### T10/08-415 revision 3

Date: 12/2/08 To: T10 Committee (SCSI) From: George Penokie (LSI) Subject: SBC-3/SPC-4 : Adding a Protection Information Interval

## 1 Overview

As logical block sizes get larger there is a concern that the CRC defined in the LOGICAL BLOCK GUARD field is not robust enough to adequately find errors in these larger logical blocks. This proposal adds the option to place protection information at regular intervals within a single logical block for types 1, 2, and 3 protection.

This proposal also leaves room for, but does not define, the option of defining new protection types that would span multiple logical blocks.

## 2 SPC-4 changes

Add a P\_I\_I\_SUP bit into a bit position (editors choice) of the Extended INQUIRY Data VPD page.

The protection information interval supported (P\_I\_SUP) bit set to one indicates that the logical unit supports protection information intervals (SBC-3). A P\_I\_SUP bit set to zero indicates that the logical unit does not support protection information intervals.

# 3 SBC-3 changes

**3.0.1 protection information:** Fields appended to each logical block <u>or added at specified intervals within a</u> <u>logical block</u> that contain a cyclic redundancy check (CRC), an application tag, and a reference tag. See 4.17.

**3.0.2** <u>protection information interval</u>: The length of user data that occurs within a logical block before protection information.

# 3.1 Protection information model

### 3.1.1 Protection information overview

The protection information model provides for protection of user data while it is being transferred between a sender and a receiver. Protection information is generated at the application layer and may be checked by any object associated with the I\_T\_L nexus (see SAM-4). Once received, protection information is retained (e.g., written to medium, stored in non-volatile memory, or recalculated on read back) by the device server until overwritten. Power loss, hard reset, logical unit reset, and I\_T nexus loss shall have no effect on the retention of protection information.

Support for protection information shall be indicated in the PROTECT bit in the standard INQUIRY data (see SPC-4).

If the logical unit is formatted with protection information and the EMDP bit is set to one in the Disconnect-Reconnect mode page (see SPC-4), then checking of the logical block reference tag within a service delivery subsystem without accounting for modified data pointers and data alignments may cause false errors when logical blocks are transmitted out of order.

Protection information is also referred to as the data integrity field (DIF).

### 3.1.2 Protection types

I

### 3.1.2.1 Protection types overview

The content of protection information is dependent on the type of protection to which a logical unit has been formatted.

The type of protection supported by the logical unit shall be indicated in the SPT field in the Extended INQUIRY Data VPD page (see SPC-4). The current protection type shall be indicated in the P\_TYPE field in the READ CAPACITY(16) command (see 5.13).

An application client may format the logical unit to a specific type of protection using the FMTPINFO field and the PROTECTION FIELD USAGE field in the FORMAT UNIT command (see 3.1.8.1).

An application client may format the logical unit to place protection information at intervals other than on logical block boundaries using the PROTECTION INTERVAL EXPONENT field in the FORMAT UNIT command (see 3.1.8.2).

The medium access commands are processed in a different manner by a device server depending on the type of protection in effect. When used in relation to types of protection, the term "medium access commands" is defined as the following commands:

- a) ORWRITE;
- b) READ (10);
- c) READ (12);
- d) READ (16);
- e) READ (32);
- f) VERIFY (10);
- g) VERIFY (12);
- h) VERIFY (16);
- i) VERIFY (32);
- j) WRITE (10);
- k) WRITE (12);
- I) WRITE (16);
- m) WRITE (32);
- n) WRITE AND VERIFY (10);
- o) WRITE AND VERIFY (12);
- p) WRITE AND VERIFY (16);
- q) WRITE AND VERIFY (32);
- r) WRITE SAME (10);
- s) WRITE SAME (16);
- t) WRITE SAME (32);
- u) XDWRITE (10);
- v) XDWRITE (32);
- w) XDWRITEREAD (10);
- x) XDWRITEREAD (32);
- y) XPWRITE (10);
- z) XPWRITE (32);
- aa) XDREAD (10); and
- ab) XDREAD (32).

The device server may allow the READ (6) command (see 5.7) and the WRITE (6) command (see 5.26) regardless of the type of protection to which the logical unit has been formatted.

### 3.1.2.2 Type 0 protection

Type 0 protection defines no protection over that which is defined within the transport protocol.

A logical unit that has been formatted with protection information disabled (see 5.2) or a logical unit that does not support protection information (i.e., the PROTECT bit set to zero in the Standard INQUIRY data (see SPC-4)) has type 0 protection.

If type 0 protection is enabled and the RDPROTECT field, WRPROTECT field, VRPROTECT field, or ORPROTECT field is set to a non-zero value, then media commands are invalid and may be terminated by the device server with CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN CDB.

If type 0 protection is enabled and the RDPROTECT field, WRPROTECT field, VRPROTECT field, or ORPROTECT field is set to a zero value, then the following media commands are invalid and shall be terminated by the device server with CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID COMMAND OPERATION CODE:

- a) READ (32);
- b) VERIFY (32);
- c) WRITE (32);
- d) WRITE AND VERIFY (32); and
- e) WRITE SAME (32).

### 3.1.2.3 Type 1 protection

Type 1 protection:

- a) defines the content of the each LOGICAL BLOCK GUARD field;
- b) does not define the content of the any LOGICAL BLOCK APPLICATION TAG field; and
- c) defines the content the each LOGICAL BLOCK REFERENCE TAG field.

If type 1 protection is enabled, then the following media commands are invalid and shall be terminated by the device server with CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID COMMAND OPERATION CODE:

- a) READ (32);
- b) VERIFY (32);
- c) WRITE (32);
- d) WRITE AND VERIFY (32); and
- e) WRITE SAME (32).

For valid medium access commands in which the RDPROTECT field, WRPROTECT field, VRPROTECT field, or ORPROTECT field is set to:

- a) zero, the data-in buffer and/or data-out buffer associated with those commands shall consist of logical blocks with only user data; or
- b) a non-zero value, the data-in buffer and/or data-out buffer shall consist of logical blocks with both user data and protection information.

### 3.1.2.4 Type 2 protection

Type 2 protection:

- a) defines the content of the each LOGICAL BLOCK GUARD field;
- b) does not define the content of the any LOGICAL BLOCK APPLICATION TAG field; and
- c) defines, except for the first logical block addressed by the command, the content of the each LOGICAL BLOCK REFERENCE TAG field.

If type 2 protection is enabled and the RDPROTECT field, WRPROTECT field, VRPROTECT field, or ORPROTECT field is set to a non-zero value, then the following media commands are invalid and shall be terminated by the device server with CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID COMMAND OPERATION CODE:

- a) ORWRITE;
- b) READ (10);

L

- c) READ (12);
- d) READ (16);
- e) VERIFY (10);
- f) VERIFY (12);
- g) VERIFY (16);
- h) WRITE (10);
- i) WRITE (12);
- j) WRITE (16);
- k) WRITE AND VERIFY (10);
- I) WRITE AND VERIFY (12);
- m) WRITE AND VERIFY (16);
- n) WRITE SAME (10);
- o) WRITE SAME (16);
- p) XDWRITE (10);
- q) XDWRITE (32);
- r) XDWRITEREAD (10);
- s) XDWRITEREAD (32);
- t) XPWRITE (10);
- u) XPWRITE (32);
- v) XDREAD (10); and
- w) XDREAD (32).

For valid medium access commands in which the RDPROTECT field, WRPROTECT field, VRPROTECT field, or ORPROTECT field is set to:

- a) zero, the data-in buffer and/or data-out buffer associated with those commands shall consist of logical blocks with only user data; or
- b) a non-zero value, the data-in buffer and/or data-out buffer shall consist of logical blocks with both user data and protection information.

### 3.1.2.5 Type 3 protection

Type 3 protection:

- a) defines the content of <u>theeach</u> LOGICAL BLOCK GUARD field within the logical blocks of the data-in buffer and/or data-out buffer;
- b) does not define the content of the any LOGICAL BLOCK APPLICATION TAG field; and
- c) does not define the content of the any LOGICAL BLOCK REFERENCE TAG field.

If type 3 protection is enabled, then the following media commands are invalid and shall be terminated by the device server with CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID COMMAND OPERATION CODE:

- a) READ (32);
- b) VERIFY (32);
- c) WRITE (32);
- d) WRITE AND VERIFY (32); and
- e) WRITE SAME (32).

For valid medium access commands in which the RDPROTECT field, WRPROTECT field, VRPROTECT field, or ORPROTECT field is set to:

- a) zero, the data-in buffer and/or data-out buffer associated with those commands shall consist of logical blocks with only user data; or
- b) a non-zero value, the data-in buffer and/or data-out buffer shall consist of logical blocks with both user data and protection information.

### T10/08-415 revision 3

I

### 3.1.3 Protection information format

Table 1 defines the placement of protection information in a logical block with one protection information interval (i.e., PROTECTION INTERVAL EXPONENT field is set to zero in the FORMAT UNIT command (see 3.1.8.2))-.

## Table 1 — User data and protection information format with one protection information interval

Bit Byte	7	6	5	4	3	2	1	0
0			USER DATA -					
n - 1		-						
n	(MSB)		LOGICAL BLOCK GUARD -					
n + 1		-						
n + 2	(MSB)		LOGICAL BLOCK APPLICATION TAG					
n + 3		-						(LSB)
n + 4	(MSB)		LOGICAL BLOCK REFERENCE TAG					
n + 7		-	LUG		(LSB)			

Table 2 shows an example of the placement of protection information in a logical block with more than one protection information interval (i.e., PROTECTION INTERVAL EXPONENT field is set to a non-zero value in the FORMAT UNIT command (see 3.1.8.2)).

I

I

I

I

I

I

I

# Table 2 — Example of user data and protection information format with more than one protection information interval

<u>Bit</u> Byte	<u>7</u>	<u>6</u>	<u>5</u>	<u>4</u>	<u>3</u>	<u>2</u>	1	<u>0</u>	
<u>0</u>			<u>USER DATA (first)</u>						
<u>n - 1</u>		-							
<u>n</u>	<u>(MSB)</u>		LOGICAL BLOCK GUARD (first)						
<u>n + 1</u>		-	<u>L(</u>	JGICAL BLUC	K GUARD (IIIS	<u>st)</u>		<u>(LSB)</u>	
<u>n + 2</u>	<u>(MSB)</u>					(first)			
<u>n + 3</u>		-	LUGICA	AL BLOCK APP	LICATION TAG	<u>s (IIISt)</u>		<u>(LSB)</u>	
<u>n + 4</u>	<u>(MSB)</u>			AL BLOCK RE		(firet)			
<u>n + 7</u>		-	LUGIC	AL BLOCK RE	FERENCE TAC	<u>3 (IIISt)</u>		<u>(LSB)</u>	
<u>n + 8</u>					(second)			_	
<u>m - 1</u>		<u>USER DATA (second)</u>							
<u>m</u>	<u>(MSB)</u>								
<u>m + 1</u>		LOGICAL BLOCK GUARD (second)						<u>(LSB)</u>	
<u>m + 2</u>	<u>(MSB)</u>	LOCION DIOCK ADDITION TAG (20000d)					_		
<u>m + 3</u>		-	LOGICAL BLOCK APPLICATION TAG (second)					<u>(LSB)</u>	
<u>m + 4</u>	<u>(MSB)</u>	LOGICAL BLOCK REFERENCE TAG (second)				_			
<u>m + 7</u>						<u>(LSB)</u>			
			<u></u>						
			<u>USER DATA (last)</u>						
<u>z - 1</u>									
<u>z</u>	<u>(MSB)</u>	-	L	OGICAL BLOC	k guard (las	st)			
<u>z + 1</u>		LOGICAL BLOCK GUARD (last)					<u>(LSB)</u>		
<u>z + 2</u>	<u>(MSB)</u>								
<u>z + 3</u>		LOGICAL BLOCK APPLICATION TAG (last)					<u>(LSB)</u>		
<u>z + 4</u>	<u>(MSB)</u>	_	LOGICAL BLOCK REFERENCE TAG (last)						
<u>z + 7</u>			LUGIC	AL BLUUK KE	I LINUE TAU	<u>, (ast)</u>		<u>(LSB)</u>	

SBC-3/SPC-4 : Adding a Protection Information Interval 6

### T10/08-415 revision 3

L

The Each USER DATA field shall contain user data. The contents of the USER DATA field shall be used to generate and check the CRC contained in the LOGICAL BLOCK GUARD field.

The Each LOGICAL BLOCK GUARD field contains the a CRC (see 3.1.4). Only of the contents of the USER DATA field immediately preceding THE LOGICAL BLOCK GUARD field. (i.e., the user data between the preceding logical block reference tag, if any, and the current logical block guard) shall be used to generate and check the CRC contained in the LOGICAL BLOCK GUARD field.

The Each LOGICAL BLOCK APPLICATION TAG field is set by the application client. If the device server detects a:

- a) LOGICAL BLOCK APPLICATION TAG field set to FFFFh and type 1 protection (see 3.1.2.3) or type 2 protection (see 3.1.2.4) is enabled; or
- b) LOGICAL BLOCK APPLICATION TAG field set to FFFFh, LOGICAL BLOCK REFERENCE TAG field set to FFFF\_FFFh, and type 3 protection (see 3.1.2.5) is enabled,

then the device server disables checking of all protection information for the logical block the associated <u>protection information interval</u> when reading from the medium. Otherwise, the contents of the logical block application tags are not defined by this standard.

The LOGICAL BLOCK APPLICATION TAG field may be modified by a device server if the ATO bit is set to zero in the Control mode page (see SPC-4). If the ATO bit is set to one in the Control mode page, then the device server shall not modify the LOGICAL BLOCK APPLICATION TAG field.

- The contents of the<u>a</u> LOGICAL BLOCK APPLICATION TAG field shall not be used to generate or check the CRC contained in the LOGICAL BLOCK GUARD field.
- The-<u>first</u>LOGICAL BLOCK REFERENCE TAG field of the first logical block in the data-in buffer and/or data-out buffer shall contain the value specified in table 3.

# Table 3 — Contents of the <u>first</u> LOGICAL BLOCK REFERENCE TAG field of the first logical block in the data-in buffer and/or data-out buffer

I	Protection Type	Content of the- <u>first</u> LOGICAL BLOCK REFERENCE TAG field of the first logical block in the data-in buffer and/or data-out buffer					
I	Type 1 protection_ <sup>a</sup> (see 3.1.2.3)	The least significant four bytes of the LBA contained in the LOGICAL BLOCK ADDRESS field of the command.					
I	Type 2 protection (see 3.1.2.4)	The value in the EXPECTED INITIAL LOGICAL BLOCK REFERENCE TAG field of the command.					
I	Type 3 protection (see 3.1.2.5)	Not defined in this standard. However, this field may be modified by the device server if the ATO bit is set to zero in the Control mode page (see SPC-4). If the ATO bit is set to one in the Control mode page, then the device server shall not modify this field.					
I	<sup>a</sup> The length of the protection information interval is equal to the logical block length (see 3.1.8.1).						

The<u>Subsequent</u> LOGICAL BLOCK REFERENCE TAG fields subsequent logical blocks in the data-in buffer and/or data-out buffer shall be set as specified in table 4.

 Table 4 — Setting the-subsequent LOGICAL BLOCK REFERENCE TAG fields

 blocks
 in the data-in buffer and/or data-out buffer

1	Protection Type	The content of thesubsequent LOGICAL BLOCK REFERENCE TAG fields of each- subsequent logical block in the data-in buffer and/or data-out buffer
I	Type 1 protection (see 3.1.2.3) and Type 2 protection (see 3.1.2.4)	The <u>previous log</u> ical block reference tag <del>of the previous logical block</del> plus one.
I	Type 3 protection (see 3.1.2.5)	Not defined in this standard. However, this field may be modified by the device server if the ATO bit is set to zero in the Control mode page (see SPC-4). If the ATO bit is set to one in the Control mode page, then the device server shall not modify this field.

The contents of <u>thea</u> LOGICAL BLOCK REFERENCE TAG field shall not be used to generate or check the CRC contained in the LOGICAL BLOCK GUARD field.

### 3.1.4 Logical block guard

I

I

I

### 3.1.4.1 Logical block guard overview

The<u>A</u> LOGICAL BLOCK GUARD field shall contain a CRC that is generated from the contents of <u>only</u> the USER DATA field <u>immediately preceding the LOGICAL BLOCK GUARD field</u>.

Table 5 defines the CRC polynomials used to generate the logical block guard from the contents of the USER DATA field.

Function	Definition
F(x)	A polynomial representing the transmitted USER DATA field, which is covered by the CRC. For the purposes of the CRC, the coefficient of the highest order term shall be byte zero bit seven of the USER DATA field and the coefficient of the lowest order term shall be bit zero of the last byte of the USER DATA field.
F'(x)	A polynomial representing the received USER DATA field.
G(x)	The generator polynomial: $G(x) = x^{16} + x^{15} + x^{11} + x^9 + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$ (i.e., $G(x) = 18BB7h$ )
R(x)	The remainder polynomial calculated during CRC generation by the transmitter, representing the transmitted LOGICAL BLOCK GUARD field.
R'(x)	A polynomial representing the received LOGICAL BLOCK GUARD field.
RB(x)	The remainder polynomial calculated during CRC checking by the receiver. RB( $x$ ) = 0 indicates no error was detected.
RC(x)	The remainder polynomial calculated during CRC checking by the receiver. RC(x) = 0 indicates no error was detected.
QA(x)	The quotient polynomial calculated during CRC generation by the transmitter. The value of $QA(x)$ is not used.
QB(x)	The quotient polynomial calculated during CRC checking by the receiver. The value of $QB(x)$ is not used.
QC(x)	The quotient polynomial calculated during CRC checking by the receiver. The value of $QC(x)$ is not used.
M(x)	A polynomial representing the transmitted USER DATA field followed by the transmitted LOGICAL BLOCK GUARD field.
M'(x)	A polynomial representing the received USER DATA field followed by the received LOGICAL BLOCK GUARD field.

### Table 5 — CRC polynomials

### 3.1.4.2 CRC generation

The equations that are used to generate the CRC from F(x) are as follows. All arithmetic is modulo 2.

The transmitter shall calculate the CRC by appending 16 zeros to F(x) and dividing by G(x) to obtain the remainder R(x):

$$\frac{(x^{16} \times F(x))}{G(x)} = QA(x) + \frac{R(x)}{G(x)}$$

R(x) is the CRC value, and is transmitted in the LOGICAL BLOCK GUARD field.

M(x) is the polynomial representing the USER DATA field followed by the LOGICAL BLOCK GUARD field (i.e., F(x) followed by R(x)):

$$M(x) = (x^{16} \times F(x)) + R(x)$$

### 3.1.4.3 CRC checking

I

M'(x) (i.e., the polynomial representing the received USER DATA field followed by the received LOGICAL BLOCK GUARD field) may differ from M(x) (i.e., the polynomial representing the transmitted USER DATA field followed by the transmitted LOGICAL BLOCK GUARD field) if there are transmission errors.

The receiver may check M'(x) validity by appending 16 zeros to F'(x) and dividing by G(x) and comparing the calculated remainder RB(x) to the received CRC value R'(x):

$$\frac{(x^{16} \times F'(x))}{G(x)} = QB(x) + \frac{RB(x)}{G(x)}$$

In the absence of errors in F'(x) and R'(x), the remainder RB(x) is equal to R'(x).

The receiver may check M'(x) validity by dividing M'(x) by G(x) and comparing the calculated remainder RC(x) to zero:

$$\frac{M'(x)}{G(x)} = QC(x) + \frac{RC(x)}{G(x)}$$

In the absence of errors in F'(x) and R'(x), the remainder RC(x) is equal to zero.

Both methods of checking M'(x) validity are mathematically equivalent.

### 3.1.4.4 CRC test cases

Several CRC test cases are shown in table 6.

Table 6 — CRC test cases

Pattern	CRC
32 bytes each set to 00h	0000h
32 bytes each set to FFh	A293h
32 bytes of an incrementing pattern from 00h to 1Fh	0224h
2 bytes each set to FFh followed by 30 bytes set to 00h	21B8h
32 bytes of a decrementing pattern from FFh to E0h	A0B7h

### 3.1.5 Application of protection information

Before an application client transmits or receives logical blocks with protection information it shall:

- determine if a logical unit supports protection information using the INQUIRY command (see the PROTECT bit in the standard INQUIRY data in SPC-4);
- 2) if protection information is supported, then determine if the logical unit is formatted to accept protection information using the READ CAPACITY (16) command (see the PROT\_EN bit in 5.13); and
- 3) if the logical unit supports protection information and is not formatted to accept protection information, then format the logical unit with protection information enabled.

If the logical unit supports protection information and is formatted to accept protection information, then the application client may use commands performing read operations that support protection information and should use commands performing write and verify operations that support protection information.

### 3.1.6 Protection information and commands

The enabling of protection information enables fields in some commands that instruct the device server on the handling of protection information. The detailed definitions of each command's protection information fields are in the individual command descriptions.

### T10/08-415 revision 3

The commands that are affected when protection information is enabled are listed in table 13 (see 5.1).

Commands that cause a device server to return the length in bytes of each logical block (e.g., the MODE SENSE commands and the READ CAPACITY commands) shall cause the device server to return the <u>combined</u> length of the USER DATA field(s) <u>contained in the logical block</u>, not including the length <u>of theany</u> protection information (i.e., the LOGICAL BLOCK GUARD field(s), the LOGICAL BLOCK APPLICATION TAG field(s), and the LOGICAL BLOCK REFERENCE TAG field(s) (e.g., if the user data plus the protection information is equal to 520 bytes <u>and there is one protection information interval</u>, then 512 is returned).

### 3.1.7 FORMAT UNIT command overview

...

### 3.1.8 FORMAT UNIT parameter list

### 3.1.8.1 FORMAT UNIT parameter list overview

;;;

### 3.1.8.2 Parameter list header

The parameter list headers (see table 7 and table 8) provide several optional format control parameters. Device servers that implement these headers provide the application client additional control over the use of the four defect sources, and the format operation. If the application client attempts to select any function not implemented by the device server, 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 PARAMETER LIST.

If the LONGLIST bit is set to zero in the FORMAT UNIT CDB, then the short parameter list header (see table 7) is used.

Bit Byte	7	6	5	4	3	2	1	0
0	Reserved						CTION FIELD	USAGE
1	FOV	DPRY	DCRT	STPF	IP	Obsolete	IMMED	Vendor- specific
2	(MSB)				ST LENGTH			
3		-						(LSB)

### Table 7 — Short parameter list header

If the LONGLIST bit is set to one in the FORMAT UNIT CDB, then the long parameter list header (see table 8) is used.

Bit Byte	7	6	5	4	3	2	1	0
0			Reserved			PROTE	CTION FIELD	USAGE
1	FOV	DPRY	DCRT	STPF	IP	Obsolete	IMMED	Vendor- specific
2		Reserved						
3	Reserved         P_I_INFORMATION         PROTECTION INTERVAL EXPONENT						IENT	
4	(MSB)							
7		DEFECT LIST LENGTH (LSB)						

### Table 8 — Long parameter list header

The PROTECTION FIELD USAGE field in combination with the FMTPINFO field (see table 9) specifies the requested protection type (see 4.17.2).

Device server indication		Application client specification		Description	
SPT <sup>a</sup>	PROTECT b	FMTPINFO	PROTECTION FIELD USAGE	Description	
xxxb	0	00b	000b	The logical unit shall be formatted to type 0 protection <sup>c</sup> (see 4.17.2.2) resulting in the P_TYPE field <sup>d</sup> being set to 000b.	
xxxb	0	00b	>000b	Illegal <sup>e</sup>	
xxxb	0	01b	xxxb	Illegal <sup>f</sup>	
xxxb	0	1xb	xxxb	Illegal <sup>f</sup>	
xxxb	1	00b	000b	The logical unit shall be formatted to type 0 protection <sup>c</sup> (see 4.17.2.2) resulting in the P_TYPE field <sup>d</sup> being set to 000b.	
xxxb	1	00b	>000b	Illegal <sup>e</sup>	

### Table 9 — FMTPINFO field and PROTECTION FIELD USAGE field (part 1 of 2)

<sup>a</sup> See the Extended INQUIRY Data VPD page (see SPC-4) for the definition of the SPT field.

<sup>b</sup> See the standard INQUIRY data (see SPC-4) for the definition of the PROTECT bit.

- <sup>c</sup> The device server shall format the medium to the logical block length specified in the mode parameter block descriptor of the mode parameter header (see SPC-4).
- <sup>d</sup> See the READ CAPACITY command (see 3.6.1) for the definition of the P\_TYPE field.
- <sup>e</sup> 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 PARAMETER LIST.
- <sup>†</sup> 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.

<sup>g</sup> The device server shall format the medium to the logical block length specified in the mode parameter block descriptor of the mode parameter header plus eight <u>bytes for each protection information interval</u> (e.g., if the logical block length is <del>512</del><u>2</u> 048 and the PROTECTION INTERVAL EXPONENT field set to two, then there are four 512 byte protection information intervals each followed by eight bytes of protection information intervals each followed by eight bytes of protection information resulting in a formatted logical block length of 2 080 bytes), then the formatted logical block length is <del>520</del>. Following a successful format, the PROT\_EN bit in the READ CAPACITY (16) parameter data (see 3.6.1) indicates whether protection information (see 4.17) is enabled.

	Device server indication		ition client	Description
SPT <sup>a</sup>	PROTECT b	FMTPINFO	PROTECTION FIELD USAGE	
xxxb	1	01b	xxxb	Illegal <sup>f</sup>
000b 001b 011b	1	10b	000b	The logical unit shall be formatted to type 1 protection <sup>g</sup> (see 4.17.2.3) resulting in the P_TYPE field <sup>d</sup> being set to 000b.
000b 001b 011b	1	10b	>000b	Illegal <sup>e</sup>
000b	1	11b	xxxb	Illegal <sup>f</sup>
001b	1	11b	000b	The logical unit shall be formatted to type 2 protection <sup>g</sup> (see 4.17.2.4) resulting in the P_TYPE field <sup>d</sup> being set to 001b.
001b	1	11b	>000b	Illegal <sup>e</sup>
011b	1	11b	000b	Illegal <sup>e</sup>
011b	1	1	001b	The logical unit shall be formatted to type 3 protection. <sup>g</sup> (see 4.17.2.5) resulting in the P_TYPE field <sup>d</sup> being set to 010b.
011b	1	11b	>001b	Illegal <sup>e</sup>
010b	1	1xb	xxxb	Reserved
1xxb	1	1xb	xxxb	Reserved

<sup>a</sup> See the Extended INQUIRY Data VPD page (see SPC-4) for the definition of the SPT field.

<sup>b</sup> See the standard INQUIRY data (see SPC-4) for the definition of the PROTECT bit.

<sup>c</sup> The device server shall format the medium to the logical block length specified in the mode parameter block descriptor of the mode parameter header (see SPC-4).

<sup>d</sup> See the READ CAPACITY command (see 3.6.1) for the definition of the P\_TYPE field.

<sup>e</sup> 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 PARAMETER LIST.

f 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.

<sup>g</sup> The device server shall format the medium to the logical block length specified in the mode parameter block descriptor of the mode parameter header plus eight <u>bytes for each protection information interval</u> (e.g., if the logical block length is <u>5122</u> 048 and the PROTECTION INTERVAL EXPONENT field set to two, then there are four 512 byte protection information intervals each followed by eight bytes of protection information information intervals each followed by eight bytes of protection information logical block length is <u>520</u>). Following a successful format, the PROT\_EN bit in the READ CAPACITY (16) parameter data (see 3.6.1) indicates whether protection information (see 4.17) is enabled.

....

For a type 1 protection formation request if the PROTECTION INTERVAL EXPONENT field is not set to zero, 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 PARAMETER LIST.

For a type 2 protection or type 3 protection format requests the protection interval exponent determines the length of user data to be sent before protection information is transferred (i.e., the protection information interval).

The protection information interval is calculated as follows:

protection information interval = logical block length / 2<sup>(protection interval exponent)</sup>

where:

logical block length	is the length in bytes of a logical block as specified in the mode
	parameter block descriptor of the mode parameter header (see
	<u>6.3.2.1xx)</u>

protection interval exponent is the contents of the PROTECTION INTERVAL EXPONENT field

If the protection information interval calculates to a value that is not an even number (e.g.,  $520/2^3 = 65$ ) or not a whole number (e.g.,  $520/2^4 = 32.5$  and  $520/2^{10} = 0.508$ ), 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 PARAMETER LIST.

The P\_I\_INFORMATION field shall be set to zero.

## 3.2 ORWRITE command

•••

If the check of protection information read from the medium and check protection information transferred from the data-out buffer is successful, then the device server shall set the protection information (see 4.17) as it writes each logical block to the medium as follows:

- a) the<u>each</u> LOGICAL BLOCK GUARD field set to a CRC properly generated (see 4.17.4) by the device server;
- b) the<u>each</u> LOGICAL BLOCK REFERENCE TAG field set to the same LOGICAL BLOCK REFERENCE TAG field received from the data-out buffer; and
- c) the <u>each</u> LOGICAL BLOCK APPLICATION TAG field set to the same LOGICAL BLOCK APPLICATION TAG field received from the data-out buffer.

The order of the user data and protection information checks and comparisons is vendor-specific.

The device server shall check the protection information read from the medium based on the ORPROTECT field as described in table 10.

Code	Logical unit formatted with protection information	Field in protection information g	Extended INQUIRY Data VPD page bit value <sup>f</sup>	If check fails <sup>de</sup> , additional sense code		
		LOGICAL	GRD_CHK = 1	LOGICAL BLOCK GUARD CHECK FAILED		
	Yes	BLOCK GUARD	grd_chk = 0	No check performed		
		LOGICAL BLOCK APPLICATION TAG LOGICAL BLOCK	арр_снк = 1 <sup>с</sup>	LOGICAL BLOCK APPLICATION TAG CHECK FAILED		
000b			APP_CHK = 0	No check performed		
			REF_CHK = 1 <sup>h</sup>	LOGICAL BLOCK REFERENCE TAG CHECK FAILED		
		REFERENCE TAG	ref_chk = 0	No check performed		
	No	No protection information on the medium to check. Only user data is checked.				

Table 10 — ORPROTECT field - checking protection information read from the medium (part 1 of 4)

Table 10 — ORPROTECT field - checking protection information read from the medium (part 2	2 of 4)
---	---------

Code	Logical unit formatted with protection information	Field in protection information g	Extended INQUIRY Data VPD page bit value <sup>f</sup>	If check fails <sup>de</sup> , additional sense code	
		LOGICAL	GRD_CHK = 1	LOGICAL BLOCK GUARD CHECK FAILED	
		BLOCK GUARD	grd_chk = 0	No check performed	
		LOGICAL BLOCK	АРР_СНК <b>=</b> 1 <sup>с</sup>	LOGICAL BLOCK APPLICATION TAG CHECK FAILED	
001b 101b <sup>b</sup>	Yes	APPLICATION TAG	APP_CHK = 0	No check performed	
1010		LOGICAL BLOCK REFERENCE TAG	REF_CHK = 1 <sup>h</sup>	LOGICAL BLOCK REFERENCE TAG CHECK FAILED	
			ref_chk = 0	No check performed	
	No	Error condition	a		
	Yes	LOGICAL BLOCK GUARD	No check performed		
		LOGICAL BLOCK Yes TAG	АРР_СНК = 1 <sup>с</sup>	LOGICAL BLOCK APPLICATION TAG CHECK FAILED	
010b <sup>b</sup>			APP_CHK = 0	No check performed	
		LOGICAL BLOCK REFERENCE TAG	REF_CHK = 1 <sup>h</sup>	LOGICAL BLOCK REFERENCE TAG CHECK FAILED	
			ref_chk = 0	No check performed	
	No	Error condition	а		

Code	Logical unit formatted with protection information	Field in protection information g	Extended INQUIRY Data VPD page bit value <sup>f</sup>	If check fails <sup>de</sup> , additional sense code	
		LOGICAL BLOCK GUARD	No check perform	ned	
011b <sup>b</sup>	Yes	LOGICAL BLOCK APPLICATION TAG	No check performed		
		LOGICAL BLOCK REFERENCE TAG	No check performed		
	No	Error condition	а		
		LOGICAL	GRD_CHK = 1	LOGICAL BLOCK GUARD CHECK FAILED	
	Yes	BLOCK GUARD	grd_chk = 0	No check performed	
100b <sup>b</sup>		LOGICAL BLOCK APPLICATION TAG	No check perform	ned	
		LOGICAL BLOCK REFERENCE TAG	No check performed		
	No	Error condition	n <sup>a</sup>		
110b to 111b	Reserved				

I

I

defined by this standard.

Table 10 — ORPROTECT field - checking protection information read from the medium (part 4 of 4)

Code	Logical unit formatted with protection information	Field in protection information g	Extended INQUIRY Data VPD page bit value <sup>f</sup>	If check fails <sup>de</sup> , additional sense code		
been CON INVA b If the required and f C The LOGIO stand d If the be se e If mu f See g If the a) L	formatted with DITION status LID FIELD IN ( logical unit dou ested command the additional s device server s CAL BLOCK APPL dard. device server to ABORTED the Extended II CHK bit, and the device server .OGICAL BLOCK A	protection inform with the sense k CDB. es not support p d with CHECK C ense code set to hall check the lo ICATION TAG field terminates the co O COMMAND. cur, the selection NQUIRY Data VI e REF_CHK bits. detects a: APPLICATION TAG	mation shall be terr ey set to ILLEGAL rotection informatic ONDITION status INVALID FIELD IN gical block applicat This knowledge r ommand with CHE of which error to r PD page (see SPC field set to FFFFh	tion tag if it has knowledge of the contents of the may be obtained by a method not defined by this CK CONDITION status, then the sense key shall eport is not defined by this standard. -4) for the definitions of the GRD_CHK bit, the and type 1 protection (see 4.17.2.3) is enabled;		
f then block h If typ com enat	<ul> <li>b) LOGICAL BLOCK APPLICATION TAG field set to FFFFh, LOGICAL BLOCK REFERENCE TAG field set to FFFF_FFFh, and type 3 protection (see 4.17.2.5) is enabled, then the device server shall not check any protection information in the associated logical- blockprotection information interval.</li> </ul>					

The device server shall check the protection information transferred from the data-out buffer based on the ORPROTECT field as described in table 11.

Table 11 — ORPROTECT field	checking protection	information from th	e data-out buffer (pa	t 1 of 3)
	oncoming protootion		c duta out sunoi (pa	(1010)

Code	Logical unit formatted with protection information	Field in protection information	Device server check	If check fails <sup>de</sup> , additional sense code	
000b	Yes	No protection information received from application client to check			
0000	No	No protection information received from application client to check			
	Yes	Logical Block Guard	Shall	LOGICAL BLOCK GUARD CHECK FAILED	
001b <sup>b</sup>		LOGICAL BLOCK APPLICATION TAG	May <sup>c</sup>	LOGICAL BLOCK APPLICATION TAG CHECK FAILED	
		LOGICAL BLOCK REFERENCE TAG	Shall (except for type 3) <sup>f</sup>	LOGICAL BLOCK REFERENCE TAG CHECK FAILED	
	No	Error condition <sup>a</sup>			

L

Table 11 — ORPROTECT field - checking protection information from the data-out buffer (part 2 of 3)

Code	Logical unit formatted with protection information	Field in protection information	Device server check	If check fails <sup>de</sup> , additional sense code
		LOGICAL BLOCK GUARD	Shall not	No check performed
010b <sup>b</sup>	Yes	LOGICAL BLOCK APPLICATION TAG	May <sup>c</sup>	LOGICAL BLOCK APPLICATION TAG CHECK FAILED
		LOGICAL BLOCK REFERENCE TAG	May <sup>f</sup>	LOGICAL BLOCK REFERENCE TAG CHECK FAILED
	No	Error condition a		
		LOGICAL BLOCK GUARD	Shall not	No check performed
011b <sup>b</sup>	Yes	LOGICAL BLOCK APPLICATION TAG	Shall not	No check performed
		LOGICAL BLOCK REFERENCE TAG	Shall not	No check performed
	No	Error condition a		
	Yes	LOGICAL BLOCK GUARD	Shall	LOGICAL BLOCK GUARD CHECK FAILED
100b <sup>b</sup>		LOGICAL BLOCK APPLICATION TAG	Shall not	No check performed
		LOGICAL BLOCK REFERENCE TAG	Shall not	No check performed
	No	Error condition <sup>a</sup>		
		LOGICAL BLOCK GUARD	Shall	LOGICAL BLOCK GUARD CHECK FAILED
101b <sup>b</sup>	Yes	LOGICAL BLOCK APPLICATION TAG	May <sup>c</sup>	LOGICAL BLOCK APPLICATION TAG CHECK FAILED
		LOGICAL BLOCK REFERENCE TAG	May <sup>f</sup>	LOGICAL BLOCK REFERENCE TAG CHECK FAILED
	No	Error condition a		
110b to 111b	Reserved			

Table 11 — ORPROTECT field - checking protection information from the data-out buffer (part 3 of 3)

Code	Logical unit formatted with protection information	Field in protection information	Device server check	If check fails <sup>de</sup> , additional sense code
form statu FIEL b If the requ and <sup>c</sup> The Cont BLOC d If the set to f If typ comp enab know	atted with protect is with the sense D IN CDB. e logical unit does ested command the additional sen device server ma rol mode page (s K APPLICATION TA e device server te o ABORTED COI ultiple errors occu be 1 protection is paring it to the low oled, the ATO bit is vledge of the con	ion information sha key set to ILLEGAI a not support protect with CHECK CONE use code set to INV by check <u>theeach</u> lo ee SPC-4) and if th G field. This knowle rminates the common MAND. r, the selection of v enabled, then the cover wer 4 bytes of the Los s set to one in the Cover tents of the LOGICAL	Il be terminated to L REQUEST and ction information, DITION status wit (ALID FIELD IN C gical block applic the device server edge is obtained hand with CHECK which error to rep device server che BA associated w Control mode pag BLOCK REFERENC	tion information (see 4.17) and has not been by the device server with CHECK CONDITION the additional sense code set to INVALID then the device server should terminate the h the sense key set to ILLEGAL REQUEST CDB. cation tag if the ATO bit is set to one in the has knowledge of the contents of the LOGICAL by a method not defined by this standard. CONDITION status, the sense key shall be ort is not defined by this standard. cks the logical block reference tag by vith the logical block. If type 3 protection is je (see SPC-4), and the device server has CE TAG field, then the device server may check iring this knowledge is not defined by this

standard.

# 3.3 READ (6) command

•••

I

I

The device server shall check the protection information read from the medium before returning status for the command as described in table 12.

Logical unit formatted with protection information	Shall device server transmit protection information?	Field in protection information	Extended INQUIRY Data VPD page bit value <sup>d</sup>	lf check fails <sup>bc</sup> , additional sense code
	No	LOGICAL BLOCK GUARD	grd_chk = 1	LOGICAL BLOCK GUARD CHECK FAILED
		GOARD	grd_chk = 0	No check performed
Yes		LOGICAL BLOCK APPLICATION TAG	арр_снк = 1 <sup>а</sup>	LOGICAL BLOCK APPLICATION TAG CHECK FAILED
			APP_CHK = 0	No check performed
		LOGICAL BLOCK REFERENCE TAG	REF_CHK = 1 <sup>f</sup>	LOGICAL BLOCK REFERENCE TAG CHECK FAILED
			REF_CHK = 0	No check performed
No		No protection information available to check		

Table 12 — Pro	tection information	n checking for READ (6)
----------------	---------------------	-------------------------

<sup>a</sup> The device server checks <u>theeach</u> logical block application tag only if it has knowledge of the contents of the LOGICAL BLOCK APPLICATION TAG field. The method for acquiring this knowledge is not defined by this standard.

<sup>b</sup> If the device server terminates the command with CHECK CONDITION status, then the sense key shall be set to ABORTED COMMAND.

- <sup>c</sup> If multiple errors occur, the selection of which error to report is not defined by this standard.
- <sup>d</sup> See the Extended INQUIRY Data VPD page (see SPC-4) for the definitions of the GRD\_CHK bit, APP\_CHK bit, and REF\_CHK bit.
- <sup>e</sup> If the device server detects a:

a) LOGICAL BLOCK APPLICATION TAG field set to FFFFh and type 1 protection (see 4.17.2.3) or type 2 protection (see 4.17.2.4) is enabled; or

b) LOGICAL BLOCK APPLICATION TAG field set to FFFFh, LOGICAL BLOCK REFERENCE TAG field set to FFFF\_FFFh, and type 3 protection (see 4.17.2.5) is enabled,

then the device server shall not check any protection information in the associated logicalblockprotection information interval.

If type 1 protection is enabled, then the device server checks the logical block reference tag by comparing it to the lower 4 bytes of the LBA associated with the logical block. If type 2 protection or type 3 protection is enabled, then the device server checks the each logical block reference tag only if it has knowledge of the contents of the LOGICAL BLOCK REFERENCE TAG field. The method for acquiring this knowledge is not defined by this standard.

## 3.4 READ (10) command

f

### T10/08-415 revision 3

The device server shall check the protection information read from the medium before returning status for the command based on the RDPROTECT field as described in table 13.

Code	Logical unit formatted with protection information	Shall device server transmit protection information?	Field in protection information <sup>h</sup>	Extended INQUIRY Data VPD page bit value <sup>g</sup>	lf check fails <sup>d</sup> f, additional sense code		
			LOGICAL BLOCK GUARD	grd_chk = 1	LOGICAL BLOCK GUARD CHECK FAILED		
			COARD	grd_chk = 0	No check performed		
	Yes		LOGICAL BLOCK APPLICATION TAG	арр_снк = 1 <sup>с</sup>	LOGICAL BLOCK APPLICATION TAG CHECK FAILED		
000b		No	TAG	APP_CHK = 0	No check performed		
			LOGICAL BLOCK REFERENCE TAG	REF_CHK = 1 <sup>i</sup>	LOGICAL BLOCK REFERENCE TAG CHECK FAILED		
				REF_CHK = 0	No check performed		
	No		No protection information available to check				
			LOGICAL BLOCK GUARD	grd_chk = 1	LOGICAL BLOCK GUARD CHECK FAILED		
			GUARD	grd_chk = 0	No check performed		
	Yes	Yes <sup>e</sup>	LOGICAL BLOCK APPLICATION	арр_снк = 1 <sup>с</sup>	LOGICAL BLOCK APPLICATION TAG CHECK FAILED		
001b 101b <sup>b</sup>			TAG	АРР_СНК = 0	No check performed		
			LOGICAL BLOCK REFERENCE TAG	REF_CHK = 1 <sup>i</sup>	LOGICAL BLOCK REFERENCE TAG CHECK FAILED		
			140	REF_CHK = 0	No check performed		
	No <sup>a</sup>	No protection in checking	formation availab	le to transmit to th	e data-in buffer or for		

Table 13 — RDPROTECT field (part 1 of 3)

Code	Logical unit formatted with protection information	Shall device server transmit protection information?	Field in protection information <sup>h</sup>	Extended INQUIRY Data VPD page bit value <sup>g</sup>	lf check fails <sup>d</sup> f, additional sense code	
			LOGICAL BLOCK GUARD	No check performed		
	Ma a	Vec 6	LOGICAL BLOCK APPLICATION TAG	арр_снк = 1 <sup>с</sup>	LOGICAL BLOCK APPLICATION TAG CHECK FAILED	
010b <sup>b</sup>	Yes	Yes <sup>e</sup>	TAG	арр_снк = 0	No check performed	
0100			LOGICAL BLOCK REFERENCE TAG	REF_СНК = 1 <sup>і</sup>	LOGICAL BLOCK REFERENCE TAG CHECK FAILED	
			170	ref_chk = 0	No check performed	
	No <sup>a</sup>	No protection in checking	formation availab	ole to transmit to th	e data-in buffer or for	
	Yes Yes <sup>e</sup>		LOGICAL BLOCK GUARD	No check performed		
011b <sup>b</sup>		Yes <sup>e</sup>	LOGICAL BLOCK APPLICATION TAG	No check performed		
			LOGICAL BLOCK REFERENCE TAG	No check performed		
	No <sup>a</sup>	No protection in checking	formation availab	le to transmit to th	e data-in buffer or for	
			LOGICAL BLOCK		LOGICAL BLOCK GUARD CHECK FAILED	
			GUARD	grd_chk = 0	No check performed	
100b <sup>b</sup>	Yes	Yes <sup>e</sup>	LOGICAL BLOCK APPLICATION TAG	No check performed		
			LOGICAL BLOCK REFERENCE TAG	No check performed		
	No <sup>a</sup>	No protection information available to transmit to the data-in buffer or for checking				
110b to 111b	Reserved	1				

Code	Logical unit formatted with protection information	Shall device server transmit protection information?	Field in protection information <sup>h</sup>	Extended INQUIRY Data VPD page bit value <sup>g</sup>	lf check fails <sup>df</sup> , additional sense code
forma CON INVA b If the comr addit c The c of the bit is EXPE the C d If the shall e Trans f If mu g See h If the a) L b) L f then b) L f then i f typ comp type know then i a f	atted with protect DITION status w LID FIELD IN C logical unit does mand with CHEC ional sense code device server sh e LOGICAL BLOCK set to one in the CTED LOGICAL BLOCK set to one in the CTED LOGICAL BLOCK be set to ABOR smit protection ir litiple errors occu the Extended IN CHK bit, and the device server d OGICAL BLOCK AF FFF_FFFh, and the device server e 1 protection is en portection is en contection is en c	tion information s with the sense key DB. s not support prote CK CONDITION size set to INVALID F all check the <u>each</u> APPLICATION TAG f c Control mode par OCK APPLICATION T this knowledge m erminates the corr TED COMMAND. of ormation to the corr TED COMMAND. of a control mode par OCK APPLICATION TAG file (17.2.4) is enabled PLICATION TAG file of type 3 protection of the sense of the mabled, then the mabled, then the d otents of the LOGIC may be acquired the	hall be terminated set to ILLEGAL F ection information tatus with the ser FIELD IN CDB. logical block app field. If the READ ge (see SPC-4), TAG field and the mand with CHEC data-in buffer. f which error to re page (see SPC- data-in buffer. f which error to re page (see SPC- f type 3 protection	d by the device ser REQUEST and the h, then the device ser lise key set to ILLE blication tag if it has (32) command (se then this knowledg LOGICAL BLOCK APP y a method not def CK CONDITION state of the definition and type 1 protection LOGICAL BLOCK REF is enabled, formation in the assist necks the logical blocks the act hogical blocks the logical blocks the act hogical blocks the logical blocks the logical blocks ince TAG field. If typ TED INITIAL LOGICAL	additional sense code set to server should terminate the GAL REQUEST and the s knowledge of the contents e 5.11) is used, and the ATC ge is acquired from the PLICATION TAG MASK field in fined by this standard. atus, then the sense key by this standard. Is of the GRD_CHK bit, the on (see 4.17.2.3) or type 2 FERENCE TAG field set to

# 3.5 READ (32) command

...

I

If the ATO bit is set to one in the Control mode page (see SPC-4) and checking of the LOGICAL BLOCK APPLICATION TAG field is enabled (see table 39 in 5.8), then the LOGICAL BLOCK APPLICATION TAG MASK field contains a value that is a bit mask for enabling the checking of the LOGICAL BLOCK APPLICATION TAG field in theevery instance in which protection information occurs for each logical block accessed by the command. A LOGICAL BLOCK APPLICATION TAG MASK field bit set to one enables the checking of the corresponding bit of the EXPECTED LOGICAL BLOCK APPLICATION TAG field with the corresponding bit of the LOGICAL BLOCK APPLICATION TAG field in theevery instance in which protection information occurs.

## 3.6 READ CAPACITY (16) command

### 3.6.1 READ CAPACITY (16) command overview

### 3.6.2 READ CAPACITY (16) parameter data

#### 

| |

...

...

The READ CAPACITY (16) parameter data is defined in table 14. Any time the READ CAPACITY (16) parameter data changes, the device server should establish a unit attention condition as described in 4.7.

Bit Byte	7	6	5	4	3	2	1	0
0	(MSB)		DETI			DESS		
7		-	RETURNED LOGICAL BLOCK ADDRESS (LSB)					(LSB)
8	(MSB)							
11		LOGICAL BLOCK LENGTH IN BYTES (LSB)					(LSB)	
12		Rese	erved		P_TYPE			PROT_EN
13		Reserved <sub>P</sub>	I_EXPONENT		LOGICAL BLO	OCKS PER PH	YSICAL BLOC	K EXPONENT
14	Rese	erved	(MSB)	LOWEST	ALIGNED LOGICAL BLOCK ADDRESS			
15		LOWEST ALIGN					ADDITESS	(LSB)
16		Reserved						
31		-		TC3C				

### Table 14 — READ CAPACITY (16) parameter data

The protection type (P\_TYPE) field and the protection enable (PROT\_EN) bit (see table 15) indicate the logical unit's current type of protection.

PROT_EN	P_TYPE	Description
0	xxxb	The logical unit is formatted to type 0 protection (see 4.17.2.2).
1	000b	The logical unit is formatted to type 1 protection (see 4.17.2.3).
1	001b	The logical unit is formatted to type 2 protection (see 4.17.2.4).
1	010b	The logical unit is formatted to type 3 protection (see 4.17.2.5).
1	011b to 111b	Reserved

### Table 15 — P\_TYPE field and PROT\_EN bit

### T10/08-415 revision 3

The LOGICAL BLOCKS PER PHYSICAL BLOCK EXPONENT field is defined in table 16.

Table 16 — LOGICAL B	BLOCKS PER PHYSICAL	BLOCK EXPONENT field
----------------------	---------------------	----------------------

Code	Description			
0	One or more physical blocks per logical block <sup>a</sup>			
n > 0	2 <sup>n</sup> logical blocks per physical block			
<sup>a</sup> The number of physical blocks per logical block is not reported.				

<u>The p\_i exponent field may be used to determine the number of protection information intervals placed within</u> each logical block (see 3.1.8).

The number of protection information intervals is calculated as follows:

number of protection information intervals =  $2^{(p_i exponent)}$ 

where:

### <u>p\_i exponent</u> is the contents of the P\_I EXPONENT field

The LOWEST ALIGNED LOGICAL BLOCK ADDRESS field indicates the LBA of the first logical block that is located at the beginning of a physical block (see 4.5).

NOTE 1 - The highest LBA that the lowest aligned logical block address field supports is 3FFFh (i.e., 16 383).

# 3.7 VERIFY (10) command

•••

If the BYTCHK bit is set to zero, then the device server shall check the protection information read from the medium based on the VRPROTECT field as described in table 17.

Table 17 — VRPROTECT field with BYTCHK set to zero - checking protection information read from the
medium (part 1 of 3)

Code	Logical unit formatted with protection information	Field in protection information g	Extended INQUIRY Data VPD page bit value <sup>f</sup>	If check fails <sup>de</sup> , additional sense code
		LOGICAL	GRD_CHK = 1	LOGICAL BLOCK GUARD CHECK FAILED
		BLOCK GUARD	grd_chk = 0	No check performed
	0b Yes	LOGICAL BLOCK	арр_снк = 1 <sup>с</sup>	LOGICAL BLOCK APPLICATION TAG CHECK FAILED
000b		APPLICATION TAG	APP_CHK = 0	No check performed
		LOGICAL BLOCK	REF_CHK = 1 <sup>h</sup>	LOGICAL BLOCK REFERENCE TAG CHECK FAILED
		REFERENCE TAG	ref_chk = 0	No check performed
	No	No protection i	nformation on the I	medium to check. Only user data is checked.

# Table 17 — VRPROTECT field with BYTCHK set to zero - checking protection information read from the medium (part 2 of 3)

Code	Logical unit formatted with protection information	Field in protection information g	Extended INQUIRY Data VPD page bit value <sup>f</sup>	If check fails <sup>de</sup> , additional sense code
		LOGICAL	grd_chk = 1	LOGICAL BLOCK GUARD CHECK FAILED
		BLOCK GUARD	grd_chk = 0	No check performed
		LOGICAL BLOCK	АРР_СНК <b>=</b> 1 <sup>с</sup>	LOGICAL BLOCK APPLICATION TAG CHECK FAILED
001b 101b <sup>b</sup>	Yes	APPLICATION TAG	APP_CHK = 0	No check performed
		LOGICAL BLOCK	REF_CHK = 1 <sup>h</sup>	LOGICAL BLOCK REFERENCE TAG CHECK FAILED
		REFERENCE TAG	REF_CHK = 0	No check performed
	No	Error condition	а	
		LOGICAL BLOCK GUARD	No check perforn	ned
		LOGICAL BLOCK	АРР_СНК <b>=</b> 1 <sup>с</sup>	LOGICAL BLOCK APPLICATION TAG CHECK FAILED
010b <sup>b</sup>	Yes	APPLICATION TAG	APP_CHK = 0	No check performed
		LOGICAL BLOCK	REF_CHK = 1 <sup>h</sup>	LOGICAL BLOCK REFERENCE TAG CHECK FAILED
		REFERENCE TAG	REF_CHK = 0	No check performed
	No	Error condition	а	
		LOGICAL BLOCK GUARD	No check perforn	ned
011b <sup>b</sup>	Yes	LOGICAL BLOCK APPLICATION TAG	No check perforn	ned
		LOGICAL BLOCK REFERENCE TAG	No check perforn	ned
	No	Error condition	а	
		LOGICAL	grd_chk = 1	LOGICAL BLOCK GUARD CHECK FAILED
		BLOCK GUARD	grd_chk = 0	No check performed
100b <sup>b</sup>	Yes	LOGICAL BLOCK APPLICATION TAG	No check performed	
		LOGICAL BLOCK	No check perforn	ned
		REFERENCE TAG		

Table 17 — VRPROTECT field with BYTCHK set to zero - checking protection information read from the
medium (part 3 of 3)

		mediam (part 5 or 5)								
Code	Logical unit formatted with protection information	Field in protection information g	Extended INQUIRY Data VPD page bit value <sup>f</sup>	If check fails <sup>de</sup> , additional sense code						
110b to 111b	Reserved	Reserved								
110b to Reserved										

I

I

If the BYTCHK bit is set to one, then the device server shall check the protection information read from the medium based on the VRPROTECT field as described in table 18.

I

# Table 18 — VRPROTECT field with BYTCHK set to one - checking protection information read from the medium (part 1 of 2)

Code	Logical unit formatted with protection information	Field in protection information <sup>g</sup>	Extended INQUIRY Data VPD page bit value <sup>f</sup>	If check fails <sup>de</sup> , additional sense code
		LOGICAL BLOCK GUARD	GRD_СНК = 1	LOGICAL BLOCK GUARD CHECK FAILED
		OUARD	grd_chk = 0	No check performed
0004	Yes	LOGICAL BLOCK APPLICATION TAG	АРР_СНК = 1 <sup>С</sup> g	LOGICAL BLOCK APPLICATION TAG CHECK FAILED
000b			APP_CHK = 0	No check performed
		LOGICAL BLOCK REFERENCE TAG	REF_CHK = 1 <sup>h</sup>	LOGICAL BLOCK REFERENCE TAG CHECK FAILED
			REF_CHK = 0	No check performed
	No	No protection information on the medium available to check		
001b		LOGICAL BLOCK GUARD	No check perforr	ned
010b 011b 100b	Yes	LOGICAL BLOCK APPLICATION TAG	No check perform	ned
100b 101b		LOGICAL BLOCK REFERENCE TAG	No check perform	ned
	No	Error condition a		
110b to 111b	Reserved			

I

# Table 18 — VRPROTECT field with BYTCHK set to one - checking protection information read from the medium (part 2 of 2)

Code	Logical unit formatted with protection information	Field in protection information <sup>g</sup>	Extended INQUIRY Data VPD page bit value <sup>f</sup>	If check fails <sup>de</sup> , additional sense code
form stat FIE b If th requ and c The bit i EXP CDI d If th be s e If m f See g If th a) b) ther bloc h If ty con 3 pr kno ther in a	matted with protection is with the sen is D IN CDB. The logical unit do uested command the additional sector device server is set to one in the PECTED LOGICAL B. Otherwise, the device server set to ABORTE builtiple errors of the Extended P_CHK bit, and the device server LOGICAL BLOCK protection (see LOGICAL BLOCK FFFF_FFFh, in the device server che the extended protection (see LOGICAL BLOCK FFFF_FFFh, in the device server che the to the protection is enal owledge of the c in this knowledg of VERIFY (32) c	ection information sha se key set to ILLEGA bes not support prote- ind with CHECK CONI sense code set to INV shall check theeach lo APPLICATION TAG field the Control mode pag BLOCK APPLICATION TAG field the Control mode pag terminates the comm D COMMAND. the REF_CHK bit. The REF_CHK bi	all be terminated by L REQUEST and t ction information, t DITION status with /ALID FIELD IN CI ogical block applica I. If the VERIFY (32 le (see SPC-4), the constant of the LOG e obtained by a me nand with CHECK on of which error to bage (see SPC-4) the d set to FFFFh and g or d set to FFFFh, LOG n (see 4.17.2.5) is e ny protection inform device server checks BA associated with server checks the Server checks the rough the EXPECTE f type 3 protection	Ation tag if it has knowledge of the contents of 2) command (see 5.25) is used, and the ATO en this knowledge is acquired from the SICAL BLOCK APPLICATION TAG MASK field in the thod not defined by this standard. CONDITION status, then the sense key shall report is not defined by this standard. for the definitions of the GRD_CHK bit, the type 1 protection (see 4.17.2.3) or type 2 GICAL BLOCK REFERENCE TAG field set to

If the BYTCHK bit is set to one, then the device server shall check the protection information transferred from the data-out buffer based on the VRPROTECT field as described in table 19.

I

# Table 19 — VRPROTECT field with BYTCHK set to one - checking protection information from the data-out buffer (part 1 of 2)

Code	Logical unit formatted with protection information	Field in protection information	Device server check	If check fails <sup>d</sup> <sup>e</sup> , additional sense code
000b	Yes	No protection info	ormation received	d from application client to check
	No	No protection info	ormation received	d from application client to check
		LOGICAL BLOCK GUARD	Shall	LOGICAL BLOCK GUARD CHECK FAILED
001b <sup>b</sup>	Yes	LOGICAL BLOCK APPLICATION TAG	May <sup>c</sup>	LOGICAL BLOCK APPLICATION TAG CHECK FAILED
		LOGICAL BLOCK REFERENCE TAG	Shall (except for type 3)	LOGICAL BLOCK REFERENCE TAG CHECK FAILED
	No	Error condition a		
	Yes	LOGICAL BLOCK GUARD	Shall not	No check performed
010b <sup>b</sup>		LOGICAL BLOCK APPLICATION TAG	May <sup>c</sup>	LOGICAL BLOCK APPLICATION TAG CHECK FAILED
		LOGICAL BLOCK REFERENCE TAG	May <sup>f</sup>	LOGICAL BLOCK REFERENCE TAG CHECK FAILED
	No	Error condition a		
	Yes	LOGICAL BLOCK GUARD	Shall not	No check performed
011b <sup>b</sup>		LOGICAL BLOCK APPLICATION TAG	Shall not	No check performed
		LOGICAL BLOCK REFERENCE TAG	Shall not	No check performed
	No	Error condition a		
	Yes	LOGICAL BLOCK GUARD	Shall	LOGICAL BLOCK GUARD CHECK FAILED
100b <sup>b</sup>		LOGICAL BLOCK APPLICATION TAG	Shall not	No check performed
		LOGICAL BLOCK REFERENCE TAG	Shall not	No check performed
	No	Error condition <sup>a</sup>		·

# Table 19 — VRPROTECT field with BYTCHK set to one - checking protection information from the data-out buffer (part 2 of 2)

Code	Logical unit formatted with protection information	Field in protection information	Device server check	If check fails <sup>de</sup> , additional sense code			
		LOGICAL BLOCK GUARD	Shall	LOGICAL BLOCK GUARD CHECK FAILED			
101b <sup>b</sup>	Yes	LOGICAL BLOCK APPLICATION TAG	May <sup>c</sup>	LOGICAL BLOCK APPLICATION TAG CHECK FAILED			
		LOGICAL BLOCK REFERENCE TAG	May <sup>f</sup>	LOGICAL BLOCK REFERENCE TAG CHECK FAILED			
	No	Error condition a					
110b to 111b	Reserved						
statu FIEL b If the reque and t C The c Obtai TAG M stand d If the shall e If mu f If typ comp enab field, then in a N Cont	<ul> <li>A verify operation to a logical unit that supports protection information (see 4.17) and has not been formatted with protection information shall be terminated by the device server with CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN CDB.</li> <li>If the logical unit does not support protection information, then the device server should terminate the requested command with CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN CDB.</li> <li>The device server may check theeach logical block application tag if the ATO bit is set to one in the Control mode page (see SPC-4) and if the device server has knowledge of the contents of the LOGICAL BLOCK APPLICATION TAG field. If the VERIFY (32) command (see 5.25) is used, then this knowledge is obtained from the EXPECTED LOGICAL BLOCK APPLICATION TAG field and the LOGICAL BLOCK APPLICATION TAG field in the CDB. Otherwise, this knowledge is obtained by a method not defined by this standard.</li> <li>If the device server terminates the command with CHECK CONDITION status, then the device server shall set the sense key to ABORTED COMMAND.</li> <li>If multiple errors occur, the selection of which error to report is not defined by this standard.</li> </ul>						

If the BYTCHK bit is set to one, then the device server shall perform a byte-by-byte comparison of protection information transferred from the data-out buffer with protection information read from the medium based on the VRPROTECT field as described in table 20.

I

Code	Logical unit formatted with protection information	Field	Byte-by-byte Comparison	If compare fails <sup>c d</sup> , additional sense code		
Yes         No protection information received from application of data is compared within each logical block.						
	No			m or received from application client to within each logical block.		
		LOGICAL BLOCK GUARD	Shall	LOGICAL BLOCK GUARD CHECK FAILED		
		LOGICAL BLOCK APPLICATION TAG (ATO = 1) <sup>e</sup>	Shall	LOGICAL BLOCK APPLICATION TAG CHECK FAILED		
		LOGICAL BLOCK APPLICATION TAG (ATO = 0) <sup>f</sup>	Shall not	No compare performed		
001b <sup>b</sup>	Yes	LOGICAL BLOCK REFERENCE TAG (not type 3)	Shall	LOGICAL BLOCK REFERENCE TAG CHECK FAILED		
		LOGICAL BLOCK REFERENCE TAG (type 3 and ATO = 0)	Shall	LOGICAL BLOCK REFERENCE TAG CHECK FAILED		
		LOGICAL BLOCK REFERENCE TAG (type 3 and ATO = 1)	Shall not	No compare performed		
	No	Error condition <sup>a</sup>				
		LOGICAL BLOCK GUARD	Shall not	No compare performed		
	<sub>b</sub> Yes	LOGICAL BLOCK APPLICATION TAG (ATO = 1) <sup>e</sup>	Shall	LOGICAL BLOCK APPLICATION TAG CHECK FAILED		
		LOGICAL BLOCK APPLICATION TAG (ATO = 0) f	Shall not	No compare performed		
010b <sup>b</sup>		LOGICAL BLOCK REFERENCE TAG (not type 3)	Shall	LOGICAL BLOCK REFERENCE TAG CHECK FAILED		
		LOGICAL BLOCK REFERENCE TAG (type 3 and ATO = 0)		Shall	LOGICAL BLOCK REFERENCE TAG CHECK FAILED	
		LOGICAL BLOCK REFERENCE TAG (type 3 and ATO = 1)	Shall not	No compare performed		
	No	Error condition <sup>a</sup>				

Table 20 — VRPROTECT field with BYTCHK set to one - byte-by-byte comparison requirements (part 2 of 2)

Code	Logical unit formatted with protection information	Field	Byte-by-byte Comparison	If compare fails <sup>c d</sup> , additional sense code	
		LOGICAL BLOCK GUARD	Shall	LOGICAL BLOCK GUARD CHECK FAILED	
		LOGICAL BLOCK APPLICATION TAG (ATO = 1) <sup>e</sup>	Shall	LOGICAL BLOCK APPLICATION TAG CHECK FAILED	
		LOGICAL BLOCK APPLICATION TAG (ATO = 0) f	Shall not	No compare performed	
011b 100b <sup>b</sup>	Yes	LOGICAL BLOCK REFERENCE TAG (not type 3)	Shall	LOGICAL BLOCK REFERENCE TAG CHECK FAILED	
		LOGICAL BLOCK REFERENCE TAG (type 3 and ATO = 0)	Shall	LOGICAL BLOCK REFERENCE TAG CHECK FAILED	
		LOGICAL BLOCK REFERENCE TAG (type 3 and ATO = 1)	Shall not	No compare performed	
	No	Error condition a			
	Yes	LOGICAL BLOCK GU		Shall	LOGICAL BLOCK GUARD CHECK FAILED
		LOGICAL BLOCK APPLICATION TAG (ATO = 1) <sup>e</sup>	Shall	LOGICAL BLOCK APPLICATION TAG CHECK FAILED	
101b <sup>b</sup>		LOGICAL BLOCK APPLICATION TAG (ATO = 0) f	Shall not	No compare performed	
		LOGICAL BLOCK REFERENCE TAG	Shall not	No compare performed	
	No	Error condition <sup>a</sup>			
110b to 111b	Reserved				
<ul> <li><sup>a</sup> A verify operation to a logical unit that supports protection information (see 4.17) and has not been formatted with protection information shall be terminated by the device server with CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN CDB.</li> <li><sup>b</sup> If the logical unit does not support protection information, then the device server should terminate the requested command with CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN CDB.</li> <li><sup>c</sup> If the device server terminates the command with CHECK CONDITION status, then the sense key shall be set to MISCOMPARE.</li> <li><sup>d</sup> If multiple errors occur, the selection of which error to report is not defined by this standard.</li> <li><sup>e</sup> If the ATO bit is set to one in the Control mode page (see SPC-4), then the logical block application tag shall not be modified by a device server.</li> </ul>					

<sup>f</sup> If the ATO bit is set to zero in the Control mode page (see SPC-4), then <u>theany</u> logical block application tag may be modified by a device server.

# 3.8 VERIFY (32) command

#### • • •

If the ATO bit is set to one in the Control mode page (see SPC-4) and checking of the LOGICAL BLOCK APPLICATION TAG field is enabled (see table 67, table 68, table 69, and table 70 in 5.22), then the LOGICAL BLOCK APPLICATION TAG MASK field contains a value that is a bit mask for enabling the checking of the LOGICAL BLOCK APPLICATION TAG field in the protection information every instance in which protection information occurs for each logical block accessed by the command. A LOGICAL BLOCK APPLICATION TAG MASK bit set to one enables the checking of the corresponding bit of the EXPECTED LOGICAL BLOCK APPLICATION TAG field with the corresponding bit of the LOGICAL BLOCK APPLICATION TAG field with the mathematical block accessed by the corresponding bit of the EXPECTED LOGICAL BLOCK APPLICATION TAG field with the corresponding bit of the LOGICAL BLOCK APPLICATION TAG field in the protection information every instance in which protection information occurs.

## 3.9 WRITE (6) command

...

If a WRITE (6) command is received after protection information is enabled, then the device server shall set the protection information (see 4.17) as follows as the device server writes each logical block to the medium:

- a) the each LOGICAL BLOCK GUARD field set to a properly generated CRC (see 4.17.4);
- b) theeach LOGICAL BLOCK REFERENCE TAG field set to:
  - A) the least significant four bytes of the LBA, if type 1 protection (see 4.17.2.3) is enabled;
  - B) FFFF\_FFFh, if type 2 protection is enabled (see 4.17.2.4);
  - C) FFFF\_FFFh, if the ATO bit is set to one in the Control mode page (see SPC-4), and type 3 protection (see 4.17.2.5) is enabled; or
  - D) any value, if the ATO bit is set to zero in the Control mode page (see SPC-4), and type 3 protection (see 4.17.2.5) is enabled;

and

- c) the each LOGICAL BLOCK APPLICATION TAG field set to:
  - A) FFFFh, if the ATO bit is set to one in the Control mode page (see SPC-4); or
  - B) any value, if the ATO bit is set to zero in the Control mode page (see SPC-4).

## 3.10 WRITE (10) command

•••

The device server shall check the protection information transferred from the data-out buffer based on the WRPROTECT field as described in table 21.

Code	Logical unit formatted with protection information	Field in protection information	Device server check	If check fails <sup>di</sup> , additional sense code		
000b Yes <sup>f g h</sup> No protection information		ormation receive	ed from application client to check			
0000	No	No protection information received from application client to check				

### Table 21 — WRPROTECT field (part 1 of 3)

Code	Logical unit formatted with protection information	Field in protection information	Device server check	If check fails <sup>di</sup> , additional sense code
		LOGICAL BLOCK GUARD	Shall	LOGICAL BLOCK GUARD CHECK FAILED
001b <sup>b</sup>	Yes <sup>e</sup>	LOGICAL BLOCK APPLICATION TAG	May <sup>c</sup>	LOGICAL BLOCK APPLICATION TAG CHECK FAILED
		LOGICAL BLOCK REFERENCE TAG	Shall (except for type 3) <sup>j</sup>	LOGICAL BLOCK REFERENCE TAG CHECK FAILED
	No <sup>a</sup>	No protection info	ormation availat	ble to check
		Logical Block Guard	Shall not	No check performed
010b <sup>b</sup>	Yes <sup>e</sup>	LOGICAL BLOCK APPLICATION TAG	May <sup>c</sup>	LOGICAL BLOCK APPLICATION TAG CHECK FAILED
		LOGICAL BLOCK REFERENCE TAG	May <sup>j</sup>	LOGICAL BLOCK REFERENCE TAG CHECK FAILED
	No <sup>a</sup>	No protection info	ormation availat	ble to check
	Yes <sup>e</sup>	LOGICAL BLOCK GUARD	Shall not	No check performed
011b <sup>b</sup>		LOGICAL BLOCK APPLICATION TAG	Shall not	No check performed
		LOGICAL BLOCK REFERENCE TAG	Shall not	No check performed
	No <sup>a</sup>	No protection info	ormation availat	ble to check
		LOGICAL BLOCK GUARD	Shall	LOGICAL BLOCK GUARD CHECK FAILED
100b <sup>b</sup>	Yes <sup>e</sup>	LOGICAL BLOCK APPLICATION TAG	Shall not	No check performed
		LOGICAL BLOCK REFERENCE TAG	Shall not	No check performed
	No <sup>a</sup>	No protection info	ormation availat	ble to check
		LOGICAL BLOCK GUARD	Shall	LOGICAL BLOCK GUARD CHECK FAILED
101b <sup>b</sup>	Yes <sup>e</sup>	LOGICAL BLOCK APPLICATION TAG	May <sup>c</sup>	LOGICAL BLOCK APPLICATION TAG CHECK FAILED
		LOGICAL BLOCK REFERENCE TAG	May <sup>j</sup>	LOGICAL BLOCK REFERENCE TAG CHECK FAILED
	No <sup>a</sup>	No protection info	ormation availat	ble to check
110b to 111b	Reserved			

Table 21 — WRPROTECT field (part 3 of 3)

Logical unit formatted Code with protection information	Field in protection information	Device server check	If check fails <sup>di</sup> , additional sense code
formatted with prof CONDITION status INVALID FIELD IN b If the logical unit de requested comman and the additional c The device server Control mode page APPLICATION TAG file from the EXPECTED field in the CDB. O d If the device server be set to ABORTE e Device server shal non-volatile memo f The device server field. g If the P_TYPE field protection is enable the LOGICAL BLOCK to 000btype 2 prote FFFF_FFFh into each Lus server may write a i If multiple errors of j If type 1 protection comparing it to the enabled, and the d field, then the devit then this knowledg in a WRITE (32) of Control mode page BLOCK REFERENCE	ection information s s with the sense key CDB. Des not support prot ad with CHECK CON sense code set to IN may check theeach e (see SPC-4) and if eld. If the WRITE (32 LOGICAL BLOCK APPI therwise, this knowledge therwise, this knowledge terminates the com D COMMAND. I preserve the conter ry). shall write a property s set to 000h in the ed, then the device s REFERENCE TAG field ection or type 3 protection theeach LOGICAL BLOCK APPLIC to one in the Contro DGICAL BLOCK APPLIC to one in the Contro DGICAL BLOCK APPLIC to one in the Contro DGICAL BLOCK APPLIC to use into each L ccur, the selection of is enabled, then the lower 4 bytes of the evice server checks the e may be acquired t ommand (see 5.30). e (see SPC-4), and t TAG field, then the d	hall be terminat set to ILLEGAL ection informati NDITION status IVALID FIELD I logical block ap it has knowled 2) command (se LICATION TAG file edge is obtaine mand with CHE nts of protection y generated CR READ CAPACI server shall writ I of each of the ection is enable DCK REFERENCE I mode page (s CATION TAG field OGICAL BLOCK A which error to e device server a LBA associate owledge of the ecach logical blo hrough the EXPL If type 3 protec he device server material	ion information (see 4.17) and has not been ed by the device server with CHECK . REQUEST and the additional sense code set to on, then the device server should terminate the with the sense key set to ILLEGAL REQUEST N CDB. oplication tag if the ATO bit is set to one in the ge of the contents of the LOGICAL BLOCK ee 5.30) is used, then this knowledge is obtained Id and the LOGICAL BLOCK APPLICATION TAG MASK d by a method not defined by this standard. CK CONDITION status, then the sense key shall in information (e.g., write to medium, store in C (see 4.17.4.2) into each LOGICAL BLOCK GUARD TY (16) parameter data (see 5.13)type 1 e the least significant four bytes of each LBA into written logical blocks. If the P_TYPE field is not set- id, then the device server shall write a value of TAG field of each of the written logical blocks. ee SPC-4), then the device server shall write . If the ATO bit is set to zero, then the device PPLICATION TAG field. report is not defined by this standard. checks the logical block. If type 2 protection is contents of the LOGICAL BLOCK REFERENCE TAG pok reference tag. If type 2 protection is enabled, ECTED INITIAL LOGICAL BLOCK REFERENCE TAG field tion is enabled, the ATO bit is set to one in the er has knowledge of the contents of the LOGICAL ay check the each logical block reference tag. If thiring this knowledge is not defined by this

# 3.11 WRITE (32) command

•••

I

If the ATO bit is set to one in the Control mode page (see SPC-4) and checking of the LOGICAL BLOCK APPLICATION TAG field is enabled (see table 21 in 3.10), then the LOGICAL BLOCK APPLICATION TAG MASK field contains a value that is a bit mask for enabling the checking of the LOGICAL BLOCK APPLICATION TAG field in the protection information every instance in which protection information occurs for each logical block accessed by the command. A LOGICAL BLOCK APPLICATION TAG MASK bit set to one enables the checking of the corresponding bit of the EXPECTED LOGICAL BLOCK APPLICATION TAG field with the corresponding bit of the

### T10/08-415 revision 3

LOGICAL BLOCK APPLICATION TAG field in the protection information every instance in which protection information occurs.

# 3.12 WRITE AND VERIFY (32) command

If the ATO bit is set to one in the Control mode page (see SPC-4) and checking of the LOGICAL BLOCK APPLICATION TAG field is enabled (see table 21 in 3.10), then the LOGICAL BLOCK APPLICATION TAG MASK field contains a value that is a bit mask for enabling the checking of the LOGICAL BLOCK APPLICATION TAG field in theprotection information every instance in which protection information occurs for each logical block accessed by the command. A LOGICAL BLOCK APPLICATION TAG MASK bit set to one enables the checking of the corresponding bit of the EXPECTED LOGICAL BLOCK APPLICATION TAG field with the corresponding bit of the LOGICAL BLOCK APPLICATION TAG field in the protection information every instance in which protection information occurs.

## 3.13 WRITE SAME (10) command

...

Table 22 describes the lbdata bit and the pbdata bit.

I

I

I

Table 22 — LBDATA bit and PBDATA bit			
LBDATA	PBDATA	Description	
0	0	<ul> <li>The device server shall write the single block of user data received from the data-out buffer to each logical block without modification.</li> <li>If the medium is formatted with type 1 or type 2 protection information, then: <ul> <li>a) the value in theeach LOGICAL BLOCK REFERENCE TAG field received in the single block of data from the data-out buffer shall be placed into the <u>corresponding</u> LOGICAL BLOCK REFERENCE TAG field of the first logical block written to the medium. Into each of the subsequent LOGICAL BLOCK.</li> <li>REFERENCE TAG fields logical blocks, the device server shall place into the <u>LOGICAL BLOCK REFERENCE</u> TAG field plus one.</li> <li>b) If the ATO bit is set to one in the Control mode page (see SPC-4), then theeach logical block application tag received in the single block of data shall be placed in the <u>corresponding</u> LOGICAL BLOCK APPLICATION TAG field of each logical block. If the ATO bit is set to zero, then the device server may write any value into the LOGICAL BLOCK APPLICATION TAG field(s) of each logical block; and</li> <li>c) The value in theeach LOGICAL BLOCK GUARD field received in the single block of data shall be placed in the -corresponding LOGICAL BLOCK APPLICATION TAG field(s) of each logical block; and</li> <li>c) The value into the data-out buffer shall be placed in the <u>corresponding</u> LOGICAL BLOCK REFERENCE TAG field of each logical block.</li> </ul> </li> <li>If the medium is formatted with type 3 protection information: <ul> <li>a) If the ATO bit is set to one in the Control mode page (see SPC-4), then theeach logical block is set to zero, then the device server may write any value into the LOGICAL BLOCK REFERENCE TAG field of each logical block. If the ATO bit is set to zero, then the device server may write any value into the LOGICAL BLOCK REFERENCE TAG field of each logical block. If the ATO bit is set to zero, then the device server may write any value into the LOGICAL BLOCK REFERENCE TAG field (s) of each logical block. If the ATO bit is set to zero, then</li></ul></li></ul>	
0	1 <sup>a</sup>	The device server shall replace the first eight bytes of the block received from the data-out buffer to each physical sector with the physical address of the sector being written using the physical sector format (see 5.2.2.4.5).	
1 <sup>a</sup>	0	The device server shall replace the first four bytes of the block received from the data-out buffer with the least significant four bytes of the LBA of the block being written, ending with the least significant byte (e.g., if the LBA is 7766_5544_3322_1100h, 3322_1100h is written with 33h written first and 00h written last).	
		rmatted with protection information, then <u>all the protection information shall be</u> value of FFFF_FFFF_FFFFFFFFFFFFFFFFFFFFFFFFFF	

### Table 22 — LBDATA bit and PBDATA bit

LBDATA	PBDATA	Description
1	1	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.
<sup>a</sup> If the medium is formatted with protection information, then <u>all</u> the protection information shall be written to a default value of FFFF_FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF		

### Table 22 — LBDATA bit and PBDATA bit

# 3.14 WRITE SAME (32) command

If the ATO bit is set to one in the Control mode page (see SPC-4) and checking of the LOGICAL BLOCK APPLICATION TAG field is enabled (see table 21 in 3.10), then the LOGICAL BLOCK APPLICATION TAG MASK field contains a value that is a bit mask for enabling the checking of the LOGICAL BLOCK APPLICATION TAG field in theprotection information every instance in which protection information occurs for each logical block accessed by the command. A LOGICAL BLOCK APPLICATION TAG MASK bit set to one enables the checking of the corresponding bit of the EXPECTED LOGICAL BLOCK APPLICATION TAG field with the corresponding bit of the LOGICAL BLOCK APPLICATION TAG field in the protection information every instance in which protection information occurs.