Hard Disk

Version 1.0

06.08.2009

Lubomir Cabla/CBL

http://www.hdat2.com/

Contents

Tables

Pictures

Preface

I am sorry for my English.

Appendixes

A. Standard ATA/ATAPI (PATA/SATA/FATA)

ATA (AT Attachment): ATA standard specifies the AT Attachment Interface between host systems and storage devices. It provides a common attachment interface for systems manufacturers, system integrators, software suppliers, and suppliers of intelligent storage devices. ATA defines the physical, electrical, transport, and command protocols for the internal attachment of storage devices to host systems.

ATA interface is derived from **Advanced Technology** (AT) developed initially for computer IBM PC/AT in the middle of 1980.

ATAPI (AT Attachment Packet Interface) device: A device implementing the Packet Command feature set. It is designed for removable devices like CD-ROM, DVD, ZIP, JAZZ, tape drive etc.

With regard to new standard SATA is for existing ATA standard used PATA (Parallel ATA).

IDE and EIDE are only marketing names to label device corresponded with ATA standard.

Description of transfer speed, e.g. Ultra DMA 133 indicate that data could be transfer with speed up to 133 MB/s. **ATTENTION:** it is burst transfer data rate and not average transfer data rate, which should be half or less than third of burst transfer data rate.

ATA - Advanced Technology Attachment, the standard PC Hard drive interface (IDE).

SATA - Serial ATA is a newer PC hard drive interface that is quickly taking over the "ATA" stronghold. SATA is faster than ATA and runs at about 150 MB/s. SATA rev.2.x runs at 300 MB/s.

PATA - Parallel ATA is the same as ATA or IDE. The old standard for connecting hard drives (40 pin connector).

FATA - FATA is a Fibre connectted ATA drive. Usable on systems that have a Fibre based backplane.

A FATA disk is a combination of SATA rev.2.x and Fibre Channel disk technologies that connects a dual-port FC interface directly to SATA rev.2.x disk drive hardware. This provides true dual-port drive connectivity.

Designed to meet the architectural standards of enterprise-class storage systems FATA disk drives provide high capacity as a low-cost alternative to FC disks without much sacrifice of performance, availability, or functionality. Also, when used within their recommended duty cycle, reliability is comparable to that of the FC disks.

A.1 ATA (ATA-1, IDE)

ANSI document X3.221-1994
ATA is actual standard for what is known as IDE. ATA define PIO (Programmed Input Output) modes 0, 1 and 2 and DMA (Direct Memory Access) mode 0.

AT Attachment Interface for Disk Drives (ATA-1) was withdrawn as a standard on 6 August 1999.

A.2 ATA-2 (EIDE, Fast-ATA)

ANSI document X3.279-1996
ATA-2 is standard known as EIDE.
ATA-2 introduces higher transfer modes PIO 3, 4, and Multiword DMA modes 1, 2. These modes make possible transfer rate up to 16.6 MB/s. Next introduce extended device identification (extended command Identify Drive), block transfer, LBA and some added commands.

A.3 ATA-3

ANSI document X3.298-1997
ATA-3 introduces S.M.A.R.T., security, support for ATAPI devices (CD-ROM, ZIP).
ATA-3 did not introduce any new PIO or DMA transfer modes.

AT Attachment Interface with Extensions (ATA-3) was withdrawn as a standard in 2002.

A.4 ATA/ATAPI-4

ANSI document NCITS.317-1998
ATA/ATAPI-4 introduces and change many things.

- new ATAPI commands and reset protocols
- change many old ATA commands and features (e.g. Format Track, Read/Write Long) into state 'obsolete')
- new transfer protocols Ultra DMA 0,1 and 2 with integrity data over CRC checking; transfer rate up to 33 MB/s
- new protocol for command overlapping and queuing into queue for ATA and ATAPI devices (overlapping, queuing)
- many new features for ATA and ATAPI devices

A.5 ATA/ATAPI-5

ANSI document NCITS.340-2000
ATA/ATAPI-5 cancel out some old commands, some new commands introduce.
Main change is adding 2 new and fast transfer modes Ultra DMA 2 and 3.

A.6 ATA/ATAPI-6

ANSI document NCITS.?

- extension 28-bits LBA addressing to 48-bits addressing mode
- increasing of 'Sector Count' (count of transfer sectors)
- increasing timing Ultra DMA mode (Ultra DMA 100)
- new commands for AV (Audio/Visual) applications
- ATA Removable Media Serial numbers

A.7 ATA/ATAPI-7

ANSI document NCITS.?

- Ultra DMA mod 133
- S.M.A.R.T.: Selective self-test, Conveyance self-test
- preserve words, codes and feature sets for Serial ATA
- Serial ATA Specification (SATA) 1.0
- Forced unit access (FUA) commands
- support for larger size of physical sector
- definition World Wide Name for ATA device

Table 1: Overview of ATA/ATAPI transfer modes

Name	Original standard	Max. Bus speed [MB/s]
PIO 0	ATA (ATA-1)	3.33
PIO 1	ATA	5.22
PIO 2	ATA	8.33
PIO 3	ATA-2	11.1
PIO 4	ATA-2	16.6
Single word DMA 0	ATA	2.1
Single word DMA 1	ATA-2	4.2
Single word DMA 2	ATA-2	8.3
Multi word DMA 0	ATA	4.16
Multi word DMA 1	ATA-2	13.3
Multi word DMA 2	ATA-2	16.6
Ultra DMA 0	ATA/ATAPI-3	16.6
Ultra DMA 1	ATA/ATAPI-3	25.0
Ultra DMA 2 (UDMA33, ATA/33)	ATA/ATAPI-4	33.3
Ultra DMA 3	ATA/ATAPI-5	44.4
Ultra DMA 4 (UDMA66, ATA/66)	ATA/ATAPI-5	66.6
Ultra DMA 5 (UDMA100, ATA/100)	ATA/ATAPI-6	100.0
Ultra DMA 6 (UDMA133, ATA/133)	ATA-ATAPI-7	133.0

PIO (Programmed Input Output) mode – The oldest method of transferring data over the IDE/ATA interface is through the use of programmed I/O. This is a technique whereby the system CPU and support hardware directly control the transfer of data between the system and the hard disk.

DMA mode - A better solution is to take the CPU out of the picture entirely, and have the hard disk and system memory communicate directly. **Direct memory access** or **DMA** is the generic term used to refer to a transfer protocol where a peripheral device transfers information directly to or from memory, without the system processor being required to perform the transaction. Modern IDE/ATA hard disks use **first-party** DMA transfers. The term "first party" means that the peripheral device itself does the work of transferring data to and from memory, with no external DMA controller involved. This is also called **bus mastering**, because when such transfers are occurring the device becomes the "master of the bus". Bus mastering allows the hard disk and memory to work without relying on the old DMA controller built into the system, or needing any support from the CPU. It requires the use of the PCI bus - older buses like MCA also supported bus mastering but are no longer in common use. Bus-mastering DMA allows for the efficient transfer of data to and from the hard disk and system memory. Bus mastering DMA keeps CPU utilization low, which is the amount of work the CPU must do during a transfer.

Table 2: Overview of cables

Cable	Length of cable		Used protocol
	Min.	Max.	
40-wire	-	18"	up to UDMA33
80-wire	10"	18"	for UDMA66 or high

80-wire (conductor) cable is optionally applicable for devices using Ultra DMA 2 mode (ATA/33).

B. Standard SCSI

SCSI = Small Computer System Interface ('skuzzy')

Table 3: Overview of SCSI

Business TERM	Standard TERM	Standard	Max. Bus speed [MB/s]	Bus width [bits]	Max. Length of cable [m]			Max. Device count
					SE	HVD	LVD	
SCSI-1 SCSI-2	Fast-5	SCSI SCSI-2	5	8	6	25	-	8
Fast SCSI	Fast-10	SCSI-2	10	8	-	25	-	8
Fast SCSI	Fast-10 (SPI, SIP)	SCSI-3	10	8	3	25	-	8
Fast Wide SCSI	Fast-10 (SPI, SIP)	SCSI-3	20	16	3	25	-	16
Ultra SCSI	Fast-20	F-20	20	8	1.5	25	-	8
Ultra SCSI	Fast-20	F-20	20	8	3	25	-	4
Wide Ultra SCSI	Fast-20	F-20	40	16	-	25	-	16
Wide Ultra SCSI	Fast-20	F-20	40	16	1.5	25	-	8
Wide Ultra SCSI	Fast-20	F-20	40	16	3	-	-	4
Ultra2 SCSI	Fast-40	SPI-2	40	8	-	25	12	8
Wide Ultra2 SCSI	Fast-40	SPI-2	80	16	-	25	12	16
Wide Ultra3 SCSI	Fast-80	SPI-3	160	16	-	-	12	16
Ultra 160 SCSI	Fast-80	SPI-3	160	16	-	-	12	16
Ultra 320 SCSI	Fast-160	SPI-4	320	16	-	-	12	16
Ultra 640 SCSI			640					
Serial Attached SCSI	SAS	?	?	?	?	?	?	?

SE = Single Ended
HVD = High Voltage Differential
LVD = Low Voltage Differential

Ultra SCSI: between devices is the same distance

Table 4: SCSI cables

SCSI	Min. distance between devices	Max. Length of cable [m]
SCSI-1	?	6
Differential SCSI	?	25
Fast SCSI-2	12"	3
Ultra SCSI	from 2. to 4. device	3
	from 5. to 8. device	1.5
Wide SCSI	12"	3

D. Standard SATA

Picture 1: SATA logo

Serial Advanced Technology Attachment (ATA) = SATA 1.0a 07.01.2003

Serial ATA Working Group was founded in 1999.
[http://www.serialata.org/]

Serial ATA (SATA) is obviously based on **serial signaling technology**. This means that those old ATA ribbon cables were used to send a parallel signal those forty wires were used to transfer data along many parallel routes. ATA is also known in retrospect as Parallel ATA (PATA). All these wires next to each other were sensitive to interference and caused all kinds of problems. Serial ATA signals are transferred at a more efficient voltage of 250mV compared to the 5V of Ultra/ATA and interference is cancelled out by two phase-reversed signals. It is unlikely that many more advancements could be made even if Parallel ATA were to have still an extended lifetime. In addition, Serial ATA is, obviously by name, a serial technology. It can only "talk" to one device per channel. While this might seem like a step back from the two devices per channel allowed with Ultra/ATA, this is quickly offset by other plusses that will be described shortly.

As far as controller cards go, today's PCI Serial ATA controller cards (year 2004) will be very short lived. In a couple of months Intel will release their 865 and 875 chipsets with native support for two Serial ATA channels in their ICH5 I/O Hub, eliminating the need for an add-in Serial ATA card.

The ATA command set support in a driver limits the addressable hard disk drive capacity. Because both Serial ATA and parallel ATA connections use the ATA commands, both Serial ATA and parallel ATA have the same capacity limitations.

SATA description:

- SATA supports a single device per SATA cable
- no master/slave jumper
- used topology „star" (point-to-point), without hubs
- 32-bits CRC check for data and commands
- whole bandwidth for every device (channel is not shared)
- low voltage (250mV) differential signaling with 8/10b encoding
- supply 12.0, 5.0 and 3.3 V

Since Serial ATA only requires seven wires per device, the new cables will only be 8mm wide. Another dimensional advantage is that while ATA cables could only be up to 40 cm long, Serial ATA cables will be very long at 1m. Of course, since the cables are smaller

and contain only seven wires, the connectors will be more compact as well. This will save space on motherboards and the hard drives themselves.

Table 5: SATA standards

Standard	Transfer [MB/s]
SATA rev.1.x (2002)	150 (up to 1.5 Gb/s)
SATA rev.2.x (2004)	300 (up to 3 Gb/s)
SATA rev.3.x (2007)	600 (up to 6 Gb/s)

PATA devices connected to SATA over adapter should be jumpered as master.

SATA 1.0 defines speed 1.5 Gb/s.

SATA II defines Extensions specifications to Serial ATA 1.0 and increases interface speed to 3.0 Gb/s. Multiple specification release:

- Serial ATA 1.0 Extensions, Cable Connector, Port Multiplier, Port Selector, Switch, Phy II 3.0 Gb/s

- All extensions are optional and the product must support at least one extension to claim compliance

Port Multiplier

- Allows one Serial ATA link to connect to more than one device
- Can provide a simple expansion solution for entry-level servers and RAID boxes

Port Selector

- Conceptually, a simple A/B switch that creates a redundant path to the device
- Can provide a simple fail over solution for entry-level servers

SATA II Extensions 1.0:

1. **Native Queuing**
- Defines simple and streamlined command queuing model that will enable 32 queued commands
- Returns a race-free status condition that minimizes the protocols round trip which in turn reduces the incurred overhead
- Can be used with first party DMA or with a scatter gather table

2. **Identify Device / Set Feature**
- Used by the host to determine which features the device supports
- Identify device and set features will include those commands in the Serial ATA II extensions specification that affect the device

3. **Staggered Spin up**
- Defines a method of staggering spin up of drives when used in RAID configurations with multiple drives

4. **Hot Plug / Presence Detect**
- Defines a method for the host to determine when a device is plugged into a hot bay

5. **First Party DMA**
- Defines a method of allowing the device to access the host's memory

- This may be used in conjunction with command queuing

SATA II Extensions 1.1:

1. **Device Configuration Overlay (DCO)**
- An ATA/ATAPI mechanism for disabling features in the drive. Identify Device will not support features disabled using DCO
- Defining DCO words for Serial ATA specific items:

  - Asynchronous notification
  - Native command queuing
  - Non-zero buffer offset
  - Power management support

2. **First Party DMA Auto-Activation**
- Eliminates the extra transmission of a FIS. The DMA Setup FIS will contain a bit to automatically activate the DMA controller

3. **Queuing Optimization**
- An in-order data delivery that guarantees the DMA buffer offsets are continuous when the device utilizes First Party DMA and non-zero buffer offsets in the DMA setup FIS

External SATA is a device that allows the user to add storage to his system without opening the system box.

- External Serial ATA Logic will be different from Serial ATA 1.0
- Different Voltage Swings
- Hot Plug connector

D.1 The Different Modes of SATA Controllers

The core of Serial ATA is defined by the Serial ATA 1.0 Specification, which was published in August 2001. To make adoption easier for the new interconnect, the Serial ATA 1.0 Specification defines a special mode for Serial ATA controllers so that they emulate the behavior and configuration of the parallel ATA interconnect. In this **Emulating Parallel ATA mode**, Serial ATA controllers can leverage all the existing parallel ATA drivers and infrastructure of shipped Windows operating systems.

The Serial ATA 1.0 Specification also has new features that are not compatible with all existing parallel ATA drivers. Existing parallel ATA drivers do not comprehend any newly defined features and capabilities for which there are no equivalents in parallel ATA. Therefore, such software will not utilize any such new features. A controller that takes advantage of these new features is said to be operating in **Native Serial ATA mode**. A Serial ATA controller capable of doing both emulation and native modes cannot switch between Emulating Parallel ATA and Native Serial ATA modes while Windows is running.

Additional new Serial ATA extensions are defined in the **Serial ATA II Specification**, which was published in October 2002. Serial ATA II is not the next version of Serial ATA, nor is it a mode of Serial ATA like Emulating Parallel ATA and Native Serial ATA. Serial ATA II is a set of optional extensions and features for Serial ATA that is available to Serial ATA controllers and devices that operate in Native Serial ATA mode.

Finally, a few miscellaneous specifications define additional optional new Serial ATA features for controllers that implement Native Serial ATA mode, but are not included in either the Serial ATA 1.0 or Serial ATA II specifications.

D.1.1 Emulating Parallel ATA Mode

The Emulating Parallel ATA mode defines a transfer level equivalent of parallel ATA for Serial ATA controllers. In this mode, a Serial ATA controller can emulate **master-only (device 0) parallel ATA** or **shared channel parallel ATA**. In master-only parallel ATA emulation, the Serial ATA controller presents itself to the computer as a parallel ATA controller with only a single master storage device attached to a channel. In shared channel parallel ATA emulation, the controller uses two Serial ATA channels, each only attach to a single storage device, as a single parallel channel attaching two storage devices. Both forms of emulation work with Serial ATA controllers that use Windows parallel ATA (atapi.sys) drivers.

Beyond configuration emulation requirements, Emulating Parallel ATA mode restricts the controller from doing Native Serial ATA mode functionality described in the next section. Another notable distinction is that parallel ATA has many transfer modes each with a unique transfer speed associated with it. Serial ATA emulating Parallel ATA mode controllers also must support parallel ATA transfer modes, but its transfer speed is much faster then all parallel ATA transfer modes. While Serial ATA controllers operate at higher transfer speeds, it is possible for them to claim to operate in slower parallel ATA transfer modes such as Programmed Input/Output (PIO).

D.1.2 Native Serial ATA Mode

The advanced features that can be utilized in native mode revolve around improvements to the Serial ATA interconnect. The feature that has generated the most interest in this area is **hot plugging**, which allows an end user to remove a storage device from a Serial ATA controller while a system is running. This is useful for RAID systems and notebook docking stations with built-in storage devices. However, hot plugging should not be attempted with a system's primary boot device.

The Serial ATA hot plug feature is implemented by the host controller, driver software, and storage device. On the personal computer, support for Serial ATA hot plugging can be implemented in the following two places:

1. In the Serial ATA controller driver:

Since parallel ATA controllers do not support hot plugging and Serial ATA Emulating Parallel ATA mode controllers are likely to use parallel ATA controller drivers, Emulating Parallel ATA mode controllers must find another way to support the hot plug feature. The atapi.sys driver in Windows does not support hot plugging; however, it is likely that Native Serial ATA mode drivers will.

2. In the system firmware outside of Windows:

Support from ACPI, BIOS, or a combination of the two can be used to trigger bus re-enumerations during a hot plug event. Check with the controller manufacturer for details on support for hot plugging.

Another improvement is finer grained power management. In addition to doing power management on Serial ATA storage devices, the Serial ATA controller itself can be managed so that unused parts of the controller can be put into lower power modes to conserve electricity.

Finally, there are features for Serial ATA drivers. There are new control, error and status registers that allow the Serial ATA controller to pass information to the driver about

Serial ATA specific features. In addition, the Serial ATA interconnect configures its own transfer rate so that the driver no longer needs to.

D.2 SATA II Features

The most significant Native Serial ATA mode feature defined in Serial ATA II is **native command queuing**, which is optimized for Serial ATA and is much more efficient. The intention of the new queuing method is to improve performance by eliminating handshakes, allowing aggregating of interrupts, and reducing the interface transaction count.

Other optional features try to make Serial ATA more attractive in enterprise storage markets. The **Enclosure Services feature** allows a Serial ATA controller to communicate status and control commands to the enclosure processor of the storage system. Serial ATA II also defines signal constraints for using Serial ATA as a rack backplane and a mechanism for controlling staggered spin up.

D.2.1 Native Command Queuing (NCQ)

Native Command Queuing (NCQ) is arguably the most significant advance in the Serial ATA II specification. Native Command Queuing allows the host to issue multiple commands to the device (up to 32 commands) without having to wait for the device to complete any commands. Queuing of commands allows SATA drives to look ahead at what data has been requested or needs to be written, thereby allowing the drive to optimize the order of the commands and maximize data throughput efficiency, providing a significant performance improvement.

Note: NCQ-capable host hardware and drivers must be used to take advantage of the performance gains provided by NCQ.

To enable Native Command Queuing, the Serial ATA II standard defines a method of allowing an HDD to control the order of command execution and data transfer. Using special SATA commands **READ FPDMA QUEUED** or **WRITE FPDMA QUEUED** the host will issue each command an identifier, or tag. The specification of Native Command Queuing allows for up to 32 tags (0 to 31). In order to avoid collisions and mishandled data, the HDD will only release a tag after the associated command is complete and the data has been returned to the host.

The **concept of command queuing** means that a drive does not need to return the data in the same order that the commands are requested (tags 0 to 31 can be executed in any order and data packets for those commands can be returned to the host in any order). For example, the commands may be issued in numerical order: 1, 2, 3, 4, and the data for those commands may be returned to the host in a different order: 4, 2, 1, 3 (or any other order). This allows a drive to use rotational position optimization to maximize the efficiency and overall performance of the drive.

D.2.2 Non-Zero Offsets in DMA Setup

When enabled, this feature allows the HDD to use non-zero buffer offsets, which is needed to implement out of order data delivery. Based on current bit densities and spin speeds, out-of order data delivery provides no significant performance advantage.

D.2.3 DMA Setup FIS Auto-Activate Optimization

When enabled, this feature allows the HDD to optimize DMA data transfers by reducing the required overhead to setup the DMA data transfer. Removing the overhead associated with DMA setup commands will not provide any appreciable performance improvements, as the overhead is an insignificant portion of data transfers.

D.2.4 Device-Initiated Interface Power State Transitions

The feature gives hosts the ability to prevent HDD's from automatically activating power saving features.

D.3 Serial ATA Hardware Register Interface

The Serial ATA specifications do not define a full standard hardware register interface for Serial ATA drivers. Without a standard interface, Serial ATA controller manufacturers are tasked with the redundant work of creating their own proprietary interfaces. The result of this is the possibility of many different Serial ATA interfaces and the inability for Microsoft to create a driver that works commonly over all Serial ATA controllers.

The potential for many unique interfaces could cause the same challenges for Serial ATA that is seen in small computer system interface (SCSI) controllers. Currently one public committee led by Intel, the **Advanced Host Controller Interface** (**AHCI**) Contributor Group, is working to create a public specification for a Serial ATA interface. Microsoft is a member of the AHCI Contributor Group, and encourages all Serial ATA controller manufacturers to adopt AHCI.

D.4 Naming Conventions for Serial ATA Products

Serial ATA products will continue to be identified by speed and not by feature set or specification compliance. This is the parallel ATA naming convention. Already Serial ATA 1.5 gigabytes (GB) per second are available, and in the future, there will be Serial ATA 3 GB per second, Serial ATA 6 GB per second, and so on.

There is no mandatory distinction made between Serial ATA products that support different Serial ATA and Serial ATA II features. Many people will be tempted to associate incorrectly Serial ATA II with Serial ATA 3.0 GB per second, which has not been defined yet. This should not be done.

In addition, there is no distinction made between Serial ATA products that support either version of Emulating Parallel ATA mode or for Native Serial ATA mode.

D.5 Supports for Serial ATA in Windows

Parallel Advanced Technology Attachment (ATA) controllers seen today come in two form factors: as components of chipsets and as discrete controller add-on cards. The great majority of parallel ATA controllers in chipsets can load drivers that are included with Windows while a smaller portion of add-on cards, usually PCI cards, can use drivers that are included with Windows.

Serial ATA controllers have been using and will likely continue to use, the same form factors as parallel ATA controllers. It is possible that many Serial ATA and parallel ATA controllers will be included in the same product to accommodate both Serial ATA and parallel ATA hard disk drives.

Like parallel ATA controllers, Windows support for Serial ATA controllers depends on the Serial ATA features implemented and the hardware register interface used by the controller.

With the product introduction of **Ataport**, two miniports will be introduced. The first is a default miniport driver that will work with the current **atapi.sys** supported parallel ATA controllers. The second is a new miniport driver that will support the Advanced Host Controller Interface (AHCI) Serial ATA controllers.

A Serial ATA **Emulating Parallel ATA mode** controller can load and use parallel ATA drivers that are supported by Windows. By definition, all versions of Windows before Windows Server 2003 have Emulating Parallel ATA mode support.

Not all parallel ATA controllers are supported natively in a Windows distribution so there will be some Serial ATA Emulating Parallel ATA mode controllers that must have their manufacturer's drivers to work properly.

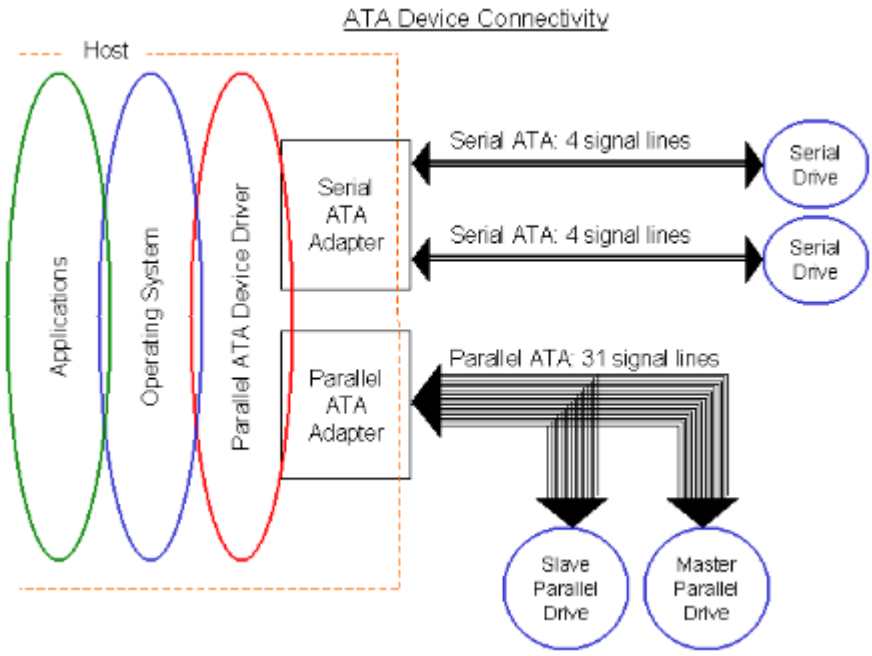
Because **Native Serial ATA mode** controllers do not appear or act like parallel ATA controllers, they cannot use the Windows parallel ATA controller drivers, and thus, Windows Server 2003 and previous do not ship with drivers that support Native Serial ATA mode controllers.

Native Serial ATA mode controllers can be made to work with all existing versions of Windows as long as the controllers come with the appropriate drivers. With these drivers, Windows can be installed on and booted from Serial ATA storage devices in the same way many parallel ATA add-on controllers do currently.

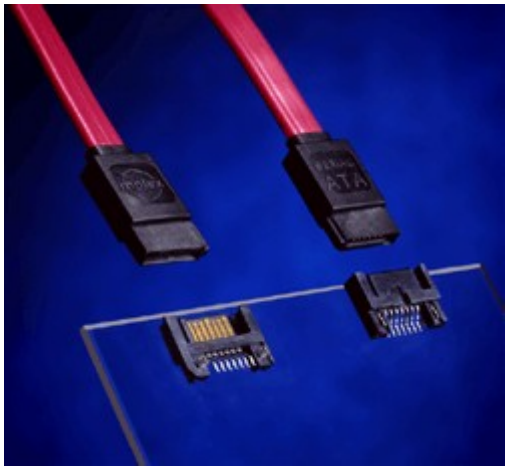
D.6 Identifying modes of SATA Controllers

Driver support for Emulating Parallel ATA mode and Native Serial ATA mode controllers is separate and distinct as these modes appear as two different and unique ATA interconnects. There will be some Serial ATA controllers that will want to implement both Emulating Parallel ATA and Native Serial ATA modes on the same device, which is acceptable. It is not acceptable to use both modes at the same time in Windows. Additionally, loading parallel ATA drivers on a Serial ATA controller in Native Serial ATA mode and vice versa will have disastrous results.

The solution to identifying the Serial ATA mode problem is using the correct PCI Sub-Class code. Appendix D of the latest PCI specification lists all the Sub-Class codes for Base Class 01h, mass storage controllers. Sub-Class code 01h, ATA controller, should be used by Emulating Parallel ATA mode controllers. A new Sub-Class code 06h, Serial ATA controller, should be used by Native Serial ATA mode controllers.



Picture 2: Serial ATA vs. Parallel ATA Device Diagram



Picture 3: Serial ATA Cables and Connectors (Source: Molex)

D.7 Notices about SATA

D.7.1 The SATA cable connector is not shielded

The basic problem is the SATA cable connector is not shielded.

- Do not operate SATA devices outside of a sealed system unit.
- Do not operate SATA devices from a power supply that is not the system unit's power supply.
- Do not tie wrap SATA cables together.
- Do not put sharp bends in SATA cables.
- Do not route SATA cables near PATA cables.

- Avoid placing SATA devices close to each other such that the SATA cable connectors are close to each other.
- Do not operate a radio transmitter (such as a cell phone) near an exposed SATA cable or device.

D.7.2 Variety of problems

Testing of SATA products are finding a variety of problems.

- timeout errors
- data compare errors
- strange status errors

The unshielded SATA cable connector is mostly like the source of many of these problems. Making things worse is the failure of the SATA specification to implement an equivalent to the ATA Soft Reset. On a PATA interface Soft Reset rarely fails to get ATA/ATAPI devices back to a known state so that a command can be retried. On a SATA interface the equivalent to this reset does not seem to reset anything and the SATA controller and device basically ignore some times it.

Today's SATA products are actually 10% to 20% slower than PATA. This is because today's SATA products are really PATA products with an extra SATA-to-PATA 'bridge chip' in the device. These bridge chips add significant overhead to the SATA protocols. In time there will real 'native' SATA devices that do not need these bridge chips - Then we can see what the true performance of SATA. However, remember SATA is a 'serial interface' and serial interfaces rarely live up to their marketing claims.

D.7.3. Specifications

SATA-1, the SATA version that will be included in ATA/ATAPI-7, is designed to emulate traditional parallel ATA. Most SATA host controllers shipping today look like and are programmed just like any other ATA host controller. These controllers are compatible with the Intel ICHx design and compatible with the T13 1510D document. This allows SATA controllers and devices to be used in systems without BIOS or OS driver changes.

The biggest problem with today's SATA host controllers is that SATA gets errors that never happened on PATA. In addition, today's SATA host controllers do a very poor job of reporting these errors to the host software. Then there is the problem of having an OS device driver for SATA that understands these errors and knows how to recover from the error conditions. As of September 2003 T13 is just now starting to talk about these problems.

There are some SATA host controllers that are not ICHx compatible and these require proprietary BIOS or OS drivers. We can only hope that the Intel AHCI specification effort is successful in bringing us a new and better SATA host controller standard.

In May 2003 Intel has issued new specification **Advanced Host Controller Specification (AHCI)** for Serial ATA host controller. AHCI support standard interface for system drivers to use extended features of SATA like command queuing, hot plug and power management.

E. Standard IEEE 1394

1394 Trade Association [http://www.1394ta.org/]
IEEE [http://www.ieee.org/]
Consumer web site [http://www.askfor1394.com/]

IEEE 1394 (High Performance Serial Bus) is an international high-performance serial-bus standard that offers the real-time data transfer of video, audio and peripheral applications through a universal I/O interface. With this technology, digital cameras, CD-ROMs, printers, hard disk drives and audio/stereo equipment can move data at high speeds to desktops and portable computers through a single cable.

IEEE 1394, High Performance Serial Bus, is electronics standard for connecting devices to your personal computer. IEEE 1394 provides a single plug-and-socket connection on which up to 63 devices can be attached with data transfer speeds up to 400 Mbps. The standard describes a **serial bus** or pathway between one or more peripheral devices and your computer's **microprocessor**. Many peripheral devices now come equipped to meet IEEE 1394. Two popular implementations of IEEE 1394 are Apple's **FireWire** and Sony's **i.LINK**. IEEE 1394 implementations provide:

A simple common plug-in serial connector on the back of your computer and on many different types of peripheral devices
A thin serial cable rather than the thicker parallel cable you now use to your printer, for example
A very high-speed rate of data transfer that will accommodate multimedia applications (100 and 200 megabits per second today; with much higher rates later)
Hot-plug and **Plug and Play** capability without disrupting your computer
The ability to chain devices together in a number of different ways without terminators or complicated set-up requirements

In time, IEEE 1394 implementations are expected to replace and consolidate today's serial and parallel interfaces, including Centronics **parallel**, **RS-232C**, and Small Computer System Interface (**SCSI**). The first products to be introduced with FireWire include **digital cameras**, digital video disks (**DVDs**), digital videotapes, digital camcorders, and music systems. Because IEEE 1394 is a **peer-to-peer** interface, one camcorder can dub to another without being plugged into a computer. With a computer equipped with the socket and bus capability, any device (for example, a video camera) can be plugged in while the computer is running.

E.1 Evolution of standard IEEE 1394

FireWire was introduced in 1990 by company Apple Computer Inc. and developed by company Texas Instruments. Originally was intended like replacement for SCSI.

1. IEEE 1394-1995:
- original standard

1394 devices on the market today conform to either the original IEEE 1394-1995 specification or its backward-compatible supplement, IEEE 1394a-2000.

2. IEEE 1394a-2000:
- max. speed 400 Mpbs, do not change basic features, and contains **„Advanced power management"**

Note that 1394a-2000 primarily clarifies the 1394-1995 specification in ways that enhance interoperability. 1394a-2000 does not increase its speeds beyond 400 Mbps or change its fundamental capabilities. One feature of 1394a-2000 of key interest to the PC industry is its advanced power management capabilities.

3. IEEE P1394b (Gigabit 1394, FireWire800):
- extension to 1394
- speed up to 800 Mbps
- support new transferal media

  - POF = plastic optical fiber
  - GOF = glass optical fiber

P1394b adds a new electrical signaling method that permits much higher speeds (800 Mbps and beyond) and easier system implementation. Protocol improvements also significantly improve the efficiency of the bus. "Loop healing" improves ease-of-use by dealing with the pathological case where a user connects the entire bus in a loop.

P1394b introduces substantial improved electrical characteristics. Backward compatibility with P1394-2000 and 1394-1995 devices is easily achieved with a low-cost chip known as a fan-out PHY (for physical layer).

P1394b also supports new transport media in addition to copper cables, including plastic optical fiber (POF), glass optical fiber (GOF), and Cat5 cable. With the new media come extended distances, e.g., 100 meters over Cat5.

4. Wireless IEEE 1394:
- The IEEE 802.15.3 Wireless Personal Area Network standard is designed to connect more than 200 wireless devices.
- Adaptation of the 1394 infrastructure to 802.15.3 makes possible the reuse of existing middleware for audio and video streaming and other multimedia applications.

Specification 1394

There are two levels of interface in IEEE 1394, one for the **backplane** bus within the computer and another for the point-to-point interface between device and computer on the serial cable. A simple bridge connects the two environments. The backplane bus supports 12.5, 25, or 50 megabits per second data transfer. The cable interface supports 100, 200, or 400 megabits per second. Each of these interfaces can handle any of the possible data rates and change from one to another as needed.

The serial bus functions as though devices were in slots within the computer sharing a common memory space. A 64-bit device address allows a great deal of flexibility in configuring devices in chains and trees from a single socket.

IEEE 1394 provides two types of data transfer: **asynchronous** and **isochronous**. Asynchronous is for traditional load-and-store applications where data transfer can be initiated and an application interrupted as a given length of data arrives in a **buffer**. Isochronous data transfer ensures that data flows at a pre-set rate so that an application can handle it in a timed way. For multimedia applications, this kind of data transfer reduces the need for buffering and helps ensure a continuous presentation for the viewer.

The 1394 standard require that a device be within 4.5 meters of the bus socket. Up to 16 devices can be connected in a single chain, each with the 4.5 meter maximum (before signal attenuation begins to occur) so theoretically you could have a device as far away as 72 meters from the computer.

There is a wealth of specifications that build on the base IEEE 1394 specifications. Here are some of the more important. The **Open Host Controller Interface (OHCI)** defines the way 1394 interfaces to a PC host. OHCI provides an operating system like Microsoft Windows a standardized way of interacting with the 1394 bus. **IEC 61883** defines the details for controlling specific audio-video device types over 1394. **Serial Bus Protocol - 2 (SBP-2)** defines standard ways of encapsulating device commands over 1394 and is essential for DVD players, printers, scanners, and other devices. **Home AV Interoperability (HAVi** [http://www.havi.org/]**)** is another layer of protocols for 1394. HAVi is directed at making 1394 devices plug-and-play interoperable in a 1394 network whether or not a PC host is present. Note that the critical pieces of OHCI, IEC61883, and SBP-2 are supported in the most recent versions of Microsoft Windows.

Digital content over 1394 can be robustly protected using **Digital Transmission Copy Protection (DTCP)**. DTCP support device authentication, content encryption, and renew ability, should a DTCP device ever be compromised. Encoding rules can be specified for content, e.g., "copy freely," "copy once," or "copy never." The motion picture industry recognizes DTCP as satisfactory to them in permitting the transmission of their content over 1394.

Comments and limitations

The Firewire cable bus is a 'non-cyclic network with finite branches", consisting of bus bridges and cable devices (nodes). Non-cyclic means that one cannot plug devices together to create a loop. Up to 16 cable hops are allowed between nodes, thus the term finite branches. 6-bit addressing allows up to 63 nodes to be connected to a single bus bridge. Thus, 63 connected devices is a limit for a conventional IEEE 1394 card in a PC.

Each node usually has three connectors, and up to 16 nodes can be connected in a 'Daisy chain" through the connectors with standard cables up to 4.5 m long for a total of 72 m. Special high-quality 'fatter' cables allow longer interconnections. Additional devices can be added in a leaf-node configuration (shown in the next figure). Physical addresses are automatically assigned to the devices on bridge power up (bus reset) and when a new node is added or removed from the system. Hot plugging of the devices is fully supported. No device ID switches are required.

Picture 4: 4-pin and 6-pin FireWire connectors

Firewire serial interface uses a simple cable with two types of small and inexpensive connectors: 4-pin and 6-pin connectors - to carry multiple channels of digital video and video data and control information plus the power.

Typical power consumption for a 1-2 port 1394 serial bus is 1-3 watts. For an Ethernet port it is 3-5 watts. The power needed for an additional port is close to 1 watt (e.g., 4 watts for a 3-port 1394 bus). There are three power class devices: power class 1, 2, 3 with power requirements of 15, 30, and 45 watts, respectively. Adding an extra port will not require a lot of extra power. Devices with 1394 ports and the ability to supply some power to other devices are considered optimal.

Any device attach/detach, or power-turn on, will cause the bus to reset. Before the link is ready for data transfer, a sequence of handshake signals (packets) will be exchanged between the two devices to complete standard procedures such as dynamic node address allocation, self-identification, and arbitration for IRM, BM, and cycle master.

The maximum number of nodes in a bus is 63.

The maximum number of cable hops is 16, where the length of each hop is 4.5 meters (per IEEE 1394-1995). Long Distance 1394 is being standardized in IEEE 1394b.

If a device is in a network, and is not in leaf node, the device must keep its PHY active even if the device is not in operation.

The devices connected in a bus should not form a closed loop where they may hang-up during the self-identification process.

Mixing slower devices with higher-speed devices may hinder performance. For example, two 200 Mbps devices separated by a 100 Mbps device can communicate with only 100 Mbps.

The 1394 serial bus architecture favors high-speed, real-time data transport (isochronous) applications. For asynchronous-only applications, lower bandwidth utilization efficiency (less than 50% of the bus bandwidth) may be expected.

Compare 1394 and USB

These buses have several common features:

Isochronous and asynchronous communication modes used for data transport.
Daisy chaining of devices allowed.
Power sourcing to peripherals with low power requirements.

However, there are some significant differences:

- Performance
- Host controller requirement
- Peer-to-peer communication
- Cost
- Interrupt capability

On the other hand, IEEE 1394-1995 is a high-speed (maximum 400 Mbps) serial bus which does not require an external host controller (such as a PC), and is designed to be used in a home environment with or without a PC. For example, homes may have an STB, a TV, and DVCR, but no PC. The 1394 bus allows peer-to-peer data communication and, depending upon need, consumer devices may be built with scaled capabilities. A device with IRM capability can act as a bus manager. Devices such as an STB, a DVD player, and a DVCR (digital VCR) will have IRM. Implementation of a 1394 port with IRM capability is much more expensive than USB ports built into peripheral devices. In general, even a minimum implementation of a 1394 port will be more expensive than that of a USB port in a slave device (such as peripherals). There is no interrupt capability in a 1394 bus.

On the USB, the host port (e.g., PC) will source power. Peripheral devices will use either their own power or sink power from the bus. Devices can be built with a 1394 port, which may sink, source, or use their own power. Each device with a 1394 port will have a power class code (0-8). During the self-identification process, this code will be included in the self-id packet. From the power class information, the bus manager recognizes what

device can source power and what device needs power. The USB uses a 4-wire cable-two for the differential signal, one for +5 volts, and the fourth for the ground. The 1394 bus uses a 6-wire cable-one pair for data, one pair for the strobe, and the third pair for power. The 1394 bus also allows a simplified low-cost cable with four conductors for data and strobe but no power. Typical voltage on the 1394 bus is 24 volts.

Table 6: IEEE 1394 vs. USB 1.1

	IEEE 1394	**USB 1.1**
Max. number of devices	63	127
Hot-swap	yes	yes
Plug and Play	yes	yes
Length of cable between devices	4.5 m	5 m
Speed	400 Mbps	12 Mbps
Need host controller (PC)	no	yes
Peer-to-peer	yes	no
Cost	expensive	simple, not expensive
Interrupt ability	no	yes
Cable	6/4-conductors	4-conductors

U. Standard USB

The **USB [Universal Serial Bus]** specification defines the Mechanical, Electrical and Protocol layers of the interface. USB defines two types of hardware, **Hubs** and **Functions**. Up to 127 devices may be connected together in a tiered **Star topology**. The limiting factor being seven address bits. Wire segments are point-to-point between a Host, Hub, or Function. The system may only have one Host.

The USB bus is a [Differential] Bi-directional serial interface cable bus. Differential NRZI data is transmitted Isochronous or Asynchronous between devices. Data is transferred at three different rates over a maximum cable length of 4 meters ~ over 4 wires, 2 of which carry data on a balanced twisted pair, one for power +5 V (0.5 A) and one for ground.

A Slow-Speed mode of 1.5Mbps is used for devices such as mice. Full-Speed mode is used by most devices and allows a transfer rate of 12Mbps. High-Speed mode [defined by USB 2.0] allows rates of 480Mbps. Transmission at the High-Speed mode requires the addition of 45 ohm termination resistors between each data line and ground. Operation at Full-Speed mode is 2.8 volts [High] to 0.3 volts [Low]. Operation at High-Speed mode is at 400mV +/-10% [High] to 0V +/- 10mV [Low]. Cable impedance for both modes is 90 ohms +/- 15% (differential).

Four different (packet) protocols are used: Control, Interrupt, Isochronous and Bulk. Each exchange contains three packets; A token packet which holds the address, a data packet which holds the data, and a handshake packet which terminate the exchange. NRZI produces a change in the signal indicating a logic zero, no change indicates a logic one. Bit stuffing is used with NRZI to stop the signal remaining in the steady state condition; if more then 6 ones are transmitted (no change in the signal) a zero is inserted to produce a transition. NRZI, with bit stuffing is self-clocking, allowing the receiver to synchronize with the transmitter.

Devices can draw power directly from the system, from an attached self-powered hub, or be connected to their own power supply.

USB is still a serial-type interface and sends bits one after another...

USB makes adding peripheral devices very easy and is quickly replacing different kinds of serial and parallel port connectors with one standardized plug and port combination. One can now connect a USB-capable mouse, keyboard, digital joystick, and a scanner, a set of digital speakers, a digital camera or a PC telephone to the computer.

In theory, a USB interface can support up to 127 individual USB peripherals at one time. The practical maximum number of devices is less since some of them reserve USB bandwidth. Additional PCI-based USB cards provide an independent USB bus so that even more peripheral devices can be connected.

For practical connection of multiple devices to the host (root), special hubs are required. Hubs notify the host when nodes (devices) attach or detach from the hub to provide the real-time reconfiguration of the system and device identification. Hubs can have up to seven connectors to nodes or other hubs. They could be self-powered or powered by the host.

USB is generally described as having a tiered star topology, however each device communicates with the host as if it had its own connection. This means that communication from the host centers around a set of hubs/devices, each of which in-turn

serves as the center for another set of hubs/devices, etc. However, the hubs are transparent to the software and the devices are addressed individually. Cables are used to create point-to-point connections between devices and USB ports, or to connect one USB hub to another. The maximum cable length is five meters long. However, a repeater hub may be used to extend the distance between the peripheral and the host. There are also special USB repeaters that can be used to extend the connection even further.

Open Host Controller Interface for USB (OHCI) specifies host controller USB rev. 1.0.

Enhanced Host Controller Interface (EHCI) specifies host controller USB rev. 2.0.

J. SI Units

Table 7: Prefixes for decimal multiplies of SI units

[http://physics.nist.gov/cuu/Units/prefixes.html]

Factor	Name	Symbol
10^{24}	yotta	Y
10^{21}	zetta	Z
10^{18}	exa	E
10^{15}	peta	P
10^{12}	tera	T
10^{9}	giga	G
10^{6}	mega	M
10^{3}	kilo	k
10^{2}	hecto	h
10^{1}	deka	da

Table 8: Prefixes for binary multiplies

[http://physics.nist.gov/cuu/Units/binary.html]

Factor	Name	Symbol	Origin	Derivation
2^{60}	exbi	Ei	exabinary $(2^{10})^6$	exa $(10^3)^6$
2^{50}	pebi	Pi	petabinary $(2^{10})^5$	peta $(10^3)^5$
2^{40}	tebi	Ti	terabinary $(2^{10})^4$	tera $(10^3)^4$
2^{30}	gibi	Gi	gigabinary $(2^{10})^3$	giga $(10^3)^3$
2^{20}	mebi	Mi	megabinary $(2^{10})^2$	mega $(10^3)^2$
2^{10}	kibi	Ki	kilobinary $(2^{10})^1$	kilo $(10^3)^1$

Examples:

1 kibibit = 1 Kibit = 2^{10} bits = 1024 bits
1 kilobit = 1 kbit = 10^3 bits = 1000 bits

1 mebibyte = 1 MiB = 2^{20} B = 1,048,576 B (binary megabyte = 1024x1024)
1 megabyte = 1 MB = 10^6 B = 1,000,000 B (decimal megabyte = 1000x1000)

1 gibibyte = 1 GiB = 2^{30} B = 1,073,741,824 B
1 gigabyte = 1 GB = 10^9 B = 1,000,000,000 B

Capacity in binary megabytes used e.g. DOS FDISK, old CMOS SETUP in ROM, and File Manager in Windows 3.x.

Capacity in decimal megabytes used DOS CHKDSK, new CMOS SETUP and manufacturers of hard drives.

Convert decimal MB to binary MB:

$$\frac{[\text{decimal MB}] \times 1{,}000{,}000}{1{,}048{,}576} = \text{binary MB}$$

Convert binary MB to decimal MB:

binary MB x 1.048576 = decimal MB

O. The BIOS Hard Disk Limitations

Interrupt 13h – Software interface of system BIOS to provide access to hard drive. The problem is in limit of fixed values of CHS (Cylinder/Head/Sector) to 1024x256x63 – capacity is limited to 8 GB.

Interrupt 13h Extensions – Extension of functions of interrupt 13h to breaking the 8 GB limit with maintenance of compatibility with old hard drives and systems. Instead of CHS is recommended LBA.

Table 9: Limitations for INT13h and ATA

Limit	Cylinders		Heads		Sectors		Max. size	Bits
INT13h	1024	10 bits	256	8 bits	63	6 bits	8.4 GB	24
Ext.INT13h	16384	14 bits	16	4 bits	63	6 bits	8.4 GB	24
ATA(28-bits)	65536	16 bits	16	4 bits	255	8 bits	137 GB	28
INT13h+ATA	1024	10 bits	16	4 bits	63	6 bits	528 MB	20
ATA(48-bits)	-	-	-	-	-	-	128 PB	48
Ext.INT13h	-	-	-	-	-	-	16 EB	64

INT13h BIOS:

16384 x 16 x 63 (Ext.INT13h)
1024 x 256 x 63 (INT13h)

= 16,515,072 sectors = 8,455,200,768 bytes = 7.9/8.4 GB

ATA specification (28-bits):

65,536 x 16 x 255 = 267,386,880 sectors = 136,902,082,560 bytes = 127.5/137 GB

Combined INT13h BIOS and ATA (28-bits):

By using the same C/H/S values for INT13h and ATA disk, we get combination of two limitations (smaller of each):

1024 x 16 x 63 = 1,032,192 sectors = 528,482,304 bytes = 504/528 MB

CHS (**C**ylinder-**H**ead-**S**ector) (**Normal**) – CHS is the oldest system for addressing sectors on an ATA-drive. The method is based on that used for ST506 drives. Since every BIOS could work with these drives, ATA was made very compatible with it. A CHS-BIOS does not perform translation, which results in a combination of the lowest values of the BIOS and drive (see table 13). Therefore, the maximum capacity for CHS-BIOS is 1024 x 16 x 63 = 504 MB.

LBA (**Logical Block Address**) – LBA provides a way to address sectors. LBA gives BIOS's a common way to address sectors. LBA does not work with sector per track, cylinders and heads, but it uses logical sector numbers. LBA can support drives to 128 GB. When LBA is turned on, the BIOS will enable geometry translation.

This translation may be done in the same way that it is done in Extended CHS or large mode, or it may be done using a different algorithm called LBA-assist translation. The translated geometry is still what is presented to the operating system for use in Int 13h calls. The difference between LBA and ECHS is that when using ECHS the BIOS translates the parameters used by these calls from the translated geometry to the drive's logical

geometry. With LBA, it translates from the translated geometry directly into a logical block (sector) number.

- 28-bits address: 2^{28} = 268,435,456 sectors (128 GB)
- 48-bits address: 2^{48} = 281,474,976,710,656 sectors (128 PB)

LBA addressing mode implement system BIOS or hard drive adaptor, which translate the CHS parameters passing to BIOS into 28/48-bits LBA address of sector to get data from hard drive. This translation could provide various utility "**dynamic drive overlay**" (software translation) also, e.g. SpeedStor (Storage Dimensions), EZ-Drive (Micro House), Disk Manager (OnTrack).

Compatibility with LBA: AMI BIOS from 25.04.1994 include
Phoenix BIOS from version 4.03 include

Some PCs are using a specific version of Phoenix BIOS 4.03, which does not support LBA.

Assisted LBA – The number of sectors/track is fixed at 63 and the number of heads is 16 or a multiple of that (32, 64, 128, 255). Then the number of cylinders is calculated by dividing the total capacity in sectors by the number of sectors per cylinder:

number of cylinders = x/(63*heads)

where x = total number of sectors
63 = number of sectors per cylinder

This translation is called the **LBA assisted method**. This translation method can only be utilized if the disk drive itself supports LBA addressing.

Table 10: LBA assisted method

Total number of sectors reported	Number of sectors reported	Number of heads reported	Number of cylinders reported	Theoretical maximum capacity
1 X 1,032,192	63	16	X/(63*16)	528.4 MB
1,032,192 X 2,064,384	63	32	X/(63*32)	1.057 GB
2,064,384 X 4,128,768	63	64	X/(63*64)	2.114 GB
4,128,768 X 8,257,536	63	128	X/(63*128)	4.228 GB
8,257,536 X 16,450,560	63	255	X/(63*255)	8.422 GB

Large/ECHS (**Extended CHS**; **CHS to CHS Translation**) – BIOS translation works by having the BIOS act as a "middleman" of sorts between the IDE/ATA hard disk and the standard BIOS Int 13h, and by taking advantage of the fact that one standard allows more heads than the other but fewer cylinders. The BIOS takes the logical geometry that the hard disk specifies according to the IDE/ATA standard, and **translates** it into an equivalent geometry that will "fit" into the maximums allowed by the BIOS Int 13h standard. This is done by dividing the number of logical cylinders by an integer, and then multiplying the number of logical heads by the same number. The technique is sometimes called **bit shift translation** (since the multiplication and division is done by shifting the cylinder and head bits). The capacity is of course unchanged, and the new geometry fits quite nicely into the BIOS limits.

Therefore, the rule for choosing the multiplier is: make the multiplier is as low as it can be to bring the cylinder and heads within the range permitted by INT13h (that is, 1024 cylinders numbered 0 to 1023). To achieve this, the BIOS do a simple loop (with some added error checking):

1. Multiplier = 1
2. Cylinder = Cylinder -1
3. Is Cylinder < 1024? If not:
   Do a right bitwise rotation on the cylinder (i.e., divide by 2)
   Do a left bitwise rotation on the multiplier (i.e., multiply by 2)

Use the multiplier on the Cylinder and Head values to obtain the translated values.
At the end of this loop, the multiplier will be as small as it can be to bring the cylinder to within the permitted range.

Table 11: Bit-Shifting translation

Interval of cylinders	Number of heads
0-1024	16
1025-2046	32
2047-4096	64
4097-8191	128
8192-16384	256
16385-32768	512

The translation in fact is nothing but a conversion of the cylinders and heads in the BIOS to their drive equivalents. If you change the translation mode of your hard disk, you risk permanent loss of all the data on the drive.

Example of translation:

Hard disk Maxtor 85120A 5.1 GB

Factory CHS = 9924 x 16 x 63 = 10,003,392 sectors = 5,121,736,704 bytes = 5.1 GB

Assisted LBA = 622 x 255 x 63 = 9,992,430 sectors = 5,116,124,160 bytes = 5.1 GB
lost = 10962 sectors = 5,612,544 bytes = 5.6 MB

Standard ECHS = 620 x 256 x 63 = 9,999,360 sectors = 5,119,672,320 bytes = 5.1 GB
lost = 4032 sectors = 2,064,384 bytes = 2.0 MB

Revised ECHS = 661 x 240 x 63 = 9,994,320 sectors = 5,117,091,840 bytes = 5.1 GB
(10585 x 15 x 63)
lost = 9072 sectors = 4,644,864 bytes = 4.6 MB

SCSI controllers using own BIOS or device driver to replace services of system BIOS at communication with SCSI hard disk and so there is no limitation for 1024 cylinders (504 MB).

ESDI (Enhanced Small Device Interface) devices using INT13h services in BIOS ROM to provide a translation of device geometry, which is compatible with ATA interface.

28/48-bit LBA in Windows

Drivers in Windows that use the 28-bit logical block addressing (LBA) ATA commands are limited to 128 gigabytes (GB). Drivers that use the new 48-bit LBA ATA commands are limited to 128 petabytes (PB). Examples are as follows:

1. Microsoft Windows XP Service Pack 1, Windows Server 2003, and later versions of Windows support 48-bit LBA ATA commands in the atapi.sys driver.

2. Windows XP (before Service Pack 1) and Windows 2000 Service Pack 3 or later support 48-bit LBA ATA commands in the atapi.sys driver, but support must be enabled by a registry key described under MS KB303013 [http://support.microsoft.com/default.aspx?scid=kb;en-us;303013].

3. All versions of Windows before Windows 2000 Service Pack 3 support 28-bit LBA ATA commands in the atapi.sys driver.

Any system that has a controller that does not use Microsoft's atapi.sys driver will have limitations dependent upon the driver's manufacturer.

O.1 Limit 504/528 MB

First limitation is a problem of fixed values of CHS in INT13h and ATA protocol. Therefore, up to 1,032,192 sectors (1024 x 16 x 63) could be addressed, and with 512 bytes per sector, this yields a maximum theoretical capacity of about 528 MB. This is the root cause of the first drive size boundary, the 528 MB barrier.

If the same values for C/H/S are used for the BIOS Int 13 call and for the ATA disk I/O, then both limitations combine, and one can use at most 1024 cylinders, 16 heads, 63 sectors/track, for a maximum total capacity of 528,482,304 bytes (528 MB), the infamous 504 MiB limit for DOS with an old BIOS. This started being a problem around 1993, and people resorted to all kinds of trickery, both in hardware (LBA), in firmware (translating BIOS), and in software (disk managers). The concept of "translation" was invented: a BIOS could use one geometry while talking to the drive, and another, fake, geometry while talking to DOS, and translate between the two.

Solution: New BIOS with LBA support or bit shift, or LBA assisted translation, eventually Ext. INT13h. Some drive overlay manager utility.

O.2 Limit 2.1 GB

BIOS developers and engineers, in order to solve the 528 MB problem, employed different methods to accomplish resolution. One such solution was to take the top 2 bits from the Int 13h head register and use them for bits 11 and 12 of the cylinder count. By doing this, the maximum head value that can be stored in the remaining 6 bits of the head register is 63 (64 heads total).

The presumption is that all bits of the head register will define the logical head count. Therefore, in order to properly translate a drive with 4,096 physical cylinders you must divide the cylinder count by four (1,024 logical cylinders) and multiply the head count by four (128 logical heads). But, since some of the early BIOS's used the top two bits of the head register as part of the cylinder count, there is no way in which to define 128 heads. A BIOS that handles drives in this fashion may hang during the system POST process, as the BIOS attempts the "Identify Drive" command and tries to set the CHS values.

Limit: $2^{12} = 4096 \rightarrow$ (0-4095 cylinders, 12 bits)

4096 x 16 x 63 = 4,128,768 sectors = 2,113,929,216 bytes = 2.1 GB (1.97 GB)

Solution: New BIOS. Some drive overlay manager utility.

O.3 Limit 3.2 GB

There was a bug in the Phoenix 4.03 and 4.04 BIOS firmware that would cause the system to lock up in the CMOS setup for drives with a capacity over 3277 MB.

Solution: New BIOS.

O.4 Limit 4.2 GB

Unlike the 528 MB barrier, which is both hardware and software related, the 4.2 GB barrier is a limitation imposed by the underlying operating system. Some operating systems store the number of heads reported as an 8-bit value. Therefore, if the BIOS reports 256 heads, then these systems will only be saving the lower eight bits, which will then result in a value of zero and a disk drive that cannot be configured. This occurs any time the device reports 16 heads and greater than 8,192 cylinders to the Bit Shift translation.

Affected operating systems: DOS, Windows 3.x/95 (version 95A); Windows NT 3.x/4.x using FAT file system

When Bit Shift Translation is used with one these operating systems, the maximum capacity that can be configured is 4.2 GB. LBA Assist Translation never reports more than 255 heads so the problem does not exist in situations where LBA Assist Translation is in use.

To access device we use **Revised ECHS** (adjusting the number of heads to 15). A simple way around the 256 head problem was to do a small "pre-translation" on the disk geometry when the number of heads exceeds 8192 that converts the number of heads from 16 to 15:

If cylinders > 8192 and heads = 16

- Heads = 15
- Cylinders = cylinders * 16 / 15 (losing the fraction component)
- Do a standard ECHS translation

To access device we use **Assisted LBA**. The BIOS achieves assisted LBA using the following algorithm (use 63 sector per cylinder always):

If cylinder > 8192

- Variable CH = Total Sectors / 63
- Divide (CH – 1) by 1024 (as an assembler bitwise right shift) and add 1
- Round the result up to the nearest of 16, 32, 64, 128 and 255. This is the value to be used for the number of heads.
- Divide CH by the number of heads. This is the value to be used for the number of cylinders.

8191 cylinders cause transfer 16 heads to 128
8192 cylinders cause transfer 16 heads to 256

Limit: 2^{13} = 8192 → 8191 cylinders (0-8190 cylinders, 13 bits)

8191 x 16 x 63 = 8,256,528 sectors = 4,227,342,336 bytes = 4227 MB (3937 MB)
(a fake geometry is CHS 1024x128x63)

O.5 Limit 7.9/8.4 GB

16384 cylinders cause transfer 16 heads to 256
16385 cylinders cause transfer 16 heads to 512, this is out of range BIOS to serve

Limit: 2^{14} = 16384 → 8191 cylinders (0-8190 cylinders, 14 bits)

16384 x 16 x 63 = 16,515,072 sectors = 8,455,716,864 bytes
= 8456 MB = 8.4 GB (8064 MB = 7.9 GB)
(a fake Assisted LBA geometry is CHS 1024xHx63, where H is a first number of 16, 32, 64, 128, and 255)

According to ATA-3 specification device with more than 8 heads should not use more than 16383 cylinders.

Program DOS FDISK does show disk size up to 8 GB.

Solution: New BIOS with Ext. INT13h support.

O.6 Limit 32 GB

Hard drives over 8.4 GB are supposed to report their geometry CHS as 16383/16/63. This in effect means that the "geometry" is obsolete, and the total disk size can no longer be computed from the geometry. Many BIOS's compute number of cylinders by dividing total capacity with value 16x63. For disk larger than 33.8 GB BIOS obtain number of cylinders greater than 65,535 what cause a failure or BIOS may hang.

O.6.1 Hardware limit

65,535 x 16 x 63 = 66059280 sectors = 33,822,351,360 bytes = 31.5 GB

Solution:
1. New BIOS without bugs.
2. Disks larger than 32 GB may have a special jumper to limit capacity up to 32 GB.
3. In CMOS set up 15 instead 16 heads.

O.6.2 Windows 95 limit

Any Windows 95 version does not support disk greater than 32 GB (see KB246818 [http://support.microsoft.com/default.aspx?scid=kb;EN-US;246818]).

Solution: none; use another operating system (upgrade to Microsoft Windows 98 or Microsoft Windows NT).

O.6.3 ScanDisk limit

If you use the protected-mode (graphical) version of ScanDisk to perform a thorough scan (which includes a surface scan) on an ATA hard disk that is larger than 32 GB in size, ScanDisk may report errors on every cluster after approximately cluster number

967,393 (see MS KB243450 [http://support.microsoft.com/default.aspx?scid=kb;EN-US;243450]).

Graphical (protected) version of ScanDisk in Windows 9x has a problem at surface scanning with disk larger than 32 GB – it may show error for every cluster after cluster number 967,393

Cause: This problem may occur on computers that use a Phoenix BIOS and use the Phoenix BitShift translation algorithm to report the geometry of large ATA hard disks. On such computers, the Windows protected-mode ATA disk driver (**Esdi_506.pdr**) may not correctly recognize the translation mode for the drive, resulting in an inability to access areas of the drive beyond the first 32 GB.

This problem does not occur if the BIOS use logical block addressing (LBA) Assist translation instead of Phoenix BitShift translation.

Solution: A supported fix is available from Microsoft Update site for Windows 98/98 SE.

O.7 Limit 64 GB

This limitation is software based only.

O.7.1 FDISK

Fdisk Does Not Recognize Full Size of Hard Disks Larger than 64 GB – see MS KB263044 [http://support.microsoft.com/default.aspx?scid=kb;en-us;263044]. When you use Fdisk.exe to partition a hard disk that is larger than 64 GB in size (68,719,476,736 bytes), Fdisk does not report the correct size of the hard disk.

The size that Fdisk reports is the full size of the hard disk minus 64 GB. For example, if the physical drive is 70.3 GB (75,484,122,112 bytes) in size, Fdisk reports the drive as being 6.3 GB (6,764,579,840 bytes) in size. Fdisk uses some 16-bit values internally to calculate the size of the drive. Some of these variables overflow when the drive size is equal to or larger than 64 GB.

Solution: A supported fix of FDISK is available from Microsoft for Windows 98/98SE (not for Windows 95). This hot fix is not designed for 48-bit logical block addressing (LBA) hard disks, and it is not supported on hard disks larger than 137 GB. Windows Me/2000/XP/2003 is not affected with this problem.

O.7.2 FORMAT

Format Displays Size of Partitions or Logical Drives Larger Than 64 GB Incorrectly in Windows 9X/Millennium – see MS KB263045 [http://support.microsoft.com/default.aspx?scid=kb;EN-US;263045]. When you use **Format.com** to format a partition or logical drive that is larger than 64 GB (68,719,476,736 bytes) in size, Format.com does not report the correct size of the drive being formatted at the beginning of the format process. However, as the formatting process progresses, the entire drive are formatted and the correct formatted size is displayed when the operation is finished.

Format.com uses some 16-bit values internally to calculate the initial displayed size of the drive. Some of these variables overflow when the drive size is equal to or larger than 64 GB. This is a display (or cosmetic) issue only; the drive is formatted to its full size.

This problem does not occur if you format the drive from within Windows Explorer.

O.8 Limit 137 GB

This is a limit of ATA specification with 28-bits logical addressing.

Solution: BIOS and device with 48-bits logical addressing support (ATA/ATAPI-6).

Implementation of 48-bit addressing required for hard drives larger than 137.4 GB.

O.9 Limit 512 GB

Fdisk.exe Unable to Partition Drives Larger Than 512 Gigabytes (see MS KB280737 [http://support.microsoft.com/default.aspx?scid=kb;en-us;280737&Product=w98])

When you attempt to partition your hard disk by using the Fdisk utility, you may be unable to create a partition that is larger than 512 GB. This issue occurs because the Fdisk utility is limited to creating partitions with a maximum size of 512 GB.

When you use the Fdisk utility to partition a drive that is larger than 512 GB, you might not receive any error messages that state that the drive was not partitioned correctly; however, the drive might not be partitioned correctly. For this reason, use alternative programs when you partition drives that are larger than 512 GB.

Cause: Fdisk utility is limited to creating partitions with a maximum size of 512 GB.

Solution: To work around this issue and create partitions that are larger than 512 GB, do not use Fdisk to partition your hard disk. You can use the Windows Millennium Edition (Me) Setup boot disk that is provided with the full version of Windows Me to partition new drives before you install Windows.

Some third-party programs also contain programs that can partition drives that are larger than 512 GB. For example, Ghost from Symantec Corporation contains a program called Gdisk that partitions drives that are larger than 512 GB.

O.10 Limit 2.2 TB

This is a limit of FAT32 file system in Windows 98.

Solution: use another file system like NTFS

O.11 Limit 128 PB

This is a limit of ATA specification with 48-bits logical addressing.

Solution: New standard with 64-bits logical addressing or another standard…

P. Parallel and Serial interface

For over 20 years, the parallel bus interface has been the mainstream storage interconnects for most storage systems. However, increasing bandwidth and flexibility demands have exposed inefficiencies in the two main parallel interface technologies: **ATA** and **SCSI**. The lack of compatibility between parallel ATA and SCSI - including different connectors, cables and software - increases costs for inventory management, R&D, training and product qualification.

Parallel technology poses still other challenges. Parallel transmissions are susceptible to crosstalk across wide ribbon cable paths. This crosstalk adds line noise and can cause signal errors, a pitfall that has been remedied by slowing the signal, limiting cable length or both. **Terminating parallel signals** is also difficult, requiring individual lines to be terminated, usually by the last drive, to avoid signal reflection at the end of a cable. Finally, parallel's large cable and connector size make it unsuitable for increasingly dense computing environments.

P.1 Introducing SAS and SATA

Serial technology, specifically **Serial ATA** (SATA) and **Serial Attached SCSI** (SAS), addresses the architectural limitations of its parallel counterparts. The technology draws its name from the way it transmits signals, that is, in a **single stream** (serially) compared with the **multiple streams** found in parallel technology. The main advantage of serial technology is that while it moves data in a single stream, it does so much faster than parallel technology because it is not tied to a particular clock speed. Serial technology wraps many bits of data into packets and then transfers the packets up to 30 times faster than parallel down the wire to or from the host.

SATA extends the ATA technology roadmap by delivering disk interconnects speeds starting at 1.5Gbps. Due to its lower cost per gigabyte, SATA will continue as the prevalent disk interface technology in desktop PCs, sub-entry servers and networked storage systems where cost is a primary concern.

SAS, the successor technology to the parallel SCSI interface, leverages proven SCSI functionality and promises to greatly build on the existing capabilities of the enterprise storage connection. SAS offers many features not found in today's mainstream storage solutions. These include drive addressability of up to 16,256 devices per port and reliable point-to-point serial connections at speeds of up to 3Gbps.

In addition, due to its small connector, SAS offers full dual-ported connections on 3.5-in. and smaller 2.5-in. hard disk drives, a feature previously found only on larger 3.5-in. Fibre Channel disk drives. This is an essential feature in applications requiring redundant drive spindles in a dense server form factor such as blade servers.

SAS improves drive addressability and connectivity using an expander that enables one or more SAS host controllers to connect to a large number of drives. Each expander allows connectivity to 128 physical links, which may include other host connections, other SAS expanders or hard disks. This highly scalable connection scheme enables enterprise-level topologies that easily support multi-node clustering for automatic failover availability or load balancing.
In one of its most significant advances, the SAS interface will also be compatible with lower-cost-per-gigabyte SATA drives, giving system builders the flexibility to integrate either SAS or SATA devices while slashing the costs associated with supporting two

separate interfaces. As the next generation of SCSI, SAS bridges the parallel technology gap in performance, scalability and affordability.

P.2 Multiple layers of compatibility

Physical layers

The SAS connector is a universal interconnection that is form-factor compatible with SATA. It allows SAS or SATA drives to plug directly into a SAS environment for mission-critical applications with high-availability and high-performance requirements or lower-cost-per-gigabyte applications such as near-box storage.

SATA connector signals are a subset of SAS signals that enable the compatibility of SATA devices and SAS controllers. SAS drives will not operate on a SATA controller and are keyed to prevent any chance of plugging them in incorrectly.

In addition, the similar SAS and SATA physical interfaces enable a new universal SAS backplane that provides connectivity to both SAS drives and SATA drives. This eliminates the need for separate SCSI and ATA drive backplanes. This consolidation of designs greatly benefits both backplane manufacturers and end users by reducing inventory and design costs.

Protocol layer

SAS consists of three types of protocols, each of which is used to transfer different types of data over the serial interface, depending on which device is being accessed. **Serial SCSI Protocol** (**SSP**) transfers SCSI commands, and **SCSI Management Protocol** (**SMP**) sends management information to expanders. Meanwhile, **SATA Tunneled Protocol** (**STP**) creates a connection that allows transmission of the SATA commands. By including all three of these protocols, SAS provides seamless compatibility with today's existing SCSI applications, management software and SATA devices.

This multi-protocol architecture support, coupled with the compatibility of SAS and SATA's physical connection, allows SAS to operate as the universal interconnection for both SATA and SAS devices.

S. File systems

S.1 File systems FAT/NTFS

FAT = File Allocation System
NTFS = NT File System
Windows NT : NTFS 4.0
Windows 2000/XP: NTFS 5.0

S.1.1 FAT16

Maximum number of files and folders within the root folder of FAT16 is 512. (Long file names can reduce the number of available files and folders in the root folder.) FAT16 supports a maximum of 65,524 clusters per volume.

FAT16 volumes larger than 2 GB are not accessible from computers running MS-DOS, Windows 95, Windows 98, Windows Me, and many other operating systems. This limitation occurs because these operating systems do not support cluster sizes larger than 32 KB, which results in the 2 GB limit.

Clusters are an allocation unit of disk space for files. Cluster size is determined by the size of the disk. Large cluster sizes mean lower disk space efficiency. For example, a 1-byte file requires 32K of disk space if the disk has 32K clusters.

Table 12: Limits of file system FAT16

Partition size	Requested cluster size for FAT16 volume
0-127 MB	2 KB = 2048 B
128-255 MB	4 KB = 4096 B
256-511 MB	8 KB = 8192 B
512-1023 MB	16 KB = 16384 B
1024-2047 MB	32 KB = 32768 B

S.1.2 FAT32

FAT32 – first version of FAT32 introduced in 1996 with Windows 95B OSR2.0.
FAT32X – second version introduced in 1997 with Windows 95C OSR2.5; support large disk over 8 GB using Extended INT13h.

A FAT32 volume must have a minimum of 65,527 clusters. Windows XP Professional can format FAT32 volumes up to 32 GB, but it can mount larger FAT32 volumes created by other operating systems.

In theory, FAT32 volumes can be about 8 terabytes; however, the maximum FAT32 volume size that Windows XP Professional can format is 32 GB. Therefore, you must use NTFS to format volumes larger than 32 GB. However, Windows XP Professional can read and write to larger FAT32 volumes formatted by other operating systems.

FAT32 has no built-in file system security or compression scheme.

The following limitations exist using the FAT32 file system with Windows operating systems:

- Clusters cannot be 64 KB or larger. If clusters were 64 KB or larger, some programs (such as Setup programs) might calculate disk space incorrectly.
- A volume must contain at least 65,527 clusters to use the FAT32 file system. You cannot increase the cluster size on a volume using the FAT32 file system so that it ends up with less than 65,527 clusters.
- The maximum possible number of clusters on a volume using the FAT32 file system is 268,435,445. With a maximum of 32 KB per cluster with space for the file allocation table (FAT), this equates to a maximum disk size of approximately 8 terabytes (TB).
- The ScanDisk tool included with Microsoft Windows 95 and Microsoft Windows 98 is a 16-bit program. Such programs have a single memory block maximum allocation size of 16 MB less 64 KB. Therefore, the Windows 95 or Windows 98 ScanDisk tool cannot process volumes using the FAT32 file system that have a FAT larger than 16 MB less 64 KB in size. A FAT entry on a volume using the FAT32 file system uses 4 bytes, so ScanDisk cannot process the FAT on a volume using the FAT32 file system that defines more than 4,177,920 clusters (including the two reserved clusters). Including the FAT's themselves, this works out, at the maximum of 32 KB per cluster, to a volume size of 127.53 GB.
- You cannot decrease the cluster size on a volume using the FAT32 file system so that the FAT ends up larger than 16 MB less 64 KB in size.
- You cannot format a volume larger than 32 GB in size using the FAT32 file system in Windows 2000. The Windows 2000 FastFAT driver can mount and support volumes larger than 32 GB that use the FAT32 file system (subject to the other limits), but you cannot create one using the Format tool. This behavior is by design. If you need to create a volume larger than 32 GB, use the NTFS file system instead.

NOTE: When attempting to format a FAT32 partition larger than 32 GB, the format fails near the end of the process with the following error: "Logical Disk Manager: Volume size too big."

Table 13: Limits of file system FAT32

Disk size	Required cluster size for FAT32 volume
16 GB	4K
64 GB	16K
128 GB	32K (max. cluster size for Windows 95/98)
256 GB	64K (Windows 95/98/Me don't support)

S.1.3 NTFS

In theory, the maximum NTFS volume size is 2^{64} clusters minus 1 cluster. However, the maximum NTFS volume size as implemented in Windows XP Professional is 2^{32} clusters minus 1 cluster. For example, using 64 KB clusters, the maximum NTFS volume size is 256 terabytes minus 64 KB. Using the default cluster size of 4 KB, the maximum NTFS volume size is 16 terabytes minus 4 KB.

Because partition tables on master boot record (MBR) disks only support partition sizes up to 2 terabytes, you must use **dynamic volumes** to create NTFS volumes over 2 terabytes. Windows XP Professional manages dynamic volumes in a special database instead of in the partition table, so dynamic volumes are not subject to the 2-terabyte physical limits imposed by the partition table. Therefore, dynamic NTFS volumes can be as large as the maximum volume size supported by NTFS. Itanium-based computers that

use **GUID partition table** (**GPT**) disks also support NTFS volumes larger than 2 terabytes.

If you use large numbers of files in an NTFS folder (300,000 or more), disable short-file name generation, and especially if the first six characters of the long file names are similar.

NTFS is so called „journal" file system. It saves boot sector into first and last sector of partition.

Because NTFS data structures are not the same for Windows NT 4.0 and Windows XP Professional, Windows NT 4.0 disk tools such as Chkdsk and Autochk do not work on NTFS volumes formatted or upgraded by Windows XP Professional. These tools check the version stamp of NTFS. After installing Windows XP Professional, you must run the updated version of these disk tools on their NTFS volumes.

Table 14: Compare NTFS 4 with 5

Parameter	NTFS 4	NTFS 5
Alternate Streams	yes	yes
Compression	yes	yes
Encryption	no	yes
Object Permissions	yes	yes
Disk Quotas	no	yes
Sparse Files	no	yes
Reparse Points	no	yes
Volume Mount Point	no	yes

S.1.4 Size limitations

FAT12: max. 4086 data clusters

Size of cluster for all file systems:

Windows 95/98/Me: a power of 2 between 512 bytes and 32,768 bytes, inclusive
Windows XP/2000/NT: a power of 2 between 512 bytes and 65,536 bytes, inclusive

Table 15: Size limitations for file systems

	FAT16	FAT32	NTFS
Max. file size	4 GB – 1 (2^{32} – 1 bytes)	4 GB – 1 (2^{32} - 1 bytes)	16 EB – 1 (2^{44} – 64 KB) (design to 2^{64} - 1 bytes)
Min. volume size	4085 clusters	65,535 clusters	1 MB
Max. volume size	2 GB (max. 65,524 clusters)	2 TB (theoretical 2^{28} clusters) **Windows 2000/XP**: format up to 32 GB, can mount/convert larger volumes **Windows Me**: up to 268,435,444 clusters (2^{28}-12) **Windows 95/98**: 4,177,918 clusters	16 EB theoretical 2^{64} clusters; actual 2^{32} clusters (4,294,967,296 clusters)
Files per volume	65,536 (2^{16})	4,177,920 (2^{28})	4,294,967,295 (2^{32} - 1)
Directory size	(2^{16}-2) 65,534 directory entries; special limit for root directory (root=512 files)	(2^{16}-2) 65,534 directory entries in single folder (root without limit)	no limit
File names	DOS 8.3	255 characters of system set	255 Unicode characters

S.1.5 DVD formats

Table 16: DVD formats

	Double-side capacity	Read/write support/sequence
DVD-Video (video playback) **DVD-ROM** (data storage only)	17 GB	Read only
DVD-R (data storage)	4.7 GB	Read and one-time write
DVD-RAM (video playback, data storage)	9.4 GB	Read and up to 100000 rewrites; random
DVD-RW (video playback, data storage)	9.4 GB	Read and up to 1000 rewrites; sequential
DVD+RW (video playback, data storage)	9.4 GB	Read and rewrites; random

Windows XP uses for DVD-RAM file system FAT32 for read/write and support use a format **UDF** (Universal Disk Format) only for read. DVD-RAM support multisession recording or LBA addressing.

S.2 MBR (Master Boot Record)

The master boot record is always located at cylinder 0, head 0, and sector 1, the first sector on the disk. The master boot record contains the following structures:

- **Master Boot Code** (boot loader, 446 bytes): The master boot record contains the small initial boot program that the BIOS loads and executes to start the boot process. This program eventually transfers control to the boot program stored on whichever partition is used for booting the PC.
- **Master Partition Table** (4x16 bytes, started at 1BEh): This small table contains the descriptions of the partitions that are contained on the hard disk. There is only room in the master partition table for the information describing four partitions. Therefore, a hard disk can have only four true partitions, also called *primary partitions*. Any additional partitions are logical partitions that are linked to one of the primary partitions. One of the partitions is marked as active, indicating that it is the one that the computer should use for booting up.

  Early MSDOS filled the partition table starting at the end. In particular, in the case of only one partition, the descriptor was stored in the fourth primary slot. These days DOS FDISK starts at the beginning, but other systems, like UnixWare, still start at the end. Also Iomega writes the single partition of a ZIP disk in the last entry (so that it has to be mounted as /dev/sda4 or /dev/hdc4 or so).
- **Boot Record Signature** (2 bytes): The sector ends with the Word-sized signature ID of **AA55h** (often called the sector's Magic number; on Intel CPU systems, hex Words are stored with the Low-byte first and the High-byte last).

DR-DOS stores a password starting at offset 1B6h.

Windows 2000 /XP to display Error Messages on screen use the three bytes at offsets 1B5h through 1B7h - these three bytes are being used to reference the offset in memory of the first byte of each Error Message that can be displayed on screen at boot up.

The four bytes from offsets 1B8h through 1BBh are called the Windows 2000/XP **Disk Signature** or Windows NT **Drive Serial Number (volume ID)** (e.g. A8h E1h A8h E1h). This four-byte Hex Word will be found in various keys of the Registry as E1A8E1A8 Hex. These particular keys:

HKEY_LOCAL_MACHINE\SYSTEM\MountedDevices
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Enum\STORAGE\Volume
HKLM\SYSTEM\ControlSet001\Control\DeviceClasses\

Disk signature is used to map drive letters to disks: in the registry item

HKEY_LOCAL_MACHINE\SYSTEM\MountedDevices

the drive letter is coupled with this disk signature. It is used as a disk label to map disk info to disks in the registry item

HKEY_LOCAL_MACHINE\SYSTEM\DISK

The Disk Administrator generates this signature when it initializes the disk, unless there already was a nonzero value there. Also LILO v20 and later preserves this area.

The OS/2 fdisk writes some strange length in the descriptor of the last extended partition. This is probably a bug. OS/2 fdisk fails to update the length of the (outer) extended partition when a primary partition is created in the free space (space not used by a logical partition) at the end of this extended partition. This can lead to overlapping partitions.

OS/2 FDISK does not know about type 0Fh, but accepts DOS Extended Partitions extending beyond cylinder 1023. When some other partition handler, like Partition Magic 4.0, changes the type of a large extended partition from 05h to 0Fh, OS/2 loses access. OS/2 Boot Manager keeps a private copy of the partition table data. This leads to problems when changing the partition table with 3rd party tools.

Windows 2000 tries to destroy OS/2 Boot Manager. Upon boot it ignores the 0Ah partition ID, and sees something resembling a FAT boot sector describing 2 FAT copies. When FASTFAT.SYS marks this partition as clean in the first reserved FAT entry, the mirror (2nd) FAT sector is also updated. However, there is no mirror FAT, and FASTFAT.SYS writes into the middle of the OS/2 Boot Manager code. This aggression was built into FASTFAT.SYS at a fairly late stage, and prerelease versions work without problems.

S.3 Type of disk partitions

The following table contains maximal sizes of partitions for MS-DOS versions:

Table 17: Type of partitions for MS-DOS

MS-DOS version	Max. Capacity of disk/partition
1.0 (1980)	no hard disk support
2.0 (1983)	disk max. 16 MB (FAT12)
2.1	disk max. 32 MB
3.0	only 1 partition max. 32 MB (FAT16)
3.3+	every partition is max. 32 MB
4.0	partition max. 512 MB
5.0+	partition max. 2 GB

MS-DOS 3.3 introduces support for more than one logical drive per hard disk. Logical drives are treated as completely separate disks under MS-DOS, even though they may occupy the same physical hard disk.

This is supported by using no bootable MS-DOS partitions known as **extended MS-DOS partitions**. Fdisk reports these as EXT DOS; other MS-DOS partitions are reported as PRI DOS (for primary MS-DOS). Each primary MS-DOS partition is a logical drive, and extended MS-DOS partitions contain from 1 to 23 logical drives (MS-DOS supports drive letters up to Z). Logical drives in extended MS-DOS partitions have the same FAT type as a primary MS-DOS partition of the same size.

Only one PRI DOS partition and one EXT DOS partition is allowed per drive. On computers with two physical hard disks, a PRI DOS partition is not required on the second physical disk. A PRI DOS partition is required on the first physical disk. (MS-DOS does not support more than two physical disks.)

NOTE: Type 0Eh is the same partition type as 06h, and 0Fh is the same as 05h. However, applications should use the (LBA) INT13h extension's read/write functions to read from or write to the drive instead of the normal Cylinder/Head/SectorPerTrack

(CHS) INT13h functions because the hard disk has more than 1024 cylinders and/or more than 16,711,680 sectors. Note that earlier versions of FDISK recognize these Windows 95 partitions as NON-DOS partitions.

Windows 95 OEM Service Release 2 (OSR2) and Windows 98 support two new partition types (0Bh and 0Ch) to support the FAT32 file system.

Some partitions ID's imply a particular method of disk access. In particular, ID's 0Ch, 0Eh, 0Fh (the LBA versions of 0Bh, 06h, 05h) go with partition table entries that have C/H/S = 1023/255/63 and expect access via the extended INT13h functions (AH=4x) of the BIOS.

MS-DOS can read only partition 01h, 04h, 05h or 06h. With type 05h DOS/Windows will not use the extended BIOS call, even if it is available.

If a partition table entry of type 42h is present in the legacy partition table, then W2K ignores the legacy partition table and uses a proprietary partition table and a proprietary partitioning scheme (**LDM** or **DDM**). As the Microsoft Knowledge Base writes: *Pure dynamic disks (those not containing any hard-linked partitions) have only a single partition table entry (type **42**) to define the entire disk. Dynamic disks store their volume configuration in a database located in a 1-MB private region at the end of each dynamic disk.*

Windows NT 4.0 or earlier will add 80h to the partition type for partitions that are part of a **Fault Tolerant set** (mirrored or in a RAID-5 volume). Thus, one gets types 86h, 87h, 8Bh, 8Ch. Windows NT does not recognize the four W95 types 0Bh, 0Ch, 0Eh, 0Fh.

Some partitions are classified as **hidden partition**. The partition ID of the hidden partitions is different from their visible counterpart only by the hex number 10h, which is added on the partition ID of the visible partition. And in order to make a partition visible only this value 10h needs to be subtracted from the partition ID. The OS/2 boot manager uses the same scheme for hiding partitions.

FDISK from **FreeDOS** hides partition 01h, 04-06h, 0Bh, 0Ch, 0Eh, 0Fh by added 8Ch.

Amoeba is a distributed operating system written by Andy Tanenbaum, together with Frans Kaashoek, Sape Mullender, Robert van Renesse and others since 1981. It runs on PCs (386 and up), Sun3, Sparc, 68030. It is free for universities for research/teaching purposes. (See ftp://ftp.cs.vu.nl/pub/amoeba)

BSDI (Berkeley Software Design, Inc.) was founded by former CSRG (UCB Computer Systems Research Group) members. Their operating system, based on Net/2, was called **BSD/386**. After the USL (Unix System Laboratories, Inc./Novell Corp.) vs. BSDI lawsuit, new releases were based on BSD4.4-Lite. Now they are announcing BSD/OS V2.0.1. This is an operating for PCs (386 and up). The current partition id is 0Fh.

386BSD is a Unix-like operating system, a port of 4.3BSD Net/2 to the PC done by Bill Jolitz around 1991. When Jolitz seemed to stop development, an updated version was called FreeBSD (1992). The outcome of a Novell vs. UCB lawsuit was that Net/2 contained AT&T code, and hence was not free, but that 4.4BSD-Lite was free. After that, FreeBSD and NetBSD were restructured, and FreeBSD 2.0 and NetBSD 1.0 are based on 4.4BSD-Lite. FreeBSD runs on PCs. See http://www.freebsd.org/FreeBSD.html. For NetBSD, it changed partition type to A9h.

BootWizard 4.0 and its new version Acronis OS Selector 5.0 use this ID BBh:

1. when hiding partitions with types other than 01h, 04h, 06h, 07h, 0Bh, 0Ch, 0Eh

2. when creating a partition without file system

REAL/32 is a continuation of DR Multiuser DOS. REAL/32 supports the standard FAT12, FAT16 partition types and will shortly support FAT32. For partitions, which have been marked as secure, we use C0h and D0h as partition markers (C0h < 32 MB, D0h >= 32 MB). REAL/32 is an advanced 32-bit multitasking & multi-user MS-DOS & Windows compatible operating system. (See http://www.imsltd.com/)

Xenix is an old port of Unix V7. Microsoft Xenix OS was announced August 1980, a portable and commercial version of the Unix operating system for the Intel 8086, Zilog Z8000, Motorola M68000 and Digital Equipment PDP-11. Microsoft introduces XENIX 3.0 in April 1983. SCO delivered its first Xenix for 8088/8086 in 1983.

Diagnostics partition ID 12h

Compaq config partition ID 12h is used by Compaq for their configuration utility partition. It is a FAT-compatible partition (about 6-40 MB) that boots into their utilities, and can be added to a LILO menu as if it were MS-DOS. ID 12 is used by the Compaq Contura to denote its hibernation partition.

NCR has used ID 12h MS-DOS partitions for diagnostics and firmware support on their WorldMark systems since the mid-90s. DataLight's ROM-DOS has replaced MS-DOS on systems that are more recent. Partition sizes were once 72 MB (MS-DOS) but are now 40 MB (ROM-DOS).

Intel has begun offering ROM-DOS based "**Service Partition**" support on many OEM systems. This support initially used ID 98h but has recently changed to ID 12h. Intel provides his own support for this partition in the form of a System Resource CD. Partition size has remained constant at 40 MB.

Plan 9 [http://plan9.bell-labs.com/plan9dist/index.html] is an operating system developed at Bell Labs for much architecture. Originally Plan 9 used an unallocated portion at the end of the disk. Plan 9 3rd edition uses partitions of type 39h, subdivided into sub partitions described in the Plan 9 partition table in the second sector of the partition

When a **PowerQuest** [http://www.powerquest.com/] product like **PartitionMagic** or **Drive Image** makes changes to the disk, it first changes the type flag to 3Ch so that the OS won't try to modify it etc. At the end of the process, it is changed back to what it was at first. So, the only time you should see a 3Ch type flag is if the process was interrupted somehow (power outage, user reboot etc). If you change it back manually with a partition table editor or something then most of the time everything is okay.

Table 18: Partition ID description

Value	Name/Description
00h	empty/unused partition table entry
01h	DOS FAT12, max. 15 MB
02h	XENIX root file system
03h	XENIX /usr file system (obsolete)
04h	DOS 3.0+ FAT16 (max. 16-32 MB)
05h	DOS 3.3+ extended partition (max. 8.4 GB)
06h	DOS 3.31+ FAT16 (Large File System, BigDOS, over 32 MB, max. 2 GB) Partitions, or at least the FAT16 file systems created on them, are at most 2 GB for DOS and Windows 95/98 (at most 65,536 clusters, each at most 32 KB). Windows NT can create up to 4 GB FAT16 file systems (using 64 KB

Value	Name/Description
	clusters), but these cause problems for DOS and Windows 95/98. Note that VFAT is 16-bit FAT with long filenames; FAT32 is a different file system.
07h	1. Windows NT NTFS It is rumored that the Windows NT boot partition must be primary, and within the first 2 GB of the disk. 2. OS/2 HPFS 3. Advanced Unix 4. QNX 2.x pre-1988 (see partition boot record; could be any of the above or others)
08h	1. OS/2 (v1.0-1.3 only) 2. AIX bootable partition 3. SplitDrive 4. Commodore DOS 5. QNX 1.x and 2.x ("qny") (see QNX Partitions) [http://www.qnx.com/literature/qnx_sysarch/fsys.html#RAWVOLUMES] 6. DELL partition spanning multiple drives
09h	1. AIX data partition 2. Coherent file system (UNIX-type OS for 286/386/486 from 1980-1995) 3. QNX 1.x and 2.x ("qnz")
0Ah	1. OS/2 Boot Manager 2. Coherent swap partition 3. OPUS (Open Parallel Unisys Server)
0Bh	Windows FAT32, max. 2047 GB (see Partition Types) [http://support.microsoft.com/support/msdn/sdk/platforms/doc/sdk/win32/95guide/src/fat32ovr_4.asp]
0Ch	Windows FAT32 LBA-mapped Ext.INT 13h equivalent of 0Bh; Extended FAT32-Partition (Windows FAT32 over 8.4 GB, using LBA-mode INT 13h extensions)
0Dh	FAT16 Windows (?)
0Eh	Windows DOS FAT16 LBA-mapped logical-block-addressable VFAT (same as 06h but using LBA-mode INT 13h) Extended Fat16-Partition
0Fh	Windows Extended partition LBA-mapped logical-block-addressable VFAT (same as 05h but using LBA-mode INT 13h); type 0Fh is used instead of 05h if extended partition exceed 1024 cylinders (Windows 95B/98); Primary FAT16-Partition (Windows 95) Windows 95 uses 0Eh and 0Fh as the extended-INT13h equivalents of 06h and 05h.
10h	OPUS
11h	OS/2 Boot Manager hidden FAT12 DOS partition
12h	1. Compaq Diagnostics/hibernation FAT partition 2. Intel ROM-DOS Service Partition 3. EISA partition
13h	Reliable Systems FTFS
14h	1. (resulted from using Novell DOS 7.0 FDISK to delete Linux Native part) 2. OS/2 Boot Manager hidden FAT16 DOS partition up to 32 MB
15h	Extended partition hidden
16h	OS/2 Boot Manager hidden over 32 MB FAT16 DOS partition
17h	1. OS/2 Boot Manager hidden HPFS partition 2. Windows NTFS hidden
18h	**AST** Windows swap file (Zero-Volt Suspend/SmartSleep partition) size is 2 MB+amount of memory

Value	Name/Description
	[http://www.ast.com/]
19h	Willowtech Photon coS
1Bh	Windows FAT32 partition hidden
1Ch	Windows FAT32 partition LBA-mapped hidden (using LBA-mode INT 13h extensions)
1Eh	Windows FAT16 partition hidden (LBA VFAT)
20h	Willowsoft Overture File System (OFS1)
21h	1. officially listed as reserved (HP Volume Expansion, SpeedStor variant) 2. FSo2 (Oxygen File System)
22h	Oxygen Extended Partition Table
23h	officially listed as reserved
24h	NEC MS-DOS 3.x
26h	officially listed as reserved
31h	officially listed as reserved
32h	NOS (Alien Internet Services in Melbourne Australia)
33h	officially listed as reserved
34h	officially listed as reserved
35h	JFS on OS/2 or eComStation (non-bootable file system)
36h	officially listed as reserved
38h	Theos version 3.2 (2 GB)
39h	1. Theos version 4 spanned partition 2. Plan 9 from Bell Labs
3Ah	**Theos** version 4 (4 GB) [http://www.theos-software.com/]
3Bh	Theos version 4 extended partition
3Ch	PowerQuest PartitionMagic/DriveImage recovery partition
3Dh	Hidden NetWare
40h	VENIX 80286 A very old Unix-like operating system for PCs.
41h	1. Personal RISC Boot 2. PowerPC Reference Platform Boot 3. **Linux/MINIX** (sharing disk with DRDOS) Very old FAQ has recommended using 41h etc. instead of 81h. etc on a disk shared with DRDOS because DRDOS allegedly disregards the high order bit of the partition type.
42h	1. Windows 2000 dynamic extended partition marker (pure dynamic disks) 2. Linux swap (sharing disk with DRDOS) 3. SFS (Secure File System) for DOS SFS is an encrypted file system driver for DOS on 386+ PCs, written by Peter Gutmann.
43h	Linux native (sharing disk with DRDOS)
44h	GoBack partition **GoBack** is a utility that records changes made to the disk, allowing you to view or go back to some earlier state. It takes over disk I/O like a Disk Manager would, and stores its logs in its own partition. [http://www.goback.com/]
45h	1. **Boot-US** boot manager Occupies a single cylinder below 8 GB. This partition does not contain file system only boot manager. [http://www.boot-us.com/] 2. Priam 3. EUMEL/Ergos L3 Elan (**Elan** was the programming language used.)

Value	Name/Description
	[http://os.inf.tu-dresden.de/L4/l3elan.html]
46h	EUMEL/Ergos L3 Elan
47h	EUMEL/Ergos L3 Elan
48h	EUMEL/Ergos L3 Elan
49h	Phoenix Protected Area (PPA)
4Ah	1. AdaOS Aquila primary 2. ALFS/THIN lightweight file system for DOS (Mark Aitchison)
4Ch	Oberon This partition type (decimal 76) is used for the Aos file system. Type 4Fh is used for the Nat file system. One may have several partitions of this type. (see http://www.oberon.ethz.ch/betadocu.html#PM)
4Dh	QNX 4.x
4Eh	QNX 4.x 2nd part
4Fh	1. QNX 4.x 3rd part QNX is a POSIX-certified, microkernel, distributed, fault-tolerant OS for the 386 and up, including support for the 386EX in embedded applications. ID 07h is outdated - QNX2 used 07h, QNX4.x uses 4Dh, and optionally 4Eh and 4Fh for additional QNX partitions on a single drive. (see http://www.qnx.com/, ftp://ftp.qnx.com/, QNX Partitions, Neutrino file systems) 2. **Oberon** boot/data partition [http://www.oberon.ethz.ch/native/]
50h	1. **OnTrack** Disk Manager (older versions), read-only partition Disk Manager is program from OnTrack to access ATA disks larger than 504 MB under DOS. Linux kernel version older than 1.3.14 cannot be used together with DM. [http://www.ontrack.com/] 2. **Lynx** RTOS [http://www.lynuxworks.com/] 3. Native Oberon (alt)
51h	1. OnTrack Disk Manager (DM6 Aux1), read/write partition 2. Novell52
52h	1. CP/M 2. Microport System V/386
53h	OnTrack Disk Manager 6.0 Aux3, write-only partition?
54h	OnTrack Disk Manager 6.0 DDO (Dynamic Drive Overlay)
55h	StorageSoft EZ-BIOS - EZ-Drive, Maxtor, MaxBlast, and DriveGuide (see also INT 13h/AH=FFh "EZ-Drive") EZ-Drive is another disk manager (**MicroHouse**, 1992). **StorageSoft** is a new mark for EZDrive and DrivePro. [http://www.storagesoft.com/] Linux kernel version older than 1.3.29 cannot be used with EZD.
56h	1. GoldenBow VFeature Volume (Disk Manager type) This is non-standard DOS volume. 2. StorageSoft DM converted to EZ-BIOS
57h	1. StorageSoft DrivePro 2. Netware VNDI Partition
5Ch	Priam EDISK (Disk Manager type)
61h	SpeedStor (Disk Manager type)
63h	1. Unix System V/386, 386/ix (SCO, ISC Unix, UnixWare...) A Unixware 7.1 partition must start below the 4GB limit. (If the /stand/stage3.blm is located past this limit, booting will fail with "FATAL BOOT ERROR: Can't load stage3".) 2. Mach, MtXinu BSD 4.3 on Mach

Value	Name/Description
	3. GNU HURD
64h	1. Novell Netware 286/2.xx 2. PC-ARMOUR protected partition by Dr. A. Solomon 3. SpeedStore
65h	Novell Netware 386/3.xx/4.xx
66h	Novell Netware SMS Partition SMS (Storage Management Services)
67h	Novell/Wolf Mountain
68h	Novell
69h	Novell Netware 5+, Novell Netware NSS Partition NSS (Novell Storage Services)
70h	DiskSecure Multi-Boot
71h	officially listed as reserved
73h	officially listed as reserved
74h	1. officially listed as reserved 2. Scramdisk partition (disk encryption software)
75h	IBM PC/IX
76h	officially listed as reserved
77h	1. M2FS/M2CS partition 2. Novell VNDI Partition
78h	XOSL Boot loader file system
7Eh	F.I.X.
80h	Minix v1.1 - 1.4a Minix is a Unix-like operating system for PC (8086 and up). (see ftp://ftp.cs.vu.nl/pub/minix)
81h	1. Minix v1.4b+ 2. Linux (early version) 3. Mitac Advanced Disk Manager
82h	1. Linux Swap partition 2. Prime 3. Solaris x86
83h	Linux native file system (usually ext2fs/xiafs)
84h	1. Hibernation partition (Microsoft APM 1.1f, MKS2D utility) 2. OS/2-renumbered type 04h partition (related to hiding DOS C: drive)
85h	Linux EXT (extended) partition
86h	1. Windows NT Legacy Fault Tolerant or volume/stripe set FAT16 volume 2. Old Linux RAID partition super block
87h	1. HPFS Fault-Tolerant mirrored partition 2. Windows NT Legacy Fault Tolerant or volume/stripe set NTFS volume
8Ah	Linux Kernel Partition (used by AiR-BOOT)
8Bh	Windows NT Legacy Fault Tolerant FAT32 volume
8Ch	Windows NT Legacy Fault Tolerant FAT32 volume using BIOS ext. INT 13h
8Dh	FreeDOS FDISK hidden Primary DOS FAT12 partition
8Eh	Linux Logical Volume Manager (LVM) partition (see http://linux.msede.com/lvm)
90h	FreeDOS FDISK hidden Primary DOS FAT16 partition
91h	FreeDOS FDISK hidden DOS extended partition
92h	FreeDOS FDISK hidden Primary DOS large FAT16 partition
93h	1. Hidden Linux native partition 2. Amoeba file system
94h	Amoeba bad block table (BBT)
95h	MIT **EXOPC** native partitions [http://www.pdos.lcs.mit.edu/exo/]
97h	FreeDOS FDISK hidden Primary DOS FAT32 partition
98h	1. FreeDOS FDISK hidden Primary DOS FAT32 partition (LBA)

Value	Name/Description
	2. **Datalight** ROM-DOS Super Boot [http://www.datalight.com/rom-dos-v.htm] 3. Intel ROM-DOS Service Partition
99h	Mylex DCE376 EISA SCSI logical drive beyond the 1024th cylinder (like DOS extended partition)
9Ah	FreeDOS FDISK hidden Primary DOS FAT16 partition (LBA)
9Bh	FreeDOS FDISK hidden DOS extended partition (LBA)
9Fh	**BSD/OS** [http://www.bsdi.com/]
A0h	Laptop hibernation partition Reported for various laptops like IBM ThinkPad, Phoenix Note BIOS, Toshiba under names like zero-volt suspend partition, suspend-to-disk partition, save-to-disk partition, power-management partition, and hibernation partition. Usually at the start or end of the disk area. (This is also the number used by Sony on the VAIO. Recent VAIOs can also hibernate to a file in the file system, the choice being made from the BIOS setup screen.) Phoenix Note BIOS Power Management "Save-to-Disk" partition
A1h	1. officially listed as reserved 2. Laptop hibernation partition Reportedly used as "Save-to-Disk" partition on a NEC 6000H notebook. Types A0h and A1h are used on systems with Phoenix BIOS; the Phoenix PHDISK utility is used with these. 3. HP Volume Expansion (SpeedStor variant) According to **PowerQuest** ID's 21h, A1h, A3h, A4h, A6h, B1h, B3h, B4h, B6h are for HP Volume Expansion (SpeedStor variant).
A3h	1. officially listed as reserved 2. HP Volume Expansion (SpeedStor variant)
A4h	1. officially listed as reserved 2. HP Volume Expansion (SpeedStor variant)
A5h	BSD/386, 386BSD, NetBSD, FreeBSD
A6h	**OpenBSD** [http://www.openbsd.org/]
A7h	**NextStep** [http://www.next.com/] Based on Mach 2.6 and features of Mach 3.0, is a true object-oriented operating system and user environment.
A8h	Mac OS-X Apple's OS-X uses this type for its file system partition (a UFS file system, in NeXT flavor, only differing from the BSD formats in the first 8 KB). See also type ABh.
A9h	**NetBSD** [http://www.netbsd.org/] NetBSD is one of the children of BSD. It runs on PCs and a variety of other hardware. Since 19-Feb-98, NetBSD uses A9h instead of A5h. It is freely obtainable - see http://www.netbsd.org/Sites/net.html.
AAh	Olivetti FAT 12 1.44Mb Service Partition Contains a bare DOS 6.22 and a utility to exchange types 06h and AAh in the partition table.
ABh	1. MAC OS-X boot partition 2. GO! partition
AEh	ShagOS file system
AFh	ShagOS swap partition
B0h	BootStar Dummy The boot manager BootStar manages his own partition table, with up to 15 primary partitions. It fills unused entries in the MBR with **BootStar** Dummy values. If you use this, do not use a disk manager, do not put LILO in the MBR and do not use fdisk. [http://www.star-tools.com/english/]
B1h	1. officially listed as reserved

Value	Name/Description
	2. HP Volume Expansion (SpeedStor variant)
B3h	1. officially listed as reserved 2. HP Volume Expansion (SpeedStor variant)
B4h	1. officially listed as reserved 2. HP Volume Expansion (SpeedStor variant)
B6h	1. officially listed as reserved 2. HP Volume Expansion (SpeedStor variant) 3. Windows NT mirror set (master), FAT16 file system
B7h	1. Windows NT mirror set (master), NTFS file system 2. BSDI BSD/386 file system (secondarily swap)
B8h	BSDI BSD/386 swap partition (secondarily file system)
BBh	Boot Wizard hidden
BEh	Solaris 8 boot partition
C0h	1. DR-DOS/Novell DOS secured partition 2. Novell NTFT Partition 3. CTOS (Convergent Technologies OS) 4. REAL/32 (DR Multiuser DOS) secure small partition up to 32 MB
C1h	DR DOS 6.0 LOGIN.EXE-secured FAT12 partition
C2h	1. Linux hidden 2. Reserved for DR-DOS 7+ According to **PowerQuest** Id's C2h, C3h, C8h, C9h, CAh, CDh are reserved for DR-DOS 7+.
C3h	Linux swap hidden
C4h	DR-DOS 6.0 LOGIN.EXE-secured FAT16 partition up to 32 MB
C5h	DR-DOS/secured (extended)
C6h	1. Windows NT FAT16 volume/stripe set (corrupted) 2. Windows NT FAT16 mirror set (slave) 3. DR-DOS 6.0 LOGIN.EXE-secured Huge FAT16 partition over 32 MB DR-DOS 6.0 will add C0h to the partition type for a LOGIN.EXE-secured partition (so that people cannot avoid the password check by booting from an MS-DOS floppy). Otherwise, it seems that the types C1h, C4h, C5h, C6h and D1h, D4h, D5h, D6h are used precisely like 01h, 04h, 05h, and 06h.
C7h	1. Windows NT NTFS volume/stripe set (corrupted) 2. Windows NT NTFS mirror set (slave) 3. Syrinx Boot
C8h	Reserved for DR-DOS 7+
C9h	Reserved for DR-DOS 7+
CAh	Reserved for DR-DOS 7+
CBh	Reserved for DR-DOS/OpenDOS secured FAT32
CCh	Reserved for DR-DOS secured FAT32 (LBA)
CDh	1. CTOS Memdump ? 2. Reserved for DR-DOS 7+
CEh	Reserved for DR-DOS secured FAT16 (LBA)
D0h	REAL/32 (DR Multiuser DOS) secured FAT over 32 MB
D1h	Old Multiuser DOS secured FAT12
D4h	Old Multiuser DOS secured FAT16 (< 32M)
D5h	Old Multiuser DOS secured extended partition
D6h	Old Multiuser DOS secured FAT16 (>= 32M)
D8h	CP/M-86
DAh	Non-FS Data
DBh	1. Concurrent DOS, Digital Research CP/M, Concurrent CP/M 2. CTOS (Convergent Technologies OS - Unisys) 3. **KDG Telemetry** SCPU boot KDG Telemetry uses type DBh to store a protected-mode binary image of the code to be run on an x86-based SCPU (Supervisory CPU) module from the

Value	Name/Description
	DT800 range. [http://www.telemetry.co.uk/]
DDh	Hidden CTOS Memdump ?
DEh	Dell PowerEdge Server utilities (FAT file system)
DFh	DG/UX virtual disk manager partition
E0h	**STMicroelectronics** file system ST AVFS [http://www.st.com/]
E1h	SpeedStor FAT12 extended partition (DOS access)
E2h	DOS read-only (Florian Painke's XFDISK 1.0.4)
E3h	1. DOS read-only 2. Storage Dimensions/SpeedStor
E4h	SpeedStor FAT16 extended partition up to 1024 cyl.
E5h	1. officially listed as reserved 2. Tandy DOS with logical sectored FAT
E6h	officially listed as reserved
EBh	**BeOS** BFS (BFS1) [http://www.be.com/]
EDh	Reserved for Matthias Paul's Sprytix Sprytix is currently a project name for an OS related project of mine, partially based on DOS and Linux technologies.
EEh	Indication that this legacy MBR is followed by an EFI header
EFh	Partition with an EFI (Extensible Firmware Interface) file system MS plans to use EEh and EFh in the future for support of non-legacy BIOS booting. These types are used to support the Extensible Firmware Interface specification (EFI); go to developer.intel.com and search for EFI. (For the types EEh and EFh, see Tables 16-6 and 16-7 of the EFI specification, EFISpec_091.pdf.)
F0h	Linux/PA-RISC boot loader The F0h partition will be located in the first 2GB of a drive and used to store the **Linux/PA-RISC** boot loader and boot command line, optionally including a kernel and ramdisk. [http://www.parisc-linux.org/]
F1h	Storage Dimensions/SpeedStor
F2h	DOS 3.3+ secondary partition (Unisys DOS with logical sectored FAT)
F3h	1. officially listed as reserved 2. Storage Dimensions/SpeedStor
F4h	1. SpeedStor large partition 2. Prologue single-volume partition
F5h	**Prologue** multi-volume partition The type F4h partition contains one volume, and is not used anymore. The type F5h partition contains 1 to 10 volumes (called MD0 to MD9). It supports one or more systems (Prologue 3, 4, 5, Twin Server). Each volume can have as file system the NGF file system or TwinFS file system. NGF (old): volume size at most 512 MB, at most 895 files per directory, at most 256 directories per volume. TwinFS (new): volume size up to 4 GB. No limit in number of files and directories. [http://www.prologue-software.com/]
F6h	1. officially listed as reserved 2. Storage Dimensions/SpeedStor
F9h	**pCache** [http://www.alcpress.com/articles/pcache.html] We propose using the F9h partition type as a pCache partition, which is our name for an "ext2/ext3 persistent cache partition".
FAh	Bochs MandrakeSoft's **Bochs** x86 emulator (similar to VMWare) uses FAh as a partition identifier. [http://bochs.sourceforge.net/]

Value	Name/Description
FBh	VMware File System partition
FCh	VMware Swap partition
FDh	Linux raid partition with auto detect using persistent superblock
FEh	1. Windows NT Disk Administrator hidden partition Windows NT Disk Administrator marks hidden partitions (i.e. present but not to be accessed) as type FEh. A primary partition of this type is also used by IBM to hold an image of the "Reference Diskettes" on many of their machines, particularly newer PS/2 systems (at a rough guess, anything built after about 1994). This clash can cause major confusion and grief if running NT on IBM kit. When this Reference Partition is activated, it changes its type into 1 (FAT12) and hides all other partitions by adding 10h to the type. 2. SpeedStor over 1024 cyl. 3. LANstep 4. IBM PS/2 IML (Initial Microcode Load) partition (image of the Reference Diskettes) (located at the end of disk) 5. Linux LVM (Logical Volume Manager) partition (old) This has been in use since the early LVM days back in 1997, and has now (Sept. 1999) been renamed 8Eh.
FFh	Xenix bad block table (BBT)

G. Glossary

G.1 Buses

ISA (Industry Standard Architecture)
8- or 16-bits bus

EISA (Extended ISA)
32-bits bus, compatible with ISA slots

MCA (Micro-Channel Architecture)
16- or 32-bits bus, IBM, from 1987

VL-Bus (VESA Local Bus)
32-bits bus, bus mastering, uses ISA slots + special connector

PCI (Peripheral Component Interconnect)
PCI is the not terminated bus, the signal relay on signal reflections to attain there final value.
32-bits bus, PCI is so-called „root technology" (AGP, CardBus, SmallPCI, PCI-X); bus mastering, uses PCI slots

1. PCI: the original specification 'Peripheral Component Interface'

2. PCI-X (PCI extended): the Next Generation of Backward-Compatible PCI

PCI-X is backward compatible with existing PCI cards. It improves upon the speed of PCI from 133 MBps to as much as 1 GBps. PCI-X was designed jointly by IBM, HP and Compaq to increase performance of high bandwidth devices, such as Gigabit Ethernet and Fibre Channel, and processors that are part of a cluster.

To achieve the very high frequencies of PCI-X 2.0, lower voltage signal swings were required. As a result, PCI-X 266 and PCI-X 533 require new 1.5V signaling. However, to maintain compatibility with previous-generations of 3.3V PCI technologies, the I/O buffers have been carefully designed to support both signal levels.

The PCI-X 2.0 specification includes **ECC** (**Error Correcting Codes**) to provide additional fault tolerance.

3. PCI Express: is a third-generation, high-performance I/O bus used to interconnect peripheral devices in computing and communication platforms. The PCI Express architecture retains much of the PCI software interface, including the configuration and device driver interfaces, to ease the transition from PCI to PCI Express. However, unlike its predecessor, which is a parallel multi-drop bus, PCI Express is a serial point-to-point bus. The already high PCI Express performance can be scaled up by increasing the data path width and, later, by increasing the clock frequency as well. Currently, the aggregate data transfer rates range from 0.5 GB/s to 16 GB/s.

While PCI Express will mainly be around in desktop systems, PCI-X will remain the prevailing high-performance interface for high-end workstations and server systems. Finally, PCI-X 1066 will be able to provide up to 8.5 GB/s. However, PCI Express is intended to replace AGP.

PCI Express is also called **3GIO** (Third Generation I/O) and occasionally **Arapahoe**.

Table 19: Compare of PCI buses

Standard	Bus width	Clock	Transfer
PCI 2.3	32 bit	33 MHz 66 MHz	133 MB/s 266 MB/s
PCI 64	64 bit	33 MHz 66 MHz	266 MB/s 533 MB/s
PCI-X 1.0	64 bit	66 MHz 100 MHz 133 MHz	533 MB/s 800 MB/s 1066 MB/s
PCI-X 2.0 (DDR)	64 bit	133 MHz	2132 MB/s
PCI-X 2.0 (QDR)	64 bit	133 MHz	4264 MB/s
PCI Express	1 line, 8 bits	2.5 GHz	512 MB/s
PCI Express	2 lines, 8 bits	2.5 GHz	1 GB/s (duplex)
PCI Express	4 lines, 8 bits	2.5 GHz	2 GB/s (duplex)
PCI Express	8 lines, 8 bits	2.5 GHz	4 GB/s (duplex)
PCI Express	16 lines, 8 bits	2.5 GHz	8 GB/s (duplex)
PCI Express	32 lines, 8 bits	2.5 GHz	16 GB/s (duplex)

AGP (Accelerated Graphics Port) – AGP is a PCI-like bus interface targeted for high-performance 3d graphic. AGP supports only memory read/write operation and single-master single-slave one-to-one only. The AGP uses both rising and falling edge of the 66 MHz clock and produces 66 MHz x 4 byte x 2 = 528 MB/s data transfer rate.

PCMCIA (PC card) – The Personal Computer Memory Card International Association (PCMCIA) has created a standard for small form factor peripherals called PC Cards. Standardizes packages for memory and input/output (modems, LAN cards, etc.) for computers, laptops, palmtops, etc. For each there is a specific set for each bus to use. As a bus it is the network/or circuitry's placement in which all the devices are attached to it and all signals pass through each device, but only the targeted device will receive and recognize the signal intended for it.

CardBus – The 32-bit version of the PCMCIA PC Card standard. In addition to supporting a wider bus (32 bits instead of 16 bits), CardBus also supports bus mastering and operation speeds up to 33 MHz.

USB (Universal Serial Bus) – A serial bus standard promoted by Intel for communication between an IBM PC and external peripherals over an inexpensive cable using biserial (in two rows or series) transmission. USB works at 12 Mbps with specific cost consideration for low cost peripherals. It supports up to 127 devices and both isochronous and asynchronous data transfers. Cables can be up to 5 meters long and it includes built-in power distribution for low power devices. It supports daisy chaining through a tiered star multidrop topology.

IEEE 1394 – serial bus. IEEE 1394, formerly FireWire. A 1995 Macintosh/IBM PC serial bus interface standard offering high-speed communications and isochronous real-time data services.
1394 can transfer data between a computer and its peripherals at 100, 200, or 400 Mbps, with a planed increase to 2 Gbps. Cable length is limited to 4.5 m but up to 16 cables can be daisy-chained yielding a total length of 72 m.

It cans daisy chain together up to 63 peripherals in a tree-like structure (as opposed to SCSI's linear structure). It allows peer-to-peer device communication, such as communication between a scanner and a printer, to take place without using system memory or the CPU. It is designed to support plug-and-play and hot swapping. Its 6-wire cable is not only more convenient than the SCSI cables but can supply up to 60 watts of power, allowing low-consumption devices to operate without a separate power cord.

Table 20: Compare of buses

Bus		Speed [Mbps]	Bandwidth		Power [W]	Type	Topology	Length device/ total [m]
			Bits	Hz				
AGP	1x	2112	32	66M		parallel	bus	
	2x	4224						
	4x	8000						
	8x	16000						
Fire Wire	a	100, 200, 400	64	66M		serial	tree	
	b	800, 1600, 3200						
USB	1.1	1.5,12		480 M	1.5	serial	tree	3-5/30
	2.0	180						
Bluetooth		1 (real 720K)		2.4G	0.25 100m		wireless	0.1-10/ 100
802.11	a	54		5G			wireless	1-33.3
	b	11 (real 7)		2.4G				50/115 (in-out)
PCI		1060	32/64	33/ 66M		parallel	bus	

G.2 Basic terms

BIOS (Basic Input/Output Services)

CMOS (Complimentary Metal-Oxide Semiconductor)

PIO (Programmed Input Output)

DMA (Direct Memory Access)
(known like bus master access)

ST506 – first standard for hard disk introduced from Seagate in 1980; need physical installation, setup configuration in CMOS, low-level and high-level formatting.

ESDI (Enhanced Small Device Interface)
This type was introduced in 1983. He has integrated controller in device drive. Configuration is similar like at ST506.

IORDY (Input Output Ready) – IORDY is a bus-signal used for high-speed transfer between drive-cache and system. Since local-bus adapters can often transfer data faster then a drive can handle, the IORDY signal has been implemented. This signal can be asserted by the drive to set wait-cycles on the bus.
As the drive will report the system when it is ready to receive or send a data-word, the drive can process data at its own speed. On drives that do not support IORDY the worst-case scenario will determine the minimum transfer cycle (max. transfer per second). PIO modes 3 and 4 use IORDY, so it's not clear what the average transfer speed will be, since the drive can always assert the IORDY signal to slow down the total transfer.

EHCI (Enhanced Host Controller Interface)

interface for host controller USB rev. 2.0

OHCI (Open Host Controller Interface)
OHCI for USB rel. 1.0a, 1996 – interface USB
1394 OHCI rel. 1.1, 2000 - interface IEEE 1394 to PC host

RAID (Redundant Array of Independent Disks)
A feature of a controller designed to protect against hard drive failure or improve storage subsystem performance or both by making multiple hard drives act as a single hard drive.

Z. References

[1] **ATA/ATAPI/SATA/SATAPI standards**
[http://www.t13.org/]

[2] **SCSI Storage Interfaces**
[http://www.t10.org/]

[3] **BIOS Enhanced Disk Drive Services (EDD) T13/1484D**

[4] **Standard BIOS 32-bit Service Directory Proposal**
Revision 0.4, 18.06.1993
Phoenix Technologies Ltd., PC Division, Desktop Product Line

[5] **Compaq/Phoenix/Intel: Plug and Play BIOS Specification**
v1.0A 05.05.1994

[6] **Compaq/Phoenix/Intel:**
EXTENDED SYSTEM CONFIGURATION DATA SPECIFICATION (ESCD)
v1.02A 31.05.1994, Part Number 485547-001

[7] **Compaq/Phoenix/Intel: BIOS Boot Specification (BBS)**
v1.01 11.01.1996

[8] **International System of Units (SI)**
[http://physics.nist.gov/cuu/Units/index.html]

[9] **Enhanced S.M.A.R.T. - Get S.M.A.R.T. for Reliability, 07/1999**
[http://www.seagate.com/docs/pdf/whitepaper/enhanced_smart.pdf]

[10] **Enhanced Host Controller Interface (EHCI) specification rev. 1.0**
[http://www.intel.com/technology/usb/download/ehci-r10.pdf]

[11] **Partition types: List of partition identifiers for PCs**
[http://www.win.tue.nl/~aeb/linux/partitions/partition_types-1.html]

[12] **ATA/ATAPI Host Adapters Standard (ATA-Adapter)**
T13/1510D rev.1.0 17.01.2003
[http://www.t13.org/]

[13] **S.M.A.R.T. Applications Guide for the ATA Interface SFF-8055i rev.1.2**
26.04.1996

[14] **Seagate Advanced SCSI Architecture II Technology Paper** [HTML]

[15] **Hale Landis: ATA-ATAPI**
[http://www.ata-atapi.com/]

[16] **G-Force Protection**
[http://www.seagate.com/support/kb/disc/gf_protect.html]

[17] **SMART Attribute Annex**
[http://www.t13.org/docs2005/e05148r0-ACS-SMARTAttributesAnnex.pdf]