

How It Works: Facts and Fiction

[\[Home\]](#) [\[Q&A\]](#) [\[History\]](#) [\[Help\]](#) [\[SATA\]](#) [\[SCSI/1394\]](#) [\[HowItWorks\]](#) [\[Search\]](#)

[\[Hale's Test Software\]](#) [\[ATACT\]](#) [\[ATADEMO\]](#) [\[ATAMDT\]](#) [\[ATADRVR\]](#) [\[ADVDRVR\]](#)

[\[HIW Topics\]](#) [\[CHSxlat\]](#) [\[DOSboot\]](#) [\[FAQ\]](#) [\[FNF\]](#) [\[MBR\]](#) [\[OS2boot\]](#) [\[PartTable\]](#)

Facts and Fiction

- [What is a Mega Anyway?](#)
- [Standard\(?\) I/O Port Addresses and Interrupt Numbers](#)
- [Logical Block Addressing \(LBA\)](#)
- [Which is better: LARGE or LBA?](#)
- [Even More About LBA](#)
- [Low Level Formating](#)
- [Zone Bit Recording \(ZBR\)](#)
- [Dual Channel ATA Host Adapters and Data Corruption](#)
- [Why Disk Drives Break](#)
- [Plug and Play \(PnP\)](#)
- [Thermal Calibration and AV Drives](#)

What is a Mega anyway?

And here is something that Hale really hates...

Someone once wrote:

HD manufacturers think 1MB = 1e6 bytes, not 1048576 bytes.

And Hale said:

The HD manufacturers are right: 1MB is 1000000 bytes - it IS NOT 1048576 bytes! Mega means 1000000 (just like kilo means 1000).

Is a 1MHz a frequency of 1048576 cycles/second? NO! It is 1000000 cycles/second. Is a

kilometer 1024 meters? NO! It is 1000 meters.

Mega, Kilo, etc are ISO defined terms that predate the computer industry by many decades!

For more information read [Definitions of the SI units The twenty SI prefixes](#) and also read [Definitions of the SI units The binary prefixes](#) . Note that the mass storage industry generally uses the correct definition of K and M.

Standard(?) I/O Port Addresses and Interrupt Numbers

The following table shows the most commonly supported I/O port addresses and IRQ numbers used for PC ATA (IDE/EIDE) host adapters.

Interface Number	CS0-Decode	CS1-Decode	IRQ number
1	01F0h-01F7h	03F6h-03F7h	14
2	0170h-0177h	0376h-0377h	15
3	01E8h-01EFh	03EEh-03EFh	12 or 11
4	0168h-016Fh	036Eh-036Fh	10 or 9

Now some history and notes...

Interface number 1 -- The Primary

The addresses and IRQ number for the first interface have been well established since the first IBM PC/AT system in 1984. This is the address and IRQ used by the MFM controller in the original IBM PC/AT. All PC software (BIOS and OS) support the primary host adapter (because they support an MFM, RLL or ESDI controller using this configuration). Remember that ATA (IDE/EIDE) is designed to operated just like an old MFM controller (of course there are new features in ATA that MFM, RLL and ESDI did not support).

Interface number 2 -- The Secondary

During 1995 support for second interface has become very well established. Some implementations of this interface number have used IRQ's other than 15 in the past. Today,

IRQ 15 has become the accepted standard for the secondary interface.

The other interfaces

The addresses and IRQ's used by the 3rd and 4th interfaces is not very standard. Usually it is the ATA (IDE) interface on some other type of card, such as a sound card, that is used for the 3rd or 4th ATA host adapter in a system. There is no standardized BIOS or OS support for such configurations and usually an OS device driver is required to access these ATA host adapter addresses.

Logical Block Addressing (LBA)

LBA DOES NOT SOLVE THE >528MB PROBLEM!

There continues to be a large amount of very false information floating around in newsgroups concerning LBA addressing on ATA (IDE) drives.

Please, lets get this correct...

- LBA has NOTHING to do with fixing the >528MB problem.
- LBA does NOT solve the >528MB problem.
- It is not true that LBA is needed to support a drive >528MB.

An INT 13H BIOS that does CHS translation is the ONLY way to support a drive >528MB today. CHS translation enables the use of >16 heads with <=1024 cylinders at the INT 13H interface while using <=16 heads and >1024 cylinders (or maybe LBA) at the drive interface.

FDISK uses CHS. It does not use LBA. It does not know (or care) if LBA is being used at the device interface. FDISK can not address any part of a disk that can not be accessed in CHS mode.

Microsoft operating systems WILL NOT use any part of a disk that can not be accessed in CHS mode. Read the Microsoft documentation carefully. OS/2 and Linux do have a special FDISK "hack" that allows partitions to span or be beyond the 528MB boundary even on systems that do not have a CHS translating BIOS and even on drives that do not support LBA. Be very careful when using this feature of OS/2 or Linux.

LBA provides only a slight performance improvement for some protected mode operating systems, HOWEVER, those systems MUST still boot in CHS mode and use CHS mode to understand the drive configuration data returned by INT 13H AH=08H and to understand the partition layout of the drive. If an OS decides to use LBA addressing it MUST not attempt to access any sectors that are beyond those that can be accessed by the CHS mode returned by

INT 13H AH=08H. Therefore, the size of a drive in LBA mode is the same as the size in CHS mode. If the INT 13H BIOS does not support drives >528MB then LBA mode can not be used make the drive look >528MB either.

So, once again, LBA does not solve the >528MB problem. Do not be confused by the the BIOS vendors that claim LBA solves the >528MB problem. They are confusing you by using LBA to explain that their BIOS is really doing CHS translation. It is very unfortunate that they are using LBA so incorrectly. We can probably thank Western Digital's EIDE Implementation Guide for starting this confusion.

NOTE: If you are about to send me email to tell me how wrong I am, please see my "How It Works" series of documents that describe how the MBR and boot sectors work, or, disassemble the master boot record on your system and look at how your system boots. You will find that it does not use LBA at all. If after doing this you still think I am wrong, then send me email. Thanks!

Which is better: LARGE or LBA?

Hmmm... The truth is this: The LBA option is not stable and is not the "standard" that will survive in the future. I have posted several articles in the past about this but the misleading information comes faster than I and others can keep up.

The truth is that the IBM/MS/Phoenix extended/enhanced BIOS specification **is** the future "standard" and it is being widely implemented. This "standard" does not require the use of LBA at the drive interface. Using LBA at the device interface never did and never will solve the >528MB (>1024 cylinder) problem. Only CHS translation at the INT 13 interface or LBA at the INT 13 interface solves this problem.

The truth is that the LBA BIOS option is a poorly designed and very poorly documented Western Digital idea that should have never been adopted by the PC industry. Even WD does not actively support it any more.

Maybe this table will help everyone understand this problem...

drive size	INT 13 interface	IDE/EIDE drive interface
<528MB and <1024 cyl	any old BIOS works -- no CHS translation is required.	CHS used here is the same as the CHS used at the INT 13 interface.

<p>>528MB or >1024 cyl but less than 8GB</p>	<p>CHS translation required -- keep cylinders under 1024 and use more than 16 heads</p> <p>implementation #1 is described by IBM/MS/ Phoenix documents.</p> <p>implementation #2 is described by the WD EIDE guide.</p>	<p>two possible implementations exist:</p> <p>#1 the "standard" or LARGE implementation uses CHS at the drive interface and generally gives better performance.</p> <p>#2 the WD EIDE BIOS or LBA implementation uses LBA at the drive interface (for no good reason). This is not the "standard" of the future and it can confuse some OS device drivers -- this is why so many versions of the Windows FastDisk (32-bit disk access) driver exist today. This BIOS implementation does not work for drives of 8GB or bigger.</p>
<p>8GB or bigger</p>	<p>IBM/MS/Phoenix "standard" is the only thing that works. LBA is used at the INT 13 interface!</p>	<p>either CHS or LBA can be used at the device interface.</p>

Note: The IBM/MS/Phoenix "standard" works for all drive sizes including drives with up to 2^{64} sectors (really big drives!). The WD LBA BIOS implementation fails when drives get to 8GB and it also does not include any of the Plug-and-Play stuff that future operating systems will require.

I really think that someone at WD got confused about how LBA should be used. It appears to me that this WD person (or persons) thought that using LBA at the device interface would solve the big disk problem. What they overlooked was this: LBA at the INT 13 interface is what really solves the big disk problems. LBA at the device interface is not needed to solve any big disk problem (even for 100GB drives)!

As I have said before, those of you that are using the LBA option in your BIOS setup **may** find that some day you will have to back up all your data and switch to the LARGE option in order to remain compatible with the real BIOS "standard".

Even More About LBA

What does LBA do for me?

Some people in this message thread seem to be confused here about the use of LBA to:

1. get around the 528MB problem,
2. reformatting the disk because of LBA,
3. performance increase due to LBA.

I am not sure were all this started but, lets take a brief look at each of these...

1. LBA does not solve the >528MB problem. Only a BIOS with an INT 13 that does CHS translation solves this problem. How the BIOS sends sector addresses to the disk drive, as CHS or as LBA, should not change the CHS that is used at the INT 13 interface.

Yes, there is at least one BIOS type out there that has a problem with this: any BIOS based on the Western Digital EIDE Implementation Guide may change the CHS used at the INT 13 interface when the use of LBA at the device interface is turned on or off. This is a very flawed implementation and should be avoided OR if you have such a BIOS, NOT do change the LBA setting in the CMOS setup once you have partitioned your disk and installed your software.

2. As long as the CHS at the INT 13 interface does not change, it should make not difference if CHS or LBA is being used at the drive's interface. CHS and LBA are completely interchangeable at the drive interface on a command by command basis.

However, beware of these WD EIDE BIOS types. These BIOS incorrectly change the CHS used at the INT 13 interface when LBA use at the drive interface is changed. This should not be. This is the major flaw in the WD EIDE BIOS type. It represents a serious threat to your data!

3. LBA use will not give any great performance increase. In fact, on some drives it may decrease performance. The disk drive industry has had many years to optimize the use of CHS at the drive's interface. Many drives do NOT convert the CHS at the interface to an LBA internally. In these drives, using LBA at the drive's interface just causes the drive more overhead as it must convert the LBA to a CHS before it can proceed with the command.

Is LBA slower?

Someone once posted this on UseNet:

>..., no it is not slower, just a different geometry translation, >and at the same speed as long as the system supports LBA, >and is set up correctly.

Beware... LBA **can** be slower for two reasons:

1. Conversion of INT 13 CHS input to an LBA at the device interface (when LBA is used in BIOS setup) usually requires more instructions than conversion of INT 13 CHS input to another CHS at the device interface (when CHS translation is used in BIOS setup, aka LARGE).
2. LBA processing in many drives is slower than CHS processing (this will change as drive vendors optimize their hardware and firmware for LBA in future drives). Many people can measure a slight difference in performance between CHS and LBA today.

Low Level Formatting

What does low-level format do on an ATA (IDE) drive?

Answer: Depends on how old the ATA drive is and who made it.

Lets talk about what low-level formatting is...

In the old days of MFM, RLL and ESDI, a new hard disk drive was just like a new unformatted floppy - there was nothing on it. The drive had to be connected to a controller and the controller had to be told how many sectors per track to lay down on each track - this is the same thing you do when you format a floppy these days. Formatting of each track creates the inter sector gaps, the sector ID fields and the sector data fields. Most MFM/RLL/ESDI controllers include a small low-level formatting program in ROM. You generally use DOS DEBUG to enter this program.

Lets talk about what high-level formatting is...

High level formatting creates a file system within a partition of a hard disk or on a floppy. This process writes the initial version of the boot record, root directory and file allocation table (FAT).

Here is a bit of confusion - when you use the DOS FORMAT command to format a floppy, you are doing BOTH a low-level and high level format at the same time. When you use the DOS FDISK and FORMAT commands to format a hard disk, all you are doing is the high level format.

Back to hard disks...

MFM/RLL/ESDI and some older ATA drives use the same controller command code, known as Format Track, to do the low-level format of a track. This command is issued once for each track on the hard disk. This command writes the inter sector gaps, the sector ID fields and the

sector data fields. Each sector on a track has an ID - an 8-bit binary number usually starting at 1. The order in which the sector IDs are written determines the interleave. If the sector IDs are written as 1, 2, 3, ..., n, this is 1-to-1 interleave. When written as 1, n, 2, n+1, 3, ..., n-1, you have 2-to-1 interleave. MFM controllers usually used 2-to-1 or 3-to-1 interleave with 17 sectors per track. RLL usually used 26 sectors per track.

MFM/RLL/ESDI also would allow "marking" a sector "bad" as the sector ID field was being created. This would cause a very special kind of error on any read or write command that attempted to access this sector - a Bad Block error. When you run the DOS FORMAT program, it reads every sector in the partition looking for "bad" sectors and sectors with uncorrectable data errors. Such sectors then cause clusters of sectors in the FAT to be marked bad and DOS will never use them. Other systems, such as OS/2 HPFS, Unix, etc, keep bad block lists as part of their file system data and they also do not access these "bad" sectors.

MFM/RLL/ESDI drive technology was generally based on using a stepper motor to position the read/write heads. Over time the bearings in the drive would wear and read/write errors would appear because the stepper motor was no longer positioning the read/write heads in the right place. The solution was to do a low-level format again.

New drives do not use stepper motors and instead use embedded servo bursts in between the sectors. These bursts are used to locate tracks and sectors and to keep the read/write heads properly positioned even when the drive's bearings are worn. More important is keeping the read/write heads properly positioned under a wide range of temperatures. Thermal expansion can change the length of the arm the read/write head is attached to by as much as 5 or 10 tracks! This is the reason you hear new drives doing a lot of thermal calibration (that strange seeking noise you hear when there should be no disk activity).

The servo bursts are written at the factory in a very controlled environment using some very expensive equipment. The drive alone can not recreate these servo data bursts. Likewise, because most drives are now zone recorded (they have different number of sectors per track at different locations on the media), the inter sector gaps, sector ID fields and sector data fields are also written at the factory and can not be recreated later. Some drives may soon do away with the sector ID fields and some of the gaps in order to increase data storage capacity.

Now for the history the ATA Format Track command...

Early ATA drives did not really implement the Format Track command - it was thought to be obsolete (and it was). What was implemented was a simple writing of some data pattern into each sector of each track formatted. Most drives did not support marking sectors bad. However, the disk drive industry is driven by features and slowly, one-by-one, the hard disk vendors started implementing the command such that it did the same thing as in MFM/RLL/

ESDI. Then someone implemented the ability to "reassign" a sector's address to a different physical sector on the disk. Never mind that there were **no** programs then and few now that use the Format Track command to do this.

The original ATA specification (now at rev 4.0c, also known as ATA-1) documents the "full" function Format Track command but leaves it to the drive vendor to decide what a drive will really do. It recommends a minimum action of writing binary zero into the data field of each sector formatted. The ATA-2 specification says that the function of the command is "vendor specific" - it does not even recommend the minimum action of writing binary zero data - a major step towards (finally) making the command obsolete.

So what does low-level formatting do on a modern ATA drive?

Assuming that you can find a program that really does issue the ATA Format Track command your ATA drive probably does not do anything other than write some data pattern, maybe binary zeroes, into the data field of every sector on the drive. That is no different than just using the normal write command and writing data into every sector on the drive. In general, low-level formatting of an ATA drive is just a big waste of time.

So several things have gotten us to this time and place:

1. No stepper motors.
2. Very close track spacing.
3. Embedded servo data.
4. Zone recording.
5. Few programs that use the Format Track command.
6. The Format Track command **DOUBLES** the amount of firmware in a drive.
7. And then there is the failure rate problem... There are two major components in a disk drive - the Head/Disk Assembly (HDA, contains the media and read/write heads) and the printed circuit board with the electronics - which fails most frequently?

THE ELECTRONICS!

Wait a minute... Are you trying to say the electronics are more unreliable than the HDA (media and heads)?

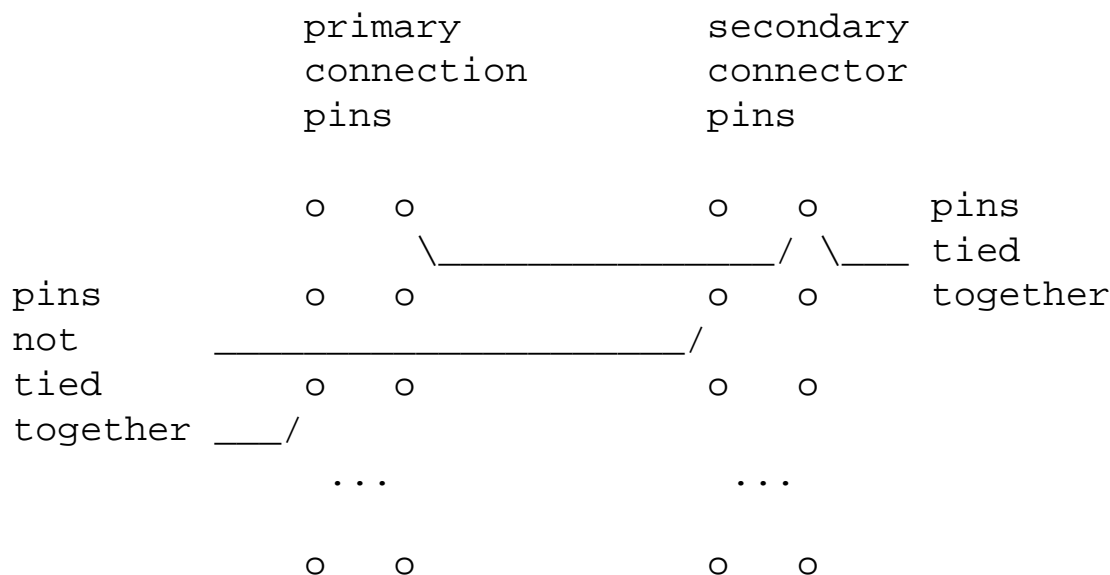
In general the failure rate for the electronics is slightly higher than the failure rate of the media and heads. High temperatures and heating/cooling cycles cause failure of the little gold wires inside the chips. This could be a good reason to make sure your drive has adequate cooling and leave it on most of the time. But heat is the big problem here - it is just a matter of time before the HDA or the electronic will fail. Of course rough handling (like dropping the drive) will probably not damage the electronics but will damage heads or media.

1. the minor reason is that the data bus exceeds the ATA 18 inch maximum length. This design makes the two cables look like one 36 inch cable.
2. the major reason is that a multitasking operating system expects to be able to perform an I/O operation at the same time on both the primary and secondary host adapters. This design will corrupt data (or cause hang conditions) because of the shared data bus.

How can you identify a flawed dual channel host adapter?

One way is to look at the printed circuit board and count the number of direct connections between the pins of the primary host adapter connector and secondary host adapter connector. This may be difficult on multilayer printed circuit boards. It is normal for a few signals (such as ground signals) to be tied together directly. However, if you see a large number of pins (more than 16) tied directly together by copper traces on the circuit board, you are looking at one of these flawed host adapters.

For example:



However, beware, this visual check is not foolproof! Pins that do not appear to be tied together in the area of the connectors could be tied together at some other location.

My advice: Talk to the host adapter (or motherboard) technical support people and ask them if the two host adapters share any signals, logic or functions. Any sharing of data bus signals, host adapter logic or functions (especially data transfer logic) would indicated a flawed design that could corrupt your data.

BUYER BEWARE!

If you run Linux, check the latest Linux IDE driver information for a version of the IDE driver that will serialize all I/O to the two host adapters in order to prevent strange things.

It is unclear at this time (May95) how WinNT, Win95 or OS/2 will deal with these host adapters.

BEWARE OF INTEL TRITON DUAL CHANNEL HOST ADAPTERS

The Triton is even worse than first look gave. True, it does not corrupt data or do strange things with interrupts, but it can have what I consider to be serious performance problems. Now that Intel has made the chip spec public, we can all find out that not only does the chip share the two ATA data buses but that combined bus is also shared with the ISA address/data bus function that is also in the chip. Intel claims "fair round robin" sharing of all the uses of the single bus. So if you have your serial or parallel ports on the Triton ISA bus and you have any COM or LPT activity going on this will be multiplexed with your two ATA interfaces on the same set of signals coming out of the Triton chip. So much for high performance.

Why Disk Drives Break

Why do hard drives fail?

There are three major (and these really are the only three real reasons hard disk drives fail):

1. they get too hot (your system cooling fan quits working).
2. they are mishandled (dropped or banged around).

Drives that have been over heated or mishandled can develop bad sectors as time goes on. Usually you will see one bad sector and then a few more and then a bunch more until the drive is basically not worth using any more.

A drive that really gets too hot may refuse to spin up because the heads get "glued" to the media by the high heat.

3. an electronics failure.

Electronics failures are usually sudden and without warning. A common time for a drive's electronics to fail are on Monday morning after the drive has been powered off all weekend. It is the heating/cooling cycles that cause breaks in the printed circuit board or breaks in the little gold wires inside the chips on the printed circuit board.

You can also zap the electronics by handling a drive when you are not properly grounded.

What can I do?

Make that your drive is properly cooled, do not drop it and be grounded before you touch a drive!

OK, every once in awhile a disk vendor will make a bunch of drives that have some kind of dirt or chemical inside that should not be there. Drives with this kind of problem usually do not last very long and usually fail during the warranty period. This is a very rare thing to have happen these days but the failure mode is usually the same as described above for over heating.

Plug and Play (PnP)

As many people have found out PnP is a joke. It is a cute marketing thing to make you think that the computer hardware and software vendors have solved all the hardware and software configuration problems for you. Just plug in a new device and the system will recognize it and allow you to use it. Ha Ha!

What is PnP really? It is a bunch of uncoordinated and proprietary solutions to specific hardware or software configuration problems. Many of the so called PnP specifications were developed by individual companies or groups of companies (aka, private clubs) to enhanced their image with the computer buying public. Most of these specifications are short sighted and do not address all of the issues.

There are so many PnP specifications floating around the industry that no one has a complete list of them. New ones appear monthly. It is a major mess.

A specific example is the PnP mess related to ATA (IDE/EIDE) devices. Several years ago various hardware specificaion groups (VESA and PCI and others) attempted to defined PnP for various type of host adapters, including ATA (IDE/EIDE) host adapters. But these groups addressed only the hardware aspects of the host adapter configuration. They forgot to address the BIOS issues!

So the PnP mess continues on and grows bigger each day.

Thermal Calibration and AV Drives

Someone once posted this on UseNet:

>The way I see it, there are (at least) two common
>myths floating around about SCSI and EIDE drives.

Myth #1: SCSI has better sustained throughput than EIDE.

>As I understand it, the SCSI 3 bus is capable of higher
>bandwidth than EIDE on a PCI bus, but that is irrelevant,
>because the hard drives themselves do not come anywhere
>near using the available bandwidth. As far as I can tell,
>the high end of SCSI drives (single drives, anyway) are
>capable of approx 7 MB/sec sustained throughput. And
>the high-end EIDE drives (PIO Mode 4) are capable of
>about the same sustained throughput. True or false?

Nearly always FALSE (due to IDE/EIDE's extremely low command overhead) except that there are real high capacity and real high performance drives that come only with the SCSI interface. This will probably change in the future.

Plus there is the old question of "how many reads are from the drive's cache?". A cache read is just a memory (in the drive) to memory (in the system) transfer and can, in theory, be done at extremely high speeds (much faster than the drive's actual media transfer rate).

Today, for the same transfer rate, EIDE is cheaper (especially when you include the cost of host adapters and cables).

Who knows what tomorrow will bring. There are many people in the disk drive industry working on faster EIDE and faster SCSI data transfer protocols. And they probably are also working on the interface that will replace both EIDE and SCSI (for disk drives). Neither EIDE or SCSI can keep up with the disk technology that is just over the hill now and moving towards us very rapidly. Expect an entirely new hard disk interface in a few years as drives approach data transfer rates of 100+MB/second.

Myth #2: Only drives sold as "AV" drives are capable of uninterrupted sustained throughput.

This is based on a problem which no longer exists.
>Older drives used to pause every so often to recalibrate
>their head positions, the so-called "thermal recal".
>As I understand it, just about all drives now use
>a system of so-called servo tracks which are written
>onto the platter surfaces. Thus, as the drives get
>hotter and the material expands, the servo tracks
>expand right along with them, and so there is no need
>for thermal recal anymore. True or false?

Well, sort of TRUE and sort of FALSE. ALL drives MUST do thermal calibration every so often.

Until a few years ago most drives had "dedicated servo" systems (all servo data on one surface and only read by one head) that required frequent calibration especially if the drives internal temperature was changing rapidly (like right after power on). The so called AV versions of this type of drive attempted to reduce the chance that the host would notice this drive activity. This activity can disturb the smooth flow of video or sound data during disk read/write commands.

Today most all drives use an "embedded servo" system (servo data mixed in with user data on every data track, every head reads servo data while reading user data). While thermal calibration is still required, its impact on the smooth flow of data can be kept to an absolute minimum. My guess is that some disk vendors will still use the AV label just because it is a good marketing tool. You may even see an AV label on an EIDE drive.

However, there is another side of this AV thing that you did not bring up (myth#3?): reduced error recovery.

A true AV drive will implement a set of reduced error recovery read commands based on the theory that your eye will notice a pause in the flow of the video data while the drive attempts its normal full error recovery (the picture will "pause" or "jump") but your eye will probably ignore a few bad bits of video data flashing across the screen (video snow). Of course you do not want your OS to use this reduced error recovery read command when it is reading real data (like a directory or your current tax return's data!).

The bottom line 99% of the time: Unless you are buying a drive that will become part of the disk array that is used to store video data, you probably do not need an AV drive. AV drives are really built for this application.

This page was last updated on 06 May 2002.