UNIVERSIDAD NACIONAL DE LA MATANZA

Departamento de Ingeniería e Investigaciones Tecnológicas

Sistemas Operativos

Trabajo Práctico N° 17: “Driver IDE/ATA”

Profesores:

Nicanor Casas
Graciela De Luca
Gerardo Puyo
Waldo Valiente
Sergio Martin
Federico Diaz
Martin Cortina

Alumnos:

Maly German	DNI: 34.890.399
Molina Rodrigo Germán	DNI: 35.025.419
Salica Matias	DNI: 35.095.388
Torres Pablo Sebastían	DNI: 34.906.129
Traverso Cristian Jorge	DNI: 35.118.774

Año: 2011

Curso: Jueves, noche

Índice

Descripción de la interfaz y características ATA

German Maly, Rodrigo German Molina, Matias Salica, Pablo Sebastian Torres, Cristian Jorge Traverso
Universidad Nacional de La Matanza

Abstract
El propósito de este documento es describir la interfaz ATA, indicando sus características principales, los modos de transmisión de datos, los registros que utiliza para la comunicación y de qué manera maneja las interrupciones generadas por los dispositivos conectados a esta interfaz.

Palabras Clave
ATA, IDE, EIDE, PIO, DMA, CHS, LBA, IRQ, Polling.

Introducción

Historia del estándar ATA

El estándar ATA fue creado en 1986 por tres compañías, una división de Control Data Corporation llamada IMprimis, Western Digital y Compaq Computer, basado en el controlador de disco duro de la PA/AT de IBM desarrollado por Western Digital en 1984.
Es la interfaz estándar que permite conectar distintos periféricos de almacenamiento a equipos de PC.
Aunque el nombre oficial de este estándar es ATA, se lo conoce en general por el término comercial IDE (Electrónica de Unidad Integrada) ó IDE Mejorado (EIDE ó E-IDE).
La interfaz fue descripta por primera vez en 1989 bajo el nombre de CAM ATA (Common Access Method).

Desarrollo

Modos de transmisión de datos

Hay dos modos de transmisión de datos utilizados, uno es el modo PIO (Entrada/Salida Programada) y el otro el modo DMA (Acceso Directo a Memoria)

- ***Modo PIO***

La transmisión de datos se realiza gracias a un protocolo llamado PIO que permite el intercambio de datos entre los periféricos y la memoria RAM, y el control de estas operaciones lo tiene el procesador.

Una desventaja de este modo es que si existen grandes transferencias de datos, se va a producir una gran carga de trabajo en el procesador reduciendo la velocidad de todo el sistema.

- ***Modo DMA***

En este modo, los periféricos acceden directamente a la memoria, sin necesidad de ocupar tiempo del procesador. Existen dos tipos de DMA:

- *DMA de palabra única:* que permite la transferencia de una sola palabra durante cada transferencia.
- *DMA de palabras múltiples:* que permite la transferencia sucesiva de varias palabras en cada sesión de transferencia.

Modos de direccionamiento

Existen tres modos de direccionamiento para seleccionar los sectores para la lectura ó escritura en un disco. Los modos son LBA de 28 bit, LBA de 48 bit y CHS. El número de bits en el modo LBA hace referencia al número de bits significativos en el sector de “dirección”, llamado LBA. En el modo de 28 bit, las direcciones desde 0 hasta 0x0FFFFFFF son legales. Esto nos da un total de 256M de sectores, ó 128 GB de espacio direccionable. Por esta razón el modo LBA de 28 bits es obsoleto para la mayoría de los dispositivos actuales. Sin embargo, el modo PIO de 28 bits es más rápido que el direccionamiento con 48 bits, por lo que puede ser una mejor opción para dispositivos ó particiones que no sobrepasen el valor máximo de direccionamiento LBA.

- ***LBA Absoluto/Relativo***

Todos los comandos ATA que usan direccionamiento LBA necesitan LBA “Absoluto”. A primera vista, parecería más eficiente almacenar los valores LBA en este mismo formato en el SO. Sin embargo, no es el caso. Siempre es necesario validar los LBA que pasan al driver, para saber si pertenecen a la partición que se está accediendo. Termina siendo más inteligente usar particiones con direccionamiento LBA relativo, ya que de esta manera no es necesario verificar si esa LBA está accediendo a la partición correspondiente, es decir, sólo es necesario hacer la mitad de las verificaciones. Esto compensa el hecho de que se necesita agregar LBA absoluta al principio de cada partición actual para cada valor LBA relativo que se le pasa al driver.

- ***Modo CHS***

El modo Cilindro, Cabeza, Sector es completamente obsoleto, pero hay ciertas cosas que es necesario conocer sobre este método.
Los dispositivos más antiguos tenían muchas “platos” y dos “cabezas” de lectura/escritura por plato. Las cabezas siempre estaban alineadas en forma vertical. Una de las cabezas de uno de los platos normalmente era usada para el “timing”. Como todos los platos rotaban, cada cabeza trazaba un círculo, y todas las cabezas juntas trazaban un “cilindro”. Cada círculo trazado por cada cabeza, denominado pista, se subdividió en una cierta cantidad de “sectores”. Cada sector podría ser usado para almacenar 512 bytes de datos. Seleccionando el Cilindro, la Cabeza y el Sector se obtenía un modo de direccionamiento.

Cambiar el cilindro significaba mover todas las cabezas juntas, lo cual se trataba de evitar en lo posible.
Pero lo más importante es que no se ha hecho así desde los últimos 20 años, excepto en las computadoras que mantienen el acceso a los datos mediante una dirección CHS artificial.
En el modo CHS, cada dispositivo tiene una “geometría”, es decir, un rango permitido de valores CHS. Generalmente los máximos valores permitidos son Cyl = 0 a 1023, Cabeza= 0 a 15, Sector= 1 a 63.

Funcionamiento en detalle del modo ATA PIO
De acuerdo a las especificaciones técnicas de ATA, el modo PIO debe ser siempre soportado por todos los drivers ATA compatibles como el mecanismo de transferencia de datos por defecto.
El modo PIO usa muchos recursos de la CPU, debido a que cada byte de datos transferido entre el disco y la CPU debe ser enviado a través de los puertos IO (de Entrada/Salida). En algunos CPU, el modo PIO puede alcanzar velocidades de transferencia del orden de los 16 MB por segundo, pero provocaría que ningún otro proceso de la máquina obtenga tiempo de CPU.
Sin embargo, cuando la computadora está en el proceso de booteo no hay ningún otro proceso ejecutándose, por lo que el modo PIO sería una interfaz simple y excelente para usar durante este estado, hasta que el sistema pase al modo multitarea.

Limpieza de la caché (Cache Flush)
En algunas unidades es necesario vaciar “manualmente” la caché de escritura de hardware luego de cada comando de escritura. Esto se realiza enviando el comando 0xE7 al Registro de Comando (esperando que el BSY esté limpio). Si el driver no hace esto, puede provocar que los siguientes comandos de escritura fallen sin ser detectados, ó se creen “sectores defectuosos temporarios” en el disco.

Sectores defectuosos
A fines prácticos, hay tres tipos de sectores defectuosos en un disco ATA:

- Sectores que no pueden ser escritos (permanentemente).
- Sectores que no pueden ser leídos (permanentemente).
- Sectores que no pueden ser leídos (temporalmente).

Algunos fabricantes de discos tiene una función que permite tener una pequeña cantidad de sectores “de reemplazo” en el disco para ser reasignados en los sectores defectuosos permanentes. Sin embargo, esta función no es un estándar y depende del fabricante específicamente. En general, un SO/file system necesitará mantener una “lista de sectores defectuosos” por cada partición de la unidad, y trabajar alrededor de los sectores defectuosos.
También existen los “sectores defectuosos temporales”. Si estos sectores son leídos, se producirá un error de hardware, como en el caso de los sectores defectuosos permanentes. Pero si se escribe en estos sectores, la escritura podría realizarse perfectamente y el sector volvería a transformarse en un sector utilizable. Los sectores defectuosos temporales pueden suceder como resultado de cachés de escritura no volcados, picos de tensión ó fallas de energía.

Detección e Inicialización

Bus flotante

El último disco que fue seleccionado (por el BIOS, durante el booteo) se supone que debe mantener el control de los valores eléctricos en cada bus IDE. Si no hay ningún disco conectado al bus, los valores eléctricos en el bus se pondrán todos en alto “high” (a +5 volts). Una computadora puede leer esto como un byte 0xFF, y es la condición que se conoce como bus “flotante”. Es una muy buena manera para descubrir si no hay ningún dispositivo conectado al bus. Antes de enviar cualquier dato a los puertos de IO, se debe leer el byte de Estado Regular. El valor 0xFF es un valor de estado ilegal, e indica que el bus no tiene dispositivos. La razón para leer el puerto antes de cualquier escritura se debe a que la escritura puede causar fácilmente que los voltajes de los cables cambien a cualquier valor en un milisegundo (ya que no hay nada conectado a los cables para controlar el voltaje), y no es posible medir el “flotante”.
La medición del “flotante” es una manera rápida de detectar si no existen dispositivos. Pero la lectura de un valor distinto a 0xFF no es completamente definitiva. El test definitivo para detectar unidades es el comando #IDENTIFY (#IDENTIFY command).

Detección de los puertos de IO del controlador

Durante el booteo, los puertos de IO que son asignados al bus ATA se supone que deben localizarse en direcciones estándar. Si no están allí, la única manera de encontrarlas es enumerando las controladoras de disco en el bus PCI. Alternativamente, si los puertos están en las direcciones estándar, la “detección” no va a dar ninguna ganancia. Sin embargo, puede verse el código de detección del controlador y es bueno reconocerlo y saber para qué sirve. Si conocemos la localización de un set de puertos de IO del controlador ATA, y hay por lo menos un dispositivo en ese bus, entonces es posible detectar esos puertos IO. La mayoría de los puertos de IO de la controladora ATA son puertos de lectura/escritura. Esto significa que si se escribe un valor, por ejemplo, en el puerto de IO “SectorCount”, se supone que es posible volver a leer ese valor nuevamente, para saber cómo está seteado. Esta función de lectura/escritura es realizada por el dispositivo maestro en el bus, a menos que exista también un dispositivo esclavo y sea seleccionado. Si se escribe un valor en un puerto IO “inexistente”, es posible volver a leer ese valor del bus, si se lee inmediatamente. Por lo tanto, generalmente, la manera en que trabaja el software de detección del puerto de IO ATA es escribir un byte en un posible puerto IO ATA, escribir un byte diferente en otro puerto IO ATA y luego leer y verificar ambos valores escritos en esos dos puertos. Si ambos son verificados correctamente, los puertos IO elegidos son realmente puertos de lectura/escritura, y se los puede considerar como puertos de IO del controlador ATA. En el bus Primario, los puertos del 0x1F2 hasta el 0x1F5 deberían ser de lectura/escritura.

Detección estándar y no-estándar

Todos los BIOS actuales tiene estandarizado el uso del comando IDENTIFY para detectar la existencia de todos los tipos de dispositivos del bus ATA (PATA, PATAPI, SATAPI, SATA).
Hay dos técnicas no-estándar que no son recomendadas. La primera es seleccionar un dispositivo (efectuando los 400ns de delay) y luego leer el Registro de Estado de ese dispositivo. Para los dispositivos ATA que no están “durmiendo”, el bit RDY siempre estará seteado. Si no hay ningún dispositivo, el valor del Estado será 0. Pero este método no funciona para detección de dispositivos ATAPI, ya que su bit DRY está siempre limpio (hasta que obtienen su primer comando PACKET).

El otro método es usar el comando Ejecutar Diagnósticos de Dispositivo (Execute Device Diagnostics) (0x90), el cual setea bits en el Registro de Error (0x1F1 en el bus Primario) para mostrar la existencia de dispositivos maestros y esclavos en el bus.

Comando IDENTIFY (IDENTIFY command)
Para usar el comando IDENTIFY, se debe seleccionar la unidad destino enviando al puerto de IO “drive select” 0xA0 para la unidad maestra, ó 0xB0 para la esclava. En el bus Primario sería el puerto 0x1F6. Luego se deben setear los puertos de IO Sectorcount, LBAlo, LBAmid y LBAhi en 0 (desde 0x1F2 hasta 0x1F5). El siguiente paso es enviar el comando IDENTIFY (0xEC) al puerto de IO de Comando (0x1F7) y leer el puerto de Estados (0x1F7) otra vez. Si el valor leído es 0, el dispositivo no existe. Si es cualquier otro valor se debe hacer un poll del puerto de Estados (0x1F7) hasta el bit 7 (BSY, valor = 0x80) y ver si están limpios. Como algunos dispositivos ATAPI no siguen las especificaciones técnicas, se deben chequear también los puertos LBAmid y LBAhi (0x1F4 y 0x1F5) para comprobar que no haya ningún cero. Si esto ocurre, el dispositivo no es ATA y se debe detener el polling. De otra manera, se continua el polling de los puertos de Estado hasta que el bit 3 (DRQ, valor = 8) esté seteado, ó hasta que el bit 0 (ERR, valor = 1) esté seteado.
En este punto, si ERR está limpio, el dato está listo para leerse del puerto de Datos (0x1F0). Se leen 256 palabras y se almacenan.

Comando Abortar (Command Aborted)
Los dispositivos ATAPI ó SATA se supone que responden a un comando IDENTIFY ATA inmediatamente reportando un error en el Registro de Estados, en lugar de establecer el bit BSY, luego el DRQ y luego enviando las 256 palabras de los datos PIO. Esos dispositivos también escriben valores específicos en los puertos de IO, que pueden ser leídos. Viendo los valores específicos ATAPI en esos puertos luego de un IDENTIFY es la prueba definitiva de que ese dispositivo es ATAPI (en el bus Primario, el puerto de IO 0x1F4 debe leerse como 0x14, y el puerto de IO 0x1F5 debe leerse como 0xEB). Si un dispositivo ATA normal debe abortar un comando IDENTIFY, los valores en esos dos puertos deberán ser 0. Pero un dispositivo SATA podrá informar los valores 0x3C y 0xC3.
Sin embargo, algunas unidades ATAPI reales no setean el flag ERR luego de abortar un comando IDENTIFY ATA. Por lo tanto no dependen completamente del flag ERR luego de un IDENTIFY.

Información que devuelve IDENTIFY
- Word 0: es usado si el dispositivo no es un disco rígido.
- Word 83: el bit 10 es seteado si el dispositivo soporta el modo LBA48.
- Word 88: los bit en la parte baja del byte informan acerca de los modos UDMA soportados, el byte superior informa que modo UDMA está activo.
- Word 93 de un dispositivo maestro en el bus: el bit 12 es seteado si el dispositivo detecta un cable de 80 pines.
- Word 60 & 61 tomadas como una DWORD contiene el número total de sectores LBA de 28 bit direccionables en el dispositivo. (Si no es cero, el dispositivo soporta LBA28).
- Words 100 hasta 103 tomadas como un QWORD contiene el número total de sectores de 48 bit direccionables en el dispositivo.

Reseteo de un dispositivo / Reseteo de Software
Para dispositivos que no son ATAPI, la única manera que tiene el driver de resetear un dispositivo luego de ocurrido un error grave, es realizando un “reseteo de software” en el bus. Para realizar esto, se debe setear el bit 2 (SRST, valor = 4) en el Registro de Control adecuado para ese bus. Esto reseteará los dos dispositivos ATA del bus. Por último, hay que limpiar ese bit nuevamente. El dispositivo maestro del bus es automáticamente seleccionado. Los dispositivos ATAPI setean los valores en sus puertos IO LBA_LOW y LBA_HIGH, pero no se utiliza para resetar o incluso terminar la ejecución del comando actual.

Hardware
Las especificaciones de disco ATA están desarrolladas en base a unas especificaciones antiguas llamadas ST506. Con ST506, cada driver de disco era conectado a una controladora mediante dos cables, un cable de datos y uno de comandos. La controladora se conectaba al bus del motherboard. La CPU se comunicaba con la controladora a través de los puertos de IO, que estaban conectados directamente al bus del motherboard.
Lo que hizo la especificación IDE original fue separar la controladora del motherboard y agregar un controlador a cada unidad de disco permanentemente. Cuando la CPU accedía al puerto de IO del disco, había un chip que hacía un cortocircuito directo de los pines del bus de IO de la CPU al cable IDE, de manera que la CPU podía acceder directamente a la controladora del driver. Los mecanismos de transferencia de datos entre la CPU y la controladora siguieron siendo los mismos, y ahora se llamaron modo PIO. Hoy en día, los chips de la controladora de disco solamente copian las señales eléctricas entre el bus del puerto de IO y el cable IDE, hasta que el driver entra en otro modo distinto del PIO.

Características de conexión de los discos ATA
Existe un cable que permite seleccionar que driver está activo en cada bus. Se realiza de forma eléctrica mediante la posición “high” ó “low”, lo que significa que nunca habrá más de dos dispositivos operando en cualquier bus ATA. Se los denomina dispositivos master (maestro) y slave (esclavo) y sus funcionalidades son completamente idénticas. Hay un bit en el puerto de IO que le permite a un driver seleccionar el dispositivo de destino para cada byte de comando.
La mayoría de los chips de las controladoras de disco actuales soportan dos buses ATA por chip, y hay un set de puertos de IO estandarizados para el control de los discos sobre cada bus. Los primeros dos buses se los llama bus ATA Primario y Secundario, y son controlados por los puertos IO del 0x1F0 hasta el 0x1F7, y del 0x170 hasta el 0x177 respectivamente. Los puertos de los Registros de Control de Dispositivo/Estado Alternativo (Device Control Registers/Alternate Status) asociados son 0x3F6 y 0x376 respectivamente. La IRQ estándar para el bus Primario es la IRQ14 y para el bus Secundario es la IRQ15.
Si los siguientes dos buses existen, normalmente son controlados por los puertos de IO del 0x1E8 hasta el 0x1EF y del 0x168 hasta el 0x16F respectivamente. Los puertos del Registro de Control de Dispositivo/Registro Alternativo son los puertos de IO 0x3E6 y 0x366.

Los registros de control actuales y las IRQ para cada bus pueden determinarse enumerando los buses PCI, encontrando todas las controladoras de disco y leyendo la información de cada espacio de configuración de la controladora PCI. Por lo tanto, técnicamente, la enumeración PCI debe hacerse antes de la detección de dispositivos ATA, pero sin embargo, este método no es totalmente confiable.
Cuando el sistema bootea, de acuerdo a las especificaciones, la controladora de disco PCI se supone que está en el modo “Legado/Compatibilidad” (Legacy/Compatibility). Esto significa que usa la configuración de los puertos de IO estándar. No hay otra opción más que depender de este hecho.

Tiempo de espera para enviar y recibir comandos
El método sugerido en las especificaciones de ATA para enviar un comando ATA nos dice que tenemos que chequear los bits BSY y DRQ antes de intentar enviar un comando. Esto significa que es necesario leer el Registro de Estados (Status Register) (El Registro Alternativo “Alternate Status” es una buena opción) del dispositivo adecuando antes de enviar el siguiente comando. Lo que implica que primero es necesario seleccionar el dispositivo correcto, antes de poder leer el estado (y luego enviar todos los otros valores a los puertos de IO de otro tipo). Es decir, antes de leer el estado, es necesario realizar una selección del dispositivo. Esto es un problema. Muchos dispositivos requieren un pequeño tiempo para responder a la selección y enviar su estado a través del bus. La sugerencia es leer el Registro de Estados cinco veces, y sólo prestar atención a los valores devueltos en la última lectura, luego de seleccionar un nuevo dispositivo maestro o esclavo. Se puede asumir que la lectura de un puerto de IO tarda aproximadamente 100 ns, por lo que con las primeras cuatro lecturas se crea un retraso de 400 ns, lo que le otorga al dispositivo el tiempo para colocar los voltajes correctos en el bus.
Leer los puertos de IO para crear retrasos desperdicia muchos ciclos de CPU. Por lo tanto, es más inteligente hacer que el driver recuerde el último valor enviado a cada puerto de IO para Selección del Dispositivo, evitando realizar selecciones de dispositivos, si el valor no cambia. Si no se envía una selección de dispositivo, lo único que hay que hacer es leer el Registro de Estados una vez.
El driver siempre necesita bloquear si el dispositivo actual está activo, modificando BSY/DRQ/ERR, y el driver del dispositivo siempre sabe que el dispositivo está en esa condición (porque el driver envió un comando al dispositivo, y éste no ha sido marcado como “completo” todavía). Una vez que la unidad completó el comando, se limpian los bits BSY y DRQ. Esto puede ser fácilmente verificable, antes de enviar el siguiente comando de selección de dispositivo, que el dispositivo seleccionado anteriormente limpió los bits BSY y DRQ correctamente al finalizar el comando. Nunca se tiene que verificar si estos bits están limpios luego de una selección de dispositivo, por lo que no se tiene que leer el Registro de Estados luego de la selección de un dispositivo.
Hay un problema similar luego de escribir en el Registro de Comando con los bits ERR/DF. Se trata de dos tipos ligeramente diferentes de errores que pueden finalizar un comando. BSY y DRQ deben ser limpiados, pero ERR ó DF deben permanecer seteados hasta que se escriba un nuevo comando en el Registro de Comando. Si se usa polling, puede ocurrir que las primeras cuatro lecturas del Registro de Estados, luego de enviar un byte de comando, tengan los bits de ERR ó DF seteados accidentalmente. (Si se usan IRQs, el Estado siempre permanecerá en forma correcta mientras se estén utilizando los servicios IRQ).

Registros usados en el bus ATA

Un bus ATA generalmente tiene 9 puertos de I/O que controlan su comportamiento. Para el bus Primario, estos puertos van del 0x1F0 hasta el 0x1F7, y el 0x3F6. Los valores en la tabla 1 son relativas a las denominadas direcciones base del puerto de I/O. Por lo tanto el valor 1 en un puerto en realidad significa 0x1F0 + 1 = 0x1F1. Esto se hace así porque la dirección base puede variar dependiendo del hardware.

Desplazamiento del Puerto	***Función***	***Descripción***
0	Data Port	Los bytes de datos son Leídos/Escritos en este puerto.
1	Features / Error Information	Usado en general por los dispositivos ATAPI
2	Sector Count	Número de sectores a leer/escribir.
3	Sector Number / LBAlo	Es específico de CHS / LBA28 / LBA48.
4	Cylinder Low / LBAmid	Dirección parcial de un sector de disco.
5	Cylinder High / LBAhi	Dirección parcial de un sector de disco.
6	Drive / Head Port	Se usa para seleccionar un dispositivo y/o una cabeza. También puede soportar bits extra de dirección/flag.
7	Command Port / Regular Status Port	Usado para enviar comandos ó leer el estado actual.

Tabla 1: Puertos de I/O

Byte de Estado (Status Byte)

Técnicamente, cuando el bit BSY es seteado, los otros bits en el byte de Estado no tienen sentido. También es una mala idea verificar el bit de “Búsqueda Completa” (Seek Complete – DSC), porque puede haber quedado obsoleto y haber sido reutilizado para otro propósito.

Bit	**Abreviatura**	**Función**
0	ERR	Indica si ocurrió un error. Se envía un nuevo comando para limpiarlo.
3	DRQ	Es seteado cuando el dispositivo tiene como modo de transferencia el modo PIO ó si está listo para aceptar datos.
4	SRV	Solicitud de servicio en modo solapado.
5	DF	Error de defecto de dispositivo (no setea el ERR).
6	RDY	Este bit está limpio cuando el dispositivo está quieto ó luego de un error. De otra manera está seteado.
7	BSY	Indica que el dispositivo está preparado para enviar/recibir datos (se debe esperar que esté “clear”). En caso de que nunca esté en el estado “clear”, se debe realizar un reseteo por software.

Tabla 2: Byte de Estado

Registro de Control de Dispositivo/Estado Alternativo
Hay un puerto de IO adicional que cambia el comportamiento de cada bus ATA, y se lo denomina Registro de Control de Dispositivo (en el bus Primario es el puerto 0x3F6). Cada bus ATA tiene su propio Registro de Control. No es posible leer este Registro de Control. En vez de eso, la lectura del puerto nos devuelve el valor del Registro de Estado Alternativo. Este valor de Estado Alternativo es el mismo que el del puerto de Estado Regular (0x1F7 en el bus Primario), pero la lectura del puerto de Estado Alternativo no afecta a las interrupciones.
Definiciones del bit de Registro de Control:

Bit	Abreviatura	Función
1	nIEN	Si este bit es seteado, se impide al dispositivo actual seguir enviando interrupciones.
2	SRST	Si este bit es seteado, se realiza un “Reseteo de Software” sobre todos los dispositivos ATA en el bus, si uno está funcionando mal.
7	HOB	Si este bit es seteado, se lee nuevamente el Byte de Orden Alto del último valor LBA48 enviado al puerto de IO.

Tabla 3: Registro de Control

Interrupciones de los dispositivos ATA (IRQs)
Al principio, la única intención de una IRQ era informar al manejador de IRQ que el dispositivo estaba listo para enviar ó aceptar datos. El manejador de IRQ sería el encargado de llevar a cabo la transferencia de datos basadas en PIO del siguiente bloque de datos, inmediatamente. Pero hoy en día el proceso es más complejo. Uno ó ambos dispositivos del bus pueden estar utilizando el modo DMA, ó tener un tamaño del bloque de datos distinto a 256 palabras. Además, se hace mucho énfasis actualmente en volver lo más rápidamente posible de la rutina del manejador de IRQ.
Si se usa IRQ compartida, es necesario chequear el byte de Estado del Busmaster PCI, para verificar que la IRQ viene del disco. Si es así, el siguiente paso es leer es Registro de Estado Regular una vez, para limpiar el flag de interrupción del disco. Si el bit ERR en el Registro de Estado esta seteado (activado) (bit 0 valor = 1), se puede leer y guardar el valor de los “detalles del error” desde el puerto de IO de Error (0x1F1 en el bus Primario).
Si la transferencia fuera una operación de lectura DMA, se debe leer el valor del Registro de Estado del Busmaster. Como el manejador de IRQ no reconoce si la operación fue una operación DMA ó no, es necesario comprobar el byte de Estado del Busmaster luego de todas las IRQs. Si este byte tiene su bit ERR seteado (bit 1, valor = 2), se puede guardar los valores actuales en los puertos de IO LBA del disco, ya que pueden decirnos qué sector del dispositivo generó el error. También es necesario limpiar el bit de error, escribiéndole un 2.
También se debe enviar un EOI (0x20) a ambos PIC’s para limpiar sus flag de interrupción. Luego hay que setear un flag para “desactivar” el driver e informarle que otra IRQ ha ocurrido, para que el driver pueda hacer cualquier transferencia de datos necesaria.

Polling del Estado vs. IRQs

Cuando un driver emite un comando PIO de lectura ó escritura, éste necesita esperar hasta que el dispositivo esté listo para la transferencia de datos. Hay dos maneras de saber cuando un dispositivo está listo para los datos. El dispositivo enviará una IRQ cuando está preparado ó el driver puede hacer un polling de uno de los puertos de Estado (ya sea el Estado Regular ó el Alternativo).
Hay dos ventajas del polling y una gran desventaja. Las ventajas son que responde más rápidamente que una IRQ. La lógica del polling es mucho más simple que la espera en una IRQ.
La desventaja es que en un ambiente multitarea, se utiliza mucho tiempo de CPU.

Si un driver siempre lee el puerto de Estado Regular luego de enviar un comando al dispositivo, la “respuesta” IRQ puede no suceder. Si se quieren recibir IRQs, se debe leer el puerto de Estado Alternativo en vez del puerto de Estado Regular. Pero algunas veces puede ocurrir que las IRQs sean un desperdicio, y sea una buena idea evitarlas.
Una manera mucho más completa para prevenir que sucedan IRQs es setear el bit nIEN en el Registro de Control de un dispositivo seleccionado. Esto evita que el dispositivo envíe IRQs por el bus, hasta que se limpie nuevamente ese bit. Pero algunas veces, puede no funcionar, ya que los dispositivos responden sólo a los valores escritos más recientemente de nIEN cuando son los dispositivos seleccionados en el bus. Es decir, si un dispositivo es seleccionado y se setea el nIEN, y luego se selecciona otro dispositivo a través del Registro de Selección de Dispositivo, se vuelve a limpiar el nIEN, por lo que el primer dispositivo debería recordar que tenía seteado ese bit.

Una manera de reducir el número de IRQs en el modo PIO multitarea es usar los comandos de múltiple lectura (0xC4) y múltiple escritura (0xC5). Este comando hace que en el buffer del dispositivo se coloquen “bloques” de sectores, y sólo se envía una IRQ por bloque, en vez de una IRQ por sector. El comando SET MULTIPLE MODE (0xC6) permite cambiar los sectores por bloque.

Estándares ATA

German Maly, Rodrigo German Molina, Matias Salica, Pablo Sebastian Torres, Cristian Jorge Traverso
Universidad Nacional de La Matanza

Abstract
El propósito de este documento es explicar los diferentes estándares ATA, indicando en cada uno, sus características, registros involucrados, comandos utilizados y las nuevas funcionalidades que agrega.

Palabras Clave
ATA, ATAPI, ANSI, Interfaz, Registros.

Desarrollo

ATA 1:

El primer estándar ATA fue reconocido por el ANSI en 1999. También es conocido bajo el nombre de IDE[1].
El estándar buscaba definir la interfaz entre los discos rígidos que poseían la controladora integrada dentro de ellos y se conectaban mediante el bus ATA. El bajo costo y tamaño de esta interfaz la convirtieron en un estándar de facto, el cual sería reglamentado con la publicación del estándar ATA.
El estándar provee la capacidad de utilizar el Bus AT mediante una “Daisy chain” conectada a dos discos, el disco 0 (maestro) y el disco 1 (esclavo). La designación se realiza mediante un interruptor en el disco, un jumper o un pin selector en el cable.
La información es transferida en forma paralela (de a 8 o 16 bits) entre la memoria del sistema Host y el buffer del disco. La operación es realizada por el disco tras haber recibido un comando de control por parte del sistema Host.
Implementación física:

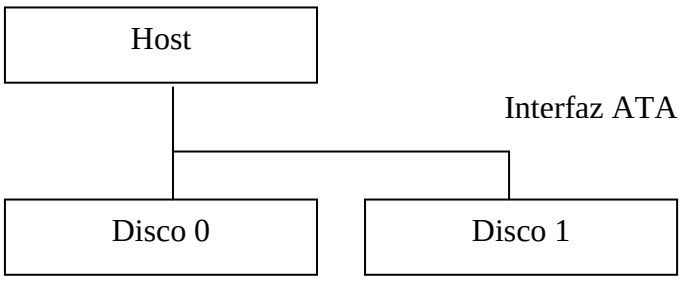

Ilustración 1: Interfaz ATA para periféricos integrados

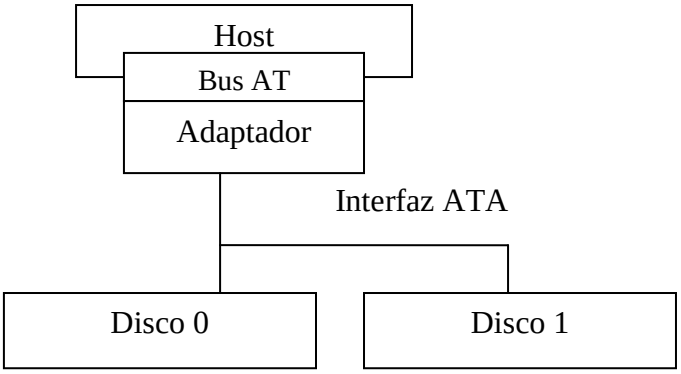

Ilustración 2: Adaptador de bus del Host y dispositivos periféricos

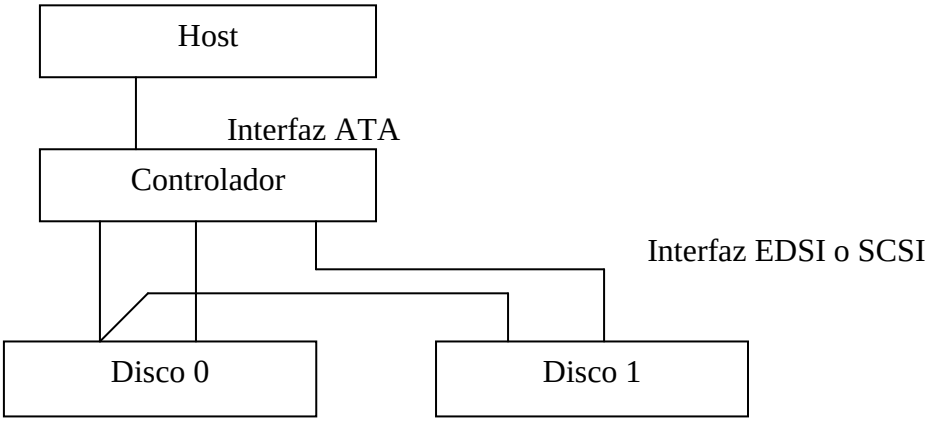

Ilustración 3: Interfaz ATA a controlador y dispositivos periféricos

Los comandos enviados a la interfaz son recibidos por ambos discos conectados a la Daisy chain, y para elegir el disco destino se utiliza el bit DRV. Si el DRV es 0, se selecciona el disco maestro, si el DRV es 1, se selecciona el disco esclavo.

Señales

Lo siguiente es una descripción de las señales transmitidas por el conector de 40-pins.
El (+) indica que la señal se activa con un nivel alto de voltaje, (-) indica nivel bajo. Por defecto, es nivel alto. Un (*) indica señales de intercomunicación entre discos.

Signal	Pin	I/O	
CS1FX-	37	I	Chip select 0 de disco
CS3FX-	38	I	Chip select 1 de disco
DA0	35	I	Bus de direcciones del disco - bit 0
DA1	33	I	- bit 1
DA2	36	I	- bit 2
DASP-	39	I/O	Disco activo/disco 1 presente
DD0	17	I/O	Bus de datos del disco - bit 0
DD1	15	I/O	- bit 1
DD2	13	I/O	- bit 2
DD3	11	I/O	- bit 3
DD4	9	I/O	- bit 4
DD5	7	I/O	- bit 5
DD6	5	I/O	- bit 6
DD7	3	I/O	- bit 7
DD8	4	I/O	- bit 8
DD9	6	I/O	- bit 9
DD10	8	I/O	- bit 10
DD11	10	I/O	- bit 11
DD12	12	I/O	- bit 12
DD13	14	I/O	- bit 13
DD14	16	I/O	- bit 14
DD15	18	I/O	- bit 15
DIOR-	25	I	Leer del disco I/O
DIOW-	23	I	Escribir en disco I/O
DMACK-	29	I	DMA aceptado
DMARQ	21	O	Petición DMA
INTRQ	31	O	Interrupción de disco
IOCS16-	32	O	16 bits I/O del disco
IORDY	27	O	Canal I/O listo
PDIAG-	34	I/O	Diagnostico exitoso
RESET-	1	I	Reset de disco
SPSYNC:	28	-	Spindle sync
CSEL	28	-	Selector del cable
Pin llave	20	-	Evita la mala conexión.

CS1FX- (Chip select 0 de disco)
Señal de selección de chip para el los registros de bloques de comando.

CS3FX- (Chip select 1 de disco)
Señal de selección de chip para el los registros de bloques de control.

DA0-2 (Bus de direcciones del disco - bit 0)
Dirección basada en 3 bits, emitida por el Host para acceder a un registro o puerto de datos en el disco.

DASP- (Disco activo/disco 1 presente)
Señal con multiplexación de tiempo que indica que el disco esta activo o que el disco 1 esta presente.

DD0-DD15 (Bus de datos del disco)
Bus de datos de 8 o 16 bits bidireccional entre Host y disco. Los 8 bits más bajos son usados para transferencias de 8 bits, por ejemplo para los registros o si es soportado por el disco, transferencia de datos de solo 8 bits.

DIOR- (Leer del disco I/O)
Señal de lectura de disco. Habilita la transferencia de datos desde el disco hacia el bus del Host, desde DD0-DD7 o DD0-DD15.

DIOW- (Escribir en disco I/O)
Señal de escritura en disco. Habilita la transferencia de datos desde el Host al disco mediante, DD0-DD7 o DD0-DD15.

DMACK- (DMA aceptado) (Opcional)
Señal en respuesta a un DMARQ, ya sea para aceptar DMA o para especificar disponibilidad por DMA.

DMARQ (Pedido DMA) (Opcional)
Señal de pedido de DMA cuando el disco esta listo para transferir datos desde o hacia el Host.
La dirección de los datos es controlada por DIOR y por DIOW. Utilizada con handshake con DMACK.
Cuando se activa operación mediante DMA, IOCS16-, CS1FX-, y CS3FX- no serán aceptados y la transferencia será de 16 bits completos.

INTRQ (Interrupción de disco)
Señal utilizada para interrumpir al sistema del Host.
En transferencias por PIO, esta señal es verificada al principio de cada bloque de datos a transferir, típicamente un solo sector, excepto cuando se declara otra cosa con el comando Set Multiple. Ocurre una excepción en los comandos Formatear Pista, Escribir Sector(es), Escribir en Buffer y Escritura Larga.
En transferencias DMA, esta señal es verificada solo una vez, luego de que el comando se termina

IOCS16- (16 bits I/O del disco)
Exceptuando DMA, IOCS16- indica al Host, que el disco esta listo para recibir o enviar palabras de 16 bits.

IORDY (Canal I/O listo) (Opcional)
Disco listo para transmisión de datos

PDIAG- (Diagnostico exitoso)
Señal desde el disco 1 al 0 para indicar que paso el diagnóstico. De acuerdo al retardo o ausencia de respuesta por parte de alguno de los discos, en señales como RESET, o del mismo diagnóstico, se activan bits de error en los registros.

SPSYNC:CSEL (Sincronización Spindle /Selector de cable) (Opcional)
Señal de doble propósito, con activación de una de las dos opciones o ambas. Si son ambas, no pueden ser concurrentes.

SPSYNC (Sincronización Spindle) (Opcional)

Si el disco es Maestro, la señal es de salida, si es esclavo es de entrada. El maestro o el Host generan SPSYNC y la transmiten. El esclavo no la genera, y es responsable de sincronizar su índice con el SPSYNC.

CSEL (Selector de cable) (Opcional)

El disco es configurado como Disco 0 o Disco 1 de acuerdo al valor de CSEL. Si CSEL esta abierto, el disco es 1, sino es 0.

El disco podía trabajar utilizando tanto el modo de direccionamiento CHS (Cylinder Head Sector) como LBA (Logical Block Addressing). Para LBA se toma la siguiente consideración:

LBA 0 = Cilindro 0 / Cabeza 0 / Sector 1.

Y la siguiente equivalencia:

LBA = [(Cilindro * número de cabezas + cabeza) * sector / pista] + (Sector – 1)

Los registros que poseen los discos ATA se diferencian en:
Registros de comando (Command Block Registers) que son usados para enviar comandos al disco o para leer su estado.
Registros de control (Control Block Registers) usados para el control del disco y para leer información alternativa de los estados del mismo.

Registros:

Alterno de estado (*de control de bloque*)
Contiene la misma información que el registro de estado del bloque de comandos, solo que leer este registro no implica interrupción.

De comando (*de comando de bloque*)
Contiene el comando enviado al disco. La ejecución del mismo comienza luego de la escritura de este registro.

Parte alta del cilindro (*de comando de bloque*)
Contiene la parte alta de los bits de dirección del cilindro. Si es LBA contiene los bits 16 a 23.

Parte baja del cilindro (*de comando de bloque*)
Contiene la parte baja de los bits de dirección del cilindro. Si es LBA contiene los bits 8 a 15.

De dato (*de comando de bloque*)
Registro de 16 bits para transferir información desde el buffer del dispositivo al Host

De control de dispositivo (*de control de bloque*)
Contiene bit de RESET y bit de activación de interrupción del disco al Host.

De dirección del disco (*de control de bloque*)
Contiene la dirección invertida del disco seleccionado y la cabeza seleccionada.

Disco/cabeza (*de comando de bloque*)
Contiene el número de disco y de cabeza.

De Error (*de comando de bloque*)
Estado del último comando ejecutado por el disco o un código de diagnostico.

De Atributos (*de comando de bloque*)
Usado para activar o desactivar atributos de la interfaz.

De contador de sectores (*de comando de bloque*)
Contiene el número de sectores de datos pedidos para transferir en una escritura o lectura.

De número de sector (*de comando de bloque*)
Contiene el sector inicial para cualquier acceso a disco para el comando que le sigue. En LBA este registro contiene los bits del 0 al 7.

De estado (*de comando de bloque*)
Especifica el estado del disco

Comandos
Se envían al disco cargando en los registros correspondientes en el bloque de comando con los parámetros necesarios. Luego se escribe el código del comando en el Registro De Comando, explicado anteriormente.
Los comandos son de implementación opcional u obligatoria.

Confirmación de cambio en el soporte (opcional)
Si el disco esta trabajando en un modo en el cuál el S.O debe confirmar un cambio en el soporte (de almacenamiento), este comando permite que se continúe con el flujo normal de operación.

Boot – post boot (opcional)
Medio para enviar información especifica del fabricante del disco.

Boot – pre boot(opcional)
Prepara un disco removible para responder al boot.

Comprobar modo de alimentación de energía
Comprueba el modo de alimentación de energía.

Cerrar puerta (opcional)
Cierra la puerta del flujo de datos si el disco esta Listo y destrabado, sino responde con un estado de No Listo

Destrabar puerta (opcional)
Destraba la puerta del flujo de datos si el disco esta Listo y trabado, sino responde con un estado de No Listo

Ejecutar diagnostico de disco
Como el nombre lo dice, ejecuta el diagnostico del disco.

Formatear pista

Este comando es relativo al fabricante del disco. Puede ser un formateo físico, inicializar el campo de datos con algún valor o no hacer nada.

Identificar disco

Permite al Host recibir información del disco. Lo siguiente, es la información que envía este comando (anteriormente se explicó este comando en la sección del funcionamiento de ATA PIO y, en general, la información que se devolvía)

Palabra	
0	Configuración general de la información por bit: 15 0 reservado para discos no magneticos 14 1=Tolerancia gap de la velocidad de formateo requerida 13 1=Opción de offset para la pista disponible 12 1= Opción de offset para los datos disponible 11 1=Velocidad de rotación tolerada es > 0,5% 10 1=Tasa de transferencia del disco> 10 Mbs 9 1= Tasa de transferencia del disco > 5Mbs pero <= 10Mbs 8 1= Tasa de transferencia del disco <= 5Mbs 7 1=disco removible 6 1=disco arreglado 5 1=Control del motor del eje implementado 4 1=Tiempo del cambio de cabeza > 15 usec 3 1=MFM sin codificar 2 1=Software sectorizado 1 1=Hardware sectorizado 0 0=Reservado
1	Numero de cilindros
2	Reservado
3	Numero de cabezas
4	Numero de bytes sin formatear por pista
5	Numero de bytes sin formatear por sector
6	Numero de sectores por pista
7-9	Específico del fabricante
10-19	Numero de serie (20 caracteres ASCII, 0000h=sin especificar)
20	Tipo de buffer
21	Tamaño del buffer en incrementos de 512 bytes (0000h= sin especificar)
22	# de ECC bytes disponibles en largos comandos leer/escribir (0000h= sin especificar)
23-26	Revisión del firmware (8 caracteres ASCII, 0000h= sin especificar)
27-46	Numero de modelo (40 caracteres ASCII, 0000h= sin especificar)
47	15-8 Especifico del fabricante 7-0 00h = Múltiples comandos de lectura/escritura sin implementar

	xxh = Numero máximo de sectores que se pueden transmitir por interrupción en comandos de múltiples lecturas/escrituras
48	0000h = No puede hacer palabra doble de I/O 0001h = Puede hacer palabra doble de I/O
49	Capacidades 15-10 0=Reservada 9 1=Soporta LBA 8 1=Soporta DMA 7- 0 Específico del fabricante
50	Reservada
51	15-8 Modo de transmisión de datos por PIO en ciclos por tiempo 7-0 Específico del vendedor
52	15-8 Modo de transmisión de datos por DMAN en ciclos por tiempo 7-0 Específico del vendedor
53	15-1 Reservado 0 1=los campos en las palabras 54-58 son válidos 0=los campos en las palabras 54-58 puede que sean válidos
54	Numero de cilindros actuales
55	Numero de cabezas actuales
56	Numero de sectores por pista actuales
57-58	Capacidad actual en sectores
59	15-9 Reservada 8 1 = Configuración de múltiples sectores es válida 7-0 xxh = Configuración actual de numero de sectores que pueden ser transferidos por interrupciones de comandos de múltiples R/W
60-61	Total de sectores direccionables del usuario (solo LBA)
62	15-8 Transmisión DMA por palabra simple activada 7-0 Modos de transmisión DMA por palabra simple soportados
63	15-8 Transmisión DMA por palabra múltiple activada 7-0 Modos de transmisión DMA por palabra múltiple soportados
64-127	Reservada
128-159	Específico del fabricante
160-255	Reservada

No hacer nada / funcionar en vacío (Idle)

Hace que el disco entre en estado Ocupado, establece el modo en Idle, limpia el estado Ocupado y genera una interrupción. Espera si se da el caso de estar ejecutándose otras acciones

No hacer nada / funcionar en vacío (Idle) inmediato

Hace que el disco entre en estado Ocupado, establece el modo en Idle, limpia el estado Ocupado y genera una interrupción. A diferencia del anterior, este es inmediato.

Inicializar parámetros del disco

Da la posibilidad al Host de inicializar el número de sectores por cabeza y el número de cabezas en -1, por cilindro.

NOP

Permite Hosts que solo pueden realizar accesos de 16 bits a los registros para verificar el estado del disco

Lectura de buffer

Permite al Host leer el contenido actual del buffer del disco.

Lectura por DMA

Funciona igual que la Lectura de Sector(es) solo que la transmisión es por DMA

Lectura larga

Similar a Lectura de Sector(es) solo que además de los datos que se leen, soporta la lectura de los bytes ECC (códigos de paridad) directamente del buffer, y el disco no los tiene que generar por si mismo.

Múltiples lecturas

Funciona igual que Lectura de Sector(es), solo que no hay una interrupción cuando se especifican los sectores a leer, permitiendo leer varios sectores en una oportunidad.

Lectura de Sector(es)

Lee desde 1 a 256 sectores especificados el registro de contador de sectores.

Verificar la lectura de Sectores(es)

Idéntico al anterior, solo que realiza una verificación de los sectores deseados.

Recalibrar

Muevo la cabeza de R/W desde cualquier lugar del disco hasta el cilindro 0.

Buscar

Inicia una búsqueda de la pista y selecciona la cabeza correspondiente en el bloque de comando.

Establecer propiedades

Permite al Host establecer ciertos parámetros que afectan a determinadas propiedades del disco y por lo tanto su ejecución, como ser activar/desactivar palabras simples, múltiples, modos de transferencias, escrituras en cache, y comandos específicos del fabricante

Establecer modo múltiple

Permite realizar operaciones de R/W múltiples.

Dormir

Es la única manera de hacer que el disco se encuentre en Modo de Dormir. El disco reduce su velocidad de giro hasta frenar, y la interfaz con el mismo se vuelva inactiva.

La única manera de recuperarse de este estado es por un RESET por software o hardware (anteriormente explicado en la sección del funcionamiento de ATA PIO).

Esperar
Hace que el disco entre en estado de espera. Espera si se da el caso de estar ejecutándose otras acciones

Esperar (inmediato)
Hace que el disco entre en estado de espera. A diferencia del anterior, este es inmediato

Escribir en buffer
Permite al Host sobrescribir el contenido del buffer del disco.

Escribir por DMA
Funciona igual que Escritura de Sector(es) solo que la transmisión es por DMA.

Escritura larga
Similar a Escritura de Sector(es) solo que además de los datos escribe los bytes ECC directamente del buffer, y el disco no los tiene que generar por si mismo.

Múltiples escrituras
Funciona igual que Escritura de Sector(es), solo que no hay una interrupción cuando se especifican los sectores a escribir, permitiendo escribir varios sectores en una oportunidad.

Escribir lo mismo
Funciona igual que Escritura de Sector(es), solo que el contenido es escrito de una a mucha veces.

Escritura de Sector(es)
Escribe desde 1 a 256 sectores según especificados en el registro de contadores de sectores.

Verificar la escritura de Sectores(es)
Idéntico al anterior, solo que realiza una verificación de los sectores escritos inmediatamente después de realizar la operación.

ATA-2
También conocido como EIDE, introdujo:

- Los modos PIO (Entrada/Salida Programada) 3 y 4 y el modo DMA de palabra múltiple (Multiword DMA) 1 y 2.
- Permitían transferir datos a una velocidad de 16 MB/s.

ATA-3
Introdujo el sistema de seguridad SMART (Tecnología Automática de Monitoreo, Análisis e Informe) que permitía mejorar la confiabilidad y prevenir posibles fallas.

Este estándar no introduce ningún modo nuevo, pero es compatible con los modos PIO 0, 1, 2 y 3 y los modos DMA 0, 1 y 2.

ATAPI

El estándar ATAPI fue desarrollado por un grupo de compañías de CD-ROMs con la ayuda de Western Digital y Oak Technology.
ATAPI (Paquete de Interfaz ATA) es una extensión del estándar ATA, que permite la interconexión de otros periféricos de almacenamiento, tales como unidades de CD o DVD, en una interfaz ATA.

ATA/ATAPI – 4

Introdujo:

- Comandos ATAPI y de reseteo nuevos.
- Un nuevo protocolo de transferencia de datos Ultra DMA (33 MB/s).
- Data integrity (utiliza comprobación de CRC).

ATA/ATAPI – 5

Con este estándar se definieron dos modos nuevos de transferencia, los modos Ultra DMA 3 y 4, pero este último también se lo denomina Ultra ATA/66 ó Ultra DMA/66.

ATA/ATAPI – 6[2]

Este estándar define el modo Ultra DMA/100, que también se lo denomina Ultra DMA modo 5 ó Ultra-ATA 100.
Aumenta el direccionamiento de bloques lógicos (LBA) de 28 a 48 bits, denominado LBA48 (Dirección Lógica de Bloques de 48 bits).
También se incrementó el Sector Count de 8 a 16 bits.
Además agrega una funcionalidad nueva llamada Gestión Acústica Automática (AAM) que le permite a las unidades ajustar automáticamente las velocidades de acceso con el objetivo de reducir el ruido operativo.

Serial Ata

SATA fue desarrollado mientras se trabajaba en ATA/ATAPI-6 por un grupo de compañías lideradas por Intel.
SATA está constituida como una estructura de capas. Consta de una capa de órdenes que es un superconjunto de la arquitectura ATA anterior, permitiendo que los nuevos dispositivos sean compatibles con los protocolos ATA tradicionales. La diferencia con la arquitectura ATA anterior se encuentra en la capa física, provocando que los nuevos dispositivos no sean físicamente compatibles con los anteriores.

ATA/ATAPI-7[3]

Define el modo Ultra DMA 6 que también es conocido con el nombre de Ultra DMA/133 ó Ultra-ATA 133.
Agrega nuevas funciones que permiten el uso de grabadoras de video digital.

ATA/ATAPI-8[4]

Agrega soporte para GPL y SCT (Comando de Transporte SMART). Realiza una especificación sobre el modo Escritura-Lectura-Verificación (como dice el nombre, se verifican los datos luego del giro del disco inmediatamente después de estas dos operaciones). Consecuentemente, agrega un campo llamado Dispositivo de Identificación para dar soporte a este modo, donde se reporta el estado del mismo.
Una de las características más sobresalientes es la inclusión del soporte a los Discos Híbridos, es decir aquellos discos que integran un cache utilizando alguna memoria de tipo no volátil. Es importante esta nueva característica ya que estos discos permiten la mejora en transmisiones a alta velocidad para archivos críticos del sistema operativo.

Referencias:

[1] AT Attachment Interface for Disk Drives, Draft 791D, Revision 4c

[2]AT Attachment with Packet Interface – 6 (ATA/ATAPI-6)

[3] AT Attachment with Packet Interface -7

[4]AT Attachment with Packet Interface -8

Descripción de la interfaz y características SATA

German Maly, Rodrigo German Molina, Matias Salica, Pablo Sebastian Torres, Cristian Jorge Traverso
Universidad Nacional de La Matanza

Abstract
El propósito principal de este documento es mostrar las distintas características del estándar SATA, la forma de conexión de los dispositivos preparados para este estándar, su compatibilidad con el estándar ATA y los diferentes avances y mejoras que se fueron desarrollando, es decir, SATA II y SATA 3.0.

Palabras Clave:
SATA, Parallel ATA, AHCI, NCQ, SATA 1.0, SATA 2.0, SATA 3.0.

Introducción

El estándar Serial ATA (S-ATA ó SATA)[1] es una interfaz de transferencia de datos que permite conectar distintos tipos de periféricos. Se creó para reemplazar el estándar Parallel ATA [2] ofreciendo muchas ventajas respecto de este, entre las cuales se encuentran: un reducido volumen de cables y costo, ya que ATA utiliza un conector de 40 conductores y SATA utiliza sólo 7 conductores, un aumento de la velocidad en la transferencia de datos producido por mayores tasas de señalización y una mayor eficiencia en la transferencia a través de los protocolos de I/O.

Desarrollo

El estándar Serial ATA se basa en una comunicación serie, utilizándose una ruta de datos para transmitir los datos y otra ruta para las confirmaciones de recepción. Los datos se transmiten mediante el modo de transmisión LDVS (Señal diferencial de bajo voltaje) que consiste en transmitir una señal a un conductor y su contrapartida a un segundo conductor para permitir que el destinatario recree la señal por diferencia. Los datos de control y los datos son transmitidos por la misma ruta, pero se los distingue mediante una secuencia específica de bits.

Por esta razón, la comunicación requiere dos rutas de transmisión, compuesta cada una por dos conductores, utilizando en total cuatro conductores para la transmisión.
SATA utiliza una comunicación serie, transmitiendo un bit a la vez, a diferencia de ATA/IDE que utiliza comunicación paralela transmitiendo de a un byte por vez.
La comunicación paralela es en sí más rápida que la serie, pero sólo a distancias muy cortas. Si las distancias aumentan, las señales que viajan en los cables paralelos se pueden desincronizar o distorsionar a causa del ruido electromagnético. El ruido no afecta a SATA, y al no tener problemas de sincronización de varios bits, puede permitirse alcanzar mayores velocidades de transferencia.

Compatibilidad con Parallel ATA

Para asegurar la compatibilidad con las aplicaciones y el software ATA anteriores, SATA utiliza el mismo set de comandos básicos de ATA y ATAPI como los dispositivos ATA y los mismos registros. Los drivers desarrollados para manejar

dispositivos ATA se encuentran con la misma interfaz al utilizar dispositivos SATA siendo invisible para ellos el cambio del modo de transferencia de paralelo a serie. El estándar SATA indica que cada dispositivo se conecta a un conector independiente, eliminando la diferencia entre disco Maestro y Esclavo. La compatibilidad hacia atrás en este aspecto se logra haciendo ver a todo dispositivo conectado como si fuera Maestro.
Las especificaciones de compatibilidad de la industria Serial ATA se originaron a través de la Organización Internacional de Serial ATA (SATA-IO). El grupo de SATA-IO en colaboración crea, revisa, ratifica y publica especificaciones de interoperabilidad y casos de prueba. Como ocurre con muchas otros estándares de compatibilidad en la industria, la propiedad del contenido SATA se transfiere a los órganos de otro sector: primeramente el subcomité ATA de INCITS T13, el subcomité SCSI de INCITS T10 y un subgrupo de T10 responsable por SAS.

Características

- La serie de especificaciones ATA incluye la lógica para la conexión en caliente (hot plugging) de dispositivos SATA. Los dispositivos y motherboards que cumplan con las especificaciones de interoperabilidad son capaces de realizar la conexión en caliente. Esta característica de conexión en caliente permite que un disco pueda ser reemplazado con la computadora prendida, lo cual es de gran utilidad para servidores y otros sistemas que no pueden darse el lujo de apagarse para realizar mantenimiento.
- Como interfaz estándar, los controladores SATA usan AHCI (Interfaz de controlador de host avanzada), permitiendo características avanzadas tales como conexión en caliente y cola de comandos nativos (native command queuing NCQ). Si AHCI no está habilitado en el motherboard y el chipset, los controladores SATA operan en modo “emulación IDE”, el cual no permite que se usen las características de los dispositivos si el estándar ATA/IDE no las soporta.

NCQ (Cola de comandos nativos – Native Command Queuing): es una tecnología que permite al disco ordenar las solicitudes de acceso de tal manera que el movimiento de la cabeza del disco sea el mínimo. Esto teóricamente aumenta el desempeño del dispositivo y su vida útil. Esta tecnología es obligatoria en SATA 2 y opcional en SATA 1.

Conectores de SATA

Los conectores son la diferencia más visible entre los dispositivos SATA y Parallel ATA. A diferencia de los Parallel ATA [3], se usa el mismo conector en los discos de computadoras de escritorio ó servidores (de 3.5 pulgadas) y los de los portátiles (2.5 pulgadas). Esto permite usar las unidades de 2.5 pulgadas en las computadoras de escritorio sin necesidad de usar adaptadores y disminuyendo los costos.
Los dispositivos SATA tienen dos tipos de conectores, el de datos y el de alimentación.

- *Conector de Datos*

El estándar SATA define un cable de datos con 7 conductores, de los cuales 3 son para conexión a tierra y 4 para la transmisión de datos, los cuales se utilizan de a pares. El conector es de 8 mm. Puede tener longitudes de hasta 1 metro y conecta un disco con un socket del motherboard. Esto se diferencia de los cables PATA, los cuales conectan uno ó dos dispositivos al motherboard, con un cable de 40 ó 80 conectores y un largo máximo de 45 centímetros, según la especificación ATA. Esto muestra que los

conectores SATA son más fáciles de colocar en lugares cerrados y producen menos obstrucciones al paso del aire.

- *Conector de Alimentación*
  - *Conector Estándar*

En los dispositivos anteriores a SATA se utilizaba un cable Molex de 4 pines, pero actualmente se utiliza un cable de 15 pines que permiten prevenir la identificación errónea y conexión de forma incorrecta del conector.

  - *Conector Slimline*

SATA 2.6 lo definió por primera vez, y se utiliza por ejemplo, en dispositivos ópticos en las notebooks.

  - *Conector Micro*

El conector micro fue originado por SATA 2.6 y es utilizado en los discos de 1.8 pulgadas (46 mm). También existe un conector de datos micro, que es similar al conector estándar, pero más delgado.

Revisiones de SATA

Revisión SATA 1.0 (SATA 1.5 Gbit/s)

Es la primera generación de interfaces ATA, conocida como SATA 1.5 Gbit/s. Se comunicaba a una velocidad de 1.5 Gbit/s. Pero la velocidad real de transferencia es de 1.2 Gbit/s (150 MB/s). Actualmente los dispositivos ATA ofrecen mejoras como NCQ, que mejoran el rendimiento en un ambiente multitarea.
Durante el período inicial luego de la finalización del SATA 1.5 Gbit/s, los fabricantes de discos y adaptadores usaron un “chip puente” para convertir los diseños PATA para uso con la interfaz SATA. Las unidades de puente tienen un conector SATA, que puede incluir uno ó ambos tipos de conectores de alimentación y funcionan en forma idéntica a sus equivalentes PATA. La mayoría no soportaba las características específicas de SATA como NCQ. Pero los productos SATA originales rápidamente opacaron a los productos puente con la introducción de la segunda generación de dispositivos SATA.

Revisión SATA 2.0 (SATA 3 Gbit/s)

La segunda generación de interfaces SATA funcionan a 3.0 Gbit/s y es la más frecuente en los discos SATA y en la mayoría de los chipset de la PC y servidores. La transferencia real de datos es de 2.4 Gbit/s (300 MB), el doble con respecto a PATA/133. Además, los dispositivos SATA ofrecen mejoras como la “cola de comandos nativos” que aumentan el rendimiento en un ambiente multitarea.
Todos los cables de datos SATA que cumplen con los requisitos SATA están clasificados para 3.0 Gbit/s y manejarán dispositivos mecánicos actuales sin pérdida de rendimiento sostenido y de transferencia de ráfaga de datos.

Revisión SATA 3.0 (SATA 6 Gbit/s)

La Organización Internacional de Serial ATA presentó el borrador de la especificación de la capa física de SATA 6 Gbit/s en Julio de 2008 y ratificó la especificación de la capa física el 18 de Agosto de 2008. El estándar 3.0 completo fue lanzado el 27 de Mayo de 2009, y provee un rendimiento pico de alrededor de 600 MB/s incluyendo el protocolo de overhead.
La nueva especificación contiene los siguientes cambios:

- 6 Gbit/s para el rendimiento escalable cuando se usan SSDs.
- Continúa siendo compatible con SAS (Serial Attached SCSI), incluyendo SAS 6 Gbit/s.
- Cola de comandos nativos Isócrono (NCQ) para permitir la transmisión de comandos de calidad de servicio de datos isócronos de transferencias para aplicaciones de contenido digital.
- Una función de gestión NCQ que ayude a optimizar el rendimiento permitiendo el manejo y procesamiento de comandos pendientes NCQ.
- Mejora de las capacidades de administración de energía.
- Un conector pequeño de “Fuerza de inserción baja” (low insertion force LIF) para dispositivos de almacenamiento más compactos que 1.8 pulgadas.
- Un conector diseñado para adaptarse a discos ópticos de 7 mm para notebooks más delgadas y livianas.
- Alineación con la INCITS ATA8-ACS estándar.

En general, las mejoras están destinadas a mejorar la calidad de servicio para streaming de video y para interrupciones de alta prioridad. Además, el estándar continúa soportando distancias de hasta 1 metro. Las nuevas velocidades pueden necesitar un mayor consumo de energía para los chips que la soportan, factores que las nuevas tecnologías de procesos y las técnicas de administración de energía esperan mitigar. La nueva especificación usa los conectores y cables SATA existentes, aunque algunos fabricantes de equipos originales esperan mejorar los conectores para las velocidades más altas. Este nuevo estándar es compatible con el estándar anterior de SATA 3 Gbit/s.

Arquitectura
El estándar divide la arquitectura de la interfaz SATA en cuatro capas [4] e indica sus características para lograr la compatibilidad:

- *Capa Física*

Brinda información acerca de los conectores y cables utilizados para la transmisión física de los datos incluyendo las especificaciones eléctricas como niveles de tensión, y las señales transmitidas y su modo de transferencia.
Brinda los siguientes servicios a la capa de Enlace:

- Transmitir una trama en serie a 3.0 Gbps o a 1.5 Gbps.
- Serializar una trama para su posterior transmisión.
- Recibir una trama serie a 3.0 Gbps o a 1.5Gbps.
- Obtener los datos de una trama serializada (de-serializar).
- Proveer un protocolo de inicialización de la interfaz SATA y realizar la negociación de la velocidad de transmisión.
- Informar el estado de los dispositivos.
- Soportar distintos modos de administración de energía (opcional).

- *Capa de Enlace*

Es la encargada de trasmitir y recibir las tramas utilizando los servicios brindados por la capa física. Se encarga también de la codificación y decodificación de los caracteres a ser enviados o recibidos utilizando 8b/10b (se agregan 2 bits a cada carácter (8 bits) y se generan los caracteres enviados y recibidos en SATA de 10 bits) y el agregado de información de control para proveer de control de flujo y detección de errores utilizando CRC (comprobación de redundancia cíclica).
Servicios brindados:

- Negociar la transmisión y resolver posibles conflictos con la capa de enlace del dispositivo destino.
- Realizar codificación y decodificación de los datos en 8b/10b.
- Realizar cálculos de CRC y su comparación con la trama recibida para control de errores.
- Proveer de los servicios de control de flujo.
- Incorporar datos de control a la trama recibida de la capa de Transporte.

- *Capa de Transporte*

Básicamente se encarga de la construcción e interpretación de una estructura conocida como Estructura de Información de Trama (FIS – Frame Information Structures). Esta estructura es la que va a ser transferida o recibida. Dentro de ella se destaca el campo Tipo de Fis el cual puede ser por ejemplo “FIS de Activación de DMA” (39h) o “FIS de configuración de PIO” (5Fh) entre otros. Este campo va a definir la longitud de la FIS a ser enviada y los campos que tendrá. Algunos ejemplos de los campos son por ejemplo: dirección LBA (dividido en parta alta, media y baja todos con posibilidad de expansión), nombre del comando o identificador del dispositivo.

- *Capa de Comandos y Aplicaciones*

La capa de aplicación brinda la interfaz con el sistema y con el fin de lograr la retrocompatibilidad, emula la interfaz clásica de ATA. La emulación se logra brindando un conjunto de registros conocidos como el “bloque de registros sombra” que encapsula las operaciones de las capas inferiores y se relacionan con el sistema como lo hacían los registros de dispositivos originales, brindando así compatibilidad con las instrucciones de la BIOS y los drivers diseñados para trabajar con ATA.
El mapeo de los registros esta emulado de la forma en que lo haría ATA, sin embargo todos los dispositivos son tratados como Maestros (SATA ignora el valor que contenga el bit DEV). De manera opcional se puede emular dispositivos Esclavos administrando en forma especial los bloques de registros sombra de dichos dispositivos con sus respectivos Maestros.

Referencias:

[1]http://es.wikipedia.org/wiki/Serial_ATA

[2]http://en.wikipedia.org/wiki/Sata

[3]http://es.kioskea.net/contents/pc/serial-ata.php3

[4]Serial ATA revisión 2.6 15 de febrero del 2007

Comparación entre las interfaces ATA y SCSI

German Maly, Rodrigo German Molina, Matias Salica, Pablo Sebastian Torres, Cristian Jorge Traverso
Universidad Nacional de La Matanza

Abstract:
El propósito de este documento es realizar una comparación entre las interfaces ATA y SCSI, mostrando sus ventajas y desventajas, así como también las diferencias existentes entre estas interfaces.

Palabras Clave:
IDE, ATA, SCSI, SAS.

Introducción

SCSI es un sistema que sirve de interfaz para la comunicación con distintos dispositivos, entre ellos los discos rígidos. Está basado en SASI (Shugart Associates System Interface)[1] un estándar creado por shugart association en conjunto con NCR en 1981. En 1986 fue ratificado por la ANSI como estándar bajo el nombre de Small Computer System Interface.

Desarrollo

Utilización del estándar SCSI

El estándar SCSI es utilizado en terminales de trabajo de alto rendimiento como en servidores, así como en computadoras antigua Macintosh. Los componentes que permite conectar son entre otros discos rígidos, escáneres, lectoras y grabadoras de CDs, impresoras y dispositivos de cinta.
Debido a su complejidad el estándar no fue utilizado en forma masiva, siendo reemplazado por el estándar IDE y luego por las interfaces USB y FireWire en el uso de un gran número de dispositivos. Las computadoras Macintosh implementaron SCSI para brindar así la capacidad de expansión, sin embargo fueron reemplazadas luego por USB y FireWire. Actualmente el uso más común de SCSI es en servidores y estaciones de trabajo como sistemas RAID.

Especificaciones

SCSI-1: (1986) Definió básicamente el sistema con sus protocolos y un conjunto de comandos de 6 y 10 bits[2]. Utilizaba un bus de 8 bits, permitía conectar hasta 8 dispositivos y podía alcanzar velocidades máximas de 3,5 MB/s en modo asincrónico y 5 MB/s en modo sincrónico.
SCSI-2: (1994) Estableció el Set de Comandos Comunes (Common Comand Set - CCS), un conjunto de comandos básicos que debían ser implementados por todos los dispositivos para garantizar la compatibilidad, y permitió duplicar la velocidad del reloj y utilizar un bus del doble de tamaño (16 bits en vez de 8 bits), aumentando el número de dispositivos que se podían conectar.

SCSI-3: (1995) Se dividió el estándar SCSI en las especificaciones de los comandos, los protocolos de transporte y la capa física como estándares separados. La especificación de como los diferentes estándares interactúan entre sí se encuentra en el documento SAM (Modelo de la Arquitectura de SCSI-3 – SCSI-3 Architecture Model). La forma más implementada fue la interfaz paralela SCSI-3 (SCSI-3 Parallel Interface - SPI), que especifica la manera de realizar la comunicación en paralelo entre dispositivos SCSI.

Características

- Puede alcanzar velocidades de transferencia muy altas (320 mbps con la Ultra320 desarrollada en el 2003)
- Es más costoso que IDE principalmente se usa en estaciones de trabajo de alto rendimiento, servidores y dispositivos RAID.[5]
- Permite conectar y desconectar dispositivos sin apagar el sistema si se utilizan placas SCSI de gama alta.
- Permite la conexión de hasta 8 o 16 dispositivos en un único bus dependiendo de la especificación que se esté utilizando.
- Utiliza señales (hand-shake) para realizar la comunicación entre la controladora y el dispositivo que trabajan a tiempos de reloj distintos y realiza detección de errores (paridad para SCSI 1y 2 y CRC32 para posteriores a SCSI-U160).
- Debe ser configurado para cada computadora en particular.
- SCSI identifica cada dispositivo con un identificador (SCSI identification number - ID) y dentro de ellos a cada unidad lógica con un numero de unidad lógica (logical unit number - LUN).

Sistema SCSI
El sistema está compuesto por 3 componentes [3]:

- La controladora
- Los dispositivos
- El cable

La controladora sirve como interfaz entre los dispositivos conectados al bus SCSI y el bus del sistema y puede estar integrado en el motherboard o como una placa de extensión. Cada dispositivo conectado al bus SCSI es identificado por un número de ID, dejando uno de estos para identificar a la controladora, teniendo la posibilidad de conectar hasta otros 7 o 15 dispositivos pudiendo ser estos internos o externos.
Los cables utilizados para conectar los dispositivos SCSI entre si varían según la especificación pudiendo ser de 50, 68 o 80 pines y se dividen en los utilizados para dispositivos internos que tienen forma de cinta, parecidos a los de IDE/ATA y los utilizados para conectar dispositivos externos que son redondeados.

Bus SCSI

El bus SCSI está formado por todos los dispositivos SCSI interconectados entre sí, incluyendo la controladora, los dispositivos internos y los externos.
Los dispositivos internos poseen solo un conector SCSI y se conectan a la controladora mediante un cable en forma de cinta. Dicho cable posee varios conectores, lo cual permite conectar varios dispositivos a la controladora utilizando el mismo cable[4].
Los dispositivos externos poseen dos conectores SCSI y se conectan a la controladora y/o a otro dispositivo formando una daisy chain.
Un detalle importante del bus SCSI es que se deben colocar terminadores conectados a los dispositivos que se encuentren en las puntas del bus para cerrarlo. Esto se debe a que si no se cierran los buses, las señales eléctricas enviadas pueden ser reflejadas al llegar al último dispositivo y causar interferencias. Los terminadores deben conectarse al último dispositivo externo y/o al último dispositivo interno, solo si se tiene alguno de estos tipos de dispositivos conectados.
Los terminadores se diferencian en pasivos y activos, utilizándose los pasivos para los dispositivos conectados a menos de un metro de la controladora y los activos para los dispositivos conectados a más de un metro de la controladora.

Comandos SCSI

Los comandos de SCSI son enviados mediante bloques, los CDB (Command Descriptor Block)[6], los cuales consisten en un byte para el código de operación seguido por cierto número de bytes determinado por el comando con sus parámetros requeridos y un último byte de control. Al final de la secuencia el dispositivo objetivo devuelve un byte con un código de estado, el cual puede ser 0x00 para éxito en la operación, 0x02 para error (llamado Check Condition) y 0x08 para dispositivo ocupado. Si se recibió una Check Condition, generalmente se envía un SCSI Request Sense Command el cual es un comando que solicita un Key Code Qualifier (KCQ), el cual brinda información sobre el error.

SAS

SAS (Serial Attached SCSI) [7] es la versión serie de SCSI. Fue rápidamente adoptado por su rendimiento mejorado y su compatibilidad con SCSI original (ambos utilizan el Set de Comandos Comunes). La conexión entre la controladora y los dispositivos se realiza de punto a punto y la comunicación es serie, no se requiere el uso de terminadores.
Incorpora el uso de Expandidores (Expanders) los cuales les permiten conectar con un máximo de 65535 dispositivos a diferencia de los 8 o 16 de SCSI.
Soporta velocidades máximas de 3 o 6 gbps.
Permite la compatibilidad con dispositivos SATA 2 conectados a un expandidor SAS utilizando el protocolo STP (SATA Tunneled Protocol).

Comparación con el estándar IDE/ATA

En primer lugar hay que destacar que a diferencia de IDE/ATA, SCSI establece un protocolo de más alto nivel al realizar el manejo de diferentes tipos de dispositivos y permitir identificarlos en forma lógica además de en forma física. SCSI identifica a sus dispositivos en forma física mediante los ID, permitiendo conectar hasta 16 dispositivos en un mismo bus, y en forma lógica cada uno de ellos están divididos en LUN, además tanto los ID como los LUN pueden ser asignados mediante un cambio en la configuración por software, IDE/ATA en cambio solo permite la conexión de dos dispositivos a un mismo cable, el maestro y el esclavo, y estos deben ser diferenciados mediante un mecanismo de hardware (jumper o selector).
SCSI permite alcanzar velocidades de transferencias más altas que IDE/ATA. Esto se ve al comparar por ejemplo la interfaz SCSI Ultra 320 desarrollada en el 2003 que alcanza una velocidad de 320 mbps comparándola con ATA/ATAPI 7 desarrollada en el 2005 que solo logra una velocidad de 133 mbps. En desventaja implementar un sistema SCSI es mucho más caro que uno IDE/ATA, los dispositivos compatibles son mucho más caros al igual que los cables y la controladora que no suele venir integrada en las placas base de las computadoras comunes. Como consecuencia de esto, SCSI se utilizó principalmente en Servidores, terminales de trabajo de alto rendimiento y sistemas RAID, donde la posibilidad de conectar varios dispositivos y alcanzar altas velocidades importa más que los costos asociados. Sin embargo con la aparición de SATA, las diferencias de velocidades se ha acortado, haciendo que SCSI sea desplazado aun más.
En cuanto a los comandos, SCSI permite utilizar comandos de más alto nivel que IDE/ATA como por ejemplo comandos para rebobinar una cinta o formatear un disco, los cuales ahorran trabajo al procesador.

Referencias:

[1]Wiki
http://en.wikipedia.org/wiki/SCSI

[2]How SCSI Works. De Jeff Tyson
http://iws45.iiita.ac.in/IIT2008052/ebook/How%20Stuff%20Work/How%20SCSI%20Works.pdf

[3]Comité t10
http://www.t10.org/

[4]Tecnología de Discos de Edman Wellington Oliveira
http://www.ic.unicamp.br/~ducatte/mo401/1s2006/T2/065819-T.pdf/
[5]configurarequipos.com

http://www.configurarequipos.com/doc370.html
[6]Draft SCSI Architecture Model 2 (SAM 2)

[7] Michael Micheletti, Product Manager, LeCroy Corp., SAS and Serial ATA Tunneling Protocol (STP)
http://www.serialstoragewire.org/Articles/2004_0225/developer_article_2_feb.htm

IMPLEMENTACION FUTURA DE DRIVER DMA

Para la implementación de DMA en dispositivos ATA se requerirá la habilitación de las interrupciones de las controladoras ATA en la configuración del PIC al iniciar Sodium, ya que hasta el momento no fueron necesarias y se encuentran enmascaradas.
Las funciones a realizar serán simples y tendrán los mismos parámetros que tienen las de Lectura y Escritura para ATA , pudiendo crearse dos nuevas funciones “leer” y “escribir” que decidan en tiempo de ejecución que estándar usar para la lectura de entre los 3 que van a existir al finalizar el TP de DMA a ser : leerSectorEnLBA28, leerSectorEnLBA48, leerSectorDMA . Lo mismo para la operación de escritura.
Se requerirán nuevas estructuras de datos a consultar con la bibliografía que deben ser añadidas, por lo pronto podemos aclarar que se necesitará de una nueva comparación contra el bloque de identificación de disco para determinar si el dispositivo soporta DMA y de ser así que versión soporta. Esto debe hacerse en la función del driver ATA: probe_ide_drives() , ya que es esta la que carga las estructuras de datos de los dispositivos.

Para mayor información de la controladora y puertos avocados a DMA ver el draft : "Information technology -AT Attachment 8 - ATA/ATAPI Command Set (ATA8-ACS)" en los tópicos de DMA y Identify Device.
Esta es la revisión del draft: Revisión 6ª
Septiembre 6, 2008
Documento: T13/1699-D
Autor: American National Standard of Accredited Standards Committee INCITS

Datos de Contacto:

German Maly | maly_89_4@hotmail.com | Universidad Nacional de La Matanza
Rodrigo German Molina | molina.rodrigo@ymail.com | Universidad Nacional de La Matanza
Matias Salica | matias.salica@yahoo.com.ar | Universidad Nacional de La Matanza
Pablo Sebastian Torres | torres.pablo.sebastian@gmail.com| Universidad Nacional de La Matanza
Cristian Jorge Traverso | traverso.cristian@yahoo.com.ar| Universidad Nacional de La Matanza

Bibliografía

AT Attachment Interface for Disk Drives, Draft 791D, Revision 4c

AT Attachment with Packet Interface – 6 (ATA/ATAPI-6)

AT Attachment with Packet Interface -7

AT Attachment with Packet Interface -8

http://es.wikipedia.org/wiki/Serial_ATA

http://en.wikipedia.org/wiki/Sata

http://es.kioskea.net/contents/pc/serial-ata.php3

Serial ATA revisión 2.6 15 de febrero del 2007

Wiki
http://en.wikipedia.org/wiki/SCSI

How SCSI Works. De Jeff Tyson
http://iws45.iiita.ac.in/IIT2008052/ebook/How%20Stuff%20Work/How%20SCSI%20Works.pdf

Comité t10
http://www.t10.org/

Tecnología de Discos de Edman Wellington Oliveira
http://www.ic.unicamp.br/~ducatte/mo401/1s2006/T2/065819-T.pdf/

configurarequipos.com
http://www.configurarequipos.com/doc370.html

Draft SCSI Architecture Model 2 (SAM 2)

Michael Micheletti, Product Manager, LeCroy Corp., SAS and Serial ATA Tunneling Protocol (STP)
http://www.serialstoragewire.org/Articles/2004_0225/developer_article_2_feb.htm