

# Hard Drive Size Limitations and Barriers

DEW Associates Corporation

## Introduction

There's nothing more frustrating than trying to add a new hard drive only to be confronted by the inability to use it's entire capacity. This inability to use the entire drive capacity is referred to as a Limitation or Barrier, but bear in mind though that it is not a limitation of the drive itself, but rather a limitation imposed by one or more other factors that have arisen during the development of computers in general.

When we began preparing this segment, we realized that hard drives, especially IDE/ATA type drives, were in a constant state of development, therefore new limitations and barriers would be discovered even as this segment was being completed. With this in mind, we decided to break this segment into two parts. The first segment, that which you are now reading, begins with the background behind some of these limitations and barriers that arose as computer systems evolve and proceed through the design and redesign process. In the second segment, we will take an in-depth look at each barrier or limitation, and that segment will evolve as new drive development occurs and new computer systems evolve.

Capacity barriers are not new to the computer industry, in fact, limitations began surfacing the the first time hard drive sizes rose above 528MB. As larger and larger disk capacities emerge in the market, end users are realizing that their older motherboard BIOS's and/or operating systems do not support the increased capacity of these new drives.

We hope the following will provide you with a better understanding of the limitations and barriers associated with higher capacity disk drives, along with some of the solutions being implemented to solve them.

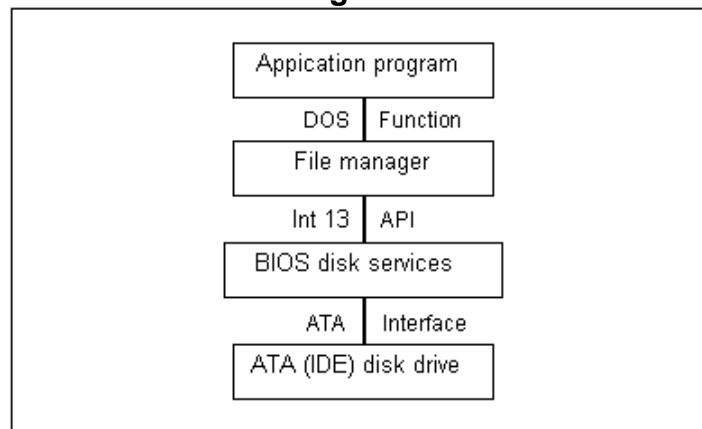
## Background

### Understanding the history behind the problem!

The I/O (input/output) system for disk drives that is in use today is still based on the original DOS-BIOS (Disk Operating System - Basic Input/Output System) developed decades ago, although this is about to change. Yes, you read it correctly, DOS-BIOS, but if you're thinking about files systems, especially the NTFS file system that is part of Windows NT/2000/XP and are thinking that this information doesn't apply to you, think again! We're not referring to the operating system and its file system here, we're referring to the I/O system that was designed around a DOS-BIOS.

Figure 1 below shows a layered model of the DOS-BIOS. This Basic I/O system remains largely unchanged from that originally developed by IBM.

Figure 1



# Hard Drive Size Limitations and Barriers

DEW Associates Corporation

## DOS-BIOS Layered Model

Applications that in any manner store or retrieve data, do so in the form of named files. As an example, when a word processing application wishes to save a document, the document is saved as a named file, and when that same application later retrieves the document, it retrieves it as a named file.

File managers, a regular part of the underlying operating system, handle the mapping of named files to locations on the disk drive, issuing read and write commands via the Int 13 (Interrupt 13) API (Application Programming Interface) to store or retrieve those files. When a file is to be written, the file manager reads the current directory on the disk to determine where the file may be written, adds an entry to the directory for the file, and then writes the file to the disk. When a file is read, the file manager reads the directory to locate the file on the disk and then reads the file. This Int 13 API interface is operating system independent so that the two upper layers (Figure 1) could be MS-DOS® based, Windows® based, Apple® MAC, Linux or some other operating system along with its associated file system.

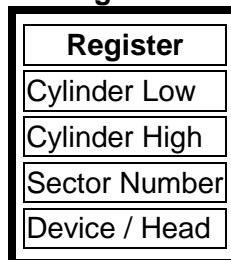
**Note:** Although we will later discuss the finite details of how data is stored on a hard drive, you should be aware that larger files may be broken up and stored in numerous locations on the hard drive rather than in any specific segment. When those large files are later retrieved, the file manager collects all of the various segments or parts of the file, and then provides them as a complete file to the application making the retrieval request.

The BIOS disk services converts the Int 13 API requests into ATA interface requests for the disk drive and then executes the actual data transfer to or from the disk drive. Let's take look at the ATA interface and the Int 13 API interface in more detail.

## The ATA Interface

The ATA interface is a register driven, parallel bus. Today, this is more commonly referred to as Parallel ATA. You will see this used more often as we discuss new technologies such as Serial ATA. To transfer data, the BIOS writes the data to defined register locations in order to set the starting address of the data on the disk and the full length of the data transfer. It then writes either a read or write command to another register location to cause the transfer of data to occur. The data on the disk is usually formatted into 512 byte sectors, and all transfers are an integer number of sectors in length. Most, if not all, of today's disk drives support both Logical Block Addressing (LBA) and Cylinder Head Sector addressing (CHS), however history shows us that BIOS's have used CHS addressing. With CHS addressing, the starting address for a data transfer is written into 8-bit registers as shown in Figure 2.

Figure 2



## ATA Address Registers

Understanding ATA address registers can be confusing at best, so take your time and re-read the explanation as necessary. Having a basic understanding of address registers will help you later in our discussions. The disk address is viewed as a number of cylinders, each of which has a specified number of heads. Each head can read or write to a number of sectors on each cylinder.

# Hard Drive Size Limitations and Barriers

DEW Associates Corporation

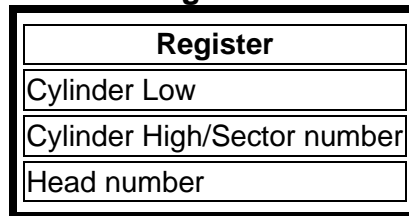
The cylinder address is a 16-bit value, broken down into two 8-bit segments. The least significant 8-bit segment is defined as the Cylinder Low register, and the most significant 8-bit segment is defined as the Cylinder High register. The head address is a 4-bit value, and is defined in the lower four bits of the Device/Head register. The sector address is an 8-bit value, and is defined in the Sector Number register. It is important to note that the first sector is defined as sector number 1 as no sector 0 (zero) exists.

Theoretically, we should be able to address up to 65,536 cylinders (2<sup>16</sup>), address up to 16 heads (2<sup>4</sup>) and address up to 255 sectors (2<sup>8</sup>-1). Therefore, up to 267,386,880 sectors (65,536 x 16 x 255) can be addressed. At 512 bytes per sector, this yields a maximum theoretical capacity of about 136.9 GB. With LBA addressing, the 28 available address bits (16 + 8 + 4) are viewed as a single LBA number. Since the value zero is included, up to 268,435,456 sectors (2<sup>28</sup>) can be addressed for a maximum capacity of about 137.4 GB. The 137-138GB limitation most system builders face today!

## The Int 13 API Interface

The Int 13 API interface is also a register driven interface. The higher layer, noted as the file manager layer in Figure 1, places a data transfer command and the parameters for that command into the host processor registers. This layer then causes an Interrupt 13 which activates the BIOS disk services causing it to execute the transfer. The starting address for the transfer is written into 8-bit registers as shown in Figure 3 immediately below.

Figure 3



## Int (Interrupt) 13 API Address Registers

The Cylinder Low register contains the least significant eight bits of the total 16-bit cylinder address. The Cylinder High/Sector number register contains the two most significant bits of the entire cylinder address in addition to a 6-bit sector number value. The Head number register contains an 8-bit head number.

As noted earlier, up to 1,024 cylinders (2<sup>10</sup>) can be addressed, up to 256 heads (2<sup>8</sup>) can be addressed, and up to 63 sectors (2<sup>6</sup>-1) can be addressed. Essentially, up to 16,515,072 sectors (1,024 x 256 x 63) can be addressed. At 512 bytes per sector, this yields a maximum theoretical capacity of about 8.456 GB. As you will see further on, 8GB happened to be one of the more recent size barriers.

With LBA addressing, the 24 available address bits (10 + 6 + 8) are viewed as a single LBA number. Since the value zero is included, up to 16,777,216 sectors (2<sup>24</sup>) can be addressed for a maximum capacity of about 8.601 GB.

## Capacity Barriers

Now that we have reviewed some of the background that has led to the first drive size limitation, let's take a look at the individual drive size barriers and limitations beginning with the 528MB barrier. This is good place to start as we will review the two methods developed to overcome the barrier, Bit Shift Translation and LBA Assist Translation. As we mentioned at the beginning, new barriers and

# Hard Drive Size Limitations and Barriers

DEW Associates Corporation

limitations are being uncovered as you read through these segments, and as quickly as those are overcome, new ones develop. We are making every effort to make these segments as all-inclusive as possible by providing updated information as soon as a new limitations or barriers are recognized and solutions are developed. At times, all-inclusive may not be possible. Let's begin at the beginning!

## The 528 MB Barrier

At the time engineers were developing the first BIOS's, the general belief was that the capacity of a hard disk drive would ever exceed 500 MB. As a result, when translating addresses from the Int 13 API to the ATA registers they simply took the 10-bit cylinder address and placed it into the ATA Cylinder High and Cylinder Low registers, always filling the upper six bits of the Cylinder High register with zeros. They took the 6-bit sector number address and placed it into the ATA Sector Number register, always filling the upper two bits with zeros. Finally, they made the assumption that the head address would never have more than four significant bits, and with that took the lower four bits and placed them in the ATA Device/Head register.

So, in this instance, up to 1,024 cylinders (210) could be addressed, up to 16 heads (24) could be addressed, and up to 63 sectors (26-1) could be addressed. Therefore, up to 1,032,192 sectors (1,024 x 16 x 63) could be addressed, and with 512 bytes per sector, this yields a maximum theoretical capacity of about 528.4 MB. This is the root cause of the first drive size boundary, the 528 MB barrier.

In order to overcome this, engineers developed a different BIOS that implemented either of two new algorithms for converting the Int 13 API address to the ATA address.

During early drive development, many disk drives did not support LBA addressing, and the first and most prevalent algorithm was known as Bit Shift Translation. This involved translating the cylinder and head addresses so that the total number of sectors remained the same but the addresses would fit within the already allotted register space. Table 1 below defines the translations for you.

**Table 1**  
**Bit Shift Translation**

ATA cylinder max address	ATA head max address	Int 13 cylinder max address	Int 13 head max address	Theoretical maximum capacity
1C1,024	1H16	C=C	H=H	528.4 MB
1,024C2048	1H16	C=C/2	H=H*2	1.057 GB
2048C4096	1H16	C=C/4	H=H*4	2.114 GB
4096C8192	1H16	C=C/8	H=H*8	4.228 GB
8192C16384	1H16	C=C/16	H=H*16	8.456 GB
16384C32768	1H8	C=C/32	H=H*32	8.456 GB
32768C65536	1H4	C=C/64	H=H*64	8.456 GB

Let's review an example of Bit Shift Translation. If the disk drive reports 16,384 cylinders and 16 heads to the BIOS disk services, the BIOS disk services reports that it has 1,048 cylinders and 256 heads to the upper layers. Note that this will yield the same number of total sectors on the disk, however the addresses for cylinder and head will now fit within the original 10-bits and 8-bits of the Int 13 register fields. The BIOS disk services then translates the 16-bit and 4-bit values when passing

# Hard Drive Size Limitations and Barriers

DEW Associates Corporation

them to the ATA registers. These translations yield the theoretical maximum capacities as shown in Table 1.

**Important!** If the disk drive cannot be configured with the number of cylinders and heads that will fit into one of the translation values as indicated in Table 1, bit shift translation will not work. In addition, since the bit shift translation only translates the cylinder and head values, should the number of sectors the drive reports be less than 63, the theoretical maximum drive capacity cannot be achieved.

Now let's take a look at the second translation method, LBA Assist Translation. This translation method can only be utilized "if the disk drive supports LBA addressing". There are no exceptions! The disk itself must support LBA, and this has nothing to do with your motherboard, drivers or anything else! With this type of translation, the BIOS disk services first determines the total number of sectors reported by the disk drive by multiplying the number of cylinders by the number of heads and then by the number of sectors reported by the drive. It then reports the configuration to the upper layers as shown in Table 2.

**Table 2**  
**LBA Assist Translation**

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

With this type of translation, the cylinder, head and sector address values will always fit into the Int 13 API registers. When the BIOS disk services gets the Int 13 address, it multiplies the three values thereby obtaining a valid LBA address for the disk drive. It then issues the resultant command to the drive using LBA addressing. With this translation of the cylinder, head and sector information, the entire capacity of the disk drive can be utilized.

## Possible Solutions:

This barrier or limitation is most often found in older legacy systems (pre-Pentium II) such as 286, 386, 486 and early Pentium I systems. Unless there is a very good reason for retaining these old units, our first recommendation would be to replace the problem system. Here are the recommended solutions.

- If the motherboard and BIOS are still supported by the manufacturer, update the BIOS to a more current version to overcome the 528MB barrier and replace the hard drive.
- Try a drive overlay such as those made available by Maxtor and Western Digital.
- Upgrade or replace the motherboard, processor and memory.
- Replace the problem system.

# Hard Drive Size Limitations and Barriers

DEW Associates Corporation

## The 2.1 GB Barrier

The 2.1 GB barrier is actually two barriers in one, a hardware barrier as well as a software barrier. Let's take a look at these two barriers and the associated problems they create.

### The Hardware Barrier

BIOS developers and engineers, in order to solve the 528 MB problem we mentioned earlier, 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). This, however, creates yet another problem as you are about to see.

Keep in mind, none of the predominant operating systems in use today use this method of translation! 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.

### The Software Barrier

The software barrier is one directly related to DOS or MS-DOS® partitioning issues which are derived from the File Allocation Table (FAT) that DOS uses to keep track of hard disk space. The FAT is only capable of working with 32,768 bytes per cluster and no more than 65,536 clusters. If you multiply the two numbers together you get the maximum partition size that DOS can use of 2,147,483,648 bytes or 2,048 MB (2,147,483,648 / 1,024). Hence, the 2,048 MB maximum partition size is the 2.1GB software barrier, and DOS, MS-DOS® and FAT are the culprits.

### Possible Solutions:

This barrier or limitation is most often found in older legacy systems (pre-Pentium® II) such as 286, 386, 486 and early Pentium® I systems. Unless there is a very good reason for retaining these old units, our first recommendation would be to replace the problem system. Here are the recommended solutions.

- If the motherboard and BIOS are still supported by the manufacturer, update the BIOS to a more current version to overcome the 528MB barrier and replace the hard drive.
- Try a drive overlay such as those made available by Maxtor and Western Digital.
- Upgrade or replace the motherboard, processor and memory.
- Replace the problem system.

As you review the remaining groups of barriers that have arisen as drive capacities increase, please keep the original barriers in mind. You will find this original historical information useful when trying to understand the overall scope of these problems.

# Hard Drive Size Limitations and Barriers

DEW Associates Corporation

## The 4.2 GB Barrier

Unlike the 528MB barrier, which is both hardware and software related, the 4.2GB 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. Referring to Table 1, this occurs any time the device reports 16 heads and greater than 8,192 cylinders to the Bit Shift translation.

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.

Operating systems affected by this issue are DOS, MS-DOS®, Windows 3x, Windows 95® (version 95A). There are some related issues with Windows NT® 3x and Windows NT® 4x when deployed on drives using the FAT file system. When deployed on NTFS volumes, the problem usually disappears.

## Possible Solutions:

This barrier or limitation is most often found in older legacy systems such as early Pentium® I and II systems. Unless there is a very good reason for retaining these old units, our first recommendation would be to replace the problem system. Here are the recommended solutions.

- If the motherboard and BIOS are still supported by the manufacturer, update the BIOS to a more current version to overcome the 528MB barrier and replace the hard drive.
- Try a drive overlay such as those made available by Maxtor and Western Digital.
- Upgrade or replace the motherboard, processor and memory.
- Replace the problem system.
- On Windows NT® based systems utilizing the FAT file system, convert the drive (volume) to NTFS.

## The 8.4 GB Barrier

One of the more recent barriers or limitations to surface is the 8.4GB Barrier. As demand for more usable drive space grew, and as drive development progressed, hard drive manufacturers and system builders faced yet another hurdle. This barrier arose just a short time after the development of Bit Translation and LBA Translation methods, and in spite beliefs that such barriers were finally over.

The LBA translation method discussed above uses ID words 1,3 and 6 of the Identify Drive command, for which the maximum values are 16,383 cylinders, 16 heads and 63 sectors for a total capacity of 8.456 GB. In order to go beyond this boundary, new Extended Int 13 functions had to be defined that no longer pass the disk drive addressing via the host registers. Instead, the address of a Device Address Packet in the hosts memory is passed. The BIOS disk service then reads this packet to determine the actual disk address. The address provided in the Device Address Packet is a 64-bit LBA address. If the disk drive supports LBA addressing, the lower 28 bits of this address may be passed directly to the ATA registers. If the device does not support LBA addressing, the host converts the LBA address to a CHS address and places that in the ATA registers.

# Hard Drive Size Limitations and Barriers

DEW Associates Corporation

Some of you who are system builders, or even those of you who build your own personal systems, may recall when these new Extended Int 13 functions first appeared. This new development enabled computer component manufacturers to again breach the drive size barriers they thought they had cured just a short time before. More important though is the fact that the redevelopment of the Extended Int 13 function would move the barrier substantially, giving drive manufacturers and BIOS developers some breathing room for the immediate future. This change in the Extended Int13 function creates a theoretical 136.9 GB capacity addressable via CHS, or 137.4 GB addressable via LBA, which can be utilized over the ATA bus.

In these first two segments we touched upon the basic drive barriers and limitations and where all of the problems began, however, the barriers and limitations we have mentioned thus far are just the tip of the iceberg. Our in-depth discussion covers most, if not all, of the known barriers and limitations through April of 2002.

Now that you have had a look at the basic barriers and limitations, you may want to take a moment and review some of the symptoms of these barriers as well as potential solutions that you might use to resolve the problem, even if only temporarily. Possible Symptoms and Solutions for Basic Barriers. If you would rather skip the next segment and move on to an in-depth review, just follow the link below.

## Hard Drive Size Limitations and Barriers In Depth

In the first segment, Hard Drive Size Limitations and Barriers - The Basics, we covered the basics of hard drive size barriers and limitations, how it began and what you might expect when upgrading to larger hard drives, especially in older (legacy) equipment. In this segment, we will cover some of those same limitations and barriers in greater depth along with all of the recently documented barriers. Bear in mind that as computer systems grow faster and drive technologies develop, new barriers will surface. As they do, we will update this area accordingly. If you find, after reviewing this material, that we have overlooked a particular barrier or limitation, or one surfaces that doesn't appear here, please write us [✉](#) and we will document the circumstances.

As older legacy equipment is discarded in favor of today's faster systems with tons of new technology, many of the older barriers will disappear. No doubt some of you will resurrect some of those old systems, and hopefully the information we have provided will enable you to resolve some of the barrier problems associated with them. Almost all of the problems we now face result either from a motherboard BIOS or operating system that hasn't evolved as quickly as drive technology. Obviously the argument can be made that most of these problems are the result of shortsightedness on the part of developers of hard disk structures, access routines and operating systems. That argument is rather hollow given that many of today's hardware and software bugs could not have been detected until hard disks grew in size exceeding some previously unknown limit or barrier. Fortunately, the solutions to most of these problems are coming nearly as fast as the problems themselves.

In discussing some of these recent barriers, we have provided a brief summary of the reasons for each of the barriers, as well as the drive capacity generally associated with it. The capacity limits are provided in both decimal (GB) and binary (GiB) formats, as most people look for specific decimal or binary numbers when identifying a particular barrier. For a complete discussion of the differences between decimal and binary measurements, see [Binary v. Decimal Measurement](#).

Important: Most of the issues we will discuss here are BIOS related, and hence involve IDE/ATA hard disks rather than SCSI hard disks. Some, however, are equally relevant for SCSI drives, especially those that involve operating system limitations. Our primary focus is that of explaining the nature of the various hard disk barriers and how they arose. We won't expand too much on how to resolve each of these barriers here, as the techniques are common to many different barrier types surfacing today.

# Hard Drive Size Limitations and Barriers

DEW Associates Corporation

## Early Size Barriers and Limitations A Refresher

The first of the hard disk barriers was the 504 MB / 528 MB barrier that showed up some time in 1994. Though not widely known, there were a number of older capacity barriers that affected hard drives well before the 504/528 limit. They, however, never made it into print! These unpublished barriers received little attention for no other reason than there were no large groups of PC users as there are today, and developing technology was still in its infancy. As technology advanced, the associated upgrades were often the reason a particular barrier problem surfaced.

Although these earlier barriers have no relevance to modern computing, we felt they might be of interest to those of you who still have some of these older machines or a nostalgic interest. Since they have no impact today, our descriptions are brief.

- **PC/XT Parameter (10.4 MiB / 10.9 MB) Barrier:**

The very first PC's that used hard disk drives were the PC/XT units produced by IBM. These were specifically designed to use a particular type of disk with 312 cylinders, 4 heads and 17 sectors per track and pre-configured to approximately 10 MB in capacity.

- **FAT12 Partition Size (16 MiB / 16.7 MB) Barrier:**

The first type of FAT format used for hard disks was that of the 12-bit FAT12 partition, which is still in use today for floppy disks. This type of FAT allows a maximum of 4,086 clusters of 4,096 bytes, for a total of 16,736,256 bytes per disk.

- **DOS 3 (32 MiB / 33.6 MB) Barrier:**

When IBM introduced their PC AT product line, they were faced with the 16 MiB barrier from the previous design. In order to work around the barrier, they modified their operating system, DOS 3.x, adding support for the FAT16 file system. This, however, created yet another new barrier. Rather than look to the future, the FAT16 implementation was merely a work around to get past the immediate 16MiB barrier. The cluster size was set to 2,048 bytes, and only 16,384 FAT entries were allowed, fixing the maximum capacity at an approximate 32 MiB. While they did introduce the ability to have multiple partitions at about the same time, each partition could only be 32 MiB or less.

- **DOS 4 (128 MiB / 134 MB) Barrier:**

DOS 4.x was an improvement over DOS 3.x as it permitted 65,526 clusters instead of 16,384, increasing the maximum partition size to about 128 MiB. Unfortunately though, the cluster size was still fixed at 2,048 bytes.

It is evident that the majority of these early limitations did not arise because of BIOS issues, but rather some short-sightedness on the part of the DOS design team. They were barely staying a year or two ahead of the hard disk technology curve! In addition to the above barriers, there was also the 512 MiB barrier, caused by the change made in DOS version 5. This operating system change from DOS 4 allowed cluster sizes in a single partition to increase to 8,192 bytes, thereby allowing a theoretical maximum partition size of about 512 MiB or 537 MB. However, most systems that had drives large enough for this to be an issue were prevented from using the full size due to the slightly smaller 504 MiB / 528 MB BIOS barrier.

## Current Size Barriers and Limitations

Although some of the early barriers we have presented thus far may appear old and irrelevant to those of you with new, or fairly new, computers, we assure you that not everyone has a fairly new computer and these barrier difficulties are extremely relevant. The information we have provided with

# Hard Drive Size Limitations and Barriers

DEW Associates Corporation

respect to these barriers will allow you to focus on system problems that may arise out of nowhere without that might not appear barrier related.

## **The 1,024 Cylinder (504 MiB / 528 MB) Barrier**

The most troubling of the hard disk barriers effecting standard IDE/ATA hard disks was the 504 MB limitation. This particular barrier began surfacing in systems in 1994. Given the larger hard disks available today, and the fact that this barrier occurred several years ago, its relevance is more historical than anything else. Many of you who are relatively new to personal computers, or at least the technical aspects of them, may not be aware of how troubling this large disk barrier was. It was a unique experience for users and technicians alike who believed that barrier limitations had been conquered.

As the result of this barrier, but only under certain circumstances, a hard disk with a size over 504 MB will appear as having only 504 MB. This results from the combination of the geometry-specification limitations of the IDE/ATA standard and the BIOS Int 13h standard. The 504 MB barrier is alternatively referred to as the 528 MB barrier, depending on whether you are looking at binary or decimal megabytes.

This barrier, and most of those that followed it that were entirely BIOS related, were eliminated in new BIOS firmware routines as out lined in a standards document released by the Technical Committee T13.

## **1226DT Enhanced BIOS Services for Disk Drives**

Abstract: This technical report describes new services provided by BIOS firmware to support ATA hard disks up to 16 mega-tera-bytes (16x10<sup>18</sup>). Older BIOS services have a compatibility limit of 528 MB and a theoretical limit of 8.4 GB.

## **How does the BIOS in your computer know the size and parameters of the installed hard disk?**

Simple, your BIOS asks the hard drive for the information, specifically its geometry. Each hard disk has the capability to disclose to the BIOS how many cylinders, heads and sectors that it has available for use. A hard disk is addressed through its geometry when a sector needs to be read from or written to, which is usually referred to as its logical geometry, and not the physical hardware geometry actually inside the hard disk assembly.

When you perform a drive parameter auto-detection in your systems BIOS setup or look in your new drives setup manual to see what the parameters are, what you see are the logical geometry values that the hard disk manufacturer has specified for the drive. Newer drives use zoned bit recording with ten or more values for sectors per track depending on which region of the disk is being examined, therefore it is not possible to set up the disk in the BIOS using the physical geometry. In addition, the BIOS has a limit of 63 sectors per track, and all newer hard disks average more than 100 sectors per track, so even without zoned bit recording, there would be a problem correctly recognizing the drive through its physical geometry.

Older hard disks have simple structures and low capacity, therefore they do not need special logical geometry. Their physical and logical geometry are the same. If you were to examine an older drive such as a 42.8 MB, which was very popular in early computers, its physical geometry consists of 820 cylinders, 6 heads, and 17 sectors, and those numbers are what is used by the system BIOS.

Newer drives cannot have their true geometries expressed using three simple numbers. To get around this issue, for disks 8.4 GB or smaller, the BIOS is provided with erroneous parameters that

## Hard Drive Size Limitations and Barriers

DEW Associates Corporation

provide the approximate capacity of the disk. These parameters combined with an intelligent hard disk controller results in an automatic translation between the logical and physical geometry. The actual physical geometry is still different, but the BIOS is completely unaware of it.

Software systems reserve a certain amount of space for specifying each of the three parameters that make up the hard disk geometry. The amount of space reserved is dictated by standards that control how IDE/ATA hard disks are supposed to work, as well as how the BIOS interprets the hard disk through its Int13h software interface.

Unfortunately though, there has been little or no planning and coordination for the development and implementation of these standards as they are not universal by any means. Each one reserves a different set of numbers of bits for the disk geometry. In order to use an IDE/ATA hard disk with a standard BIOS disk routine, the limitations of both standards must be observed, which means that only the smaller of each geometry number can be used. Here is a look at how the two standards allocate bits for the geometry:

Standard	Bits For Cylinder Number	Bits for Head Number	Bits for Sector Number	Total Bits for Geometry
IDE/ATA	16	4	8	28
BIOS Int 13h	10	8	6	24
Combination (Smaller of Each)	10	4	6	20

Since each drive geometry figure is a binary number with a specific number of bits as indicated above, the maximum number supported for any parameter is  $2^N$ , where N is the number in the table above. Therefore, when referring to the IDE/ATA standard,  $2^{16}$  or 65,536 cylinders are supported. We then multiply all the figures together to get a total number of sectors supported, and then multiply that result by 512 bytes (per sector) to get the maximum supported capacity. Here's how they look after conversion.

Standard	Maximum Cylinders	Maximum Heads	Maximum Sectors	Maximum Capacity
IDE/ATA	65,536	16	256	128 GiB
BIOS Int 13h	1,024	256	63	7.88 GiB
Combination (The smaller of each)	1,024	16	63	504 MiB

The BIOS Int 13h limit for sectors is 63, and not 64. By design, sectors are numbered starting at 1 and not 0.

Repeating what we have stated earlier, the 504 MiB figure is just  $1,024 * 16 * 63 * 512$ , which equals 528,482,304. The problem is the combination of the limitations imposed by the two standards. Due to the 16-head limitation of IDE/ATA, no IDE hard disk is ever specified with more than 16 logical heads. Instead, they must always have a large number of cylinders in order to create a large capacity drive. Unfortunately, when you install one of these large capacity drives in a system with a standard, non-

# Hard Drive Size Limitations and Barriers

DEW Associates Corporation

translating BIOS, the system can't see more than 1,024 cylinders. There are several ways that a system may react to a drive too large for the legacy system. You can view the possibilities by clicking [here](#).

The usual solution to the 504 MiB problem is to use a system (or motherboard) with a current BIOS that supports BIOS translation. However there are drive overlays that will work too! This provides a work-around for the problem by using what is little more than a software trick, but it does work. Software drive overlays, such as those made available by Maxtor and Western Digital, will also help you overcome the problem. When using drive overlays, be aware that they come with their own difficulties, therefore be careful and consider all of the implications. In our discussion, "Resolving Hard Drive Barriers and Limitations", you will find an entire section devoted to resolving this barrier.

## **The 4,096 Cylinder (1.97 GiB / 2.11 GB) Barrier**

As discussed in detail above with respect to the 504 MiB barrier, the basic problem with BIOS related capacity barriers is that the normal system (motherboard) BIOS interface on older computers is not designed to handle hard disks that employ cylinder counts above 1,024. Every hard disk made today uses more than 1,024 cylinders, which would cause a drastic reduction in available capacity due to this limitation.

Systems that use an enhanced BIOS are able to employ translation to get around the 1,024 cylinder limitation. However, some BIOS's, in spite of their support of translation, will still choke if the number of cylinders exceeds 4,095. This results in the same 504 MiB barrier problem surfacing all over again. When you do the math,  $2^{12}$  is 4,096, which means that if you go beyond 4,095 cylinders on a drive, the number will require a 13th bit to properly represent it. As an example, 4,097 in decimal is 1000000000001 in binary. Normally this wouldn't be a problem, however some inexpensive (no cheap!) motherboards rely on poorly written BIOS code, which will result in only 12 bits of the cylinder number being recognized. The actual limitation is  $4,096 * 16 * 63 * 512$  bytes, which is approximately 1.97 binary gigabytes or 2.11 decimal gigabytes. Some BIOS's evidencing this problem will show a disk with more than 4,096 cylinders as being 1.97 GB, while others will show it as substantially less. For example, 4,097 cylinders may show up as only 1 cylinder if the 13th bit is merely ignored and the lower-order 12 bits are used by alone!

Do not confuse this capacity barrier with the FAT 16 Partition Size (2.00 GiB / 2.15 GB) barrier (which is exactly 2 binary gigabytes) discussed next, as that limitation is purely a file system issue, and completely unrelated to the BIOS issue we are discussing here. This particular size barrier began surfacing on systems in 1996. There are several options available for resolving this barrier, which are similar to those used for the 504 MiB limitation. See [Resolving BIOS and Drive Size Barriers](#).

## **The FAT16 Partition Size (2.00 GiB / 2.15 GB) Barrier**

The 2 GiB capacity barrier is unlike most of the other barriers we have discussed thus far, as it is purely an operating system problem that has nothing at all to do with the BIOS. It is different than the 1.97 GiB barrier, as it is BIOS related. In many ways, it is similar to the early BIOS barriers that preceded the 504 MiB barrier as discussed at the beginning of this segment.

The 2 GiB capacity barrier is a limitation on the size of disk volumes in the FAT 16 file system. Due to the way that disks are set up using clusters, it is not possible to have more than 2 GiB in a single partition when using either the DOS, Windows 3.x or the early Windows 95 version "Windows 95A". Under Windows NT, the limit is 4 GiB instead of 2 GiB when using FAT partitions. NTFS partitions do not have this limitation!

# Hard Drive Size Limitations and Barriers

DEW Associates Corporation

Using 4 GiB FAT partitions under Windows NT requires the use of 64 kiB clusters. This is supported but non-standard, and will result in problems if you try to set up a system using other operating systems in addition to Windows NT.

We discuss all of these issues in our Operating Systems and their File Systems segment.

If you put a hard disk over 2 GiB into a machine that is using FAT 16 (16-bit FAT) under DOS, Windows 3.x or the first version of Windows 95, you can use all of the drive on the assumption that you haven't been limited by one of the other BIOS related barriers mentioned above or in our "Basics" segment. To access the full contents of the disk, however, you must create several partitions. This limitation is a function of the operating system, and it affects IDE/ATA and SCSI hard disks equally.

☑ This limitation does not apply to disks formatted using the later FAT 32 file system. FAT 32 is an enhancement created specifically to avoid this problem, and was first introduced in Windows 95 OEM SR2, (the second version of Windows 95). It is supported in Windows 98, Windows ME, Windows 2000 and Windows XP. There is also the NTFS file system, supported by Windows NT, Windows 2000 and Windows XP, which uses a completely different set of structures and can have truly enormous partitions. Review our segment on NTFS, the "New Technology File System".

## **The 6,322 Cylinder (3.04 GiB / 3.26 GB) Barrier**

The 3.04 GiB/3.26 GB barrier is one of the more obscure of the size barriers that you will find, as it affects only a small percentage of computer systems. On these particular systems, the BIOS cannot handle hard disk geometry having more than 6,322 cylinders. Setting cylinder value higher than 6,322 causes the system to hang. This barrier fixes the capacity on affected systems to about 3.04 GiB or 3.26 GiB,  $6322 \text{ cylinders} * 16 \text{ heads} * 63 \text{ sectors} * 512 \text{ bytes}$ . Oddly enough, there's no apparent significance to the 6,322 number, as it isn't a round number in either decimal or binary.

Solving this barrier is usually accomplished by either updating the BIOS or replacing the motherboard. In some cases, a drive mask can be used such as those available from Maxtor and Western Digital.

## **The Phoenix BIOS 4.03 / 4.04 Bug (3.05 GiB / 3.28 GB) Barrier**

This is another of the rather obscure size barriers, however this one that differs from the others described thus far. This particular barrier differs as it isn't the result of poor or inadequate BIOS design, nor as the result of an operating system characteristic or limitation. This particular limitation is the result of a programming error or a bug in a few types of systems manufactured during the mid 1990s. Some systems that used the Phoenix BIOS, specifically versions 4.03 or 4.04, have a problem with the BIOS routine that calculates hard disk drive sizes. BIOS code is initially written by the BIOS maker, and subsequently specifically tailored by system or motherboard manufacturers. The obscure nature of this barrier is due to the fact that not all implementations of these BIOS versions may have this bug while others may.

This is not only a difficult barrier to diagnose, but it is also very difficult to resolve. The barrier actually isn't a single value, and appears to depend on the values of the geometry parameters. Its behavior can be different based on the values entered. If you were to use standard IDE head and sector values of 16 and 63 respectively, the cylinder field can have a maximum value of 6,349 without any problems, resulting in a maximum capacity of 3.05 GiB or 3.28 GB. If you were to change the cylinder value to a value between 6,350 and 8,322, the BIOS setup program may lock up. If you were to use cylinder values of 8,323 to 14,671, the displayed drive size is incorrect. Phoenix seems to have corrected this problem in subsequent versions of this BIOS code.

# Hard Drive Size Limitations and Barriers

DEW Associates Corporation

## The 8,192 Cylinder (3.94 GiB / 4.22 GB) Barrier

After the discovery of the 504 MiB BIOS barrier, the resolution of the problem was to make use of BIOS geometry translation. This method continued until hard drives exceeded about 8 GB in size and the whole IDE/ATA geometry scheme had to be abandoned altogether. Translation works by dividing the number of cylinders or a hard drive by a binary number such as 2, 4, 8 or 16, and then multiplying the number of heads by the same number. While this allows the number of cylinders the BIOS sees fall below the Int13h limit of 1,024, it also causes a problem in some systems when using a hard disk of 4 GB or greater in size.

To understand how translation causes this particular new barrier to arise, you need to understand how BIOS translation works. [Click here for an explanation of BIOS Translation.](#)

When the number of cylinders on the drive is between 8,192 and 16,383, the typical number used for translation is 16. Using a theoretical 6.4 GB hard disk with a typical IDE/ATA specification, 16 heads and 63 sectors per track:

	Cylinders	Heads	Sectors	Capacity
<b>IDE/ATA Limits</b>	65,536	16	256	137 GB
<b>Hard Disk Logical Geometry</b>	<b>12,496</b>	<b>16</b>	<b>63</b>	<b>6.45 GB</b>
<b>BIOS Translation Factor</b>	<b>divide by 16</b>	<b>multiply by 16</b>	--	--
<b>BIOS Translated Geometry</b>	<b>781</b>	<b>256</b>	<b>63</b>	<b>6.45 GB</b>
<b>BIOS Int 13h Limits</b>	1,024	256	63	7.88 GB

Theoretically, this should work just fine, as it overcomes the BIOS issues and results in a geometry that falls within acceptable parameters. Unfortunately a problem arose when drives began exceeding 8,192 cylinders during the 1997-98 time period. MS-DOS and early versions of Windows choke when presented with a drive that had 256 heads. Hence, this barrier is caused by both the operating system and the system BIOS. The operating system should have been able to easily handle 256 heads, however the BIOS was creating the problem because of the geometry translation.

The simplest way to resolve this problem would have been to change the way the BIOS does translation. As a result, BIOS's stopped creating translated geometries that had 256 heads. One method used was to use 15 as the translation factor instead of 16, resulting in this sort of conversion:

	Cylinders	Heads	Sectors	Capacity
<b>IDE/ATA Limits</b>	65,536	16	256	137 GB
<b>Hard Disk Logical Geometry</b>	<b>12,496</b>	<b>16</b>	<b>63</b>	<b>6.45 GB</b>
<b>BIOS Translation Factor</b>	<b>divide by 15</b>	<b>multiply by 15</b>	--	--
<b>BIOS Translated</b>	<b>833</b>	<b>240</b>	<b>63</b>	<b>6.45 GB</b>

## Hard Drive Size Limitations and Barriers

DEW Associates Corporation

Geometry				
BIOS Int 13h Limits	1,024	256	63	7.88 GB

Of course, if you have a BIOS that doesn't know about the 256 head problem, you will need to address this with either a hardware or software solution. In order to avoid some of these problems, many hard disk manufacturers changed their specified geometries to use only 15 heads instead of 16. So instead of the example drive above being specified with 12,496 cylinders, 16 heads and 63 sectors, it might have been 13,329 cylinders, 15 heads and 63 sectors. With these parameters, even if the BIOS used a translation factor of 16, the resulting number of heads would only be 240.

### The 240 Head Int 13 Interface (7.38 GiB / 7.93 GB) Barrier

The Interrupt (Int) 13h interface limit normally restricts some systems to 7.88 GiB or 8.46 GiB as a result of the limits of the BIOS Int13h interface, 1,024 cylinders, 256 heads and 63 sectors of 512 bytes. This issue is covered more fully immediately below. In some systems, however, the Int13h interface restriction results in a smaller limit of only 7.38 GiB or 7.93 GB.

The reason behind this occurrence though is related to an entirely different barrier problem. As described in the discussion of the 8,192 cylinder limit above, DOS and some early Windows® versions cannot properly handle geometry translations that specifies 256 heads. BIOS and motherboard developers, in order to work around this issue, change the translation method so that only 240 heads are presented to the operating system. While this does fix the 256 head problem, it reduces the drive capacity. In spite of this work around, the 1,024 cylinder and 63 sector restrictions remain, but with only 240 heads. The maximum drive capacity becomes  $1024 * 240 * 63 = 15,482,880$  sectors of 512 bytes, or 7,927,234,560 bytes. In reality, there's little difference in how this standard Interrupt 13h problem is resolved. Remember, you still need to use Int13h extensions, which are given a little more depth in the next section.

### The Int 13 Interface (7.88 GiB / 8.46 GB) Barrier

This barrier is often characterized merely as the 8 GB barrier, however it is one of the most important barriers facing the small system builder and those upgrading their computer system. There's no hiding the fact that today's hard disk capacities have surpassed the 100GB mark and moving beyond that quickly. This particular barrier, based upon its influence upon system builders and those doing upgrades, is very much like the early 528MB barrier that suddenly got everyone's attention. This particular barrier usually surfaces when system builders or end-users attempt to upgrade systems built during the late 1990s that had hard disks of 1 GB to 4 GB in size.

Similar to many of the other drive barriers, this one is also BIOS related, although its somewhat tougher to resolve as it involves the traditional limits imposed by how hard disks are used in personal computers along with the Interrupt 13h interface. That standard allocates 10 bits for the cylinder number, with a maximum of 1,024 cylinders, 8 bits for the head number, with a maximum of 256, and 6 bits for the sector number, with a maximum of 63 (the number 0 is not used). When you multiply these together, presuming the standard of 512 bytes per sector, the result is the maximum of 8,455,716,864 bytes. This is the largest hard disk size that can be addressed using the standard Int13h interface.

Unfortunately there is no translation available that can resolve this because it doesn't result from a combination of limitations like the earliest 504 MiB barrier. Factually it is a limitation imposed as the result of how hard disks are represented using the BIOS Interrupt 13h routines used by DOS and other applications to access the hard disk. In order to resolve this particular problem we need to move

# Hard Drive Size Limitations and Barriers

DEW Associates Corporation

beyond the "standard Int13h routines and move on to the enhanced Int13h extensions in the motherboards BIOS.

Don't be misled by the confusion between standard and enhanced interrupt 13h extensions. Enhanced Int13h extensions require full support from both the BIOS and the operating system. Most older operating systems do not support Int13h extensions, and as far as we know there are no plans of providing them. All versions of non-Windows® DOS, such as 6.22 and earlier, as well as Windows NT version 3.5 do not support Int13h extensions and cannot use hard disks over 8.4 GB in size.

✍️As we have mentioned on a number of occasions throughout this segment, BIOS developers are not solely responsible for some of these limitations. After a BIOS is developed and delivered to a motherboard manufacturer, it is often modified even further for proprietary reason known only to the motherboard manufacturer. Some systems have a smaller Int13h capacity limit due to the use of modified translation to avoid presenting geometry with 256 heads to the operating system. You can review this information again by clicking [here](#) for the details.

## The Windows 95 and the 29.8 GiB / 32.0 GB Barrier

If you have done any research at all regarding this alleged barrier, you probably took note of the fact that there has been quite a bit of discussion about this barrier by a large number of reasonably savvy tech oriented web sites. Many of these tech sites espouse that in 1999 Microsoft announced that Windows® 95 would not be supporting hard disks beyond 32 GB in size as one of the reason for the barrier, some going as far as to blame Microsoft directly. Their statements, and their assumptions, are very broad and very inaccurate.

While it is true that Microsoft announced that they had no plans to support FAT 32 in the first version of Windows®, version 95A, there is support in last Windows® 95 OSR2 version, as well as Windows® 98 and Windows® ME. It is because of this erroneous information that we have decided to include this perceived barrier (for lack of a better description) in our Drive Size Barrier segment.

In order for Microsoft to provide support for FAT 32 in Windows® 95A, the kernel would have to be rewritten, among other things. In addition, Windows® 95A is still entirely reliant on MS-DOS® version 6.22, which does not support FAT 32, and like Windows® NT, it cannot see FAT 32 volumes. Beyond this, and except for test computers, most personal computers that have Windows® 95A installed have other limitations, such as standard Int13h disk access instead of support for Int13h Extensions. No doubt there would be a multitude of other BIOS related restrictions as well. Follow this link to review Microsoft's Knowledge Base Article on the subject: Q246818 Windows 95 Does Not Support Hard Disks Larger Than 32 GB.

First and foremost, this is not a 32GB issue, but rather a FAT 32 issue, and even saying that is a stretch. Here's some information that may assist you with this issue. This limitation involves the operating system itself, old MS-DOS® issues, BIOS limitations, drive geometry as well as other factors. As you read through the following, you will better understand some of the problems.

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

- Clusters cannot be 64 kilobytes (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.

# Hard Drive Size Limitations and Barriers

DEW Associates Corporation

- 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 FATs themselves, this works out, at the maximum of 32 KB per cluster, to a volume size of 127.53 gigabytes (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.
- 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."

## Windows 2000 Specific Issues:

- Setup Does Not Check for INT-13 Extensions
- 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."

## Windows XP Specific Issues:

- You cannot format a volume larger than 32 gigabytes (GB) in size using the FAT32 file system during the Windows XP installation process. Windows XP can mount and support FAT32 volumes larger than 32 GB (subject to the other limits), but you cannot create a FAT32 volume larger than 32 GB by using the Format tool during Setup. If you need to format a volume that is larger than 32 GB, use the NTFS file system to format it. Another option is to start from a Microsoft Windows 98 or Microsoft Windows Millennium Edition (Me) Startup disk and use the Format tool included on the disk.

## The 65,536 Cylinder (31.5 GiB / 33.8 GB) Barrier

This barrier is relatively recent, and along with a couple of others began showing up during the spring and summer of 1999. Although this barrier is often referred to as the 32 GB barrier similar to the one immediately above, that description is a bit of a misnomer.

This particular barrier is caused by some versions of the Award BIOS not being able to handle drives having more than 65,535 cylinders. Most hard disk parameters use 16 heads and 63 sectors, which

# Hard Drive Size Limitations and Barriers

DEW Associates Corporation

works out to a capacity of approximately 33.8 GB or 31.5 GiB. It is our understanding that on or about June of 1999, this problem had been corrected by Award. This is somewhat of an unusual barrier given that most, if not all, hard disks above 8 GB no longer use discrete geometry for access, instead Logical Block Addressing is used along with a flat sector number from 0 to one less than the number of sectors on the drive. No doubt this 65,536 cylinder problem must somehow be related to some older code that was being used, or a compatibility issue with older hard drives (or both). From everything we have been able to examine, this issue was limited to a few machines that relied upon old Award BIOS code that was subsequently corrected with an update.

## The Windows 98/98SE 64GB Barrier

This barrier is often mischaracterized as a Windows® 95/98/SE barrier, when in fact it is not a limitation of the operating system at all. Rather, the barrier (a limitation actually) is caused by the use of an old version of Microsoft's disk setup tools, Fdisk.exe and Format.com.

When you use Fdisk.exe to partition a hard disk that is larger than 64 GB (64 gigabytes, or 68,719,476,736 bytes) in size, 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.

The reason for this is that 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. Microsoft has made a fix available for this problem, which is an updated version of Fdisk.exe.

The new file information is as follows:

File Date	Time Stamp	File Size	File Name	Windows Version
05/19/00	10:30AM	64,428	Fdisk.exe	Windows 98
05/18/00	08:35AM	64,460	Fdisk.exe	Windows 98 SE

You can read more about this issue in the Microsoft Knowledge Base Article titled: Q263044 - Fdisk Does Not Recognize Full Size of Hard Disks Larger than 64 GB.

## The FAT 32 Limitation (124.55 GiB / 127.53)

Although many of the limitations imposed by the FAT 32 file system are outlined above, this particular limitation is actually imposed by one of the disk tools included with Microsoft Windows® 95, 98 and Windows® Millennium Edition.

- 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 FATs themselves, this works out, at the maximum of 32 KB per cluster, to a volume size of 127.53 gigabytes (GB).

There are no fixes or work-arounds for this issue!

# Hard Drive Size Limitations and Barriers

DEW Associates Corporation

## The ATA Interface Limit (128 GiB / 137 GB) Barrier

In order to avoid previous disk barriers and limitations, other than those imposed by the operating systems themselves, today's hard drives no longer rely upon discrete geometry (specific cylinder, head and sector numbers) and instead use logical block addressing and a sector number.

Unfortunately, even when we move away from bit addressing in favor of head and sector numbers, we still reach the limit of our ability to address all of the bits when taken together. Let's take a look at the ATA interface. There are 28 bits used for the sector number interface with the operating system, BIOS and the hard disk. This means a hard disk can have a maximum of  $2^{28}$  or 268,435,456 sectors of 512 bytes, placing the ATA interface maximum at 128 GiB or approximately 137.4 GB.

As little as one or two years ago, no one thought there would be hard drives exceeding 137.4 GB. However, as many of you have seen, hard drive capacity surpassed this mark when Maxtor released their DiamondMax Plus 540X at 160 GB on October 29, 2001. Of course, Maxtor's release of a drive at 160 GB caught many techno-geeks off guard. How could they possibly make these huge drives work? Maxtor considered the 137 GB barrier well before releasing the drive as part of an entirely new initiative to take storage capacities into the petabyte region. To conquer the 137GB barrier, when Maxtor released their new drive, they made a new Ultra ATA/133 PCI adapter card available as part of the package. Believe it or not, for a limited period, the add-in card was "free". The adapter itself, however, was Maxtor's method of bringing these new drives to market and in the process temporarily solving the barrier problem without requiring users to purchase new computers to handle the new technology.

Significant changes have occurred to the ATA interface between the hard disk and the rest of the computer system in less than a year, and more are in the making. One entity charged with the responsibility of developing the new standards for this interface (and its changes) is Technical Committee T13. It is responsible for the coordination and development of all interface standards relating to the popular AT Attachment (ATA) storage interface utilized on most personal and mobile computers today. A few years ago a number of different proposals to expand ATA addressing from 28 bits to either 48 or 64 bits were made, and over those few years the committee examined each very closely. Either of these technology changes would permit huge drive sizes. The first to surface, however, was 48 bit addressing and delivered in the form of a hard drive at 160 GB by Maxtor. Using 48-bits like Maxtor takes drive sizes 100,000 times higher than current limits. This is most definitely a signal of what lies ahead!

### Solution:

Due to BIOS limitations as well as those unique to Windows®, partitioning and formatting drives larger than 137 Gigabytes without proper driver or controller support will result in data loss when storing data to the drive beyond the 137 GB Barrier.

In order for you system to recognize more than 137 GB you will need to utilize one of the following recommended solutions:

1. If you have a motherboard that has a Intel chipset (810, 810E, 810E2, 815, 815, 815E, 815EP, 815P, 820, 820E, 830M, 830MP, 830MG, 840, 845, 850, or 860) please visit Intel's web site and download the Intel Application Accelerator. Intel's Application Accelerator supports the full capacity of drives larger than 137 GB.
2. If you do not have a motherboard that has a Intel chipset then it is recommended that you purchase an Ultra ATA 133 PCI card that supports 48 bit Logical Block Addressing (LBA). You can

# Hard Drive Size Limitations and Barriers

DEW Associates Corporation

purchase the Maxtor Ultra ATA 133 PCI Card, which supports drives that are larger than 137 GB, directly from us or your local distributor.

If you do not follow either of the above steps, and you attempt to use a drive that exceeds 137 GB and/or that relies on 48-bit Logical Block Addressing by attempting to "tweak" the system even though your ATA controller, chipset drivers and/or system BIOS do not properly support 48-bit Logical Block Addressing, data loss will occur when storing data to the drive beyond the barrier.

## What the Future Holds!

Although in development for quite some time, you might want to take a look at Serial ATA as the next major development in hard drive technology.

## How a BIOS handles Oversized Hard Disks

When you install a hard disk into a computer system that is larger than that which the BIOS is capable of handling, the system may react in a number of different ways, most of which are predictable. How a particular system responds though, will depend largely on the system, the BIOS date, and the overall quality of the BIOS routines. Most, if not all, issues noted below occur as the result of the hard disk having a total cylinder count larger than the maximum supported by the BIOS.

These are the four most common reactions you may expect with a machine having an older BIOS and a hard disk larger than it is capable of supporting, listed in the order of probability.

### Truncation:

A BIOS, when presented with a logical geometry containing more cylinders than it can handle, will simply truncate the total number of cylinders to the maximum it can support. You will usually experience this in an older BIOS that doesn't support more than 1,024 cylinders, or in some cases in a BIOS with a set maximum of 4,096 cylinders. This is commonly found in systems that do not support Int13h extensions, as these systems typically see drives larger than 8.4 GB as being just 8.4 GB in size. Although truncation defeats the purpose behind adding a larger hard disk, the risk of losing data is minimal to nonexistent, and preferable to the other possibilities outlined below.

### Wrap-Around:

Many an old BIOS will presume that the number of cylinders in a drive will always be 1,024 or below, and therefore will only look to the bottom 10 bits of the cylinder number reported by the hard disk ( $2^{10} = 1,024$ ). As a result, when drives report cylinders over 1,023, the BIOS counts up to 1,024 and then wraps around to zero and starts over. As an example, let's presume for the moment that you have purchased a brand new 60 GB hard drive for your computer. For the purposes of this example, we'll presume this new drive was manufactured by Maxtor and it has 119,150 actual cylinders. Even in a recent manufactured computer you could only manually enter a maximum of 16,383 with current ATA specifications. In this scenario, the BIOS would only see 366 cylinders. This is because the BIOS would count up to 1,024 one hundred and sixteen (116) times to yield 118,784, or wrap around 116 times, ending with a net 366 cylinders ( $119,150 \text{ minus } 118,784 = 366$ ).

A nearly identical scenario can occur with a BIOS that supports only 4,096 cylinders, as it will only look at the bottom 12 bits. A troubling problem surfaced only a few years ago when computers began upgrading storage in older systems with small (by today's standards) 2.5 GB hard disks only to learn that they had only about 400 MB of usable space showing up. Unfortunately, this was a common failure in systems that had a BIOS that didn't support more than 4,096 cylinders.

Even some BIOS's that support translation may wrap the cylinders if you disable translation. When you re-enable translation the problem may go away.

# Hard Drive Size Limitations and Barriers

DEW Associates Corporation

**BIOS Ignorance:** This is truly an ugly issue. Some systems with an older BIOS correctly reports the true number of logical cylinders of the drive, making you think the motherboard and BIOS (your system) supports the full size of the hard disk. In reality, the BIOS doesn't have a clue as to the number of cylinders or what to do with them. It's actually just reporting what the drive reports. When you attempt to partition and format the hard disk, you're faced with the stark reality of a 1,024 cylinder limitation, but this is not readily evident. It can have you chasing your tail for hours trying to determine what is wrong. Fortunately though, this is only seen in systems with an older BIOS and the 1,024 cylinder limitation.

**BIOS Failure:** While you may not view it this way, a BIOS failure when attempting to install a large hard disk in a system with a BIOS that doesn't support is probably a very fortunate situation. Obviously no one wants to deal with a barrier when upgrading their computer, but this particular failure might save you hours of searching for the reason for the failure. Many times and older BIOS will cause the system to completely lock up if you try installing a hard disk larger than can be supported. While fairly uncommon, it does occur with the more proprietary computer systems where the manufacturer withholds accurate BIOS and disk support information. Often you will experience this with some of the larger hard disk barriers and also with some of the more obscure ones.

## **What will be the Next Barrier?**

While it is true that the ATA/ATAPI-6 standard defines a method to provide a total capacity for a device of 144 petabytes, the next limit will be imposed not by the ATA devices but by many of the popular operating systems in use today. This limit will be at 2.2 terabytes (2,200 gigabytes). This barrier exists because many of today's operating systems are based on 32-bit addressing. These operating systems include many flavors of Linux, Mac OS 9.x, and Windows 95, 98, ME, NT 4, 2000, and XP (Windows XP/64-bit also has the limit because of leveraged 32-bit code).

This barrier could be real as early as 2004 if current hard drive capacity rate increases continue along the same growth trends of today.

This concludes our segment of drive barriers and limitations. If you feel that we have missed something, or that a barrier or limitation has been overlooked, please write us and tell us about it.