# Large Sector Size Proposal

**To:** **T13 Technical committee**
**From:** **Dan Colegrove**
**IBM**
**2903 Carmelo Dr.**
**Henderson, NV 89052**
**Phone: 702-614-6119**
**Fax: 702-614-7955**
**Email: dcolegro@us.ibm.com**
Date: **June 10, 2002**

## 1 Introduction

The purpose of this proposal is to provide support for two methods of increasing the sector size of disk drives, increasing the logical sector size and increasing the physical sector size. Longer logical sectors are needed to support RAID applications. Longer physical sectors, composed of multiple logical sectors, are needed to increase media format efficiency while maintaining backward compatibility.

This proposal is based on Sector Globs (d99118r0) by Hale Landis

# 2 Definitions

**physical sector:** A group of contiguous logical sectors, which must be read from or written to the device media in a single operation.

**logical sector:** A uniquely addressable set of words.

**unaligned write:** An unaligned write is a write command that does not start at the first logical sector of a physical sector or does not end at the last logical sector of a physical sector.

# 3 Specification Changes

## 3.1 Identify Device

IDENTIFY DEVICE information

| Word | O/M | F/V | | Description |
|---|---|---|---|---|
| TBD1 | O | | | Physical Sector Size |
| | | F | 15 | Shall be cleared to zero |
| | | F | 14 | Shall be set to one |
| | | F | 13 | 1 = Device has multiple logical sectors per physical sector |
| | | F | 12 | 1 = Identify word TBD2 is valid |
| | | F | 11-4 | Reserved |
| | | F | 3-0 | $2^X$ Logical Sectors per Physical Sector |
| TBD2 | O | | | Logical Sector Size |
| | | F | 15-0 | Words per Logical Sector |

### 3.1.1 Word TBD1: Physical Sector Size

Bit 13 of word TBD1 shall be set to 1 to indicate that the device has more than one logical sector per physical sector.

Bit 12 of word TBD1 shall be set to 1 to indicate that word TBD2 is valid. Bit 12 of word TBD1 shall be cleared to 0 to indicate that word TBD2 is invalid and that the logical sector size is 256 words.

Bits (11:4) of word TBD1 are reserved.

Bits (3:0) of word TBD1 indicate the size of device physical sectors in power of two logical sectors. Examples:

Bits (3:0): 0 = $2^0$ = 1 logical sector per physical sector
Bits (3:0): 1 = $2^1$ = 2 logical sectors per physical sector
Bits (3:0): 2 = $2^2$ = 4 logical sectors per physical sector
Bits (3:0:) 3 = $2^3$ = 8 logical sectors per physical sector

### 3.1.2 Word TBD2: Logical Sector Size

Bits (15:0) of word TBD2 indicate the size of device logical sectors in words.

## 3.2 Command Behavior for Non-Packet Devices with Logical Sector Length Other Than 256 Words

Table 1 describes the behavior of drives with logical sector length other than 256 words. Data transfer commands transfer either the logical sector length or 256 words depending on the command. For

example, Read and Write Extended commands transfer data in logical sectors while Read and Write Log Extended commands transfer 256 words per sector, regardless of the logical sector length

Note: Add, “Logical Block Length is 256 words” to the prerequisites for the commands marked command aborted in the table.

**Table 1 – Non-256 Word Logical Block Function**

| Command | Function | Words Per Sector Transferred |
|---|---|---|
| CFA ERASE SECTORS | Command aborted | - |
| CFA REQUEST EXTENDED ERROR CODE | Command aborted | - |
| CFA TRANSLATE SECTOR | Command aborted | - |
| CFA WRITE MULTIPLE WITHOUT ERASE | Command aborted | - |
| CFA WRITE SECTORS WITHOUT ERASE | Command aborted | - |
| CHECK MEDIA CARD TYPE | Command aborted | - |
| CHECK POWER MODE | Executable | - |
| CONFIGURE STREAM | Command aborted | - |
| DEVICE CONFIGURATION | Executable | 256 |
| DEVICE RESET | Executable | - |
| DOWNLOAD MICROCODE | Executable | 256 |
| EXECUTE DEVICE DIAGNOSTIC | Executable | - |
| FLUSH CACHE | Command aborted | - |
| FLUSH CACHE EXT | Executable | - |
| GET MEDIA STATUS | Executable | - |
| IDENTIFY DEVICE | Executable | 256 |
| IDENTIFY PACKET DEVICE | Command aborted | - |
| IDLE | Executable | - |
| IDLE IMMEDIATE | Executable | - |
| MEDIA EJECT | Executable | - |
| MEDIA LOCK | Executable | - |
| MEDIA UNLOCK | Executable | - |
| NOP | Executable | - |
| PACKET | Command aborted | - |
| READ BUFFER | Executable | 256 |
| READ DMA | Command aborted | - |
| READ DMA EXT | Executable | Identify Word TBD2 |
| READ DMA QUEUED | Command aborted | - |
| READ DMA QUEUED EXT | Executable | Identify Word TBD2 |
| READ LOG EXT | Executable | 256 |
| READ MULTIPLE | Command aborted | - |
| READ MULTIPLE EXT | Executable | Identify Word TBD2 |
| READ NATIVE MAX ADDRESS | Command aborted | - |
| READ NATIVE MAX ADDRESS EXT | Executable | - |
| READ SECTORS | Command aborted | - |
| READ SECTORS EXT | Executable | Identify Word TBD2 |
| READ STREAM DMA | Command aborted | - |
| READ STREAM PIO | Command aborted | - |
| READ VERIFY SECTORS | Command aborted | - |
| READ VERIFY SECTORS EXT | Executable | - |
| SECURITY DISABLE PASSWORD | Executable | 256 |
| SECURITY ERASE PREPARE | Executable | - |
| SECURITY ERASE UNIT | Executable | 256 |
| SECURITY FREEZE LOCK | Executable | - |
| SECURITY SET PASSWORD | Executable | 256 |
| SECURITY UNLOCK | Executable | 256 |
| SEEK | Command aborted | - |
| SERVICE | Executable | - |
| SET FEATURES | Executable | - |
| SET MAX ADDRESS | Command aborted | - |
| SET MAX ADDRESS EXT | Executable | - |
| SET MULTIPLE MODE | Executable | - |
| SLEEP | Executable | - |
| SMART DISABLE OPERATIONS | Executable | - |
| SMART ENABLE/DISABLE AUTOSAVE | Executable | - |

*(continued)*

**Table 0 – Security mode command actions** *(continued)*

| Command | Function | Words Per Sector Transferred |
|---|---|---|
| SMART ENABLE OPERATIONS | Executable | - |
| SMART EXECUTE OFF-LINE IMMEDIATE | Executable | - |
| SMART READ DATA | Executable | 256 |
| SMART READ LOG | Executable | 256 |
| SMART RETURN STATUS | Executable | - |
| SMART WRITE LOG | Executable | 256 |
| STANDBY | Executable | - |
| STANDBY IMMEDIATE | Executable | - |
| WRITE BUFFER | Executable | 256 |
| WRITE DMA | Command aborted | - |
| WRITE DMA EXT | Executable | Identify Word TBD2 |
| WRITE DMA QUEUED | Command aborted | - |
| WRITE DMA QUEUED EXT | Executable | Identify Word TBD2 |
| WRITE LOG EXT | Executable | 256 |
| WRITE MULTIPLE | Command aborted | - |
| WRITE MULTIPLE EXT | Executable | Identify Word TBD2 |
| WRITE SECTORS | Command aborted | - |
| WRITE SECTORS EXT | Executable | Identify Word TBD2 |
| WRITE STREAM DMA | Command aborted | - |
| WRITE STREAM PIO | Command aborted | - |

*(concluded)*

# 4 Appendix QQ2 Design and Programming Considerations for Large Physical Sector Devices and Non-Packet Devices with Logical Sectors Size other than 256 Words

## 4.1 Large Physical Sector Considerations

Since the inception of the ATA interface the smallest addressable unit of data has been the 512 byte sector. In hard disk drives each sector has an associated error correcting code field to allow detection and correction of read errors. Over time, error correcting code fields have been lengthened to provide greater detection and correction capability. As a result, the proportion of device media devoted to ECC fields has risen. Increasing the length of physical sectors increases the efficiency of ECC by enabling better error detection and correction using a smaller proportion of media. **Physical Sectors**

Because the 512 byte sector has been a constant since the beginning of ATA many software changes are be required when device logical sectors are made larger. To preserve the legacy software that assumes a 512 byte sector, logical addressing based on 512 byte sectors has been retained. Larger physical sectors are implemented as power of two multiples of logical sectors, 1,2,4,8,16, etc. For example, devices may have physical sectors that are 8 512 byte logical sectors long or 4096 bytes total.

### 4.1.2 Unaligned Write

While retaining the 512 byte logical sector maintains software compatibility it introduces a potential performance issue, unaligned write, which must be avoided. A physical sector must be written to the media in a single operation. To complete a write command that writes a fraction of a physical sector the device must read the entire physical sector into buffer memory update the buffer memory with the write data and then write the entire physical sector to the media. This will incur a performance penalty of at least a drive revolution.

Write commands can begin mid physical sector and end mid physical sector resulting in two unaligned writes. In this case the device has to read both the beginning and ending physical sector of the write into the buffer.

To avoid the performance penalty from an unaligned write all write operations must begin with the first sector of a physical sector and end with the last sector of a physical sector.

The first logical sector must begin at the first byte of the first physical sector on the device. This allows a host to align write operations with the physical sectors.

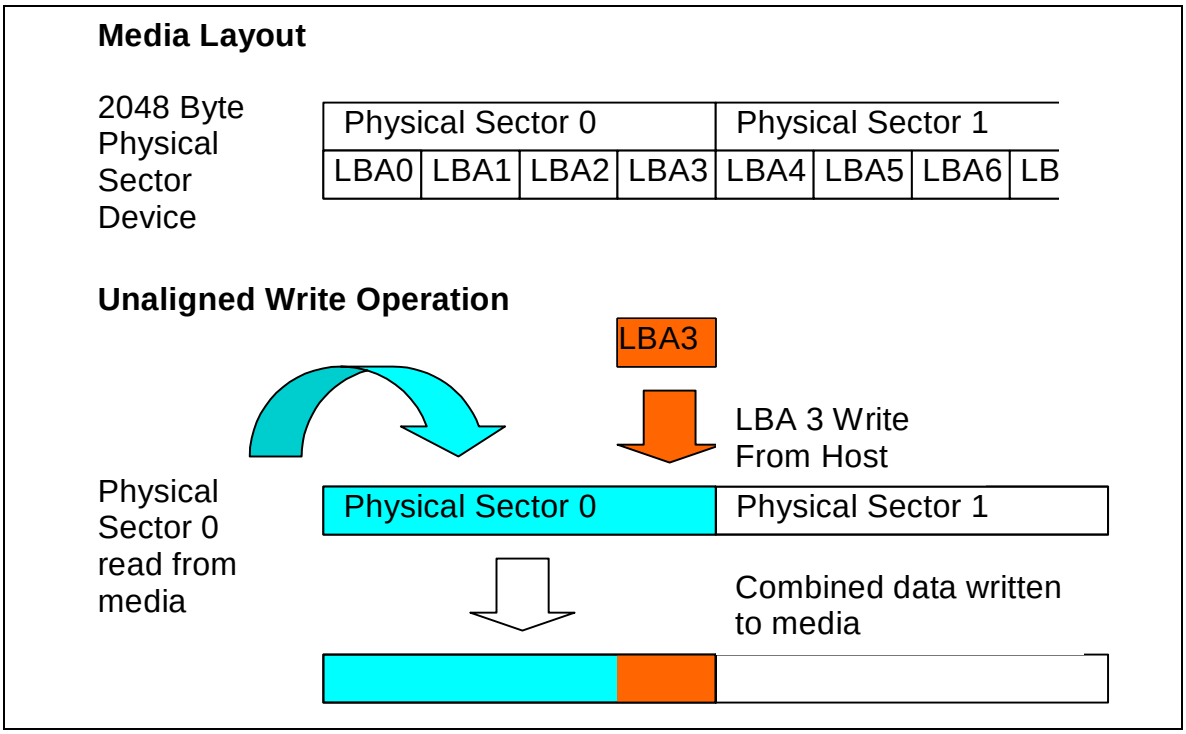


**Figure <QQ.1> – Unaligned Write Example**

Figure QQ.1 illustrates an unaligned write on a device with 2048 byte physical sectors. The First 4 logical sectors, LBA0 – LBA3, reside on physical sector 0. To write only LBA3 the host sends a conventional write command and the data for LBA3. On receipt of the write command the device seeks to the physical sector that contains LBA3, which is physical sector 0. Physical sector 0 is read into the device buffer. Then the new write data for LBA3 is placed in the buffer, overwriting a segment of the buffer. The buffer data is then written to the media, physical sector 0.

### 4.1.3 Set Max

Hosts which use the Set Max command should Set Max to the last logical sector of a physical sector to allow writes to the end of the user area without requiring an unaligned write. Devices should accommodate a Set Max setting to any LBA address to maintain compatibility.

### 4.1.4 Software Compatibility

The current specification allows devices to report up to $2^3$ or 8 logical sectors per physical sector. There are file system limitations in existing systems that restrict practical device implementations to 4096 bytes per physical sector. Operating system designers are advised to consider support for physical sectors longer than 4096 bytes in future designs.

## 4.2 Design and Programming Considerations for Non-Packet Devices With Logical Sector Length Other Than 256 Words

In RAID applications there may be a need for extra data bytes per sector for management of the array. The extra bytes are used inside the array and are not sent to the host system. For this purpose longer logical sectors are required, 520 bytes, 528 bytes, etc.

### 4.2.1 Backward Compatibility

Because the AT Attachment interface for non-packet devices has always used 512 byte fixed length sectors devices with longer or shorter logical sectors are not backward compatible with existing systems. Devices will fail due to an incorrect number of words being transferred during read and write commands. Since there is no backward compatibility to preserve many older commands are not supported on non-256 word logical sector devices. The supported commands are listed in Table 1.

It is recommended that hosts which use conventional 256 word logical sectors read the logical sector length from the Identify data and present an error message when a non 256 word logical sector length device is attached. Devices with non 256 word logical sectors are physically identical to conventional AT Attachment devices so there is a potential for incorrect installation.

### 4.2.2 Setting Logical Sector Size

The method for setting logical sector size is vendor specific.

Notes:

1. There is no mechanism for setting the logical sector size. I intend to leave this a vendor specific function. The current microcode download function is not generally used because manufacturers whish to keep control over the download process. I expect the process for changing drive logical sector length to be similar.
2. Do we want to allow non-256 word sectors for AV commands?