytes que los de un disquete, por ser más constante la velocidad de giro de un rígido, merced a un control electrónico más sofisticado. Por lo tanto, un sector de un rígido ocupa en total menos bytes que un sector de un disquete.



Los números que acompañan a las letras indican la cantidad de bytes que ocupa cada campo

Figura 2.24

Esto permite aprovechar mejor las pistas, amen de existir mayor densidad, y de usar codificación RLL. Los campos CRC en los sectores de un rígido se denominan ECC (Error Correction Code o *error checking and correcting* o *error detecting and correcting* - EDC). Se calculan y verifican del mismo modo que los CRC, pero permiten no sólo detectar errores, sino también *corregirlos*, dentro de ciertos límites. La corrección implica determinar la ubicación de los bits errados, para restaurar su valor original.

Se requiere ECC por la mayor cantidad de errores que tienen lugar en la búsqueda de pistas, y en la lectura de bytes de encabezados y de datos, por la muy alta densidad (bpi y tpi) que presentan los discos rígidos. Los programas en la ROM del microcontrolador de una unidad ATA-IDE o SCSI deben corregir cualquier tipo de error de lectura salvable, conforme los permita el código ECC.

Luego del último sector de la pista sigue un campo que indica fin de pista (GAP93, dibujado al comienzo). Otra diferencia, hace a la posibilidad que se tiene en los discos rígidos de poder indicar el mal estado de una pista o sector. A tal fin, el campo CHS, que contiene los números de cilindro, pista y sector, reserva el cuarto byte (sector flag) para indicación de sector dañado, pista mala reasignable, pista reasignada, pista mala sin reasignación. Este cuarto byte en los sectores de un disquete se usa para indicar número de bytes de datos.

### ¿Qué particularidades tiene el formateo de un disco rígido?

En la actualidad los discos rígidos vienen con el formateo físico o de "*bajo nivel*" realizado por el fabricante: cada cara tiene generadas un número predeterminado de pistas y sectores; asimismo cada sector tiene grabado su encabezamiento, y reservada una zona para datos a almacenar.

El denominado formateo lógico o de "*alto nivel*" lo realiza el usuario mediante el mismo comando "Format" que usa para formatear disquetes<sup>1</sup>, para crear al igual que en éstos el área del sistema (con el registro de arranque, la FAT y el Directorio raíz).

En definitiva, con el DOS, un disco rígido queda organizado *igual* que un disquete, con una pequeña área del sistema y otra área para datos, y con cada sector para 512 bytes. La diferencia estriba que en el disquete el formateo físico y lógico se realizan en un solo paso, mientras que un disco rígido primero el fabricante lleva a cabo el formateo físico, y a posteriori el usuario hace el formateo lógico. Además, un disquete no puede ser particionado, en varios independientes, y en un rígido ello es factible.

### ¿Qué se conoce como "interleave" de un disco y qué hace el caché de disco?

En computadoras anteriores con microprocesadores más lentos, si se quería leer sectores sucesivos de una pista, mientras se transfería a memoria los datos de un sector, el siguiente ya había pasado por la cabeza lectora, debiéndose esperar una revolución completa del disco para volverlo a leer.

Por ejemplo, sea una unidad de disco en la cual a 3600 r.p.m. una cabeza lee todos los bits de un sector en 0,001 milisegundos. Si en ese tiempo la controladora, buses, memoria y procesador son lo suficiente rápidos para permitir que los 512 bytes de datos del sector lleguen a memoria principal, podrá leerse a continuación el sector que en la pista accedida sigue al sector antes leído. Caso contrario debería esperarse otra vuelta hasta que el sector siguiente del leído vuelve a pasar frente a la cabeza.

Para solucionar esto, en el formateo, entre dos sectores con numeración correlativa existía otro no correlativo; como ser, el sector 10 se halla entre el 1 y 2, el 11 entre el 2 y 4, etc. Con esta disposición, si

<sup>1</sup> En un disquete dicho comando realiza una tras otra los formateos físico y lógico, pero con un disco rígido sólo lleva a cabo el segundo.

se ordena leer el 1 y el 2, se tiene el doble de tiempo para transferir y procesar los datos de un sector antes de que la cabeza pase por el siguiente, pues hay otro sector (el 10) entre ambos.

Se dice en este caso que los sectores sucesivos están *intercalados* con un factor de "interleaving" de dos. De esta forma, con 32 sectores por pista, en vez de esperar 32 vueltas para leer todos los sectores contiguos, son suficientes 2 vueltas.

Hoy día, merced a la existencia de controladores IDE o SCSI con caché incorporado (figura 2.25), que guardan en un cache de disco una pista o un cilindro completo, y a la velocidad de los procesadores y buses, no es necesario especificar en el formateo físico de un disco rígido el factor de interleaving requerido.

**Los discos duros actuales vienen formateados de manera "natural": el número de sector aumenta en uno de un sector al que le sigue en cada pista (interleave 1:1).**

O sea que pueden escribir o leer los datos de todos los sectores de una pista en una sola vuelta del disco. El caché de disco (como ser de 2 MB) recibe todos los sectores de una pista, a la velocidad de transferencia interna de la unidad de disco. Después, los 512 bytes de cada sector se transfieren del caché a memoria, con la velocidad que permite el bus. Así sólo se requiere tiempos tseek y tlat para acceder al primero de varios sectores consecutivos, y luego los siguientes se transfieren desde el caché a memoria a gran velocidad.

Los disquetes, por su menor velocidad de giro, en general siempre se han usado con un interleave de 1:1.

### ¿Qué significa que un disco está muy fragmentado ?

Hemos visto que para minimizar los tiempos de acceso a los sectores que componen un archivo, cada cluster se corresponde en el disco con sectores físicos sucesivos, al igual que clusters con numeración progresiva, los cuales muy probablemente estarán en un mismo cilindro (o parte en el siguiente). Por tal motivo, cada vez que en una FAT un SO asigna clusters libres a un archivo, normalmente éstos tienen numeración consecutiva. Pero cuando el archivo crece, si en el interín el SO asignó clusters a otros archivos, los nuevos clusters que le asigna a este archivo van en numeración consecutiva a partir del primer número de cluster que quedó libre. Entonces sus clusters dejarán de tener numeración progresiva y por ende le corresponderán sectores (sucesivos para cada cluster) que estarán esparcidos en distintos cilindros.

Esto puede ocurrir cada vez que crece un archivo, lo cual trae aparejado más demoras en la lectura de archivos pues el cabezal debe realizar muchos movimientos de posicionamiento para ir de un cilindro a otro. Resulta así una distribución azarosa de porciones de archivos por distintos cilindros, conocida como "fragmentación" externa.

Para lograr que todos los archivos de un disco tengan sus sectores en un mismo cilindro o en cilindros vecinos, se recurre a un programa para "desfragmentar", cuando un disco se vuelve muy lento.

### ¿Cómo están organizados físicamente los sectores en las pistas de un disco rígido ?

La organización de la figura 2.3 con igual número de sectores en cada pista, desperdicia capacidad de almacenamiento, pues las pistas exteriores podrían tener más que el doble de sectores que las más internas, de menor radio. La mayor densidad de bits de éstas (bpi) determina y limita el número de sectores que tendrán otras pistas más alejadas del centro.

En la mayoría de las unidades de disco actuales (tipo IDE o SCSI) se emplea igual densidad de grabación en todas las pistas (*constant density recording = CDR*), y "grabación zonal" ("zone recording"), que consiste en formar desde el centro del disco hacia afuera, varias zonas de cilindros, cada una con más sectores por pista que la más interna anterior. Así se logra hasta un 50% más de capacidad que con la otra disposición. Puesto que las subrutinas del BIOS "ven" cualquier disco con igual número de sectores por pista, una IDE o SCSI deben simularlo. Por ello si un disco tiene dos platos físicos, cuando se enciende un computador puede aparecer como que tiene 6 platos lógicos (12 cabezas) para compensar la mayor capacidad real. Dado que las pistas más externas tienen más sectores por pista, y que la velocidad de giro es constante, los archivos guardados en estas pistas se benefician por una mayor velocidad de transferencia interna.

### ¿Cómo localiza el cabezal más rápidamente un cilindro en un rígido ?

Hoy día las unidades de disco rígido de más de 80 MB, no usan como las disqueteras un motor paso a paso para ubicar el cabezal en cada pista de un cilindro. El cabezal (figura 2.10.a) no avanza en línea recta, sino que pivota alrededor de un eje, como el brazo de los tocadiscos con púa. La armadura se mueve de forma parecida al de la bobina de un parlante ("voice coil" identifica este sistema de posicionamiento). Sobre la armadura se tiene una bobina, la cual está sometida a un fuerte campo magnético creado por un imán

permanente que está fijo fuera de la armadura un pequeño campo magnético, que rueda. Cuando el sistema de control envía una determinada corriente por la bobina, ésta también genera al accionar con el campo existente, creado por el imán permanente, hace mover bobina, y por ende la armadura hasta la pista (cilindro) seleccionada. Si la cabeza no se encuentra justo sobre dicha pista, tiene lugar un ajuste fino automático de su posición, merced a la existencia de información extra de servocontrol escrita (*servowriter*) antes de cada sector o en una cara de un plato dedicada a esa información, donde no se almacenan archivos. Si estas señales al ser sensadas no tienen la amplitud suficiente, la controladora varía la corriente de la bobina hasta que el cabezal esté justo sobre la pista. Esto permite la localización exacta de cada pista, con independencia de cualquier variación de las dimensiones de los platos por la temperatura. Resulta así que las cabezas hacen un "seguimiento" de las pistas, de donde deviene su denominación "*track following system*". A tal efecto el sistema realiza en forma automática periódicas autocalibraciones (cada 5 ó 25 minutos) con los discos girando, actualizando datos sobre variaciones en la memoria de la controladora IDE o SCSI. Cuando la unidad de disco se apaga, el cabezal se estaciona automáticamente (*automatic head parking*) fuera de las pistas con datos, merced a que un resorte lleva la armadura a una posición fija, que el campo del imán permanente ayuda a mantener. Al encender el equipo, la fuerza que se origina al circular corriente por la bobina de la armadura (para posicionar el cabezal) estira dicho resorte y mueve la misma.

### ¿Qué funciones realiza una unidad de disco inteligente ATA-IDE?

Unidades de discos rígidos anteriores, adecuadas al estándar ST506, requerían una interfaz-controladora cuya circuitería estaba en una placa insertable en un zócalo ("slot"), con funciones análogas a las descriptas al tratar la interfaz-controladora de disquetera (figuras 2.22 a y b). Por las razones que se expondrán, fue necesario que la interfaz-controladora esté localizada junto a la unidad de disco rígido, *integrada* con la electrónica de este periférico (figuras 2.25), de donde provienen las siglas IDE de "integrated drive electronics". Las siglas ATA (AT Attachment, de la PC AT) son sinónimas de IDE (marca registrada), por lo cual se designan ATA/IDE.

Dadas las actuales capacidades de los discos rígidos, y las velocidades de acceso y de transferencia de una unidad de disco rígido (*drive*), se requiere que la electrónica ligada a ella sea "inteligente", conteniendo un microcontrolador, con un programa en su ROM, y una RAM veloz para *buffer* del periférico.

El microcontrolador maneja los sistemas con servowriter citados anteriormente, corrige sobre la marcha errores de lectura de un sector, soporta un caché de disco, simula hacia el exterior un disco compatible con el sistema operativo y BIOS existentes, y realiza rápidamente otras tareas complejas. También incluye la mayoría de las funciones de la interfaz controladora descriptas anteriormente para la unidad de disquetes. Asimismo en el presente puede operar como "bus master" en el bus PCI, a los efectos de leer o escribir sectores por ADM sin necesidad del controlador de ADM.

La proximidad física entre la interfaz y las cabezas evita retardos e interferencias (ruidos eléctricos) en la lectura o escritura, que se producirían si se quiere transmitir a gran velocidad información entre la electrónica de la unidad de disco y una interfaz más alejada, como la existente para una unidad ST506. Una unidad IDE es una buena solución de compromiso entre velocidad y costo para sistemas monotarea corrientes. No requiere de una placa interfaz especial en la "mother" como la SCSI. Acorde con lo anterior, la electrónica de una unidad "inteligente" de disco IDE puede incorporar funciones tratadas en la interfaz-controladora de disquetera, en particular en lo concerniente a la existencia de registros direccionables ("ports") para enviarle un block de comandos y para recabar el estado de la unidad<sup>1</sup>, mediante la ejecución de subrutinas del BIOS. El microcontrolador de la unidad de disco detecta y lleva a cabo estos comandos (del tipo posicionar las cabezas en un cilindro, leer o escribir un sector, etc.) mediante la ejecución de instrucciones contenidas en su ROM.

En relación con el port de datos, en la electrónica de la unidad existe un "sector buffer", o sea un buffer con capacidad para un sector del disco, para dar tiempo a la corrección de datos leídos, que realiza el microcontrolador, usando el área ECC del sector (figura 2.24). Sólo si los datos leídos son correctos, se realiza la transferencia hacia memoria a través del bus PCI (figura 1.80 de la Unidad 1 de esta obra). Según se planteó, luego de acceder al disco para leer un sector solicitado, y sin que se mueva el cabezal, se van leyendo los siguientes sectores de la pista o cilindro (pues es probable que luego se solicite su lectura), los cuales pasan al cache de disco, constituido por una memoria DRAM manejada por el microcontrolador.

<sup>1</sup> Estos 7 registros "ports" de 8 bits (con direcciones 01F0 a 01F6) constituyen el denominado "AT task file" (archivo de tareas AT). Un octavo registro de dirección 01F7 es el "port de datos", de 16 bits, para lectura o escritura.

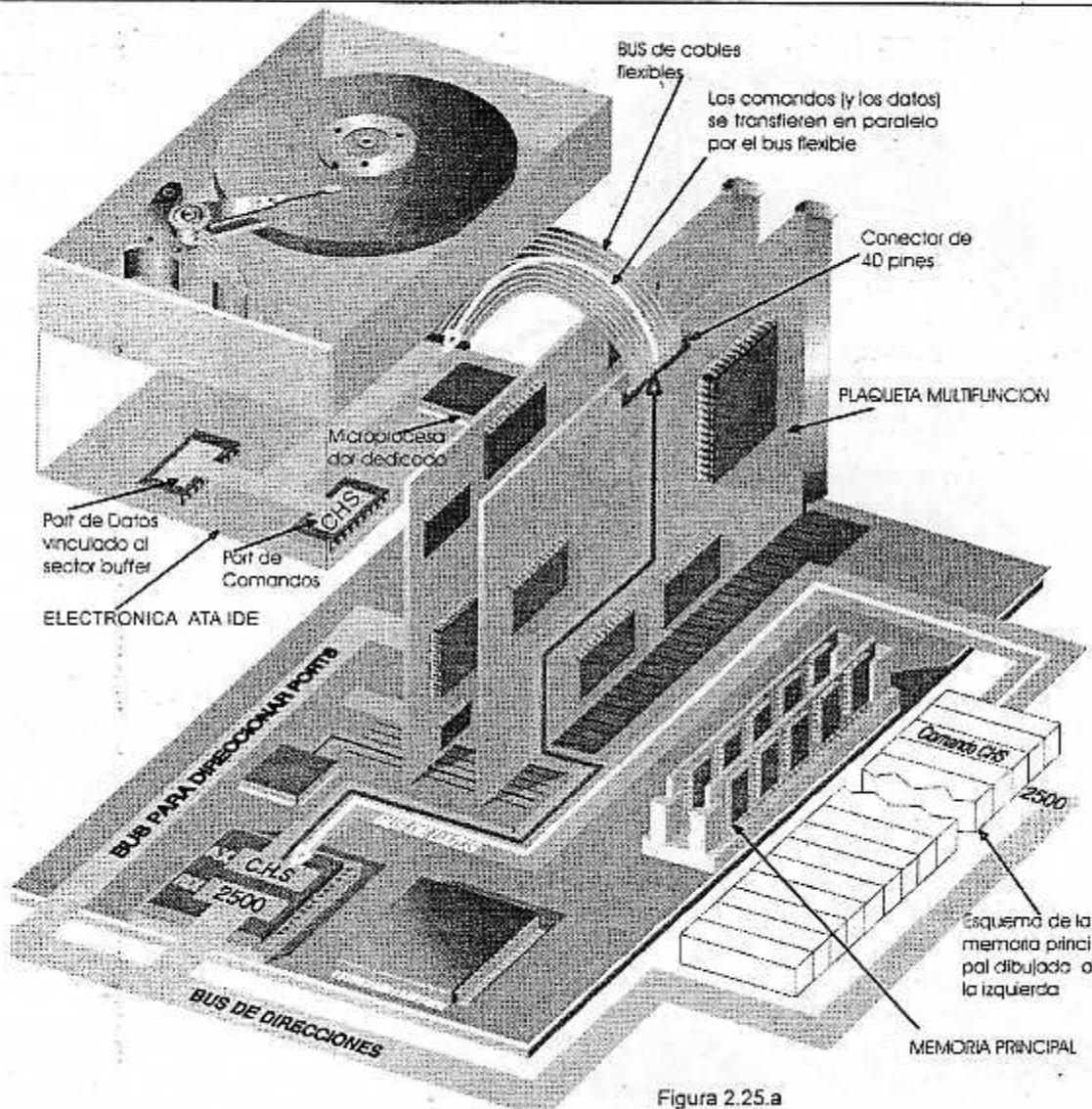


Figura 2.25.a

Si se ordena escribir un sector, deben llegar desde memoria y a través del bus PCI al "sector buffer" 512 bytes para ser escritos, a través del port de datos citado. En caso que se envíen datos para ser escritos en sectores sucesivos, los mismos pueden guardarse transitoriamente en el caché citado.

Una unidad IDE realiza funciones de interfaz (figuras 2.25), la cual se conecta a las líneas de datos, direcciones e IRQ del bus PCI mediante un cable plano terminado en un conector con agujeros para 40 terminales, para conectarse a igual número de "agujas" ("pines") que llegan a circuitos vinculados al "puente" con bus PCI (fig 1.80 citada). Dichas "agujas" van a un conector en la "mother" y anteriormente iban a la placa "multifunción" que también contenía la controladora de disquetera, citada al tratar ésta.

Mas en detalle (figura 2.25.a), a la electrónica IDE le llegan comandos, que ordenan leer o escribir un sector, del cual se indican sus números de CHS. Merced a la ejecución de subrutinas del BIOS<sup>1</sup> estos comandos que estaban en memoria principal, pasan al registro AX de la UCP, y de éste a los registros "ports de comandos" de la interfaz IDE, a través del bus de datos que llega a ésta.

La electrónica IDE, después de recibir estos comandos realiza las siguientes acciones (figura 2.25.b):

- Traduce dichos comandos en señales para que el cabezal se posicione en el cilindro elegido; y que luego la pista correspondiente a la cabeza seleccionada sea leída por ésta hasta encontrar el sector buscado.
- La cabeza lee el número de cada sector que encuentra en la pista que va leyendo, el cual es transmitido a la electrónica IDE, para determinar si es o no el comienzo del sector buscado, a fin de escribir o leer –según sea la orden- los datos en la zona correspondiente del sector buscado

<sup>1</sup> En definitiva estas directivas son enviadas por el sistema operativo, dado que la Rom Bios puede considerarse parte del mismo

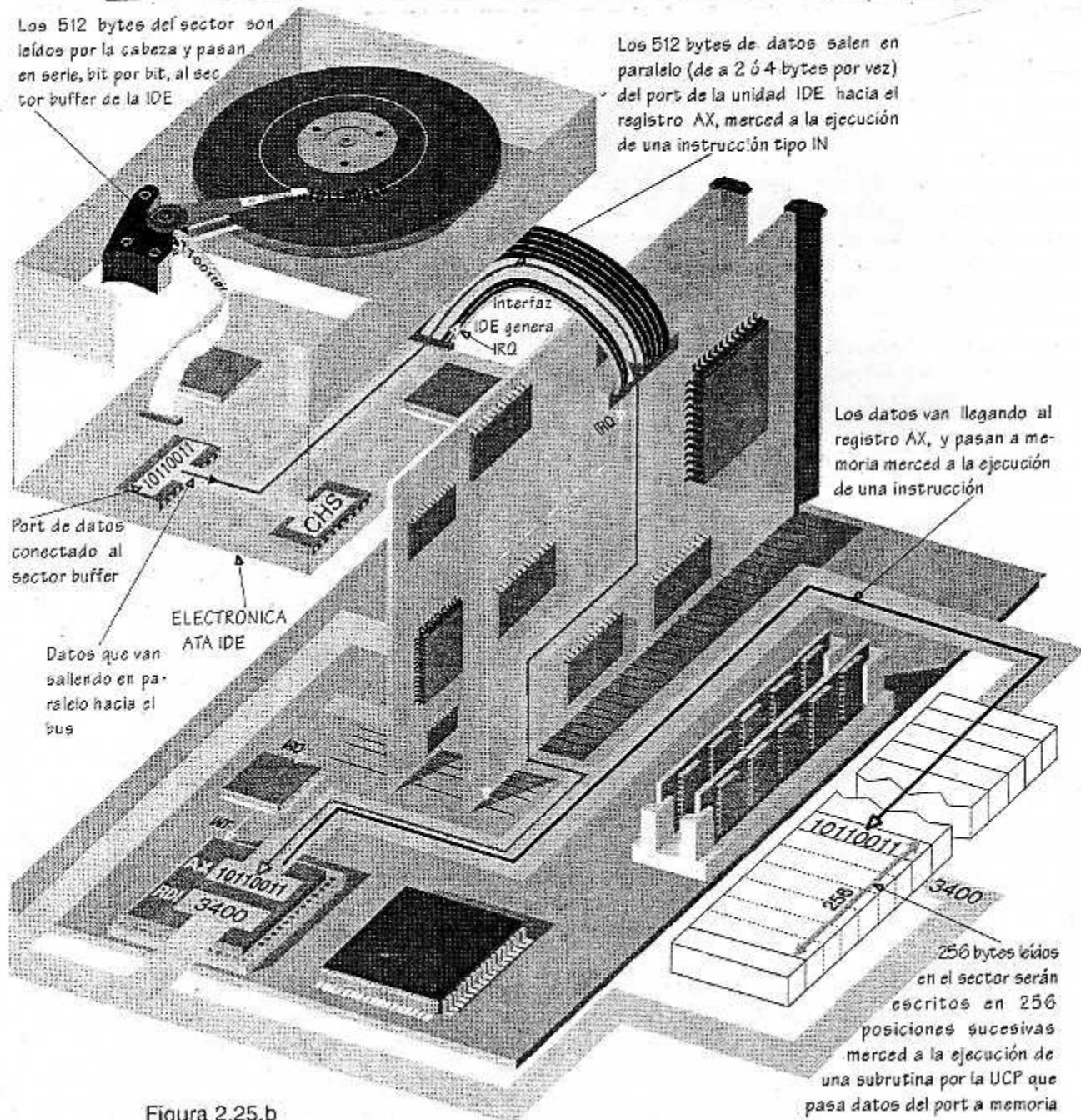


Figura 2.25.b

- Si es una orden de *lectura*, todos los bits del sector son leídos en serie por la cabeza. A medida que son leídos se realiza la verificación ECC (semejante a la CRC) y pasan al sector buffer de la electrónica, para ser corregidos de ser necesario.
- En caso de que dicha lectura sea correcta, en modo PIO una forma consiste en que la sección interfaz de la electrónica IDE active su línea IRQ, para que la UCP interrumpa el programa en ejecución, y ejecute una subrutina del BIOS para hacer sucesivos AIM a través del bus PCI, de modo de ir sacando del sector buffer los datos hacia AX y luego pasarlo a memoria. En caso de hacerse ADM, serán necesarios sucesivos ADM para llevar los 512 B a memoria.
- De manera inversa, luego de una orden de *escritura*, los bytes a escribir en el sector van llegando (de a 2 ó 4) por el bus (desde la memoria) al port de datos, y de éste al sector buffer, luego de lo cual se activa la línea IRQ<sup>1</sup> si se quiere operar en PIO. Luego los bits pasan en serie hacia la cabeza, para que los escriba en el sector.

<sup>1</sup> Si este buffer está deshabilitado, IRQ se activa recién cuando los datos fueron escritos en el disco rígido.

- A medida que escribe o lee los bits de un sector, realiza el cálculo del ECC (figura 2.24), cuyo valor graba a continuación de la zona de datos en una escritura.
- Realiza en una lectura o escritura, el manejo y control de errores, codificando en un registro port el tipo de error ocurrido.

En definitiva, subrutinas del BIOS al enviar comandos a los ports de la interfaz IDE (o SCSI) dan origen a lecturas y escrituras en el disco, siendo que las velocidades máximas de transferencia (externa) están determinados por los circuitos de la interfaz, y por los buses intervenientes.

El tiempo denominado "*I/O read and write cycle time*", es determinante de la velocidad de transferencia, siendo el mínimo lapso que puede mediar entre dos escrituras o lecturas sucesivas del registro port de datos de la interfaz ATA (IDE) de una unidad de disco rígido; registro que tiene un tamaño de 16/32 bits = 2/4 B. Por ejemplo, de los 5 modos PIO (AIM), en el modo 0 que es el más lento, dicho tiempo de ciclo es de 600 nanoseg. Conocido este tiempo, puede deducirse que la máxima velocidad de transferencia para este modo es de 3,3 MB/seg., como se indica a continuación.

Suponiendo que teóricamente en forma ininterrumpida se lean o escriben 2 bytes del port de datos cada 600 nseg = 0,0000006 seg., en un segundo podrían hacerse 1/0,0000006 transferencias de 2 bytes por AIM.

O sea podrían transferirse 2 bytes cada 1/0,0000006 seg. = 3.333.333,33 bytes/seg. = 3,3 MB/seg., dado que 1 MB = 1.048.576 bytes. Lo anterior equivale a decir que en un segundo se transferirían:

$3,3 \times 2048$  sectores = 6758 sectores, siendo que 2048 sectores de 512 bytes = 0,5 KB conforman 1 MB.

En el presente existen unidades IDE estándar ATA-2, con PIO modo 4, tiempo de ciclo de 120 nseg, lo cual implica una velocidad de transferencia máxima teórica de 16,5 MB/seg.<sup>1</sup> (5 veces mayor que el modo 0).

Cabe mencionar que esta velocidad de transferencia máxima para PIO es la misma que para ADM modo 3, dado que depende del tiempo de ciclo con que opera la unidad IDE. Si en lectura o escritura deben mediar 120 nseg entre dos direccionamientos al port de datos de 16 bits, este tiempo de ciclo debe respetarse, ya sea que los direccionamientos se hagan para efectuar transferencias por AIM (PIO) o ADM (DMA).

Que la transferencia entre memoria y dicho port (o viceversa) convenga hacerla por AIM o ADM, dependerá del sistema operativo en uso, y si se trabaja o no en "multitasking".

Las unidades con electrónica no normalizadas designadas E-IDE (enhanced IDE) ó Fast ATA (1 ó 2) permiten además comandos para *escritura o lectura múltiple*, los que dan lugar al "Block Mode".

Envíando a los ports correspondientes uno de estos comandos y la cantidad de sectores a transferir (de 2 a 128) se evita que la electrónica active la línea de interrupción IRQ, con cada sector que debe ser pasado del sector buffer a memoria o en sentido inverso. Con esto se evitan las pérdidas de tiempo involucradas en cada interrupción (guardar registros de la UCP en la pila, llamar y ejecutar una subrutina, y volver a restaurar dichos registros en la UCP). Así, hasta 128 sectores pueden ser transferidos con un solo comando, con lo cual es factible ganar un máximo de 30% de tiempo.

Es factible conectar más de una unidad IDE a un bus de una PC (sean dos discos rígidos, un rígido y una lectora de CD, etc.), debiendo actuar el más rápido de ellos como "master", y el otro como "slave". Esto se define conectando los "jumpers" (puentes de contacto) como indica el manual de instalación.

Direccionando y escribiendo el valor (1 ó 0) de un bit del registro drive/head, se selecciona si un comando es para el "master" o el "slave".

En los modos Ultra DMA (UDMA-66) o Ultra ATA (UATA) se transfieren 16 bits en las dos transiciones de la señal reloj: cuando ésta va de 0 a 5 volts (por ejemplo en la mitad del ciclo), y cuando pasa de 5 a 0 volts (al final del ciclo). Así para un tiempo de ciclo reloj dado se duplica la velocidad de transmisión.

Dadas las elevadas velocidades de transmisión (33 ó 66 MB/seg), para detectar errores se hizo necesario enviar luego de los datos el código CRC de los mismos (de la forma vista en el formateo de discos) hacia el extremo receptor. Este calcula el CRC de los bits enviados, y si no es igual al recibido, los datos transmitidos a través del cable IDE son erróneos.

En UDMA se requiere un cable IDE de 80 conductores: 40 para señales y 40 para masas. Además la IDE, la interfaz controladora IDE/ATA y el BIOS y/o el sistema operativo deben soportar UDMA.

Esto es general: dado que una interfaz IDE/ATA hace de intermediario entre la unidad de disco por un lado y por otro con el chipset de la "mother" y el BIOS, se requiere que en ambos extremos exista soporte para el modo de transferencia o protocolo que se quiera usar.

<sup>1</sup> Estas velocidades con Fast ATA deben configurarse en la tabla de opciones del BIOS CMOS, o usar unidades de disco ya preparadas. "Mothers" con nuevas versiones del BIOS realizan esta configuración en forma automática.

Los PIO modos 3 y 4 tienen el aditamiento IORDY, para indicar que la unidad IDE puede retardar la velocidad de transferencia, desactivando temporalmente la línea IORDY del bus, a fin de no abortar transferencias rápidas cuando existen problemas en el sistema.

## ¿Qué es el DMA bus mastering o "first party" DMA y cómo se lleva a cabo?

El bus PCI (figura 1.80 de la Unidad 1 de esta obra) permite que un dispositivo conectado al mismo puede hacerse "master" del bus, y realizar ADM sin emplear el Controlador de ADM descripto en el disquete.

Un dispositivo que debe leer o escribir la memoria solicita ser master del bus. (A su vez puede ser slave de la UCP actuando como master cuando ésta ejecuta una subrutina para inicializar ports de dicho dispositivo).

De esta forma la electrónica de un periférico, si está preparado para ello (como el controlador IDE, SCSI, tarjetas de video y sonido, tarjetas de red, canales de ADM, etc), puede acceder a memoria igual que la UCP si está preparada para ello. Sólo puede haber un master por vez.

Un controlador IDE con **bus mastering DMA** (como el UDMA) permite transferir directamente ("first party") por ADM en forma más rápida que PIO o ADM clásico, que requieren ("third party") respectivamente de la UCP y de un controlador de ADM. Mediante "third party" se requiere al menos un ciclo de bus PCI para leer y otro para escribir. Para bus mastering, la electrónica IDE incorpora al anterior bloque de registros ports (para comandos, datos y status) nuevos ports para su "controladora" que se definen más abajo. En la "mother" el chip set debe soportar esta tecnología, al igual que el sistema operativo, el cual debe presentar un "driver" (subrutina de manejo de periférico) apropiado.

### Preparación:

- a) El software prepara en memoria el Physical Region Descriptor (PRD) con la dirección donde comienza la zona de memoria hacia (desde) donde se transferirán los datos desde (hacia) el disco, y la cantidad (bloque) de bytes a transferir (hasta 64 KB).
- b) El software mediante las instrucciones MOV y OUT escribe ports del controlador IDE (cuya dirección varía según el dispositivo sea primario o secundario): en el registro PRD la dirección donde dejó la PRD (que el controlador IDE deberá traer) y en el de Comandos la orden de lectura o escritura así como la orden de que la IDE sea bus master; en el port de Status habilitará la activación de la IRQn, y pondrá en cero el bit de error. También a los otros ports debe llegar la dirección del primer sector a acceder y la cantidad de sectores sucesivos a los que se accederá luego del mismo.

**Sincronización:** cuando la unidad está lista para recibir o enviar los datos avisa con DRQ al controlador. Cuando se lee el disco los datos deben ser guardados transitoriamente antes de ser transferidos.

**Transferencia:** con el DRQ el controlador solicita al árbitro del bus PCI ser "master" para la transferencia por sucesivos ADM del bloque de datos hacia o desde la memoria.

**Escritura:** si se trata de una escritura la información que llega a un buffer de datos es escrita en el disco.

**Verificación:** al final de la transferencia el controlador activa la línea de interrupción IRQn para que una subrutina (driver) verifique que la misma se ha realizado correctamente. Para ello lee los ports de status del controlador y de la unidad del disco para determinar si hubo algún error.

Si se detecta que la transferencia no se completó, en el registro Sector Count de la IDE se determina cuántos sectores faltan acceder, se genera una nueva PRD y se completa la transferencia.

## ¿Qué es la ATA Packet Interface (ATAPI)?

Para los discos ópticos y almacenamientos removibles que sean "IDE devices" se desarrolló un protocolo específico más complejo que el ATA, denominado ATAPI. "Packet interface" se refiere a que los comandos para estos dispositivos se envían en grupos designados "packets". Los actuales sistemas operativos presentan software (drivers) para comunicarse con los dispositivos ATAPI, y el BIOS permite bootear desde ellos.

## ¿Qué es la interfaz Serial ATA (S-ATA)?

Al tratar los buses (sección 1.13 de la Unidad 1) se vió que un bus paralelo podía transmitir 8, 16, 32, ... bits simultáneamente, pero debido al acoplamiento electromagnético de las señales eléctricas que por él viajaban, su longitud se limitaba más a medida que se quería transmitir más Bytes/seg. En general un bus de este tipo como el PCI, es compartido por varios dispositivos de distinto tipo identificables por su dirección, lo cual lo hace económico. Pero debe existir una circuitería extra para acceder al bus, para que no puedan comunicarse más de dos dispositivos por vez, y un protocolo de señales que implica tiempos de direccionado, de aceptación, etc.

Las exigencias de multimedia (gráficos en 3D, vídeo, etc.) y Gigabit Ethernet, entre otras, requieren cada vez más ancho de banda (Bytes/seg) para enviar desde dispositivos hacia memoria y UCP, por lo que el bus PCI se está convirtiendo en un cuello de botella. Asimismo buses de mayor cantidad de líneas requieren chips de gran cantidad de pines para comunicarse entre sí, lo cual aumenta la radiación electromagnética y la disipación.

Para menores anchos de banda ya se está dando el reemplazo del bus paralelo ISA por el USB (Unidad 1).

Los fabricantes de procesadores, chip sets y periféricos incorporan nuevas tecnologías de transmisión buscando aumentar el ancho de banda para enviar información desde/hacia periféricos (inclusive entre procesadores) disminuyendo

los tiempos de respuesta, el número de buses y la disipación. Como en el USB, se transmiten paquetes de bits en serie por dos líneas: en cada instante el receptor sensa la diferencia de tensión entre esas dos líneas (**tensión diferencial**). Así se compensan los ruidos electromagnéticos que pudieran generarse: cuando ocurre un ruido éste afecta por igual a las dos líneas, o sea que ambas durante cortos lapsos pueden subir o bajar igual valor de tensión en relación a masa, pero estas variaciones no modifican el valor de la diferencia neta de tensión entre los 2 conductores. Conforme a esta tendencia, S-ATA transmite por dos líneas, en forma diferencial, pulsos de tensión de tan solo 500 mV pico a pico (contra 5 volts de Parallel ATA). Esta menor excursión de las señales puestas en juego permite pasar rápidamente de un nivel a otro de la señal, y así transmitir más pulsos por segundo. De esta manera usando transmisión de paquetes en serie mediante un cable con 2 pares de conductores (que puede tener hasta un metro de longitud para conectar dispositivos IDE fuera del gabinete) se pueden transmitir hoy hasta 150 MB/sec (con proyecciones a 300 y 600) en lugar del cableado de 80 conductores de Ultra ATA.

### ¿Qué nuevas funciones soportan las unidades de disco SCSI para servers?

En el presente para servidores existen discos SCSI con "cola de múltiples comandos optimizada". Pueden recibir múltiples comandos de lectura o escritura, operar con cierto grado de paralelismo interno auxiliándose con su caché interno, y decidir en qué orden llevarlos a cabo. Para ello se tiene en cuenta su orden de llegada, y la optimización de los tiempos seek y latencia. En cada instante, en función del cilindro donde se encuentra el cabezal decide qué comando atenderá a fin de que el próximo cilindro sea el más cercano. Asimismo tiene en cuenta el ángulo de rotación, pues puede ser que convenga otro cilindro más alejado en el cual el sector se encuentre más rápidamente.

### ¿Cómo es la numeración lógica de sectores por LBA?

No hace mucho, el disco más grande que podía manejar una PC con interfaz IDE era de 500 MB. Correspondía a un disco de 1024 cilindros, 16 cabezas (heads) y 63 sectores de 512B = 0,5 KB, con lo cual la capacidad era exactamente de  $1024 \times 16 \times 63 \times 0,5 = 504$  MB = 528 millones de bytes (si se considera 1 MB = 1.000.000 bytes)

Este límite se debe en principio, a que por un lado, cuando la subrutina del BIOS debe enviar al drive IDE los números de cilindro, cabeza (head) y sector, abreviados en inglés CHS, para los mismos tiene establecidos 10, 8 y 6 bits, respectivamente, número de bits que también están reservados en la Tabla de Particiones (MBR). Por lo tanto, para el BIOS y la Tabla de Particiones, los números máximos que se pueden formar son:

$2^{10} = 1024$  cilindros;  $2^8 = 256$  cabezas;  $2^6 = 64$  sectores, que son 63, pues el nro 0 de sector no se usa.

A su vez, un drive IDE o EIDE para CHS está limitado a 16, 4 y 6 bits respectivamente. Resulta así, que los números máximos que puede manejar son:

$2^{16} = 65536$  cilindros;  $2^4 = 16$  cabezas;  $2^6 = 64$  sectores, que van hasta el número 63

Compatibilizando ambas limitaciones, resultan 1024 cilindros, 16 cabezas y 63 sectores, que hacen el límite de los 504 MB calculados; aunque El BIOS por separado permitiría  $1024 \times 256 \times 63 \times 0,5 = 8$  GB, mientras que por su parte, una interfaz IDE permitía una capacidad de hasta  $65536 \times 16 \times 63 \times 0,5 = 128$  GB.

En 1994 las normas IDE de 1984 pasaron a ser las E-IDE. A fin de maximizar la compatibilización entre BIOS e IDE para poder operar con discos de mayor capacidad que 504 MB, se crearon algunos artificios matemáticos que pueden ser llevados a cabo por la UCP ejecutando subrutinas de un BIOS actualizado, o por el microprocesador de la unidad IDE ejecutando subrutinas de su firmware.

Una forma de compatibilizar discos de más de 504 MB era la siguiente. Sea un disco de 1 GB que físicamente presenta 2 platos (4 cabezas), y cuyas pistas tienen un número distinto de sectores, que aumenta desde el centro hacia el borde, superando los 63, como se trató más atrás. Este disco para la unidad IDE se caracterizaría físicamente como equivalente a un disco de 4096 cilindros, 4 cabezas, y 63 sectores de 0,5 KB por pista, dado que  $8192 \times 4 \times 63 \times 0,5 = 1$  GB. La geometría del disco real resulta así invisible al exterior.

Si se divide 8192 por N=8 resulta 1024, y si se multiplica 4 por N=8 resulta 32. De este modo, un disco que tuviera 1024 cilindros, 16 cabezas y 63 sectores por pista también tendría  $1024 \times 32 \times 63 \times 0,5 = 1$  GB, pero los valores 1024 y 32 son compatibles para subrutinas del BIOS, según se describió.

La unidad IDE se encarga de trasladar cada número de sector, cabeza y sector del disco lógico que suponen las subrutinas del BIOS, en otro número de sector, cabeza y sector para el disco físico de  $8192 \times 4 \times 63 \times 0,5$

También es factible que dicha traslación sea llevada a cabo por dichas subrutinas del BIOS actualizado. Esto se conoce como ECHS (Extended CHS translation) o "large". Existen varias formas de realizar esto.

No debe confundirse los números lógicos de cilindro, pista y sector –que simula el drive de un disco para un sistema operativo (y para la ROM BIOS)– con el formateo lógico, destinado a reservar sectores que serán usados por dicho sistema, ni con la estructura lógica con que el DOS "ve" a un disco (antes tratada).

LBA es el mecanismo actual más usado para operar discos con más de 504 MB. También se conoce como BIOS 13 extensions. Las unidades de disco SCSI y las IDE permiten que se identifique los sectores mediante números consecutivos, comenzando desde 0 (donde está el MBR), siendo que físicamente se tiene números de cilindro, cabeza y sector. Esto es, esas subrutinas en lugar de enviar números de cilindro, cabeza y sector para acceder a un sector, pueden enviar un número que lo identifica denominado LBA (Logic Block Address o sea dirección lógica del bloque).

Una unidad ATA/IDE con la opción de operar con CHS o LBA, requiere asignar LBA 0 al cilindro 0, cabeza 0, sector 1. En un disco de 504 MB el LBA del último sector sería 1.065.456.

La técnica LBA de 28 bits permite acceder a unos  $2^{28}$  sectores (de 0,5 KB), o sea 137 GB en un IDE.

El uso de LBA puede implementarse en un BIOS actualizado o en una unidad IDE<sup>1</sup>, siendo que hoy día operan así por default. No siempre significa una mejor performance de un disco. Muchos sistemas operativos pueden operar directamente con LBA, pero el DOS y otros sistemas deben usar la geometría CHS.

Pero dado que BIOS INT 13 estándar para LBA sólo permite 24 bits apareció otra barrera para manejar discos grandes (de las 10 que acá no se mencionan), pues  $2^{24} \times 512B = 8,4$  GB. El BIOS extendido permite LBA de 64 bits, con lo cual el nuevo límite es  $2^{64} \times 512B = 9,4$  Tera Gigabytes!. A su vez ATA-6 define LBA de 48, lo que permite  $2^{48} \times 512B = 144$  Petabytes (PB).

## ¿Qué fases podemos distinguir en la lectura o escritura de un sector en un disco de una unidad ATA-IDE usando el método PIO?

En principio, en cualquier computador, para que tenga lugar una operación de escritura o lectura de algún medio (disco, cinta, papel), es necesario que durante la ejecución de un programa se ejecute una instrucción que llama a la subrutina del SO, o de la ROM BIOS, preparada para tal fin. Podemos considerar al BIOS formando parte del SO, siendo que éste se encarga del manejo de los periféricos (Apéndice 2 de la U1)<sup>2</sup>.

Por ejemplo, si en una pantalla Windows indicamos con el cursor del mouse que queremos abrir o guardar un archivo en un disco o disquete, se pasará a ejecutar una secuencia de instrucciones. En ésta existirá una, que expresada en assembler será del tipo INT 13<sup>3</sup>, para llamar al SO, si la UCP de la PC es un 80x86 (o un clon de Intel). Lo mismo si mediante el teclado ordenamos la apertura o cierre de archivo.

Igualmente, si en un lenguaje de alto nivel se tiene una sentencia como READ, WRITE, GET, PUT u otra que ordena escribir o leer un archivo, el programa compilador la traduce a una secuencia de instrucciones de máquina, entre las cuales habrá una que llama al SO. Esta será del tipo INT 13, si el compilador de Microsoft genera código de máquina a ejecutar por un procesador 80x86 o Pentium<sup>4</sup>.

Trataremos aspectos centrales de un proceso de escritura de un archivo en un sector en un disco por PIO, que dividiremos en las siguientes fases:

Preparación => Sincronización => Transferencia => Escritura

Se tratará la forma PIO ("Programmed I/O"), o sea la realización por AIM (memoria => AX => port datos), de la fase 3 de transferencia, en una E/S; en este caso en la escritura de un sector de un disco de una unidad IDE. Más arriba dimos cuenta de las fases de dicho proceso para la metodología DMA Bus Mastering. Conforme a lo anterior, para que este proceso tenga lugar, en un programa en ejecución debe existir una instrucción de llamada al SO, que en assembler es INT 13 si el procesador es de Intel y el SO es de Microsoft. Entonces, las fases en cuestión serán:

<sup>1</sup> Esto puede decidirse por software, direccionando y poniendo en 1 el bit "L" del registro port Drive/Head (citado en respuesta anterior), para lo cual se le debe enviar un comando mediante una instrucción tipo OUT.

<sup>2</sup> De un modo más general: las operaciones de E/S son preparadas, activadas y controladas por subrutinas del SO.

<sup>3</sup> El número nn y valores que previamente se cargan en registros de la UCP determinan cual subrutina será llamada (ver Unidad 3).

<sup>4</sup> Para otros procesadores, esta instrucción de llamada al SO tiene otro nombre: SVC, Call, Trap, etc.

- Fase de preparación:** cuando la UCP ejecuta el código de máquina de INT 13<sup>1</sup> comienza esta fase, primera de las cinco fases en que hemos dividido la escritura de un sector. Dicho código ordena que la UCP pase a ejecutar una subrutina del SO cuyo cometido es llevar a cabo la presente fase.

Algunas de las numerosas acciones preparatorias, que realizará esta subrutina del SO y otras subrutinas del sistema que ella a su vez ordena ejecutar, serán:

- división del archivo a grabar en "clusters". El tamaño de éstos depende del tamaño de la partición.
- lectura del directorio raíz de la partición, o de un subdirectorio (para conocer si el nombre del archivo no existía antes), así como la lectura de la copia de la FAT en memoria (para determinar el número del primero y los siguientes "clusters" asignados al archivo);
- escritura en el subdirectorio del nombre asignado al archivo, y en la FAT, los "clusters" que ocupa;
- conversión de cada número de "cluster" en los números de SL que lo componen, y éstos en los correspondientes sectores físicos, identificados por su LBA o CHS en el disco rígido.

Una vez realizadas las acciones anteriores, la ejecución de instrucciones OUT, que también forman parte de subrutinas del BIOS permite enviar –a través de líneas de datos del bus PCI (figura 1.80 de la Unidad 1) – comandos desde el registro AX de la UCP hacia registros ports de la electrónica IDE (figura 2.27 a). Entre ellos estará el LBA o CHS del primer sector a acceder.

Suponiendo que los platos de la unidad estén girando, dichos comandos son cumplimentados en orden por el microcontrolador de la unidad IDE –merced a la ejecución de subrutinas que están en la ROM de la IDE.

- Sincronización:** transcurre desde que el LBA ó CHS se escribió (mediante MOV y OUT) en el port de control, hasta que la cabeza indicada detecte el principio del sector seleccionado. Como se vio, cuando esto ocurre en ADM, la IDE activa la línea DRQ para que el hardware correspondiente lleve a cabo la fase de transferencia. En PIO esta fase que es por AIM debe realizarse por software, mediante la ejecución de instrucciones del tipo MOV (memoria => AX) y OUT (AX => port datos). El momento en que la cabeza llegó al sector, y que por lo tanto la transferencia debe comenzar, puede indicarse por la unidad IDE de dos maneras:

- Cambiando en su port de status el valor del bit R ("Ready"). La subrutina que realiza los AIM, previamente debe dirigir este port para leer el valor de este bit. Durante tseek y tlat (10 msesg) la UCP debe ejecutar una secuencia repetida de instrucciones (que sigue a la secuencia que realiza la fase 1) que lee dicho port para detectar el momento en que el bit R cambia. Entonces otra subrutina efectuará la transferencia.
- Activando el cable de solicitud de interrupción IRQn (típicamente n=14) para que se pase a ejecutar una subrutina que lleve a cabo la transferencia. A fin de no interrumpir por cada 512 bytes transferidos se envía un bloque de bytes (como ser del tamaño de un cluster), y la memoria se lee por ráfagas.

- Fase de transferencia (por AIM)<sup>2</sup>** de los bytes a escribir desde memoria al registro AX de la UCP, y de éste al port de datos de la electrónica IDE, con destino al sector buffer de ésta (figura 2.27 b).

Para tal fin, si el bus PCI tiene 16 líneas de datos, la UCP ejecutará una subrutina que realizará 256 AIM de dos bytes cada uno, para transferir los 512 bytes de memoria al port de datos de la IDE, desde donde pasarán al sector buffer.

- Escritura del sector (a cargo del microcontrolador de la IDE):** los bytes que están en el sector buffer son convertidos a bits en serie. Estos bits son considerados en el cálculo que el microcontrolador realiza para determinar el ECC, y son enviados con codificación RLL a la cabeza que los grabará en el sector accedido.

Luego de escribir la zona de datos del sector, el microcontrolador enviará los bits que la cabeza deberá escribir en la zona del ECC (figura 2.24).

Al finalizar, la electrónica IDE registrará en su port de status si la operación de escritura ordenada fue exitosa, o si debe codificar algún tipo de error sucedido que su inteligencia no pudo subsanar.

<sup>1</sup> Esto varía un poco para una interfaz E IDE.

<sup>2</sup> Existen unidades IDE con la opción para hacer ADM, de manera semejante a la descripta para disqueteras. En sistemas para "multitasking" puede ser conveniente que en la fase de transferencia, en vez de que la UCP ejecute una subrutina para hacer sucesivos AIM, en su lugar ella ejecute otro programa. La controladora de ADM se encarga de realizar por hardware la transferencia, al mismo tiempo que la UCP ejecuta el otro programa. Luego, se debe ejecutar una subrutina para verificar que el ADM se hizo correctamente.

	INT 13	Llama a subrutina del SO Tra instr de fase preparación	Para las fases 1, 2.a) y 3 que se realizan por software, se indican las instrucciones más significativas de cada una, suponiendo conocimientos básicos de Assembler (Unidad 3 de esta obra, y el salto ya se plantea en la Unidad 1).
Fase de preparación	MOV AX, [mem] OUT Port Control	Lleva de mem a AX el CHS Lleva CHS de AX al P Control Última instrucción fase preparación	Es importante notar que en este proceso (usado en las PC personales) la UCP está ocupada en todas las fases, siendo que la más larga es la de sincronismo (9 mseg) en la cual se repiten millones de veces las tres instrucciones indicadas hasta que el bit R cambie. La fase de transferencia es muy corta en relación con la de sincronismo. El ciclo repetitivo durante el sincronismo sirve para coordinar, en el inicio de la
Fase de sincronización	Otro IN Port Status AND, máscara	Lleva copia P. Status a AX AX "and" máscara para determinar el valor del bit R	
Fase de transferencia	JZ Otro MOV AX,[datos]	Volver a Otro si R no cambia R cambió, por lo que van 2 bytes (de los 512) de mem a AX	
	OUT Port Datos	Van 2 bytes de AX al P. Datos	(Estas 2 últimas instrucciones se repiten 256 veces)

transferencia, el hardware con el software.

A diferencia, si el proceso se hace por ADM, la UCP sólo estará ocupada en el mismo durante la preparación y verificación, por lo que en las fases 2 y 3 la UCP puede ejecutar una sucesión de programas (multitasking) que no tienen nada que ver con la escritura o lectura del disco. Así se opera necesariamente en un servidor.

Si en PIO el momento en que el sector llega a la cabeza se detecta por una solicitud de interrupción, durante dicha fase la UCP puede hacer multitasking.

En caso que en la etapa de preparación se ordene la lectura de un sector, por tratarse de una operación de entrada, el sentido del movimiento de los datos es del sector buffer hacia memoria, por lo que debe invertirse el orden de las fases 3 y 4 descriptas para la escritura de un sector.

Esto es, ahora en la fase 3 –de lectura de la zona de datos del sector accedido– a medida que la cabeza seleccionada va leyendo esta zona del sector, envía hacia la electrónica IDE bits en serie, codificados en RLL. Esta información, a medida que llega es decodificada, para determinar los bits de datos que contiene, los cuales también son considerados en el cálculo que el microcontrolador realiza para determinar el ECC.

Estos bits son convertidos en grupos de 2 bytes con destino al sector buffer, de modo que en la fase 4 –ahora fase de transferencia por sucesivos AIM– salgan en paralelo desde el port de datos hacia el registro AX, y de éste al buffer de memoria principal.

Se supone que en las fases de sincronización y trasferencia se realizan las verificaciones necesarias de la correcta marcha del proceso leyendo el port de status de la interfaz asociada a la electrónica IDE.

## ¿Cómo vuelven más veloz a un disco los cachés en hardware o software?

El concepto de caché es en esencia el mismo para disco que para memoria principal, tratado en 1.@@: se trata de una memoria dedicada a almacenar los datos que probablemente se requerirán próximamente. Esto último se lleva a cabo automáticamente, siendo manejado por el “controlador de caché”.

Un caché de disco simula un disco de igual capacidad, pero más veloz que el real.

En general, el caché de un disco (500 KB - 2 MB, o más) mejora su performance, disminuyendo el número de accesos a cilindros para el caso de lecturas sucesivas, y permite que los datos fluyan sin interrupción desde disco hacia él en caso de que el bus PCI o SCSI esté ocupado. Cada vez que se ordena leer un sector, primero se determina electrónicamente si está en el caché. De ser así no se accede al disco; caso contrario debe accederse y una copia de los sectores leídos además de transmitirse pasan al caché por si vuelven a ser solicitados, conforme opera un caché (Unidad 1).

Cuando se escribe en un disco un archivo, se trata de grabar todos los sectores que lo componen en un mismo cilindro. Así se pierde un solo tiempo de posicionamiento y latencia para encontrar el primer sector del archivo. Luego serán escritos los siguientes sectores que entran en dicho cilindro, sin mover el cabezal y sin tiempo de latencia. Sólo existirá para cada sector a escribir el tiempo de transferencia (figura 2.10.c).

Este análisis de tiempos vale también para la lectura del archivo, para lo cual se debe repetir el mismo orden secuencial seguido al escribir los sectores que lo constituyen.

La estadística indica que si se accede a un sector que compone un archivo, para ser escrito o leído, es altamente probable que luego se deba acceder a los otros sectores que le siguen en la pista donde está dicho sector, para luego continuar con los sectores de las restantes pistas del mismo cilindro.

Por lo tanto, puede resultar conveniente luego de leer un sector de una pista, *leer de seguido* –sin movimiento alguno del cabezal, ni tiempo de latencia– *los restantes sectores de la misma*, e ir pasando automáticamente todos los sectores así leídos a una memoria tipo RAM que oficia de caché, ubicada en la electrónica IDE o SCSI (figura 2.25). Así, una pista completa pasa al caché y de este cada sector a memoria, por AIM o ADM, según sea. El tamaño del caché es típicamente de 32 KB hasta 1 MB.

Entonces, solamente se accede a la memoria de disco para encontrar el primer sector del archivo, éste y los restantes sectores se localizan y leen en el caché RAM de disco citado –de tiempo de acceso despreciable en comparación con el de un disco– cuando se ordena su lectura. En ese momento, cada sector leído en el caché se transfiere hacia un registro de la controladora, y de éste al buffer asignado a ese sector en memoria principal. Como se trató en el tema del “interleave”, este caché permite que los datos de sectores consecutivos de una pista pasen al mismo a la velocidad de transferencia interna del disco, y luego desde el caché puedan ser transferidos hacia memoria a la velocidad del bus, obteniéndose en definitiva una importante ganancia de velocidad en relación con la no-existencia de un caché. Así, la transferencia desde el caché de una interfaz IDE/ATA de 8 sectores consecutivos del tamaño de un cluster puede ser 100 veces más rápida que si los mismos fueran accedidos en el disco.

*Con el caché de disco se simula una serie de operaciones de lectura en un disco, siendo que en realidad se realizan en una RAM con la rapidez de esta memoria electrónica<sup>1</sup>.*

*El caché de disco permite mejoras en velocidad entre el 30 y el 400%, dependiendo de la aplicación, interfaz y bus usados. De este modo resulta una gran independencia del tiempo de acceso a un disco.*

Análogamente, en una escritura del disco, se pasa desde memoria principal al caché de disco (a través de un registro de la controladora), todos los datos del archivo a grabar, pudiéndose realizar la escritura de cada sector del archivo en el disco cuando éste no es requerido por un programa. Es factible diferenciar cachés de disco para lectura y para escritura.

El caché de disco que está en la RAM de la electrónica IDE o SCSI se denomina “caché de disco incluido” o “interno” o “caché de disco en hardware”. Su tamaño oscila entre 32 KB y 1 MB ó más.

También se emplea el “caché de disco por software” en memoria principal RAM (“software caché”), manejado por un programa (como el SmartDrive de Microsoft que viene con el Windows), que reserva una zona para caché de disco en memoria de tamaño definible por el usuario. En ésta se guarda una pista (o cilindro) del disco cada vez que se ordena leer un sector de la misma. Luego cada sector solicitado se toma desde esta zona de memoria principal, en lugar de tener que accederse al disco. A continuación, esos datos pasan a otra zona de la misma donde está el buffer asignado al sector (figura 2.26). Para la escritura del disco, este caché va guardando los datos que deberían ir al disco. Cuando éstos tienen determinado tamaño, o luego de un tiempo especificado, se envían de una sola vez a una pista del disco, evitando múltiples accesos al mismo. El caché de escritura tiene el problema que si se corta el suministro eléctrico o se hace un reset general, los datos que están en el mismo se pierden.

Reservando 2 MB de memoria para este tipo de caché de disco, se han observado ganancias de hasta 20 veces en relación con el disco sin ningún tipo de caché.

Los disquetes también pueden tener un caché por software, pero sólo debe ser para lectura, dado que un disco flexible puede sacarse de la unidad en cualquier instante, con la consiguiente pérdida de la información a escribir.

Pueden usarse en forma complementaria caches incluidos y por software, pero en ciertos casos operar con ambos puede significar una peor performance que con uno solo de ellos.

## Qué es un disco virtual ó “disco RAM” ?

La idea es usar un área de memoria principal RAM como disco, para escribir y leer archivos completos en cortos tiempos de acceso y sin tiempos de transferencia. Disco RAM o virtual son sinónimos.

Para ello en el Config-Sys se debe agregar un renglón que lo defina, y hacen falta programas “drivers” para que el DOS y la ROM BIOS “vean” un disco (rápido) formateado, designado como ser “D”, con su FAT, área de booteo y directorio raíz también en la RAM.

<sup>1</sup> El caché de disco suele almacenar la FAT actualizada del disco, que permite localizar todos los sectores de un archivo, dado que antes de acceder a un archivo en el disco se debe acceder a la FAT (que puede cambiar con cada escritura de archivos). Así también en lugar de acceder a la FAT del disco y pasar ésta a memoria principal, se accede rápidamente al caché de disco, y de éste la FAT pasa a memoria.

Amen de ser hoy día los programas muy voluminosos, el problema de los discos virtuales es que si se interrumpe el suministro eléctrico, se pierden los archivos almacenados. Si sólo se pasan a un disco RAM copias de utilitarios de uso intenso, como ser un compilador, no se corre este riesgo, lográndose tiempos de acceso sumamente breves.

Un disco virtual es un disco abstracto simulado, inmaterial e intangible, que cobra existencia como datos sobre la memoria principal (dispositivo físico). Del mismo modo que el software, debe fijarse en la memoria para materializarse como algo utilizable. Se le puede asignar un nombre lógico, como ser "D".

A diferencia de un caché por software realizado en memoria principal –donde se almacena como ser una pista entera, cuando se accede a un sector de ella en el disco real– un disco virtual simula en memoria principal un disco rígido compuesto de varias pistas.

### ¿En qué consiste la compresión de un disco?

El objetivo de la compresión de datos en un disco es simular un disco con capacidad doble, triple o más de la real. Existen distintos tipos de programas que realizan tal cometido, y su rendimiento depende del tipo de archivo que se trate, pues se basan en la existencia de patrones de datos repetitivos.

Así es factible, si se tiene por ejemplo 50 bytes repetidos que representan la letra A, reemplazarlos por un byte de la letra A y otro byte que indique el número 50. Técnicas que son aptas para comprimir archivos de texto tienen baja performance para archivos gráficos, y viceversa. El programa DoubleSpace del DOS 6.0 trabaja en tiempo real<sup>1</sup>, independientemente del tipo de datos de un archivo busca patrones que se repiten, y los reemplaza por un número que indica dónde un patrón se repitió antes, y la longitud del mismo. En la descompresión estos números se reemplazan por los patrones originarios.

Los archivos con programas en código de máquina, o con datos encriptados (por ser éstos de carácter aleatorio), son poco comprimibles, a diferencia de los existentes en bases de datos, que presentan repeticiones.

### ¿Qué son las unidades de discos RAID - Redundant Array of Inexpensive Disks?

La idea originaria fue agrupar económicamente discos rígidos formando una disposición ("array") que sea visto por el sistema operativo como un disco de muchos Terabytes (como ser para una base de datos), posibilitando que además los discos puedan leerse simultáneamente. Esto equivale a que varias cabezas de un disco grande escriban o lean al mismo tiempo, lográndose transferir más bytes por segundo que con un solo disco.

Puesto que al existir más elementos móviles aumenta la probabilidad de fallas, al conjunto de discos posteriormente se le dió mayor confiabilidad con el agregado de información redundante para protección de los datos almacenados.



Figura 2.25.c

Los datos a almacenar son divididos en segmentos consecutivos o tiras ("stripes", designándose "striping" este método) del tamaño de bloques, sectores o bytes, que se distribuyen por todos los distintos discos: la primer tira del archivo se escribe en el primer disco, la segunda va al segundo, etc., repitiéndose luego este patrón (figura 2.25.c). De esta forma, si una lectura requiere tiras consecutivas, si existen  $n$  discos, se puede acceder en paralelo a  $n$  tiras. Inclusive si llegan a la controladora dos órdenes de lectura para archivos diferentes, es probable que las tiras a acceder estén en distintos discos, por lo que podrán leerse juntas.

Asimismo se emplean algoritmos para reducir tiempos seek y de latencia aprovechando la ventaja de poder acceder a varios discos en paralelo. También pueden sincronizarse los discos de manera que un instante determinado todas las cabezas estén en la misma vertical. Si las tiras son de corta longitud el tiempo de respuesta está determinado por estos tiempos (mseg), siendo despreciable el tiempo de transferencia hacia o desde memoria. Por otra parte es factible organizar los discos a fin de mejorar la performance de las lecturas en relación con las escrituras.

"Byte level striping size" significa que el archivo está formado por elementos de tamaño de un byte (o un número limitado de bytes); "Block-level striping size" implica que cada archivo se divide en bloques de igual tamaño (de 2 KB a 1 MB y más) que se distribuyen en los distintos discos. "Striping width" se refiere al número de segmentos de archivos que pueden ser escritos o leídos simultáneamente. Por ejemplo 6 para 6 discos, siendo que con más discos aumenta el rendimiento.

Se definen 6 "levels" normalizados de RAID que no implican necesariamente un orden creciente de desempeño, sino distintas alternativas para aplicaciones, siendo los más corrientes los niveles 0, 3 y 5. En algunas aplicaciones RAID 5 es mejor que RAID 0, en otras es a la inversa, siendo también un factor importante el costo.

<sup>1</sup> O sea comprimen los datos a medida que son enviados al disco, y los descomprimen cuando vuelven a memoria en la lectura de archivos.

**RAID 0:** Usa striping sin paridad para protección de fallas, lo cual implica mayor tiempo de recuperación si un disco tiene problemas. Cada archivo es dividido en tiras de tamaño definido por el usuario. Es de bajo costo y apto para datos estables que no sean críticos. Puede combinarse con otros niveles RAID.

**RAID 1:** los datos se guardan 100% por duplicado ("disk mirroring") en discos duplicados, o sea que sólo se aprovecha el 50% de la capacidad del conjunto. No se usa striping. En una lectura puede ser accedido uno cualquiera de los discos que contenga los datos pedidos. Si un disco falla simplemente se accede a su disco duplicado. Una mejora del RAID 1 es el *duplexing* que duplica el número de discos y el de controladores de disco.

**RAID 1** conjuga gran tolerancia a las fallas con bajo costo. Puede usarse en las aplicaciones donde la duplicación de datos es más segura que el uso de paridad, en bases de datos pequeñas, y en sistemas para transacciones en los que predominen las lecturas.

**RAID 3:** Emplea byte level striping con paridad (al igual que RAID 7) aprovecha la capacidad como RAID y mejora la tolerancia a los fallos reservando un disco del conjunto con información de paridad para la integridad de los datos de los restantes discos. El número de bytes de cada tira por lo general es menor que 1024. La información de paridad es guardada en un disco dedicado, lo cual puede afectar la performance. Si falla un disco se accede al disco de paridad y se reconstruyen "al vuelo" los datos con las tiras de los discos restantes, realizando la electrónica del controlador RAID operaciones lógicas "Or excluyente".

Apto para guardar archivos grandes con buena velocidad de transferencia dado que las tiras son de corta longitud, como puede requerirse en multimedia. Dado que en cada lectura o escritura se accede a todos los discos, sólo es factible una lectura o escritura por vez, por lo que en un sistema para transacciones la performance no es tan buena.

**RAID 5:** usa block-level striping (como RAID 4 y RAID 6) con striping parity para corregir errores. Así a la par que se consigue buena performance en relación con niveles que tienen un disco sólo para paridad, se logra capacidad de recuperación ante la falla de un disco del arreglo, siendo que cada bloque de datos y su información de paridad se guardan en discos separados. Si un disco falla, se comunica a otro, sin pérdida de datos o servicio. Por su relación costo/performance, RAID 5 es ampliamente usado para transacciones, bases de datos relacionales. Si bien para las lecturas al azar y secuenciales tiene excelente performance, no ocurre lo mismo con las escrituras.

Otra propiedad que interesa es la *disponibilidad* ("availability") de la información, pues para muchas empresas la imposibilidad de que usuarios o empleados no puedan acceder más allá de un cierto tiempo a la información puede significar una pérdida mucho más significativa que el costo del RAID.

RAID 1 y Parity RAID (RAID 3, 4, 5 y 6) pueden proveer Extended Data Availability and Protección (EDAP), esto es, capacidad de un RAID para proteger sus datos y proveer inmediato acceso on line a estos, como se necesita por ejemplo para los cajeros automáticos.

En general mientras que RAID 1 requiere duplicar los datos, el uso de información redundante de paridad sólo ocupa un 10 a 33% más de capacidad. RAID 1, 2, 3, 4, 5 y 7 necesitan la inutilización de un disco.

Los niveles simples de RAID emplean controladores IDE o SCSI corrientes, para 2 ó 4 discos, mientras que multi-niveles más complejos requieren tipos especiales de controladores, que también pueden manejar muchos discos.

Niveles RAID pueden combinarse para lograr una mejor performance. Así, RAID 01 ("mirror of stripes") divide los 10 discos en dos mitades. Una mitad se configura RAID 0, y la otra es un duplicado de ella (RAID 1). En RAID 10 ("stripe of mirrors") esos discos se dividirían en 5 grupos de 2 discos con igual información. Los strips son distribuidos por los 5 grupos. RAID 1+5 ó 5+1 combina mirroring más striping con paridad..

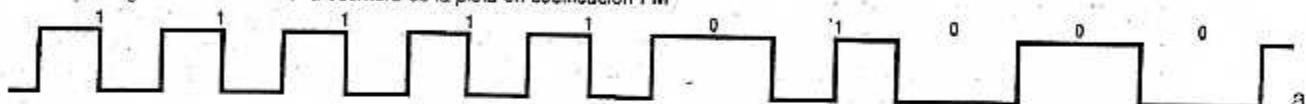
## ¿Qué son los métodos de grabación MFM y RLL ?

Según se expuso antes (figuras 2.4 y 2.5), una cabeza (bobina) mientras graba magnetiza en la pista grupos de pequeñas partículas microscópicas de óxido de hierro (no una sola), dando lugar a pequeños imanes que originan campos magnéticos en la superficie del disco, cuya polarización (S-N o N-S) depende del sentido de la corriente de la bobina. En las figuras 2.26 b, e, h aparecen pistas grabadas dibujadas rectilíneas, siendo que en las mismas se enfrentan dos polos iguales cuando cambia de nivel la señal eléctrica que se aplica a la cabeza (figuras 2.26 a, d, g), lo cual hace cambiar el sentido de la corriente que circula.

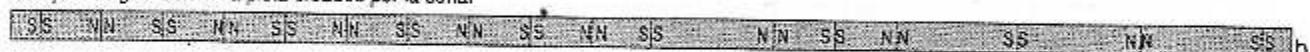
Cuando la misma cabeza debe leer, senza dichos campos, detectando campos magnéticos existentes debidos a imanes enfrentados (norte *contra* norte, sur *contra* sur). Vale decir, no detecta la existencia ó no de campo magnético, sino *inversiones* en el flujo (campo) magnético, cuando se enfrentan dos polos iguales. En una lectura, *al pasar la cabeza por cada una de estas inversiones, se genera en la bobina una corriente eléctrica que da lugar a una señal constituida por un breve pulso eléctrico* (figura 2.26 c, f, i). Los pulsos así generados, al ser decodificados por la electrónica correspondiente, permiten reconstruir la señal que excitó la bobina de la cabeza durante la escritura de la pista, y así decodificar los ceros y unos en el sector leído.

Una codificación emplea inversiones de flujo extra para separar bits, y otra los usa sólo para separar ceros. Estas inversiones usadas para demarcar bits -que en correspondencia requieren

Señal que llega a la cabeza en una escritura de la pista en codificación FM



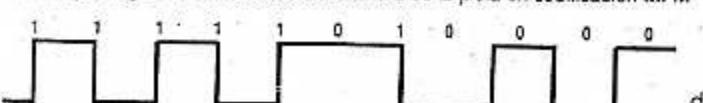
Campos magnéticos en la pista creados por la señal



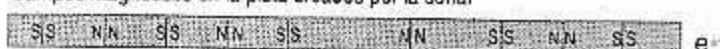
Señal que sale de la cabeza en una lectura de la pista



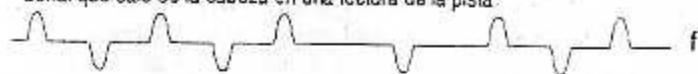
Señal que llega a la cabeza en una escritura de la pista en codificación MFM



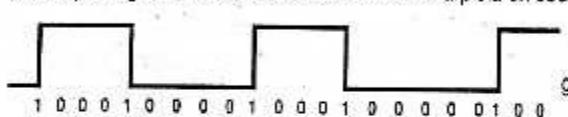
Campos magnéticos en la pista creados por la señal



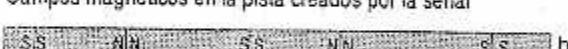
Señal que sale de la cabeza en una lectura de la pista



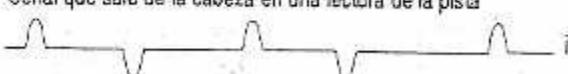
Señal que llega a la cabeza en una escritura de la pista en codificación RLL 2,7



Campos magnéticos en la pista creados por la señal



Señal que sale de la cabeza en una lectura de la pista



Grupo de bits a escribir	Codificación RLL 2,7
000	000100
0010	00100100
0011	00001000
010	100100
011	001000
10	0100
11	1000

Figura 2.26

cambios de nivel en las señales eléctricas que se aplican a una cabeza - se denominan "clocks", en el sentido que sirven para autosincronismo, a fin de poder determinar tiempos de duración de bits.

En la grabación de discos se usa principalmente el método de codificación conocido como MFM (Modulación de Frecuencia Modificada). En los rígidos la técnica anterior se ha reemplazado por otra conocida como RLL ("Run Length Limited", traducible como "longitud limitada de ceros corridos" o sea sucesivos), que permite hasta un 50% más de densidad de grabación. Ambas codificaciones son mejoras sucesivas del denominado método de grabación FM ("Frecuencia Modulada").

En la grabación FM (figuras 2.26 a, b, c) se emplea *siempre* una inversión de flujo antes de cada bit a escribir, sea uno o cero; y además se debe emplear otra inversión por cada bit de valor uno a escribir, inversión que se da a mitad de camino entre la inversión que indica su comienzo y la del comienzo del bit siguiente. O sea, que para escribir un uno se requiere *dos* cambios de nivel en la señal que recibe la cabeza: un cambio para indicar que empieza un bit, y otro para señalar que se trata de un uno.

A diferencia, la escritura de un cero implica sólo *un* cambio de nivel, para indicar el comienzo de dicho bit, siendo que la ausencia de otro cambio inmediatamente después identifica que se trata de un cero.

El número máximo de inversiones sucesivas de flujo magnético por centímetro o pulgada cuadrada debe permitir escrituras o lecturas seguras. Está limitado por las características del material magnético, por el ancho del entrehierro, y la sensibilidad de la cabeza.



Para un número máximo dado de tales inversiones, de lo que se trata, en principio, es codificar una mayor cantidad de unos y ceros por centímetro de pista, habiéndose desarrollado para tal fin varios métodos, que implicaron sucesivas mejoras en la densidad de almacenamiento. En todos ellos —como se planteó— en una escritura, cada cambio de nivel de la señal eléctrica que se aplica a una cabeza produce una inversión en el flujo magnético de la superficie de la pista que está siendo escrita. Por lo tanto, se busca codificar la mayor cantidad de unos y ceros con el menor número de cambios de nivel en dicha señal.

Los tres métodos de codificación que se discutirán tienen en común:

- Los unos y ceros a grabar están separados igual intervalo de tiempo entre sí; y
- Cada bit de valor uno a escribir le corresponde siempre en la pista una inversión del campo magnético; mientras que en correspondencia con cada cero a escribir, no existe ninguna inversión de campo. Pero esta convención sin más no permite en la lectura detectar cuántos ceros sucesivos han sido grabados.

La denominación FM proviene de que en la codificación de unos sucesivos resulta una frecuencia de pulsos mayor que la existente para ceros sucesivos, o sea que existen dos frecuencias distintas para unos y ceros.

Dado que en la codificación FM para grabar un uno se necesita dos inversiones de campo magnético en la pista, fue reemplazada por la MFM, que permite codificar un uno con una sola inversión de campo, siendo que sólo usa inversión para indicación de comienzo de bit cuando un cero está precedido por otro cero.

Esta convención permite codificar, como se exemplifica, la misma secuencia de unos y ceros antes exemplificada (11111010000) con la mitad de inversiones de flujo que con FM, por lo que en MFM se puede duplicar el número de bits por pulgada de pista, para una cantidad máxima de inversiones posibles por pulgada (que depende del material magnético usado).

Para los discos rígidos de gran capacidad fue necesario aumentar la densidad de grabación, para lo cual se creó la codificación RLL 2,7 que permite con un menor número de inversiones de flujo codificar una mayor cantidad de bits (hasta 50% más que con MFM). A tal fin, una sucesión de bits a escribir se descompone, a partir del primero, en sucesivos grupos de bits cuya codificación en RLL 2,7 se recuadra en la figura 2.26.

Esta recodificación minimiza el número de unos a grabar, y por ende, inversiones de flujo, siendo que en MFM también se necesitan inversiones cuando hay ceros consecutivos (en RLL sólo se usan para los unos).

En nuestro ejemplo, los datos a escribir 11111010000 se descomponen en los grupos 11 11 10 10 000 codificados en la figura 2.26 g, siendo que en RLL sólo se produce una inversión de flujo si hay un uno, sin emplear inversiones de comienzo de bit para los ceros en ninguna circunstancia.

La lectura de una pista exige una electrónica sofisticada, como la IDE o SCSI, para determinar correctamente en función del tiempo transcurrido, cuántos ceros existen entre la detección de dos "unos".

La figura 2.26 g permite apreciar que si bien la codificación RLL requiere el doble de bits que cada grupo de bits a escribir, el número de inversiones de flujo es menor que en MFM, resultando en comparación una ganancia en la densidad de bits almacenados, que estadísticamente puede llegar al 50%.

Las siglas 2,7 de la codificación RLL 2,7 resultan de la tabla anterior. Después de un uno puede haber dos ceros como mínimo, y tres ceros como máximo; y antes de un uno como máximo pueden darse cuatro ceros. Por consiguiente, entre dos unos como mínimo pueden haber dos ceros, y como máximo siete ceros.

Ocho o más ceros seguidos, se descomponen en grupos de tres ceros, cada uno codificable como 000100.

Existen también las codificaciones RLL 1,7 y RLL 3,9 también conocidas como ARLL (Advanced RLL), que permiten hasta un 90% de ganancia de densidad en relación con MFM.

## ¿Qué son los disquetes y unidades ZIP?

Las unidades ZIP ("Zip Drive"), por las capacidades de sus disquetes, por su confiabilidad, y por su velocidad de transferencia están a mitad de camino entre las unidades de disquete y las de disco duro, aunque más próximas a esta última. Así, su velocidad de giro es del orden de 3000 r.p.m, lo cual redundaría en una mayor velocidad de transferencia. El gabinete del ZIP drive es externo al gabinete del computador.

La conexión del ZIP drive generalmente se hace en el port paralelo que usa la impresora, debiéndose desconectar ésta de dicho port, y volverla a conectar al gabinete del ZIP drive en un conector preparado.

Los disquetes para ZIP drive son flexibles, y pueden almacenar en sus dos caras magnetizables 100/200 MB, empleándose comúnmente para back-up del disco rígido. Las cabezas de escritura/lectura están en contacto con las superficies de ambas caras, siendo más pequeñas en tamaño que las usadas en una disquetera, lo cual permite grabar y sensar con densidades de grabación mayores.