Capítulo 8

ENTRADA/SALIDA Y CANALES

8.1 CANALES (*BUSES*)

Para que los componentes de una arquitectura de Von Neumann puedan constituir una computadora funcional es necesario que se comuniquen. Evidentemente entre el CPU, la memoria y los dispositivos de entrada y salida se deben intercambiar datos. Para ello se necesita un canal de comunicación: un *bus*. Físicamente esto no es más que un conjunto de cables o pistas en un circuito impreso, conectores entre los dispositivos y el canal y un sistema regulador que controla quien y cómo usa esos cables en un momento determinado.

El esquema general de la comunicación es el mostrado en la figura 8.1. Como se puede ver, es necesario que los diferentes componentes se transmitan tres cosas diferentes:

- Datos. Lo más obvio es que se deben intercambiar datos. Datos la memoria le proporciona al CPU para operar con ellos, o datos que el CPU almacena en la memoria; otroa que pueden ser transferidos de la memoria a algun dispositivo de salida o bien de un dispositivo de entrada a la memoria.

- Direcciones. Los datos están almacenados en algún lugar antes de ser transferidos o deben ser almacenados en algún lugar diferente. La localzación de un dato es, en el sentido más general del término, *su dirección*. Solemos decirle dirección al índice de

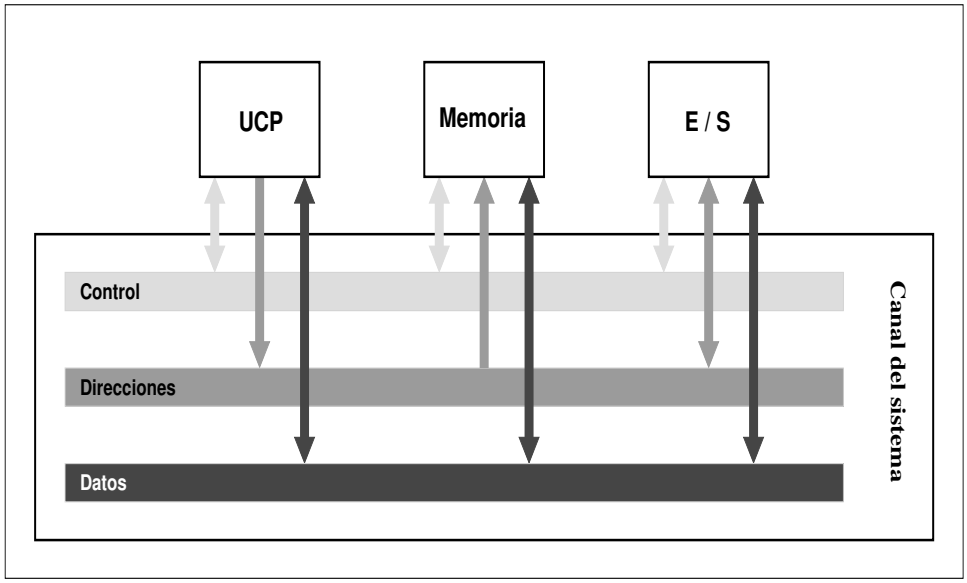

Figura 8.1. Esquema general de comuniación entre los diferentes componentes de la arquitectura de Von Neumann usando un canal.

una celda de memoria, eso es correcto, pero no es el único tipo de dirección posible. Las coordenadas de locaclización de un dato en un disco o en el buffer de una interfaz de red son también direcciones. Por supuesto para solicitar la lectura o escritura de un dato se debe porporcionar su ubicación y eso debe viajar por el canal.

- Señales de control. A los dispositivos conectados al canal se les debe indicar qué hacer con lo datos y las direcciones porporcionadas: a la memoria, por ejemplo, se le debe indicar si la dirección proporcionada debe leerse o escribirse. A través de las señales de control del canal se puede también indicarle a un dispositivo que la operación solicitada ya se llevó a cabo o enviar acuse de recibo (*acknowledge*) por los datos.

Un canal puede tener lineas exclusivas para cada cosa o un sólo conjunto de líneas para todo, cuyo uso se divide en el tiempo para cada de las categorias descritas. En este último caso se dice que el canal es *multiplexado*. Desde hace mucho tiempo los canales siempre tienen líneas de control independientes y entonces el término multiplexado se refiere solamente a las líneas de datos y direcciones. Esta característica se denomina el *ancho del canal*. Por supuesto se obtiene un mejor

desempeño si el canal tiene ancho para líneas de datos y de direcciones independientes y menor costo si es multiplexado.

No se debe confundir el ancho del canal descrito en el párrafo previo y lo que se suele denominar *ancho de banda*. El ancho de banda es una caracteristica de cualquier medio de comunicación, en particular de los canalas de una computadora. Es la medida de qué tan alto es el desempeño del medio de comunicación. Se mide en unidades de información sobre unidades de tiempo. Mega bits por segundo, por ejemplo (Mbps) o Gigabytes por segundo (GB/s)[1].

En el canal de un sistema de cómputo se llevan a cabo intercambios de datos entre diferentes componentes: la unidad central de proceso, por ejemplo, puede requerir de un operando almacenado en alguna dirección de memoria y procede entonces a emitir una solicitud destinada a la memoria con la dirección de dato que requiere. Más tarde la memoria le responderá con el dato solicitado y se habra llevado a cabo entonces lo que se denomina *transacción*. Una transacción está compuesta entonces por dos partes:

1. Petición. Un dispositivo requiere de un dato que no posee y sabe de alguna manera la dirección del dato en algún dispositivo. Procede entonces a emitir una señal de petición del dato y la dirección que permite ubicarlo en el dispositivo que lo contiene.

2. Respuesta. El dispositivo que posee el dato ya lo encontró y procede a enviarlo a quien lo solicitó.

En principio, el canal está desocupado entre la petición y la respuesta. Hay canales en los que durante este intervalo se pueden llevar a cabo, iniciar o terminar otras transacciones. A esto se le denomina *transacciones fragmentadas* y bajo ese esquema el uso del canal es más eficiente.

Otra variable a considerar en un canal es el número de dispositivos que pueden iniciar una transacción. A lo dispositivos con este privilegio se les denomina *bus masters*. Por supuesto la unidad central de proceso siempre debe poder iniciar transacciones, así que es un bus master, pero puede no ser el único. Si lo es entonces todo acceso al canal debe pasar por el CPU, lo que lo convierte potencialmente en un cuello de botella. Si no es así entonces debe existir una política de acceso al canal, es decir, un procedimiento que determine, en el caso de que varios dispositivos requieran acceder al canal simultáneamente, a cuál

[1]Se han ejemplificado aquí las dos notaciones usuales: poniéndo "ps" para decir "por segundo" o bien poniendo el cociente explícito. Normalmente se usa "b" minúscula para denotar bits y la mayúscula para los bytes.

Característica	Mejor desempeño	Menor costo
Ancho del canal	datos+direcciones	multiplexado
bus masters	múltiples	único
Transacciones	fragmentadas	conexión continua
Sincronización	síncrono	asíncrono

Tabla 8.1. Alternativas de diseño de canales.

de ellos se le concede el uso del canal. En una transacción es entonces un bus master quien la inicia y en ese momento quien recibe la petición hace las veces de *esclavo*.

Al procedimiento para regular el acceso al canal se le denomina *esquema de arbitraje* y, de ser necesaria una entidad central reguladora, se le denomina, claro, *árbitro* del canal.

Una última alternativa en el diseño de un canal es su sincronización. Si existe un reloj que marca el ritmo en el canal, de tal forma que, por ejemplo, sólo sea posible iniciar una transacción cuando el reloj sube de 0 a 1, entonces la interfaz de todos los dispositivos con el canal debe operar a la misma fecuencia o un múltiplo de ella. Se dice entonces que el canal es *síncrono*. Tiene la ventaja de que cualquier dispositivo que inicia una transacción sabe que su contraparte está "despierto" también. En un canal más permisivo, en el que cada dispositivo puede operar a su propia frecuencia, entonces la sincronización entre el maestro y el esclavo debe hacerse en un nivel de abstracción ligeramente superior, mediante la ejecución de un protocolo de sincronización normalmente llamado de *apretón de manos* o *handshake*. Un canal síncrono es más simple de diseñar. Uno asíncrono puede ser más largo porque no hay atenuación de señal de reloj por la que preocuparse.

En todas estas disyuntivas de diseño hay, claro, un compromiso entre el costo y el desempeño o la flexibilidad. En la tabla 8.1 se sintetizan las opciones.

8.2 PROTOCOLO DE SINCRONIZACIÓN

Supongamos que un dispositivo A le solicita a otro B la transferencia de datos indicándole la localización de los mismos, es decir la dirección donde se encuentran. Se procede como sigue:

1. A levanta una señal de petición de datos DataReq en las líneas de control del canal y coloca la dirección de los datos requeridos en las lineas de direcciones.
2. B se dá cuenta de que le es solicitada información al ver la señal DataReq para él y lee la dirección de las líneas de direcciones. Levanta entonces una señal de reconocimiento Ack.
3. A ve el Ack y libera la señal de petición y las lineas de direcciones.
4. B baja la señal Ack y busca los datos solicitados (de hecho esto lo empieza a hacer desde el paso 2).
5. B coloca los datos leídos en las lineas de datos del canal y envía una señal DataRdy a A.
6. A ve la señal DataRdy, lee los datos del canal y levanta Ack otra vez.
7. B se percata del Ack, baja la señal DataRdy y libera las lineas de datos.
8. A ve abajo DataRdy y baja Ack lo que completa la transacción.

Es la ejecución de este protocolo lo que hace más lentos los canales asíncronos.

8.2.1 Otra clasificación

Los canales también pueden caracterizarse por su uso. En esta clasificación las categorías son las siguientes:

- Canales memoria-CPU. El ancho de banda del canal es igual al rendimiento (*throughput*) de la memoria. Ningun otro dispositivo se conecta al canal. El ancho del canal es también igual al tamaño de bloque entregado por la memoria.
- Canal de panel trasero. Se llaman asi porque tradicionalmente los conectores del canal estaban dispuestos en un panel trasero del chasís de la computadora (como hoy día los conectores del bus sobre la tarjeta madre o *motherboard*). Pretenden ser un punto intermedio entre los canales de memoria y los de entrada/salida, poseen un ancho de banda suficientemente alto como para conectar en ellos la memoria sin que el desempeño sufra demasiado y al mismo tiempo permiten la coexistencia de el resto de los dispositivos compartiendo el canal. En general requieren de lógica adicional para admitir las conexiones de los distintos dispositivos.

- Canales de entrada/salida. Los canales anteriores son, en general, cortos, todos los dispositivos deben estar físicamente cerca, en los canales de entrada/salida esto no necesariemente es así, pueden ser bastante largos y casi no requieren de lógica adicional para la conexión de los diferentes dispositivos.

8.2.2 Esquemas de arbitraje

Cuando hay múltiples *bus masters* debe haber un árbitro que regule el acceso al canal. Hay diferentes esquemas para definir qué dispositivo accede al canal. En general hay dispositivos que tienen mayor prioridad de acceso, como ya hemos dicho.

- Arbitraje *Daisy chain*. Por el canal fluye, en una sola dirección, una señal de permiso de acceso, los dispositivos se conectan al canal en orden decreciente de prioridad, cuando un dispositivo ve la señal de permiso pasando frente a él, si tiene transacciones pendientes, atrapa la señal e inicia su transacción. Por supuesto esto hace que los dispositivos de prioridad más alta puedan hablar antes que los de prioridad más baja, dado que ven la señal antes que estos últimos. Lo malo es que si algun dispositivo deja de funcionar correctamente y se queda con la señal permanentemente, el canal completo se vuelve inútil.

- Centralizado. Hay un árbitro que recibe, ya sea por líneas de control específicas o por líneas del canal, peticiones de acceso de los diferentes dispositivos. El árbitro sabe cuáles dispositivos tienen mayor prioridad y responde especificamente al dispositivo seleccionado, que puede iniciar su transacción. El problema es que el árbitro puede volverse un cuello de botella. PCI es un canal centralizado.

- Arbitraje distribuido por autoselección. Cada dispositivo sabe su propia prioridad y se dá cuenta de las prioridades del resto de los dispositivos que hayan solicitado acceso simultáneo al canal, si es el de mayor prioridad sabe que tiene derecho a iniciar su transacción, si no, sabe que debe esperar. SCSI y el canal de las Apple Macintosh II son de autoselección.

- Arbitraje distribuido por detección de colisión. Es como el CSMA/CD de las redes Ethernet. Si un dispositivo requiere iniciar una transacción lo intenta, si hay más de un intento de acceso al canal todos se percatan de ello y aquellos que comenzaron sus

transacciones al mismo tiempo esperan un tiempo antesde volver a intentar.

Hoy en día hay muchos y muy variados canales, algunos de ellos se han convertido en un estándar en la industria, primero en un estándar de facto y más tarde en estándar formal, por ejemplo SCSI. Algunos de los estándares más famosos han surgido del mundo de las computadoras personales más usuales, las de procesador de Intel 80X86.

Canales de PC.

El primer canal de una PC de IBM fue el llamado S-100 (100 lineas), 2.3 MHz de frecuencia, 8 bits de capacidad. Más tarde los 8 bits se convirtieron en 16 y en 1987 el IEEE promovió el diseño del bus como un estándar llamado ISA (*Industry Standard Architecture*). ISA era un canal de 16 bits a una frecuencia de 8 MHz, una tasa de transferencia de 8Mbytes por segundo. El canal ISA fue usado por la IBM PC AT. A ISA se conectaban el procesador, los periféricos y la memoria inclusive, asi que resultaba bastante lento para nuestros estándares de hoy día. Cada dispositivo conectado a bus lo hace a través de un controlador, de tal forma que hay muchos *bus masters* (cada controlador lo és). Para acceder a la memoria todos debían pasar por procesador, algo que disminuye el desempeño notablemente, dado que el procesador debe mediar siempre entre el controlador del dispositivo que requiere la memoria y esta última.

Como a pesar de todo el bus era lento, IBM decidió diseñar un nuevo canal mucho más rápido, el resultado fué la arquitectura de microcanal (MCA o *MicroChannel Architecture*) bastante más rápida, mucho más costosa e incompatible con ISA. Normalmente los dueños de computadoras desean poder actualizar estas y poder reutilizar: el software (de allí la eterna compatibilidad hacia atrás de Intel) y los periféricos. El hecho de no poder reutilizar los periféricos hizo poco rentable usar máquinas con MCA y el resultado fue que IBM acabó haciendo máquinas que no eran *IBM compatibles*. MCA surgió en 1987 y fue usado sólo en las PS/2 de IBM, transfería 32 bits de una sola vez y su tasa de transferencia era de unos 40 Mb por segundo, era un canal asíncrono.

Hoy en día nuestros canales más usuales son PCI (*Peripheral Component Interface*), creado en 1992. PCI opera a 33MHz transfiere 32 bits de una sola vez y su máxima tasa de transferencia es de 132 Mbps, es un canal síncrono, *backplane*, con direcciones y datos multiplexados (ambas cosas por las mismas líneas), admite múltiples *bus masters*, de arbitraje centralizado y admite la conexión de hasta 32 dispositivos por segmento, aunque es posible conectar más de un segmento y colgar

hasta 1024 dispositivos. Actualmente se trabaja (Compaq o mejor dicho HP) en una extensión de 64 bits para PCI llamada PCI-X que será capaz de transferir hasta 1Gbps con transferencias de 64 bits de una sola vez. La especificación actual más reciente es PCI 2.2, también de 64 bits a 66 MHz y con transferencia máxima de 533 Mbps. Detrás de PCI hay un organismo, el grupo de interés especial en PCI (*PCISig*) que se encarga de formular los estándares de PCI.

Otro canal famoso es el SCSI, del que hablamos en el contexto de discos. SCSI es un canal de entrada/salida, datos y direcciones se multiplexan, admite múltiples *bus masters*, es de arbitraje distribuido de autoselección, hay versiones síncronas y asíncronas de SCSI, también hay distintas versiones con diferentes límites de conexión: 7, 15 y 31 dispositivos, por ejemplo; la longitud puede ser de hasta 25 metros.

Otros buses exitosos usados hoy en día son los punto a punto usados por Alpha (21264) y el Athlon de AMD, el bus es el EV6 de 100 y 200 MHz nominales y 200 MHz y 400 MHz efectivos, 32 bits.

8.3 DISCOS

El dispositivo de almacenamiento por excelencia es, hasta hoy, el disco magnético fijo, al que comúnmente llamamos disco duro.

Estructura física de un disco duro.

Un disco duro consiste de una pila de uno o más platos[2] de aluminio recubiertos de oxido ferroso para hacerlos susceptibles a la magnetización. Los platos miden desde 1.2 hasta unas 5 pulgadas, lo más popular, por el momento, parecen ser los de 3 1/2 pulgadas, el diámetro de un disco flexible, en las *nootebooks* son usuales los de 1.2. La capacidad de almacenamiento, actualmente, va desde 4.3 Gb. hasta alrededor de 80 Gb[3]. Las labores de escritura y lectura en el disco están a cargo de un conjunto de cabezas capaces de desplazarse en una linea recta radial.

Para escribir en el disco la cabeza recibe corriente del exterior, positiva o negativa dependiendo del bit a escribir, esta corriente induce un campo magnético que alinea las moléculas del disco hacia un lado u otro (0 o 1, izquierda o derecha). Para leer la operación es inversa, el campo magnético de la superficie del disco induce una corriente en la cabeza que se traduce en 0 o 1 dependiendo del signo de la corriente inducida. La cabeza lectora/escritora no toca la superficie del disco[4] y lee o escribe la secuencia de bits conforme el plato gira. Las velocidades

[2] Algunos llegan hasta 12 platos, p.ej. el Seagate Cheetah 18LP SCSI de 36Gb.
[3] El Quantum Atlas 10K II es de 73Gb (SCSI) y el IBM DTLA307075 es de 76 Gb (IDE)
[4] En los discos flexibles si la toca.

de giro actuales más comunes son de 5400 RPM o 7200 RPM, pero hay discos de 10,000 RPM SCSI y seguramente habrá algunos más veloces.

En general la densidad de almacenamiento de los discos magnéticos, durante los 1990 se triplico cada dos años (al menos) y el costo ha descendido de 100 USD el megabyte en 1983 a un rango de 6 a 9 USD el gigabyte actualmente (discos IDE).

Los datos en el disco se organizan en pistas (*tracks*) concéntricas, cada pista es dividida en tramos llamados sectores y el conjunto de sectores en la misma posición en todos los platos se denomina cilindro. La información de cada sector va precedida de un preámbulo (generalmente una secuencia de bits con valores alternados) para sincronizar la cabeza lectora, luego el bloque de datos (512 bytes es un tamaño usual) y luego un código corrector de errores (ECC, generalmente Reed-Solomon). Para separar cada sector se deja entre ellos un pequeño espacio vacío (*gap*). Generalmente la capacidad reportada de los discos ya excluyen toda la que se utiliza en preámbulos, corrección de errores y *gaps*, en lo que se invierte un 15% de la capacidad absoluta.

Seek time y latencia rotacional.

El desempeño de los discos depende de cuanto tiempo tome colocar las cabezas en posición, y cuanto tiempo tome que los datos pasen por debajo de la cabeza. Así que hay dos periodos de tiempo a medir:

- El tiempo usado para poner las cabezas a la distancia adecuada del centro para leer/escribir la pista que se requiere. A esto se le llama *seek time*.
- El tiempo que hay que esperar para que el sector requerido pase debajo de la cabeza. A este se le llama *latencia rotacional*.

Además hay que considerar el tiempo requerido para transferir los datos de la cabeza al canal de donde serán leídos (lo que involucra su verificación de integridad), pero ciertamente lo más tardado es la operación mecánica del disco, los dos factores mencionados arriba son los que dominan el tiempo de acceso.

Densidad variable.

El lector observador se estará preguntando "y el disco ¿tiene más sectores por pista mientras más lejos del centro esté la pista?". Dado que un circulo de mayor radio tiene mayor circunferencia que uno de radio menor, esta pregunta es lógica y representa un problema, dado que los discos giran a una velocidad angular constante (evidente dado que se usan RPM para medir la velocidad). En principio cabrían más sectores en la pista más exterior que en la más interior, así que hay que decidir que hacer. Lo más sencillo resulta ver cuantos sectores caben en la más interior de las pistas y grabar siempre a esa densidad. Esto hace que las pistas más externas se desperdicien mucho, pero bueno,

esa fue la solución adoptada por todos hasta hace poco. Actualmente, para minimizar el desperdicio se dividen las pistas en categorías (de 10 a 30 categorías diferentes) cada categoría caracterizada por el número de sectores que caben en la pista más interior del conjunto, luego todas las pistas en la categoría se graban a distinta densidad variando a saltos entre categorías.

Asi que resulta mejor para el desempeño tener discos de mayor velocidad de giro y de menor tiempo de posicionamiento de cabezas. La medida de desempeño debe ser entonces el tiempo promedio requerido para leer o escribir un sector en el disco. Por ejemplo: con un disco Seagate Barracuda IDE de 7200 RPM con un tiempo promedio de posicionamiento de cabezas de 8.2 ms. una capacidad de transferencia de 45.5 Mb/seg y tiempo de transferencia (*overhead*) de 1 ms.[5] obtenemos:

$$7200\ RPM = 120\ Rev/seg = 0.12\ Rev/ms \tag{8.1}$$

Si, como es usual, cada sector tiene 512 bytes (0.5 Kb) entonces, cada vez que se lee un sector, como se transfiere a una tasa de 45.5 Mb/seg = 46592 Kb/seg = 46.592 Kb/ms, el tiempo requerido para transferirlo es:

$$\frac{0.5\ Kb}{46.592\ Kb/ms} = 0.0107\ ms \tag{8.2}$$

dada la posición de las cabezas y la del sector a ser leído, habrá veces que el sector tendrá que dar toda una vuelta para llegar bajo la cabeza, habrá veces que no tenga que girar nada, en promedio tendrá que dar media vuelta. Así que el tiempo promedio de latencia rotacional, usando el resultado obtenido en 8.1 es[6]:

$$\frac{0.5\ Rev}{0.12\ Rev/ms} = 4.166\ ms \tag{8.3}$$

Ahora, usando los resultados de 8.2 y 8.3 y añadiendo el tiempo de posicionamiento promedio y el *overhead* obtenemos el tiempo promedio de acceso:

$$\overline{T}_{disco} = 8.2\ ms + 0.0107\ ms + 4.166\ ms + 1\ ms = 13.377\ ms \tag{8.4}$$

Esta es una estimación del promedio, el *seek time* promedio que reportan los fabricantes del disco es el promedio sobre todas las posibles posiciones de las cabezas, no es un promedio medido sobre el uso real

[5]Este valor es el único inventado en este ejemplo, pero no es descabellado, generalmente esta entre 1 y 2 ms.
[6]Por cierto el valor obtenido coincide con el reportado por Seagate.

del disco, este es de un 25 a un 35 % del *seek time* promedio dado por el fabricante, dependiendo del que tanto del disco haya sido usado y de la fragmentación de la información en el mismo (aquí el sistema operativo tiene un papel importante en el desempeño). Si usáramos el 30 % de los 8.2 ms (es decir 5.74 ms) dados por Seagate tendríamos:

$$\overline{T}_{disco} = 5.74\ ms + 0.0107\ ms + 4.166\ ms + 1\ ms = 10.9167\ ms \quad (8.5)$$

Recalibración térmica.

Ahora es evidente que, mientras más rápido gire un disco, mejor será su desempeño porque disminuye su latencia rotacional. Sin embargo, dado que los discos giran muy rápido hay fricción, esto significa calor, así que los discos tienden a calentarse y como son de metal (aluminio, que además es buen conductor de calor) se dilatan, ol que hace que las pistas del disco, luego de un tiempo, ya no están donde deberían estar, sino un poco más afuera (además la diferencia no es constante, es una función lineal del radio de la pista). Así que los discos duros deben autorecalibrarse con frecuencia, lo que hace que, de pronto algunos accesos del disco se retarden inusualmente. Para recalibrarse los discos avanzan sus cabezas a lo largo de todo su desplazamiento de ida y vuelta. Para evitar este inconveniente hay discos que se hacen de vidrio en vez de aluminio, como el vidrio es muy frágil se recubre de cerámica, ninguno de los dos materiales se dilata.

IDE Vs. SCSI.

Todos los discos funcionan igual, sin embargo se clasifican en dos categorías dependiendo de la interfaz del disco con el bus: IDE (*Integrated Drive Electronics*), también llamado ATA (*Attachment*) y SCSI (*Small Computer Systems Interface*). Generalmente los discos SCSI tienen tiempos de posicionamiento menor y pueden girar a velocidades mayores, esto no se debe a limitaciones mecánicas de los IDE sino a que estos, o mejor dicho sus controladores, requieren de la intervención del CPU del sistema mientras que los controladores SCSI son independientes y están, por entero dedicados al disco. Un disco IDE no puede entregar tan rápido porque su controlador es más simple. Los controladores SCSI son además suficientemente inteligentes como para manejar eficientemente solicitudes diversas, encolarlas y procesarlas con traslape de tiempo para aprovechar al máximo las posiciones relativas de los diversos accesos y no perder tanto tiempo de *seek*. Además SCSI contempla la posibilidad de tener muchos dispositivos conectados en cascada en la misma interfaz, 7 en algunas versiones de SCSI, en otras hasta 15 dispositivos, sin que se vuelva un cuello de botella la interfaz (en la medida de lo posible). Esto hace que SCSI sea la alternativa idónea para sistemas con muchos discos o varios dispositivos en general, que deben poder ser accedidos concurrentemente en un ambiente

Versión	Bus (bits)	Transferencia (Mb/seg)
SCSI-1	8	5
Fast SCSI	8	10
Fast Wide SCSI	16	20
Ultra SCSI	8	20
Wide Ultra SCSI	16	40
Ultra 2 SCSI LVD	8	40
Wide Ultra 2 SCSI lvd	16	80
Wide Ultra SCSI 3 LVD	16	160

Tabla 8.2. Las diferentes versiones de SCSI. LVD significa *Low Voltage Differential* una estrategia para proveer de redundancia al bus de SCSI y así poder inmunizarlo a un amplio rango de errores, lo que hace posible tener un cable de hasta 25 metros confiable.

multitarea (como servidores, por ejemplo). En ambientes más sencillos y, en general, en máquinas con uno o pocos usuarios y una carga no muy pesada, los discos IDE son la mejor opción, tienen tiempos de posicionamiento de dos o tres milisegundos más que los SCSI, pero cuestan la mitad.

SCSI sí es un estándar y ha pasado por varias versiones desde que se estableció en 1986. Las diversas versiones se encuentran listadas en la tabla 8.2.

8.4 EL SISTEMA OPERATIVO Y LA E/S

Así como el desempeño del sistema, en particular del CPU, depende en mucho de los compiladores, el desempeño del sistema de E/S depende en mucho del sistema operativo.

En general el sistema operativo debe:

- Garantizar que los procesos tengan acceso a los dispositivos si tienen permiso para ello. Es decir, el sistema debe poder poner el semáforo verde a los procesos.
- Asegurar que ningun proceso no autorizado para acceder a un dispositivo tenga acceso a él. Es decir, debe poder poner el semáforo rojo.

- Proveer de un conjunto de abstracciones que hagan manejables los dispositivos y oculten las complicaciones de hardware innecesarias (es más fácil leer un archivo que leer el sector X del cilindro Y, pista Z).
- Proveer de servicios a los dispositivos. Servicios indispensables en el momento preciso. Proveer por ejemplo, de las herramientas necesarias para el manejo de interrupciones y las rutinas de servicio asociadas o proveer los mecanismos necesarios para el monitoreo frecuente (*polling*).
- Optimizar el acceso a los recursos de E/S para mejorar, en lo posible, el desempeño. Por ejemplo que el diseño del sistema de archivos sea tal que minimice la fragmentación de datos en el disco y con ello los viajes largos de las cabezas de lectura/escritura.
- Proveer de servicios de alto nivel a los programadores. Por ejemplo las llamadas al sistema Unix que permiten hacer E/S de manera casi uniforme a dispositivos muy diversos.
- Garantizar la consistencia de la jerarquía de memoria cuando se tienen dispositivos de acceso directo a memoria (DMA). Garantizar que si el dato de la localidad X, que es escrito mediante acceso directo por un dispositivo, se encuentra alojado también en el cache el contenido de estos caches sea consistente con la memoria principal.