Date: Dec 21, 1999

To: T10 Committee (SCSI)

From: George Penokie (IBM)

Subject: Beyond 2TBytes

Overview

Subsystems connected to parallel SCSI and Fibre Channel are rapidly approaching sizes that will require SCSI commands that will address more than 2 TBytes of data. Many of the SCSI commands defined today for direct-access type SCSI devices are limited to addressing 2 TBytes when the block size is set to 512 bytes. Already some UNIX operating systems support an 8 byte address space, so where possible, this proposal will modify the LBA fields to 8 bytes.

This proposal will only address the direct-access type SCSI device command set.

Proposed changes

This proposal would make 16 byte commands out of any CDB that contains an LBA field. Those LBA fields would be make into 8 byte fields with the format of the CDB as shown in table 1.

.

Table 1 -Typical CDB for large LBA 16-byte commands

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE							
1	Reserved			MISC. CDB INFORMATION				
2	(MSB)	LOGICAL BLOCK ADDRESS						
9								(LSB)
10	(MSB)	TRANSFER LENGTH (if required) PARAMETER LIST LENGTH (if required) ALLOCATION LENGTH (if required)						
11								
12								
13								(LSB)
14	Reserved							
15	CONTROL							

The commands that would use the above format are listed in table 2.

Table 2 -CDB and parameter list changes

Command Name	**Op code**	**Type**	**Standard**	**Comment**
EXTENDED COPY	83h	O	SPC-2	Already in SPC-2
FORMAT UNIT	04h	M	SBC-2	CDB OK - Need new Defect List Format (see table 8)
LOCK-UNLOCK CACHE(16)***		O	SBC-2	Use format from table 1 for extended LBA and number of blocks.
PRE-FETCH(16)***		O	SBC-2	Use format from table 1 for extended LBA and transfer length.
READ(16)*		O	SBC-2	Use format from table 1 for extended LBA and transfer length.
READ CAPACITY(16)*		O	SBC-2	Use format from table 1 for extended LBA and use table 5 for the parameter data's extended LBA.
READ LONG	N/C	O	SBC-2	Diagnostic command - large LBA range not needed.
REASSIGN BLOCKS	07h	O	SBC-2	CDB OK - New new option for defect list to add in 8-byte LBAs (see table 8).
REBUILD(long)	N/C	O	SBC-2	No proposed change. A set of variable length CDBs should be defined for XOR commands.
REGENERATE (long)	N/C	O	SBC-2	No proposed change. A set of variable length CDBs should be defined for XOR commands.
SET LIMITS(16)	N/C	O	SBC-2	No proposed change. No used.
SYNCHRONIZE CACHE(16)***		O	SBC-2	Use format from table 1 for extended LBA and number of blocks.
VERIFY(16)**		O	SBC-2	Use format from table 1 for extended LBA and verification length.
WRITE(16)*		O	SBC-2	Use format from table 1 for extended LBA and transfer length.
WRITE AND VERIFY(16)*		O	SBC-2	Use format from table 1 for extended LBA and transfer length.
WRITE LONG	N/C	O	SBC-2	Diagnostic command - large LBA range not needed.
WRITE SAME(16)**		O	SBC-2	Use format from table 1 for extended LBA and number of blocks.
XDREAD(long)	N/C	O	SBC-2	No proposed change. A set of variable length CDBs should be defined for XOR commands.
XDWRITE(long)	N/C	O	SBC-2	No proposed change. A set of variable length CDBs should be defined for XOR commands.
XDWRITE EXTENDED(long)	N/C	O	SBC-2	No proposed change. A set of variable length CDBs should be defined for XOR commands.
XPWRITE(long)	N/C	O	SBC-2	No proposed change. A set of variable length CDBs should be defined for XOR commands.

Mode page header changes

In addition to the commands and parameters listed above the mode page header is another area where the LBA for direct-access SCSI device has only an eight byte field. To allow 8-byte LBAs to be used a bit must be added to the mode page header (10) to indicate the mode parameter block descriptor length is 16. Then a new mode block descriptor would be defined that could handle 8-byte LBAs. The following changes would be required:

.

Table 3 -Mode parameter header (10)

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)	MODE DATA LENGTH						
1								(LSB)
2	MEDIUM TYPE							
3	DEVICE-SPECIFIC PARAMETER							
4	Reserved							LONGLBA
5	Reserved							
6	(MSB)	BLOCK DESCRIPTOR LENGTH						
7								(LSB)

<u>The long lba (LONGLBA) bit of zero indicates the mode parameter block descriptors are eight bytes long. A LONGLBA bit of one indicates the mode parameter block descriptors are 16 bytes long.</u>

The BLOCK DESCRIPTOR LENGTH field specifies the length in bytes of all the block descriptors. It is equal to the number of block descriptors times eight<u> or 16</u>, and does not include pages or vendor-specific parameters, if any, that may follow the last block descriptor. A block descriptor length of zero indicates that no block descriptors are included in the mode parameter list. This condition shall not be considered an error.

<u>The mode parameter block descriptor format for all device types use long LBAs is shown in table 4.</u>

.

Table 4 -Long mode parameter block descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)	NUMBER OF BLOCKS						
7								(LSB)
8	DENSITY CODE							
9	Reserved							
10	Reserved							
11	Reserved							
12	(MSB)	BLOCK DESCRIPTOR LENGTH						
15								(LSB)

<u>Block descriptors specify some of the medium characteristics for all or part of a logical unit. Support for block descriptors is optional. Each block descriptor contains a DENSITY CODE field, a NUMBER OF BLOCKS field, and</u>

a BLOCK LENGTH field. Block descriptor values are always current (i.e., saving is not supported). A unit attention condition (see x.x and SAM-2) shall be generated when any block descriptor values are changed.

The NUMBER OF BLOCKS field specifies the number of logical blocks on the medium to which the DENSITY CODE and BLOCK LENGTH fields apply. A value of zero indicates that all of the remaining logical blocks of the logical unit shall have the medium characteristics specified.

If the SCSI device doesn't support changing its capacity by changing the NUMBER OF BLOCKS field (via a MODE SELECT command), the value in the NUMBER OF BLOCKS field is ignored. If the device supports changing its capacity by changing the NUMBER OF BLOCKS field, then the NUMBER OF BLOCKS field is interpreted as follows:

a) If the number of blocks is set to zero, the device shall retain its current capacity if the block size has not changed. If the number of blocks is set to zero and the block size has changed, the device shall be set to its maximum capacity when the new block size takes effect;
b) If the number of blocks is greater than zero and less than or equal to its maximum capacity, the device shall be set to that number of blocks. If the block size has not changed, the device shall not become format corrupted. This capacity setting shall be retained through reset events or power cycles.
c) If the number of blocks field is set to a value greater than the maximum capacity of the device and less than FFFFFFFFFFFFFFFFh, then the command is terminated with a CHECK CONDITION status. The sense key is set to ILLEGAL REQUEST. The device shall retain its previous block descriptor settings;
d) If the number of blocks is set to FFFFFFFFFFFFFFFFh, the device shall be set to its maximum capacity. If the block size has not changed, the device shall not become format corrupted. This capacity setting shall be retained through reset events or power cycles.

Note 1 - There may be implicit association between parameters defined in the pages and block descriptor. For direct-access devices, the block length affects the optimum values (the value that achieves the best performance) for the sectors per track, bytes per physical sector, track skew factor, and cylinder skew factor fields in the format parameters page. In this case, the target may change parameters not explicitly sent with the MODE SELECT command. A subsequent MODE SENSE command may be used to detect these changes.

The DENSITY CODE field is unique for each device type. Refer to the mode parameters clause of the specific device type command standard (see x.x.x) for definition of this field. Some device types reserve all or part of this field.

The BLOCK LENGTH field specifies the length in bytes of each logical block described by the block descriptor.

Additional Read Capacity changes

The following statement needs to be added to the current Read Capacity command to cover the case were a target receives a Read Capacity command and the values that would be returned are too large to fit in the RETURNED LOGICAL BLOCK ADDRESS field of the Read Capacity data parameter list.

If the number of logical blocks exceeds the maximum value that may be specified in the RETURNED LOGICAL BLOCK ADDRESS field the device server shall transfer no data and return a CHECK CONDITION status and the sense key shall be set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN CDB.

The format for the Read Capacity (16) data parameter list is:

Table 5 -Read Capacity data

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)	RETURNED LOGICAL BLOCK ADDRESS						
7								(LSB)
8	(MSB)	block length in bytes						
11								(LSB)

Additional FORMAT UNIT defect descriptor

An additional Format unit defect descriptor will have to be added to allow returning a block format defect desiccator that can return the larger LBAs. The following additions will be needed to the FORMAT UNIT defect descriptor format and requirements table.

Table 6 -FORMAT UNIT defect descriptor format and requirements

FMTDATA	CMPLST	Defect List Format	Defect List Length	Type	Comments
Block Formats					
1	0	011b	>0	O	See notes (2) and (3)
1	1	011b	>0	O	See notes (2) and (4)

The following FORMAT UNIT text would be added:

Each block format defect descriptor format specified as 000b (see table 7) specifies a four-byte defective block address that contains the defect. Each block format defect descriptor format specified as 011b (see table 8) specifies an eight-byte defective block address that contains the defect. Use of the Block format is vendor-specific.

Table 7 -DEFECT DESCRIPTOR - Block format (000b)

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)	DEFECTIVE BLOCK ADDRESS						
3								(LSB)

Table 8 -DEFECT DESCRIPTOR - Block format (011b)

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)	DEFECTIVE BLOCK ADDRESS						
7								(LSB)

Commands allowed in the presence of various reservations

SBC commands

This clause should be placed into the model clause of the next version of the SBC standard when, and if, a new version of that standard is published. It should replace all the individual command descriptions of how reservations work.

Reservation restrictions are placed on commands as a result of access qualifiers associated with the type of reservation. The details of which commands are allowed under what types of reservations are described in tables 9, 10 and 11. For the reservation restrictions placed on commands for the Reserve/Release management method see tables 9, 10 and 11 column [A]. For the reservation restrictions placed on commands for the Persistent Reservations management method, see the columns under [B] in tables 9, 10 and 11.

In tables 9, 10 and 11 the following key words are used:

allowed: Commands issued by initiators not holding the reservation or by initiators not registered when a registrants only persistent reservation is present should complete normally.

conflict: Commands issued by initiators not holding the reservation or by initiators not registered when a registrants only persistent reservation is present shall not be performed and the device server shall terminate the command with a RESERVATION CONFLICT status.

Commands from initiators holding a reservation should complete normally. The behavior of commands from registered initiators when a registrants only persistent reservation is present is specified in tables 9, 10 and 11.

A command that does not explicitly write the medium shall be checked for reservation conflicts before the command enters the current task state for the first time. Once the command has entered the current task state, it shall not be terminated with a RESERVATION CONFLICT due to a subsequent reservation.

A command that explicitly writes the medium shall be checked for reservation conflicts before the device server modifies the medium or cache as a result of the command. Once the command has modified the medium, it shall not be terminated with a RESERVATION CONFLICT due to a subsequent reservation.

For each command, this standard, SPC-2, or a related command standard defines the conditions that result in RESERVATION CONFLICT. Depending on the particular command standard the conditions are defined in that standard’s device model clause or in the clauses that define the specific commands. An annex in SPC-2 contains the RESERVATION CONFLICT information for some of the command sets.

Table 9 -SBC direct access commands that are allowed in the presence of various reservations

Command	Addressed LU is reserved by another initiator [A]	Addressed LU has this type of persistent reservation held by another initiator [B]				
		From any initiator		From registered initiator (RO all types)	From initiator not registered	
		Write Excl	Excl Access		Write Excl – RO	Excl Access – RO
LOCK/UNLOCK CACHE(16)	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
PRE-FETCH(16)	Conflict	Allowed	Conflict	Allowed	Allowed	Conflict
READ(6)/READ(10)/READ(12)/ READ(16)	Conflict	Allowed	Conflict	Allowed	Allowed	Conflict
READ CAPACITY(16)	Allowed	Allowed	Allowed	Allowed	Allowed	Allowed
SYNCHRONIZE CACHE(16)	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
VERIFY(16)	Conflict	Allowed	Conflict	Allowed	Allowed	Conflict
WRITE(6)/WRITE(10)/WRITE(12)/ WRITE(16)	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
WRITE AND VERIFY(16)	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
WRITE SAME(16)	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
Key: LU=Logical Unit, Excl=Exclusive, RO=Registrants Only, <> Not Equal						

Table 10 -SBC optical memory commands that are allowed in the presence of various reservations

Command	Addressed LU is reserved by another initiator [A]	Addressed LU has this type of persistent reservation held by another initiator [B]				
		From any initiator		From registered initiator (RO all types)	From initiator not registered	
		Write Excl	Excl Access		Write Excl – RO	Excl Access – RO
LOCK/UNLOCK CACHE(16)	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
PRE-FETCH(16)	Conflict	Allowed	Conflict	Allowed	Allowed	Conflict
READ(6)/READ(10)/READ(12)/ READ(16)	Conflict	Allowed	Conflict	Allowed	Allowed	Conflict
READ CAPACITY(16)	Allowed	Allowed	Allowed	Allowed	Allowed	Allowed
READ LONG(16)	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
Key: LU=Logical Unit, Excl=Exclusive, RO=Registrants Only, <> Not Equal						

Table 10 -SBC optical memory commands that are allowed in the presence of various reservations

Command	Addressed LU is reserved by another initiator [A]	Addressed LU has this type of persistent reservation held by another initiator [B]				
		From any initiator		From registered initiator (RO all types)	From initiator not registered	
		Write Excl	Excl Access		Write Excl – RO	Excl Access – RO
SYNCHRONIZE CACHE(16)	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
VERIFY(10)/VERIFY(12)/ VERIFY(16)	Conflict	Allowed	Conflict	Allowed	Allowed	Conflict
WRITE(6)/WRITE(10)/WRITE(12)/ WRITE(16)	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
WRITE AND VERIFY(10)/ WRITE AND VERIFY(12)/ WRITE AND VERIFY(16)	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
WRITE LONG(16)	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
Key: LU=Logical Unit, Excl=Exclusive, RO=Registrants Only, <> Not Equal						

Table 11 -SBC write-once commands that are allowed in the presence of various reservations

Command	Addressed LU is reserved by another initiator [A]	Addressed LU has this type of persistent reservation held by another initiator [B]				
		From any initiator		From registered initiator (RO all types)	From initiator not registered	
		Write Excl	Excl Access		Write Excl – RO	Excl Access – RO
LOCK/UNLOCK CACHE(16)	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
PRE-FETCH(16)	Conflict	Allowed	Conflict	Allowed	Allowed	Conflict
READ(6)/READ(10)/READ(12)/ READ(16)	Conflict	Allowed	Conflict	Allowed	Allowed	Conflict
READ CAPACITY(16)	Allowed	Allowed	Allowed	Allowed	Allowed	Allowed
READ LONG(16)	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
SYNCHRONIZE CACHE(16)	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
VERIFY(10)/VERIFY(12)/ VERIFY(16)	Conflict	Allowed	Conflict	Allowed	Allowed	Conflict
WRITE(6)/WRITE(10)/WRITE(12)/ WRITE(16)	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
WRITE AND VERIFY(10)/ WRITE AND VERIFY(12)/ WRITE AND VERIFY(16)	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
WRITE LONG(16)	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
Key: LU=Logical Unit, Excl=Exclusive, RO=Registrants Only, <> Not Equal						