To: T10 Technical Committee
From: Rob Elliott, HP (elliott@hp.com)
Date: 12 November 2007
Subject: 07-451r1 SBC-3 SPC-4 WRITE LONG COR_DIS and WR_UNCOR interaction

Revision history
Revision 0 (24 October 2007) First revision
Revision 1 (12 November 2007) Incorporated comments from November 2007 CAP WG

Related documents
sbc3r11 - SCSI Block Commands - 3 (SBC-3) revision 11
spc4r11 - SCSI Primary Commands - 4 (SPC-4) revision 11
05-374r2 - SBC-3 SPC-4 Disabling reassign on WRITE LONG logical blocks (George Penokie, IBM) - incorporated into sbc3r03 (November 2005)
06-034r5 - SBC-3 Physical blocks (Rob Elliott, HP) - incorporated into sbc3r07 (September 2006)
07-447 - SBC-3 Read-Write Error Recovery page clarifications (Rob Elliott, HP)

Overview
As currently defined, COR_DIS=1 has the same behavior as WR_UNCOR=1 - both force the specified logical block to report an unrecoverable error on read commands, regardless of whether the other bits in the CDB and the provided write data (if any) cause an unrecoverable error. There should not be two bits with the same meaning.

One way to make them coexist better would be to preface the description of returning CHECK CONDITION status with “if an unrecoverable error occurs, ”. COR_DIS=1 would just disable error recovery, reallocation, and reporting if an error does occur, but not force an error itself.

The author of the COR_DIS proposal prefers to have it continue to force an error, though. So, the next distinction that can be applies is that COR_DIS=1 still allows the WRITE LONG command to transfer data based on the BYTE TRANSFER LENGTH field, while WR_UNCOR=1 ignores the BYTE TRANSFER LENGTH field and does not perform any data transfer. Since the BYTE TRANSFER LENGTH field can be set to 0, the write data may or may not exist. If it does exist, it may result in valid data, data with a recoverable error, or data with an unrecoverable error being placed on the medium.

The interaction of COR_DIS and PBLOCK is also unclear. The table defining WR_UNCOR and PBLOCK is expanded to include COR_DIS so all combinations are clearly defined.

A rule is added that the self-test defined in SPC-4 and the background scan defined in SBC-3 ignores pseudo unrecovered errors with correction disabled and honor them with correction enabled.

A WU_SUP bit is added to the Extended INQUIRY Data VPD page to indicate WR_UNCOR=1 is supported (and implicitly that the logical unit is implementing both COR_DIS and WR_UNCOR per this proposal). The COR_D_SUP bit is rename CD_SUP to match the shortening convention of WU_SUP (the SPC-4 editor indicated COR_DIS_SUP and WR_UNCOR_DIS won’t fit, so matching the SBC field name is not viable).

Suggested changes to SBC-3 (as modified by 07-447)

3.1.xx pseudo read data: Data that is transferred by the device server based on the setting of the RC bit or the TB bit in the Read-Write Error mode page (see 6.3.5) in order to maintain a continuous flow of data or to transfer the amount of data requested for a command even though an unrecovered read error (see 3.1.xx) occurred while the device server was processing the command. Psuedo read data may be erroneous or fabricated by the device server (e.g., data already in a buffer or any other vendor-specific data).

3.1.xx pseudo unrecovered error: A simulated error (e.g., created by the WRITE LONG command) for which the device server reports that it is unable to read or write a logical block, regardless of whether the data on the medium is valid, recoverable, or unrecoverable.

3.1.xx pseudo unrecovered error with correction enabled: A pseudo unrecovered error (see 3.1.xx) for which the device server performs the maximum error recovery as specified in the Read-Write Error Recovery mode page (see 6.3.5). See 4.14.2.

3.1.xx pseudo unrecovered error with correction disabled: A pseudo unrecovered error (see 3.1.xx) for which the device server performs no error recovery. See 4.14.2.

3.1.xx unrecovered error: An error in which the device server is unable to read or write a logical block within the recovery limits specified in the Read-Write Error Recovery mode page (see 6.3.5) and the Verify Error Recovery mode page (see 6.3.6).

3.1.xx recovered error: An error in which the device server is able to read or write a logical block within the recovery limits specified in the Read-Write Error Recovery mode page (see 6.3.5) and the Verify Error Recovery mode page (see 6.3.6).

4.5 Physical blocks

A physical block is a set of data bytes on the medium accessed by the device server as a unit. A physical block may contain:

a) a portion of a logical block (i.e., there are multiple physical blocks in the logical block)(e.g., a physical block length of 512 bytes with a logical block length of 2 048 bytes);
b) a single complete logical block; or
c) more than one logical block (i.e., there are multiple logical blocks in the physical block)(e.g., a physical block length of 4 096 bytes with a logical block length of 512 bytes).

Each physical block includes additional information not normally accessible to the application client (e.g., an ECC) that the device server uses to manage storage and retrieval.

If the device server supports the COR_DIS bit and/or the WR_UNCOR bit in the WRITE LONG command (see 5.35 and 5.36), the device server shall have the capability of marking individual logical blocks as containing pseudo uncorrectable errors with correction enabled or with correction disabled.

...

4.9 Medium defects

Any medium has the potential for defects that cause data to be lost. Therefore, each logical block may contain additional information that allows the detection of changes to the user data and protection information, if any, caused by defects in the medium or other phenomena, and may also allow the data to be reconstructed following the detection of such a change (e.g., ECC bytes).

A defect causes a recoverable error if the device server is able to retrieve the data by retrying or using the additional information to reconstruct the data. A defect causes an unrecoverable error if the device server is unable to retrieve the data.

Direct-access block devices may allow the application client to examine and modify the additional information by using the READ LONG commands and the WRITE LONG commands (see 5.16, 5.17, 5.35, and 5.36). The application client may use the WRITE LONG commands to ~~induce a defect~~ alter the additional information to test the defect detection logic of the direct-access block device or to emulate an unrecoverable logical block when generating a mirror copy. This may induce a recoverable error or an unrecoverable error.

Direct-access block devices may allow the application client to ~~disable error correction and automatic reallocation on specific logical blocks by using the WRITE LONG command (see 5.35 and 5.36)~~ use the features of the WRITE LONG commands (see 5.35 and 5.36) to:

a) disable error correction on specific logical blocks or physical blocks;
b) disable automatic reallocation on specific logical blocks or physical blocks; and
c) mark specific logical blocks or physical blocks as containing pseudo unrecoverable errors with correction enabled or with correction disabled;

~~This allows~~ These features provide methods for an application client to prevent logical blocks from being ~~included in the algorithm used for~~reported as information exception conditions and~~, there by, preventing unwarranted information exception condition trips and unnecessary reassignments~~.

During a self-test operation (see SPC-4), the device server shall ignore pseudo unrecoverable errors with correction disabled and shall honor pseudo unrecoverable errors with correction disabled. See 4.19.1 for rules on background scans.

Defects may also be detected and managed during processing of the FORMAT UNIT command (see 5.2). The FORMAT UNIT command defines four sources of defect information: the PLIST, CLIST, DLIST, and GLIST.

These defects may be reassigned or avoided during the initialization process so that they do not affect any logical blocks. The sources of defect location information (i.e., defects) are defined as follows:

a) Primary defect list (PLIST). This is the list of defects, which may be supplied by the original manufacturer of the device or medium, that are considered permanent defects. The PLIST is located outside of the application client accessible logical block space. The PLIST is accessible by the device server for reference during the format operation, but it is not accessible by the application client except through the READ DEFECT DATA commands (see 5.12 and 5.15). Once created, the original PLIST shall not change;
b) Logical unit certification list (CLIST). This list includes defects detected by the device server during an optional certification process performed during the FORMAT UNIT command. This list shall be added to the GLIST;
c) Data defect list (DLIST). This list of defects may be supplied by the application client to the device server during the FORMAT UNIT command. This list shall be added to the GLIST; and
d) Grown defect list (GLIST). The GLIST includes all defects sent by the application client (i.e., the DLIST) or detected by the device server (i.e., the CLIST). The GLIST does not include the PLIST. If the CMPLST bit is set to zero, the GLIST shall include DLISTs provided to the device server during the previous and the current FORMAT UNIT commands. The GLIST shall also include:
    A) defects detected by the format operation during medium certification;
    B) defects previously identified with a REASSIGN BLOCKS command (see 5.18); and
    C) defects previously detected by the device server and automatically reallocated.

The direct-access block device may automatically reassign defects if allowed by the Read-Write Error Recovery mode page (see 6.3.5).

Defects may also occur after initialization. The application client issues a REASSIGN BLOCKS command (see 5.18) to request that the specified LBA be reassigned to a different part of the medium. This operation may be repeated if a new defect appears at a later time. The total number of defects that may be handled in this manner is vendor-specific.

Defect management on direct-access block devices is vendor-specific. Direct-access block devices not using a removable medium may optimize the defect management for capacity or performance or both. Some direct-access block devices that use a removable medium do not support defect management or use defect management that does not impede the ability to interchange the medium.

4.11 Caches

...

During write operations, the device server uses the cache to store data that is to be written to the medium at a later time. This is called write-back caching. The command may complete prior to logical blocks being written to the medium. As a result of using a write-back caching there is a period of time when the data may be lost if power to the SCSI target device is lost and a volatile cache is being used or a hardware failure occurs. There is also the possibility of an error occurring during the subsequent write operation. If an error occurred during the write operation, it may be reported as a deferred error on a later command. The application client may request that write-back caching be disabled with the Caching mode page (see 6.3.4) to prevent detected write errors from being reported as deferred errors. Even with write-back caching disabled, undetected write errors may occur. The VERIFY commands and the WRITE AND VERIFY commands may be used to detect those errors.

...

Commands may be implemented by the device server that allow the application client to control other behavior of the cache:

a) the PRE-FETCH commands (see 5.4 and 5.5) cause a set of logical blocks requested by the application client to be read into cache for possible future access. The logical blocks fetched are subject to later replacement;

b) the SYNCHRONIZE CACHE commands (see 5.20 and 5.21) force any write data in cache in the requested set of logical blocks to be written to the medium. These commands may be used to ensure that the data is written and any detected errors reported;
c) the Caching mode page (see 6.3.4), writable by the MODE SELECT commands, allows control of cache behavior.

4.14 Error reporting

4.14.1 Error reporting overview

If any of the conditions listed in table 4 occur during the processing of a command, the command shall be terminated with CHECK CONDITION status with the sense key set to the specified value and the additional sense code set to the appropriate value for the condition. Some errors may occur after the completion status has already been reported. For such errors, SPC-4 defines a deferred error reporting mechanism. Table 4 lists some error conditions and the applicable sense keys. The list does not provide an exhaustive enumeration of all conditions that may cause CHECK CONDITION status.

Table 1 — Example error conditions

Condition	Sense key
Invalid LBA	ILLEGAL REQUEST
Unsupported option requested	ILLEGAL REQUEST
Logical unit reset, I_T nexus loss, or medium change since last command from this application client	UNIT ATTENTION
Self diagnostic failed	HARDWARE ERROR
Unrecovered ~~read~~ error	MEDIUM ERROR or HARDWARE ERROR
Recovered read error	RECOVERED ERROR
Pseudo unrecovered error	MEDIUM ERROR
Over-run or other error that might be resolved by repeating the command	ABORTED COMMAND
Attempt to write on write-protected medium	DATA PROTECT

...

When a recovered read error is reported, the information field of the sense data shall contain the LBA of the last logical block accessed by the command on which a recovered read error occurred.

When an unrecovered error is reported, the information field of the sense data shall contain the LBA of the logical block on which the unrecovered error occurred.

4.14.2 Handling pseudo unrecovered errors

If a pseudo unrecovered error is encountered on a logical block with correction disabled (e.g., by a command, a background scan, or a background self-test), the device server shall:

a) perform no error recovery on the affected logical blocks, including any read error recovery enabled by the Read-Write Error Recovery mode page (see 6.3.5) or the Verify Error Recovery mode page (see 6.3.6);
b) perform no automatic reallocation of the affected logical blocks, including any automatic reallocation enabled by the Read-Write Error Recovery mode page;
c) not consider errors on the affected logical blocks to be informational exception conditions as defined in the Information Exceptions Control mode page (see SPC-4); and
d) not log errors on the affected logical blocks in the Error Counter log pages (see SPC-4);
e) in any information returned for the error (e.g., in sense data, the Background Scan Results log page (see 6.2.2)), set the sense key set to MEDIUM ERROR and the additional sense code to either:
   A) READ ERROR – LBA MARKED BAD BY APPLICATION CLIENT; or

B) UNRECOVERABLE READ ERROR.

The additional sense code should be set to READ ERROR – LBA MARKED BAD BY APPLICATION CLIENT.

Editor's Note 1: SAT implementations will not be able to completely honor a) above; ATA8-ACS only recommends that the ATA device perform no error recovery, not requires that it perform no error recovery.

If a pseudo unrecovered error is encountered on a logical block with correction enabled (e.g., by a command, a background scan, or a background self-test), the device server shall:

a) if enabled by the Read-Write Error Recovery mode page (see 6.3.5) or the Verify Error Recovery mode page (see 6.3.6), perform error recovery on the affected logical blocks;
b) perform no automatic reallocation of the affected logical blocks, including any automatic reallocation enabled by the Read-Write Error Recovery mode page;
c) consider errors on the affected logical blocks to be informational exception conditions as defined in the Information Exceptions Control mode page (see SPC-4); and
d) log errors on the affected logical blocks in the Error Counter log pages (see SPC-4);
e) in any information returned for the error (e.g., in sense data, the Background Scan Results log page (see 6.2.2)), set the sense key set to MEDIUM ERROR and the additional sense code to either:
   A) READ ERROR – LBA MARKED BAD BY APPLICATION CLIENT; or
   B) UNRECOVERABLE READ ERROR.

   The additional sense code should be set to READ ERROR – LBA MARKED BAD BY APPLICATION CLIENT.

4.14.~~2~~3 Block commands sense data descriptor

...

4.19 Background scanning operations

4.19.1 Background scanning overview

During background scanning, the device server, without using any bandwidth on a service delivery subsystem, reads logical blocks from the medium for the purpose of:

a) identifying logical blocks that are difficult to read or unreadable;
b) logging read problems; and
c) when allowed, taking a vendor-specific action to make the logical block readable again.

If a logical block is readable but requires extra actions (e.g., retries or application of a correction algorithm) to be read (i.e., the data is recoverable), the device server may resolve the problem using vendor-specific means. The ARRE bit in the Read-Write Error Recovery mode page (see 6.3.5) controls whether the device server ~~may automatically repair or relocaterecoverable read errors~~ performs automatic reallocation of defective logical blocks during read operations.

If a logical block is unreadable (i.e., the data is unrecoverable) the device server may mark the logical block as bad so it may be relocated. The AWRE bit in the Read-Write Error Recovery mode page (see 6.3.5) controls whether the device server ~~may relocate~~ performs automatic reallocation of defective logical blocks during write operations. If allowed by the AWRE bit setting, logical blocks that have previously been noted as unrecoverable are reassigned at the start of the next write operation to that logical block.

During a background scan, the device server may scan the logical blocks in any order (e.g., based on physical block layout). The device server should not retain any logical blocks in cache memory after they are read.

During a background scan, the device server shall ignore pseudo unrecoverable errors with correction disabled, and shall honor pseudo unrecoverable errors with correction enabled.

5.35 WRITE LONG (10) command

The WRITE LONG (10) command (see table 83) requests that the device server mark a logical block or physical block as containing an error, or transfer data for a single logical block or physical block from the data-out buffer and write it to the medium. The data written shall be the same length and shall be in the same order as the data returned by the READ LONG (10) command (see 5.16). The device server shall write the logical block or physical block to the medium, and shall not return GOOD status until the logical block has actually been written on the medium.

[Table 83 - WRITE LONG (10) command]

The OPERATION CODE field is defined in SPC-4 shall be set to the value defined in table 83.

~~A correction disabled (COR_DIS) bit set to zero specifies that, when the specified logical block is affected logical blocks are read, the device server shall perform normal error recovery on that logical block. A COR_DIS bit set to one specifies that, when the specified logical block is affected logical blocks are read (e.g., with a READ command), the device server shall:~~

~~d) perform no error recovery on that logical block~~the affected logical blocks ~~including any read error recovery enabled by the Read-Write Error Recovery mode page (see 6.3.5);~~

~~e) perform no automatic reallocation of that logical block~~the affected logical blocks ~~including any automatic reallocation enabled by the Read-Write Error Recovery mode page; and~~

~~f) not consider errors on logical blocks~~the affected logical blocks ~~to be informational exception conditions as defined in the Information Exceptions Control mode page (see SPC-4): and~~

~~g) if an unrecoverable error occurs, return CHECK CONDITION status with the sense key set to MEDIUM ERROR and the additional sense code set to either:~~

~~A) READ ERROR – LBA MARKED BAD BY APPLICATION CLIENT; or~~

~~B) UNRECOVERABLE READ ERROR.~~

~~The additional sense code should be set to READ ERROR – LBA MARKED BAD BY APPLICATION CLIENT.~~

~~The For each affected logical block, the condition established by the COR_DIS bit being set to one shall remain in effect until the logical block is written by any means (e.g., any WRITE command, WRITE SAME command, FORMAT command, or another WRITE LONG command specifying the same logical block with the COR_DIS bit set to zero).~~

The correction disable (COR_DIS) bit, the write uncorrectable error (WR_UNCOR) bit, and the physical block (PBLOCK) bit are defined in table 84. If there are more than one logical block per physical block (i.e., the

LOGICAL BLOCKS PER PHYSICAL BLOCK EXPONENT field in the READ CAPACITY (16) data (see 5.13.1) is set to a non-zero value), the device server shall support the WR_UNCOR bit and the PBLOCK bit.

~~**Table 84 — WR_UNCOR bit and PBLOCK bit**~~

~~WR_UNCOR~~	~~PBLOCK~~	~~More than one logical block per physical block~~	~~Description~~
~~0~~	~~0~~	~~yes [a] or no~~	~~Write only the specified logical block~~
~~0~~	~~1~~	~~yes [a]~~	~~Write the entire physical block containing the specified logical block~~
~~1~~	~~0~~	~~no~~	~~Return CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN CDB~~
~~1~~	~~0~~	~~yes [a] or no~~	~~Mark only the specified logical block as containing an uncorrectable error, ignore the byte transfer length field, and transfer no data~~
~~1~~	~~1~~	~~yes [a]~~	~~Mark the entire physical block containing the specified logical block as containing an uncorrectable error (i.e., mark all the logical blocks in the same physical block as the specified logical block as containing an uncorrectable error), ignore the byte transfer length field, and transfer no data~~
~~1~~	~~1~~	~~no~~	~~Return CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN CDB~~

~~[a] The LOGICAL BLOCKS PER PHYSICAL BLOCK EXPONENT field in the READ CAPACITY (16) data (see 5.13.1) is set to a non-zero value~~

Replacement table:

Table 84 — WR_UNCOR bit and PBLOCK bit (part 1 of 3)

COR_DIS	WR_UNCOR	PBLOCK	More than one logical block per physical block [a]	Description
0	0	0	yes or no	Honor the BYTE TRANSFER LENGTH field and write only the specified logical block.
		1	no	Return CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN CDB
			yes	Honor the BYTE TRANSFER LENGTH field and write the entire physical block containing the specified logical block.

[a] An entry of “yes” means the LOGICAL BLOCKS PER PHYSICAL BLOCK EXPONENT field in the READ CAPACITY (16) data (see 5.13.1) is set to a non-zero value. An entry of “no” means it is set to zero.

Table 84 — WR_UNCOR bit and PBLOCK bit (part 2 of 3)

COR_DIS	WR_UNCOR	PBLOCK	More than one logical block per physical block [a]	Description
0	1	0	yes or no	Mark only the specified logical block as containing a pseudo unrecovered error with correction enabled (see 4.14.2) in a manner that causes the device server to perform the maximum error recovery as specified in the Read-Write Error Recovery mode page (see 6.3.5). Ignore the BYTE TRANSFER LENGTH field and transfer no data.
		1	no	Return CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN CDB
			yes	Mark the entire physical block containing the specified logical block as containing a pseudo unrecovered error (i.e., mark all the logical blocks in the same physical block as the specified logical block as containing a pseudo unrecovered error) with correction enabled (see 4.14.2) in a manner that causes the device server to perform the maximum error recovery as specified in the Read-Write Error Recovery mode page (see 6.3.5). Ignore the BYTE TRANSFER LENGTH field and transfer no data.
1	0	0	yes or no	Mark only the specified logical block as containing a pseudo unrecovered error with correction disabled (see 4.14.2). Honor the BYTE TRANSFER LENGTH field and write only the specified logical block.
		1	no	Return CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN CDB
			yes	Mark the entire physical block containing the specified logical block as containing a pseudo unrecovered error (i.e., mark all the logical blocks in the same physical block as the specified logical block as containing a pseudo unrecovered error) with correction disabled (see 4.14.2). Honor the BYTE TRANSFER LENGTH field and write the specified physical block.

[a] An entry of “yes” means the LOGICAL BLOCKS PER PHYSICAL BLOCK EXPONENT field in the READ CAPACITY (16) data (see 5.13.1) is set to a non-zero value. An entry of “no” means it is set to zero.

Table 84 — WR_UNCOR bit and PBLOCK bit (part 3 of 3)

COR_DIS	WR_UNCOR	PBLOCK	More than one logical block per physical block [a]	Description
1	1	0	yes or no	Mark only the specified logical block as containing a pseudo unrecovered error with correction disabled (see 4.14.2). Ignore the BYTE TRANSFER LENGTH field and transfer no data.
		1	no	Return CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN CDB
			yes	Mark the entire physical block containing the specified logical block as containing a pseudo unrecovered error (i.e., mark all the logical blocks in the same physical block as the specified logical block as containing a pseudo unrecovered error) with correction disabled (see 4.14.2). Ignore the BYTE TRANSFER LENGTH field and transfer no data.

[a] An entry of “yes” means the LOGICAL BLOCKS PER PHYSICAL BLOCK EXPONENT field in the READ CAPACITY (16) data (see 5.13.1) is set to a non-zero value. An entry of “no” means it is set to zero.

Any pseudo unrecovered error with correction disabled shall remain in effect until the logical block is written by any means (e.g., another WRITE LONG command with the COR_DIS bit set to zero and the WR_UNCOR bit set to zero that writes to the same logical block, any WRITE command that specifies the writes to the same logical block, or a FORMAT UNIT command).

In the Extended INQUIRY Data VPD page (see SPC-4), the CD_SUP bit indicates if the logical unit supports the COR_DIS bit being set to one and the WU_SUP bit indicates if the logical unit supports the WR_UNCOR bit being set to one.

The LOGICAL BLOCK ADDRESS field specifies an LBA. If the specified LBA exceeds the capacity of the medium the device server shall terminate the command with CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to LOGICAL BLOCK ADDRESS OUT OF RANGE.

If table 84 indicates the BYTE TRANSFER LENGTH field is honored, the ~~The~~ BYTE TRANSFER LENGTH field specifies the number of bytes of data that the device server shall transfer from the data-out buffer and write to the specified logical block or physical block~~, if the WR_UNCOR bit is set to zero~~. If the BYTE TRANSFER LENGTH field is not set to zero and does not match the data length that the device server returns for a READ LONG command, then the device server shall terminate the command with CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN CDB. In the sense data (see 4.14 and SPC-4), the ILI and VALID bits shall be set to one and the INFORMATION field shall be set to the difference (i.e., residue) of the requested length minus the actual length in bytes. Negative values shall be indicated by two's complement notation. A BYTE TRANSFER LENGTH field set to zero specifies that no bytes shall be written. This condition shall not be considered an error.

Suggested changes to SPC-4

7.6.4 Extended INQUIRY Data VPD page

The Extended INQUIRY Data VPD page (see table 370) provides the application client with a means to obtain information about the logical unit.

Table 370 — Extended INQUIRY Data VPD page

Byte\Bit	7	6	5	4	3	2	1	0
0	PERIPHERAL QUALIFIFER			PERIPHERAL DEVICE TYPE				
1	PAGE CODE (86h)							
2	Reserved							
3	PAGE LENGTH (3Ch)							
4	Reserved		SPT			GRD_CHK	APP_CHK	REF_CHK
5	Reserved			GROUP_SUP	PRIOR_SUP	HEADSUP	ORDSUP	SIMPSUP
6	Reserved				~~Reserved~~ WU_SUP	~~COR_D_SUP~~ CD_SUP	NV_SUP	V_SUP
7	Reserved							LUICLR
8	Reserved							
63								

...

A write uncorrectable supported (WU_SUP) bit set to zero indicates that the device server does not support application clients setting the WR_UNCOR bit to one in the WRITE LONG command (see SBC-3). A WU_SUP bit set to one indicates that the device server supports application clients setting the WR_UNCOR bit to one in the WRITE LONG command.

A correction disable supported (~~COR_D_SUP~~CD_SUP) bit set to zero indicates that the device server does not support application clients ~~disabling read error checking on a logical block written using a~~ setting the COR_DIS bit to one in the WRITE LONG command (see SBC-3). A ~~COR_D_SUP~~CD_SUP bit set to one indicates that the device server supports application client~~s disabling read error checking on a logical block written using a~~ setting the COR_DIS bit to one in the WRITE LONG command.