



ISO/IEC 14776-333

Edition 1.0 2013-09

INTERNATIONAL STANDARD

**Information technology – Small Computer System Interface (SCSI) –
Part 333: SCSI Stream Commands – 3 (SSC-3)**



THIS PUBLICATION IS COPYRIGHT PROTECTED

Copyright © 2013 ISO/IEC, Geneva, Switzerland

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either IEC or IEC's member National Committee in the country of the requester.

If you have any questions about ISO/IEC copyright or have an enquiry about obtaining additional rights to this publication, please contact the address below or your local IEC member National Committee for further information.

IEC Central Office
3, rue de Varembe
CH-1211 Geneva 20
Switzerland

Tel.: +41 22 919 02 11
Fax: +41 22 919 03 00
info@iec.ch
www.iec.ch

About the IEC

The International Electrotechnical Commission (IEC) is the leading global organization that prepares and publishes International Standards for all electrical, electronic and related technologies.

About IEC publications

The technical content of IEC publications is kept under constant review by the IEC. Please make sure that you have the latest edition, a corrigenda or an amendment might have been published.

Useful links:

IEC publications search - www.iec.ch/searchpub

The advanced search enables you to find IEC publications by a variety of criteria (reference number, text, technical committee,...).

It also gives information on projects, replaced and withdrawn publications.

IEC Just Published - webstore.iec.ch/justpublished

Stay up to date on all new IEC publications. Just Published details all new publications released. Available on-line and also once a month by email.

Electropedia - www.electropedia.org

The world's leading online dictionary of electronic and electrical terms containing more than 30 000 terms and definitions in English and French, with equivalent terms in additional languages. Also known as the International Electrotechnical Vocabulary (IEV) on-line.

Customer Service Centre - webstore.iec.ch/csc

If you wish to give us your feedback on this publication or need further assistance, please contact the Customer Service Centre: csc@iec.ch.



ISO/IEC 14776-333

Edition 1.0 2013-09

INTERNATIONAL STANDARD

**Information technology – Small Computer System Interface (SCSI) –
Part 333: SCSI Stream Commands – 3 (SSC-3)**

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

PRICE CODE **XC**

ICS 35.200

ISBN 978-2-8322-1128-1

Warning! Make sure that you obtained this publication from an authorized distributor.

AVA Documents
Document Sharing Hub

CONTENTS

FOREWORD	9
INTRODUCTION	11
1 Scope	12
2 Normative references	13
3 Terms, definitions, acronyms, keywords and conventions	14
3.1 Terms and definitions.....	14
3.2 Acronyms	22
3.3 Keywords	22
3.4 Editorial conventions.....	23
3.5 Notation conventions – State diagrams	24
4 General concepts	26
4.1 Overview	26
4.2 Sequential-access device model	26
4.2.1 Sequential-access device model overview	26
4.2.2 Physical elements.....	26
4.2.3 Removable volumes	29
4.2.4 Device entity	29
4.2.5 Early-warning	32
4.2.6 Programmable early warning	32
4.2.7 Partitions within a volume	33
4.2.8 Logical objects	34
4.2.9 Logical files	35
4.2.10 Object buffering	35
4.2.11 Synchronize operation behavior	36
4.2.12 Direction and position definitions	36
4.2.13 Error reporting.....	37
4.2.14 Write protection.....	39
4.2.15 Progress indication	41
4.2.16 Tagged command queuing	42
4.2.17 Block address mode	43
4.2.18 TapeAlert application client interface	52
4.2.19 READ ATTRIBUTE and WRITE ATTRIBUTE command support	58
4.2.20 Reservations.....	59
4.2.21 WORM volume and WORM mode.....	61
4.2.22 Logical block encryption	61
4.2.23 External data encryption control	72
4.2.24 Logical block encryption key protection	78
4.2.25 Appending data to a volume containing encrypted logical blocks	78
4.2.26 Self-test operations.....	79
4.2.27 Capability-based command (CbCS) security	79
5 Explicit address command descriptions for sequential-access devices	81
5.1 Summary of commands for explicit address mode	81
5.2 ERASE(16) command.....	84
5.3 READ(16) command.....	85
5.4 READ REVERSE(16) command.....	89
5.5 VERIFY(16) command.....	90
5.6 WRITE(16) command	92
5.7 WRITE FILEMARKS(16) command.....	94
6 Implicit address command descriptions for sequential-access devices	96
6.1 Summary of commands for implicit address mode	96
6.2 ERASE(6) command.....	99
6.3 LOCATE(10) command	100

6.4	READ(6) command.....	101
6.5	READ REVERSE(6) command.....	103
6.6	SPACE(6) command.....	103
6.7	VERIFY(6) command.....	106
6.8	WRITE(6) command	107
6.9	WRITE FILEMARKS(6) command.....	108
7	Common command descriptions for sequential-access devices	110
7.1	FORMAT MEDIUM command	110
7.2	LOAD UNLOAD command	111
7.3	LOCATE(16) command	113
7.4	PREVENT ALLOW MEDIUM REMOVAL command	114
7.5	READ BLOCK LIMITS command	115
7.6	READ POSITION command	117
7.6.1	READ POSITION command description	117
7.6.2	READ POSITION DATA format, short form.....	118
7.6.3	READ POSITION data format, long form	120
7.6.4	READ POSITION data format, extended form	122
7.7	RECOVER BUFFERED DATA command	123
7.8	REPORT DENSITY SUPPORT command	124
7.8.1	REPORT DENSITY SUPPORT command description.....	124
7.8.2	REPORT DENSITY SUPPORT header.....	124
7.8.3	Density support report	125
7.8.4	Medium type support report.....	127
7.9	REWIND command.....	129
7.10	SET CAPACITY command	130
7.11	SPACE(16) command.....	130
8	Parameters for sequential-access devices	134
8.1	Diagnostic parameters.....	134
8.2	Log parameters.....	134
8.2.1	Log parameters overview	134
8.2.2	Sequential-Access Device log page	135
8.2.3	TapeAlert log page	137
8.2.4	Device Statistics log page.....	138
8.2.5	Tape Diagnostic Data log page	140
8.2.6	Current Service Information log page	144
8.2.7	Requested Recovery log page	149
8.3	Mode parameters.....	152
8.3.1	Mode parameters overview	152
8.3.2	Data Compression mode page	155
8.3.3	Device Configuration mode page	159
8.3.4	Medium Partition mode page	163
8.3.5	Read-Write Error Recovery mode page	166
8.3.6	Informational Exceptions Control mode page.....	168
8.3.7	Medium Configuration mode page.....	169
8.3.8	Device Configuration Extension mode page.....	170
8.4	Vital product data (VPD) parameters	172
8.4.1	VPD parameters overview and page codes	172
8.4.2	Sequential-access Device Capabilities VPD page	173
8.4.3	Manufacturer-assigned Serial Number VPD page.....	173
8.4.4	TapeAlert Supported Flags VPD page	174
8.4.5	Automation Device Serial Number VPD page	174
8.5	Security protocol parameters	175
8.5.1	Security protocol overview.....	175
8.5.2	SECURITY PROTOCOL IN command specifying Tape Data Encryption security protocol.....	175

8.5.3 SECURITY PROTOCOL OUT command specifying Tape Data Encryption security protocol.....	190
8.5.4 SECURITY PROTOCOL IN and SECURITY PROTOCOL OUT descriptors	201
Annex A (informative) Application client recommendations for using TapeAlert	205
Annex B (informative) Security environment	211
Annex C (informative) Example keyless copy operation flowchart	213
Annex D (informative) Sense logical block information for error conditions	215
BIBLIOGRAPHY	220

Figure 1 — SCSI document relationships	12
Figure 2 — Example state diagram	25
Figure 3 — Typical volume layout	27
Figure 4 — Typical medium track layout	27
Figure 5 — Serpentine recording example	28
Figure 6 — Parallel recording example	28
Figure 7 — Helical scan recording example	28
Figure 8 — UML example of SCSI target device and device entity	30
Figure 9 — Early-warning example	32
Figure 10 — Programmable early warning example	32
Figure 11 — Partitioning example - one partition per track group	33
Figure 12 — Partitioning example - one partition per two track groups	34
Figure 13 — Partitioning example - two partitions per track group	34
Figure 14 — Block address mode state diagram, overview	45
Figure 15 — Block address mode state diagram, Idle state	46
Figure 16 — Block address mode state diagram, Explicit Address Mode - Neutral	48
Figure 17 — Block address mode state diagram, Explicit Address Mode - Write Capable	50
Figure 18 — Block address mode state diagram, Implicit Address Mode	51
Figure B.1 — Simple security deployment environment	211
Figure C.1 — Example keyless copy operation flowchart	214

Table 1 — Numbering conventions examples	24
Table 2 — Device entity attributes	30
Table 3 — Stream commands sense data descriptor	37
Table 4 — Information sense data descriptor	38
Table 5 — Error conditions and sense keys	38
Table 6 — Write protect additional sense code combinations	40
Table 7 — Commands providing progress indication without changing ready state.....	41
Table 8 — Commands changing ready state and providing progress indication	42
Table 9 — TapeAlert flags severity	52
Table 10 — TapeAlert flags	53
Table 11 — TapeAlert flag activation conditions.....	57
Table 12 — Device common attributes.....	58
Table 13 — Medium common attributes	59
Table 14 — SSC-3 commands that are allowed in the presence of various reservations	60
Table 15 — Default I_T_L nexus logical block encryption information	69
Table 16 — Logical block encryption parameters for encryption request policies	74
Table 17 — Logical block encryption parameters for decryption request policies	75
Table 18 — Logical block encryption parameters for encryption request indicator settings	76
Table 19 — Logical block encryption parameters for decryption request indicator settings	76
Table 20 — Logical block encryption period timer expired indicator.....	77
Table 21 — Association between commands and CbCS permissions	79
Table 22 — Explicit address command set for sequential-access devices.....	81
Table 23 — ERASE(16) command	84
Table 24 — METHOD field	85
Table 25 — READ(16) command	86
Table 26 — READ REVERSE(16) command	89
Table 27 — VERIFY(16) command	90
Table 28 — WRITE(16) command.....	92
Table 29 — WRITE FILEMARKS(16) command	94
Table 30 — Implicit address command set for sequential-access devices.....	96
Table 31 — ERASE(6) command	99
Table 32 — LOCATE(10) command	100
Table 33 — READ(6) command	101
Table 34 — READ REVERSE(6) command	103
Table 35 — SPACE(6) command	104
Table 36 — CODE field.....	104
Table 37 — VERIFY(6) command	106
Table 38 — WRITE(6) command.....	107
Table 39 — WRITE FILEMARKS(6) command	108
Table 40 — FORMAT MEDIUM command.....	110
Table 41 — FORMAT field	111
Table 42 — LOAD UNLOAD command.....	112
Table 43 — LOCATE(16) command	113
Table 44 — DEST_TYPE field.....	114
Table 45 — PREVENT ALLOW MEDIUM REMOVAL command.....	114
Table 46 — PREVENT field	115
Table 47 — READ BLOCK LIMITS command	115
Table 48 — READ BLOCK LIMITS data.....	116
Table 49 — READ POSITION command.....	117
Table 50 — READ POSITION service action codes.....	117
Table 51 — READ POSITION data format, short form	118
Table 52 — READ POSITION data format, long form	120
Table 53 — READ POSITION data format, extended form	122
Table 54 — RECOVER BUFFERED data command.....	123
Table 55 — REPORT DENSITY SUPPORT command.....	124
Table 56 — REPORT DENSITY SUPPORT header	124
Table 57 — Density support data block descriptor	125
Table 58 — Medium type descriptor	128
Table 59 — REWIND command	129
Table 60 — SET CAPACITY command.....	130

Table 61 — SPACE(16) command	131
Table 62 — Space positioning information	132
Table 63 — Diagnostic page codes	134
Table 64 — Log page codes	134
Table 65 — Parameter codes for Sequential-Access Device log page	135
Table 66 — TapeAlert log page	137
Table 67 — TapeAlert parameter format	137
Table 68 — Device Statistics log page	138
Table 69 — Device Statistics log parameter codes	138
Table 70 — Device statistics data counter log parameter format	139
Table 71 — Medium type log parameter format	140
Table 72 — Medium type parameter format	140
Table 73 — Tape Diagnostic Data log page	141
Table 74 — Tape diagnostic data log parameter format	142
Table 75 — Current Service Information log page	144
Table 76 — Service information log parameter format	144
Table 77 — Service information descriptor	145
Table 78 — SERVICE INFORMATION DESCRIPTOR TYPE field	145
Table 79 — Vendor-specific service information descriptor	145
Table 80 — Device information descriptor	146
Table 81 — DEC field	146
Table 82 — DEVICE REQUESTED RECOVERY field	147
Table 83 — Volume information descriptor	147
Table 84 — VIC field	147
Table 85 — VICQ field	148
Table 86 — TapeAlert flag specific information descriptor	149
Table 87 — Requested Recovery log page	150
Table 88 — REQUESTED RECOVERY LOG PARAMETER CODES	150
Table 89 — Requested recovery log parameter format	150
Table 90 — RECOVERY PROCEDURES	151
Table 91 — Device-specific parameter	152
Table 92 — Buffered modes	152
Table 93 — SPEED field	153
Table 94 — Sequential-access density codes	153
Table 95 — Mode page codes and subpage codes	154
Table 96 — Data Compression mode page	155
Table 97 — Possible boundaries and resulting sense keys due to data compression	156
Table 98 — Compression algorithm identifiers	158
Table 99 — Device Configuration mode page	159
Table 100 — EOD DEFINED field	160
Table 101 — WTRE field	161
Table 102 — REWIND ON RESET field	162
Table 103 — Medium Partition mode page	163
Table 104 — PSUM field	164
Table 105 — MEDIUM FORMAT RECOGNITION field	165
Table 106 — Read-Write Error Recovery mode page	166
Table 107 — Informational Exceptions Control mode page	168
Table 108 — TEST bit and TEST FLAG NUMBER field	168
Table 109 — Medium Configuration mode page	169
Table 110 — WORM MODE LABEL RESTRICTIONS field	170
Table 111 — WORM MODE FILEMARKS RESTRICTIONS field	170
Table 112 — Device Configuration Extension mode page	170
Table 113 — SHORT ERASE MODE field	172
Table 114 — Sequential-access device VPD page codes	172
Table 115 — Sequential-access Device Capabilities VPD page	173
Table 116 — Manufacturer-assigned Serial Number VPD page	173
Table 117 — TapeAlert Supported Flags VPD page	174
Table 118 — Automation Device Serial Number VPD page	174
Table 119 — SECURITY PROTOCOL SPECIFIC field	175
Table 120 — Tape Data Encryption In Support page	176

Table 121 — Tape Data Encryption Out Support page	176
Table 122 — Data Encryption Capabilities page	177
Table 123 — EXTDECC field	177
Table 124 — CFG_P field	178
Table 125 — Logical block encryption algorithm descriptor.....	178
Table 126 — DECRYPT_C field	179
Table 127 — ENCRYPT_C field	179
Table 128 — AVFCLP field	180
Table 129 — NONCE_C field	180
Table 130 — DKAD_C field	181
Table 131 — EEMC_C field.....	181
Table 132 — RDMC_C field	182
Table 133 — Supported Key Formats page.....	182
Table 134 — Data Encryption Management Capabilities page	183
Table 135 — Data Encryption Status page.....	184
Table 136 — PARAMETERS CONTROL field	185
Table 137 — Next Block Encryption Status page	186
Table 138 — COMPRESSION STATUS field	187
Table 139 — ENCRYPTION STATUS field.....	187
Table 140 — Random Number page	189
Table 141 — Device Service Key Wrapping Public Key page	189
Table 142 — PUBLIC KEY TYPE field	190
Table 143 — SECURITY PROTOCOL SPECIFIC field	190
Table 144 — Set Data Encryption page.....	191
Table 145 — SCOPE field.....	192
Table 146 — CEEM field.....	192
Table 147 — RDMC field	193
Table 148 — ENCRYPTION MODE field	194
Table 149 — DECRYPTION MODE field	195
Table 150 — LOGICAL BLOCK ENCRYPTION KEY FORMAT field.....	196
Table 151 — KEY field format with KEY FORMAT field set to 00h	197
Table 152 — KEY field format with KEY FORMAT field set to 01h	198
Table 153 — KEY field format with KEY FORMAT field set to 02h	198
Table 154 — PARAMETER SET field	198
Table 155 — ECIES-HC REQUIREMENTS AND PARAMETERS FOR ECIES-KEM	200
Table 156 — ECIES-HC REQUIREMENTS AND PARAMETERS FOR ECIES-DEM	200
Table 157 — SA Encapsulation page	201
Table 158 — Tape Data Encryption descriptor format.....	201
Table 159 — KEY DESCRIPTOR TYPE field.....	202
Table 160 — AUTHENTICATED field.....	202
Table 161 — Wrapped Key descriptor format.....	203
Table 162 — WRAPPED KEY DESCRIPTOR TYPE field	203
Table A.1 — TapeAlert log page parameter codes	205
Table B.1 — Security environment threats	212
Table D.1 — Sense logical block error indications for read and write operations	215
Table D.2 — INFORMATION field and position for read and write operations	217
Table D.3 — Summary of length error conditions on read type commands	219

INFORMATION TECHNOLOGY -

Small Computer System Interface (SCSI) -

Part 333: SCSI Stream Commands - 3 (SSC-3)

FOREWORD

- 1) ISO (International Organization for Standardization) and IEC (International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards. Their preparation is entrusted to technical committees; any ISO and IEC member body interested in the subject dealt with may participate in this preparatory work. International governmental and non-governmental organizations liaising with ISO and IEC also participate in this preparation.
- 2) In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.
- 3) The formal decisions or agreements of IEC and ISO on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC and ISO member bodies.
- 4) IEC, ISO and ISO/IEC publications have the form of recommendations for international use and are accepted by IEC and ISO member bodies in that sense. While all reasonable efforts are made to ensure that the technical content of IEC, ISO and ISO/IEC publications is accurate, IEC or ISO cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.
- 5) In order to promote international uniformity, IEC and ISO member bodies undertake to apply IEC, ISO and ISO/IEC publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any ISO/IEC publication and the corresponding national or regional publication should be clearly indicated in the latter.
- 6) ISO and IEC provide no marking procedure to indicate their approval and cannot be rendered responsible for any equipment declared to be in conformity with an ISO/IEC publication.
- 7) All users should ensure that they have the latest edition of this publication.
- 8) No liability shall attach to IEC or ISO or its directors, employees, servants or agents including individual experts and members of their technical committees and IEC or ISO member bodies for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication of, use of, or reliance upon, this ISO/IEC publication or any other IEC, ISO or ISO/IEC publications.
- 9) Attention is drawn to the normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.
- 10) Attention is drawn to the possibility that some of the elements of this International Standard may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

International Standard ISO/IEC 14776-333 was prepared by subcommittee 25: Interconnection of information technology equipment, of ISO/IEC joint technical committee 1: Information technology.

A list of all currently available parts of the ISO/IEC 14776 series, under the general title *Information technology – Small computer system interface (SCSI)*, can be found on the IEC web site.

This International Standard has been approved by vote of the member bodies and the voting results may be obtained from the address given on the second title page.

This publication has been drafted in accordance with the ISO/IEC Directives, Part 2.

For any information concerning Technical Committee T10 (SCSI Storage Interfaces), contact ANSI/INCITS (American National Standards Institute, <http://www.ansi.org/>).

INTRODUCTION

The SCSI Stream Commands - 3 (SSC-3) standard is divided into eight clauses:

Clause 1 is the scope.

Clause 2 enumerates the normative references that apply to this standard.

Clause 3 describes the definitions, acronyms, keywords, and conventions used in this standard.

Clause 4 describes an overview and model of the sequential-access type device.

Clause 5 describes the explicit address command set for sequential-access type devices.

Clause 6 describes the implicit address command set for sequential-access type devices.

Clause 7 describes the common command set for sequential-access type devices.

Clause 8 describes the parameters for sequential-access type devices.

The annexes provide information to assist with implementation of this standard.

INFORMATION TECHNOLOGY -

Small Computer System Interface (SCSI) -

Part 333: SCSI Stream Commands - 3 (SSC-3)

1 Scope

This part of ISO/IEC 14776 defines the command set extensions in order to facilitate operation of the sequential-access device type. This part of ISO/IEC 14776, implemented in conjunction with the requirements of the SCSI Architecture Model-4 (SAM-4) standard and the applicable clauses of the SCSI Primary Commands-4 (SPC-4) standard, fully specify the standard command set for the sequential-access device type.

This standard provides the following:

- a) it permits an application client to communicate over a SCSI service delivery subsystem, with a logical unit that declares itself to be a sequential-access device in the PERIPHERAL DEVICE TYPE field of the standard INQUIRY data (see SPC-4);
- b) it defines commands unique to the sequential-access device type; and
- c) it defines commands in order to manage the operation of the sequential-access device type.

Figure 1 shows the relationship of this standard to the other standards and related projects in the SCSI family standards as of the publication of this standard.

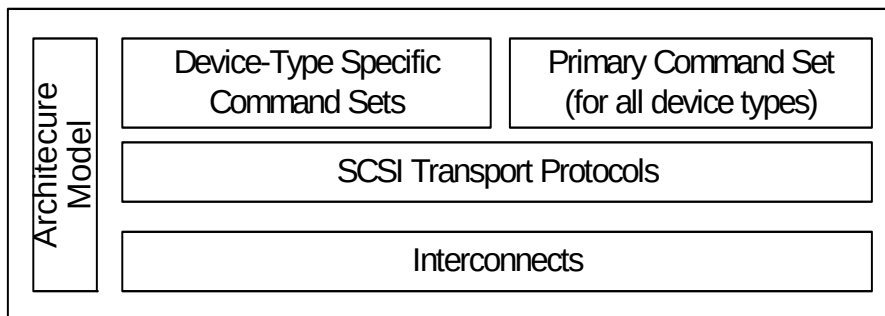


Figure 1 — SCSI document relationships

The roadmap in figure 1 is intended to show the general applicability of the documents to one another. The figure is not intended to imply a relationship such as a hierarchy, protocol stack, or system architecture. It specifies the applicability of a standard to the implementation of a given SCSI protocol.

This standard makes obsolete the following concepts from previous versions of this standard:

- a) the GAP SIZE field;
- b) RSMK and setmarks; and
- c) Attached Media Changer model.

The term SCSI is used to refer to the family of standards described in this clause.

2 Normative references

The following documents, in whole or in part, are normatively referenced in this document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 14776-414, *Information technology – Small Computer System Interface (SCSI) – Part 414: Architecture model - 4 (SAM-4)*

ISO/IEC 14776-452, *Information technology – Small Computer System Interface (SCSI) – Part 452: Primary Commands - 2 (SPC-2)*

ANSI X9.63:2001, *Public Key Cryptography for the Financial Services Industry – Key Agreement and Key Transport Using Elliptic Curve Cryptography*

ANSI INCITS 382-2004, *Information technology – SCSI Media Changer Commands - 2 (SMC-2)*

ANSI INCITS 513-2004, *Information technology – SCSI Primary Commands - 4 (SPC-4)* [T10/1731-D]

RFC 3447, *Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1*

3 Terms, definitions, acronyms, keywords and conventions

3.1 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

3.1.1

additional sense code

see SPC-4

3.1.2

application client

object that is the source of SCSI commands and task management requests

Note 1 to entry: Further definition of an application client may be found in SAM-4.

3.1.3

authorization white list

set of identifiers, typically public keys, for entities that are authorized to perform some operation

3.1.4

auxiliary memory

memory component that is accessible to the device server

Note 1 to entry: This memory is usually non-volatile and independent of the main function of the device server.

3.1.5

BOx

beginning-of-medium (see 3.1.6) or beginning-of-partition (see 3.1.7)

3.1.6

beginning-of-medium

BOM

extreme position along the medium in the direction away from the supply reel that is accessible by the device

Note 1 to entry: This position may be different from a beginning-of-partition position.

3.1.7

beginning-of-partition

BOP

position at the beginning of the permissible recording region of a partition

Note 1 to entry: This position may not coincide with a beginning-of-medium position.

3.1.8

block address mode

mode of operation that the logical unit is currently supporting (see 4.2.17)

Note 1 to entry: The block address mode is either the explicit address mode (see 3.1.22) or the implicit address mode (see 3.1.30).

3.1.9

buffered mode

mode of logical block transfer in write operations that facilitates tape streaming

Note 1 to entry: Buffered mode is specified by a non-zero value (1h or 2h) in the BUFFER MODE field in the mode parameter header (see 8.3). Buffered mode is the opposite of unbuffered mode (see 3.1.80).

3.1.10**byte**

8-bit construct

3.1.11**command**

request describing a unit of work to be performed by a device server

Note 1 to entry: A detailed definition of a command may be found in SAM-4.

3.1.12**command descriptor block****CDB**

structure used to communicate commands from an application client to a device server

Note 1 to entry: A command descriptor block may have a fixed length of up to 16 bytes or a variable length of between 12 bytes and 260 bytes.

3.1.13**common command**

command that is contained in both the explicit and implicit address command sets

3.1.14**device entity**

object in a SCSI target device that performs operations on a volume, stores parameters, and communicates between device servers

Note 1 to entry: See 4.2.4.

3.1.15**device server**

object within a logical unit that processes SCSI tasks according to the rules of task management

Note 1 to entry: See SAM-4.

3.1.16**device type**

type of device (or device model) implemented by the device server

3.1.17**early-warning****EW**

physical mark or device computed position near but logically before the end-of-partition, independent of physical direction (see 4.2.5)

3.1.18**encrypted logical block**

logical block containing data that has been subjected to a ciphering process by the device server

Note 1 to entry: Within this standard, a logical block is only considered encrypted if the ciphering process was performed by a device server.

3.1.19**end-of-data****EOD**

recorded indication that no valid logical objects are recorded between this position and end-of-partition (see 3.1.21)

Note 1 to entry: End-of-data is denoted in a format-specific manner (see 4.2.8.1).

3.1.20

end-of-medium

EOM

extreme position along the medium in the direction away from the take-up reel that is accessible by the device

Note 1 to entry: This position may be different from an end-of-partition position.

3.1.21

end-of-partition

EOP

position at the end of the permissible recording region of a partition

3.1.22

explicit address mode

mode of operation in which the logical unit is supporting the explicit address command set (see 3.1.23)

3.1.23

explicit address command set

command set in which read and writes contain positioning information

3.1.24

explicit command

command contained only in the explicit address command set (see table 22)

3.1.25

field

group of one or more contiguous bits, a part of a larger structure such as a CDB (see 3.1.12) or sense data (see 3.1.68)

3.1.26

filemark

special recorded logical object within a partition, not containing user data, that provides a segmentation scheme for the contents of a partition

3.1.27

fixed-block transfer

read or write type command with the FIXED bit set to one

3.1.28

format label

vendor-specific series of logical objects that contain information used to identify the volume or data set

3.1.29

generic command

explicit command (see 3.1.24) that is not a read type or write type command

3.1.30

implicit address mode

mode of operation in which the logical unit is supporting the implicit address command set (see 3.1.31)

3.1.31

implicit address command set

command set in which read and writes do not contain positioning information, and positioning is implied relative to the current position

3.1.32

implicit command

command contained only in the implicit address command set (see table 30)

3.1.33**information field**

command-specific field in the sense data (see SPC-4)

3.1.34**I_T nexus**

nexus between a SCSI initiator port and a SCSI target port (see SAM-4)

3.1.35**I_T nexus loss**

condition resulting from the events defined by SAM-4 in which the SCSI device performs the operations described in SAM-4 and this standard

3.1.36**I_T nexus lossevent**

SCSI transport protocol specific event that triggers I_T nexus loss as described in SAM-4

3.1.37**logical block**

logical object that is a unit of data supplied or requested by an application client

3.1.38**logical block encryption parameters**

set of parameters (see 4.2.22.14) that controls the logical block encryption and decryption processes in the device entity (see 3.1.14)

3.1.39**logical file**

zero or more logical blocks starting immediately after BOP or a filemark (see 4.2.9)

3.1.40**logical file identifier**

unique identifier, within a partition, for a logical file (see 4.2.9.2)

3.1.41**logical identifier**

logical object identifier or logical file identifier

3.1.42**logical object**

logical block or a filemark (see 4.2.8)

3.1.43**logical object identifier**

unique identifier, within a partition, for a logical object (see 4.2.8.2)

3.1.44**logical unit reset**

logical unit action in response to a logical unit reset event in which the logical unit performs the operations described in SAM-4

3.1.45**logical unit reset event**

event that triggers a logical unit reset from a logical unit as described in SAM-4

3.1.46**medium auxiliary memory****MAM**

auxiliary memory residing in a volume (e.g., a tape cartridge) that is accessible to the device service

Note 1 to entry: See SPC-4.

3.1.47

message authentication code

information used to validate the integrity of encrypted logical blocks (see 3.1.18)

3.1.48

native capacity

capacity assuming one-to-one compression (e.g., compression disabled), the medium is in good condition, and that the device recommended typical block size is used

3.1.49

nexus

relationship between two SCSI devices, and the SCSI initiator port and SCSI target port objects within those devices (see SAM-4)

3.1.50

nonce

unpredictable random value used only for a single instance or invocation of a cryptographic algorithm or protocol

3.1.51

one

logical true condition of a variable

3.1.52

overlength

incorrect-length condition that exists after processing a read command when the length of the actual logical block read exceeds the requested transfer length in the command descriptor block or the mode header block size field, whichever is appropriate

3.1.53

page

several commands use regular parameter structures that are referred to as pages

Note 1 to entry: These pages are identified with a value known as a page code.

3.1.54

partition

entire usable region for recording and reading in a volume or in a portion of a volume, defined in a vendor-specific or format-specific manner (see 4.2.7)

3.1.55

principal density code

principal density code is a density code selected by the device server

Note 1 to entry: The logical unit specifies the principal density code by reporting a DEFLT bit of one in the density support data block descriptor for supported densities in response to the REPORT DENSITY SUPPORT command (see 7.8). The selection of the principal density code is vendor specific.

3.1.56

programmable-early-warning zone

PEWZ

zone within a partition that has its EOP side established at early warning and extends towards BOP for a distance indicated by the PEWS field (see 8.3.8)

Note 1 to entry: See 4.2.6.

3.1.57**reservation loss**

event caused by the release of a reserve/release method reservation (see SPC-2) or by the transition within the device server from the state where a persistent reservation holder exists to the state where a persistent reservation holder does not exist (see SPC-4)

3.1.58**SCSI device**

device that contains one or more SCSI ports that are connected to a service delivery subsystem and supports a SCSI application protocol (see SAM-4)

3.1.59**SCSI domain**

interconnection of two or more SCSI devices and a service delivery subsystem

Note 1 to entry: A detailed definition of a SCSI domain may be found in SAM-4.

3.1.60**SCSI initiator device**

SCSI device containing application clients and SCSI initiator ports that originates device service and task management requests to be processed by a SCSI target device and receives device service and task management responses from the SCSI target devices

Note 1 to entry: When used this term refers to SCSI initiator devices or SCSI target/initiator devices that are using the SCSI target/initiator port as a SCSI initiator port.

3.1.61**SCSI initiator port**

SCSI initiator device object that acts as the connection between application clients and the service delivery subsystem through which requests, indications, responses, and confirmations are routed

Note 1 to entry: In all cases when this term is used it refers to a SCSI initiator port or a SCSI target/initiator port operating as a SCSI initiator port.

3.1.62**SCSI port**

SCSI device resident object that connects the application client, device server or task manager to the service delivery subsystem through which requests and responses are routed

Note 1 to entry: SCSI port is synonymous with port. A SCSI port is one of: a SCSI initiator port, a SCSI target port, or a SCSI target/initiator port.

3.1.63**SCSI target device**

SCSI device containing logical units and SCSI target ports that receives device service and task management requests for processing and sends device service and task management responses to SCSI initiator devices

Note 1 to entry: When used this term refers to SCSI target devices or SCSI target/initiator devices that are using the SCSI target/initiator port as a SCSI target port.

3.1.64**SCSI target port**

SCSI target device object that contains a task router and acts as the connection between device servers and task managers and the service delivery subsystem through which indications and responses are routed

Note 1 to entry: When this term is used it refers to a SCSI target port or a SCSI target/initiator port operating as a SCSI target port.

3.1.65

SCSI target/initiator device

SCSI device that has all the characteristics of a SCSI target device and a SCSI initiator device

3.1.66

SCSI target/initiator port

SCSI device resident object that has all the characteristics of a SCSI target port and a SCSI initiator port

3.1.67

security metadata

data used by security methods to enable user data to be returned in the form it existed prior to the application of the security methods (e.g., logical block encryption parameters, passwords, wrapped keys)

Note 1 to entry: Security meta-data may be used for vendor-specific security methods.

3.1.68

sense data

see SPC-4

3.1.69

sense key

see SPC-4

3.1.70

service delivery subsystem

part of an SCSI I/O system that transmits service requests to a logical unit or SCSI target device and returns logical unit or SCSI target device responses to a SCSI initiator device

Note 1 to entry: A detailed definition of a service delivery subsystem may be found in SAM-4.

3.1.71

spacing

act of positioning the medium on a sequential-access device while processing a SPACE command

3.1.72

status

one byte of response information sent from a device server to an application client upon completion of each command

Note 1 to entry: A detailed definition of status may be found in SAM-4.

3.1.73

synchronize operation

process of writing buffered logical objects to the medium (see 4.2.11)

3.1.74

tagged write sequence

one or more WRITE(16), WRITE FILEMARKS(16), or ERASE(16) commands delineated by the FCS and LCS bits (see 5.6, 5.7, and 5.2)

3.1.75

tape

medium on which data is recorded

Note 1 to entry: See 4.2.2.

3.1.76

TapeAlert

device server capability that provides detailed device diagnostic information using a standard interface

3.1.77**thread**

process in which the medium is being engaged for positioning on a suitable transport mechanism (e.g., spooled on to a take-up reel, wrapped around the surface of a helical scan drum)

Note 1 to entry: After threading is complete the tape device may begin positioning the medium to an initial position.

3.1.78**track**

contiguous line on the medium consisting of a pattern of recorded signals written by one write component

3.1.79**track group**

set of tracks that are recorded at the same time

3.1.80**unbuffered mode**

mode of operation where write data is written directly to the medium without being buffered

Note 1 to entry: Unbuffered mode is specified by a zero value (0h) in the `BUFFER MODE` field in the mode parameter header (see 8.3). Unbuffered mode is the opposite of buffered mode (see 3.1.9).

3.1.81**underlength**

incorrect-length condition that exists after processing a read command when the requested transfer length in the command descriptor block or the mode header block size field, whichever is appropriate, exceeds the length of the actual logical block read

3.1.82**unencrypted logical block**

logical block containing cleartext (i.e., logical block that has not been enciphered by the device server)

3.1.83**unthread**

part of the unloading process in which the medium is being disengaged from the suitable transport mechanism (e.g., de-spooled from a take-up reel, unwrapped from around the surface of a helical scan drum)

3.1.84**variable-block transfer**

read or write type command with the `FIXED` bit set to zero

3.1.85**vendor-specific control metadata**

vendor-specific information stored on the volume outside the user data area(s) that is used to control or specify how the volume is being used by application clients (e.g., directory information, partition information, EOD locations, copies of data stored in a vendor-specific manner, volume serial number information, number of logical blocks on the medium)

3.1.86**volume**

medium together with its physical carrier

Note 1 to entry: See 4.2.2.

3.1.87**zero**

logical false condition of a variable.

3.2 Acronyms

A-KAD	authenticated key-associated data
ADC	Automation Device Control
ALDC	Adaptive Lossless Data Compression: ISO/IEC 15200:1996
BOM	beginning-of-medium
BOP	beginning-of-partition
CbCS	capability-based command security
CDB	command descriptor block
DCLZ	Data Compression according to Lempel and Ziv: ISO/IEC 11558:1992
DEM	data encapsulation mechanism
ECC	error correction code
ECC	elliptic curve cryptography
ECDSA	Elliptic Curve Digital Signature Algorithm
ECIES	Elliptic Curve Integrated Encryption Scheme
ECMA	European Computer Manufacturers Association
EOD	end-of-data
EOM	end-of-medium
EOP	end-of-partition
EW	early-warning
HC	hybrid cipher
I/O	input-output
ID	identifier
IDRC	Improved Data Recording Capability
INCITS	InterNational Committee for Information Technology Standards
KDF	key derivation function
KEM	key encapsulation mechanism
LSB	least significant bit
M	mandatory
M-KAD	metadata key-associated data
MA	MAC algorithm
MAC	message authentication code
MAM	medium auxiliary memory
MSB	most significant bit
NA	not applicable
O	optional
PEWZ	programmable-early-warning zone
RSA	Rivest-Shimier-Adleman
Rsvd	reserved
SA	security association
SAM-4	SCSI Architecture Model - 4
SC	symmetric cipher
SCSI	Small Computer System Interface
SDK	supplemental decryption keys
SMC-2	SCSI Media Changer Commands - 2
SPC-2	SCSI Primary Commands - 2
SPC-4	SCSI Primary Commands - 4
SSC-3	SCSI Stream Commands - 3 (this standard)
U-KAD	unauthenticated key-associated data

3.3 Keywords

3.3.1 expected

A keyword used to describe the behavior of the hardware or software in the design models assumed by this standard. Other hardware and software design models may also be implemented.

3.3.2 invalid

A keyword used to describe an illegal or unsupported field or code value. Receipt of an invalid field or code value shall be reported as an error.

3.3.3 ignored

A keyword used to describe an unused field or code value. The contents or value of an ignored field or code value shall not be examined by the receiving SCSI device and may be set to any value by the transmitting SCSI device.

3.3.4 mandatory

A keyword indicating an item that is required to be implemented as defined in this standard.

3.3.5 may

A keyword that specifies flexibility of choice with no implied preference (equivalent to "may or may not").

3.3.6 may not

A keyword that specifies flexibility of choice with no implied preference (equivalent to "may or may not").

3.3.7 obsolete

A keyword indicating that an item was defined in prior SCSI standards but has been removed from this standard.

3.3.8 optional

A keyword that describes features that are not required to be implemented by this standard. However, if any optional feature defined by this standard is implemented, then it shall be implemented as defined in this standard.

3.3.9 reserved

A keyword referring to fields and code values that are set aside for future standardization. A reserved field shall be set to zero, or in accordance with a future extension to this standard. Recipients are not required to check reserved fields for zero values. Receipt of reserved code values in defined fields shall be reported as an error.

3.3.10 shall

A keyword indicating a mandatory requirement. Designers are required to implement all such mandatory requirements to ensure interoperability with other products that conform to this standard.

3.3.11 should

A keyword indicating flexibility of choice with a strongly preferred alternative; equivalent to the phrase "it is strongly recommended".

3.3.12 vendor specific

Items (e.g., fields, code values, etc.) that are not defined by this standard and may be vendor defined.

3.4 Editorial conventions

Certain words and terms used in this standard have a specific meaning beyond the normal English meaning. These words and terms are defined either in clause 3.1 or in the text where they first appear. Names of

commands, statuses, sense keys, additional sense codes, and additional sense code qualifiers are in all uppercase (e.g., REQUEST SENSE). Lowercase is used for words having the normal English meaning.

The names of fields are in small uppercase (e.g., ALLOCATION LENGTH). When a field name is a concatenation of acronyms, uppercase letters may be used for readability (e.g., NORMACA). Normal case is used when the contents of a field are being discussed. Fields containing only one bit are usually referred to as the name bit instead of the name field.

Numbers that are not immediately followed by lower-case b or h are decimal values.

Numbers immediately followed by lower-case b (xxb) are binary values.

Numbers or upper case letters immediately followed by lower-case h (xxh) are hexadecimal values.

The most significant bit of a binary quantity is shown on the left side and represents the highest algebraic value position in the quantity.

If a field is specified as not meaningful or the field is to be ignored, the entity that receives the field shall not take any action based on the value of that field.

Lists sequenced by letters (e.g., a-red, b-blue, c-green) show no priority relationship between the listed items. Numbered lists (e.g., 1-red, 2-blue, 3-green) show a priority ordering between the listed items.

If a conflict arises between text, tables, or figures, the order of precedence to resolve the conflicts is text; then tables; and finally figures. Not all tables or figures are fully described in the text. Tables show data format and values. Notes do not constitute any requirements for implementors.

This standard uses the following conventions for representing decimal numbers:

- a) the decimal separator (i.e., separating the integer and fractional portions of the number) is a period;
- b) the thousands separator (i.e., separating groups of three digits in a portion of the number) is a space; and
- c) the thousands separator is used in both the integer portion and the fraction portion of a number.

Table 1 shows some examples of decimal numbers represented using various conventions.

Table 1 — Numbering conventions examples

ISO/IEC	US	This standard
0,6	0.6	0.6
3,141 592 65	3.14159265	3.141 592 65
1 000	1,000	1 000
1 323 462,95	1,323,462.95	1 323 462.95

3.5 Notation conventions – State diagrams

All state diagrams use the notation shown in figure 2.

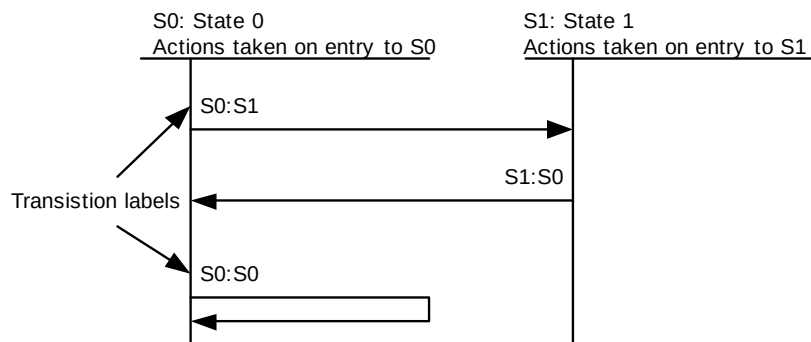


Figure 2 — Example state diagram

The state diagram is followed by a list of the state transitions, using the transition labels. Each transition is described in the list with particular attention to the conditions that cause the transition to occur and special conditions related to the transition. Using figure 2 as an example, the transition list might read as follows:

Transition S0:S1: This transition occurs when state S0 is exited and state S1 is entered.

Transition S1:S0: This transition occurs when state S1 is exited and state S0 is entered.

Transition S0:S0: This transition occurs when state S0 transitions to itself. It is particularly important to note that the actions taken whenever state S0 is entered are repeated every time this transition occurs.

A system specified in this manner has the following properties:

- time elapses only within discrete states;
- state transitions are logically instantaneous; and
- every time a state is entered, the actions of that state are started. Note that this means that a transition that points back to the same state restarts the actions from the beginning.

4 General concepts

4.1 Overview

The sequential-access device type has the characteristic of primarily handling data in a sequential manner (i.e., a stream). This does not limit the device's ability to position randomly within the data although a sequential-access device is not truly random-access.

This standard describes two modes and associated command sets for communicating with a sequential-access device:

- a) implicit address mode. Commands to read and write on a sequential-access device do not contain any positioning information fields. Instead, the device position is normally determined by previous commands; and
- b) explicit address mode. Commands to read and write on a sequential-access device contain positioning information fields.

Commands are available for absolute and relative positioning. Writing to a sequential-access device may cause all data starting at the point at which the data is written to be invalidated. There may be restrictions on where write operations may be initiated. Reading or writing data as a long string of data, as in a stream, tends to be the most efficient.

4.2 Sequential-access device model

4.2.1 Sequential-access device model overview

Sequential-access devices are described herein from the point of view of a tape device. However, other implementations are not precluded.

Sequential-access devices optimize their use in storing or retrieving user data in a sequential manner. Since access is sequential, position changes typically take a long time, when compared to random-access devices.

4.2.2 Physical elements

The medium for tape devices consists of various widths and lengths of a flexible substrate coated with a semi-permanent magnetic material. The medium may be spooled onto single reels or encapsulated into cartridges containing both a supply reel and a take-up reel. Several American National Standards exist covering the construction of reels and cartridges for interchange as well as recording techniques for many of the format or density combinations.

For a sequential-access device, a medium exists between two reels, the supply reel and take-up reel. The read/write mechanism may only access the medium between the reels. As the medium is taken out of one reel, it passes by the read/write mechanism and into the other reel. Transferring data as a stream is most efficient, since the medium may traverse the read/write mechanism producing a flow of data. To position to a given point requires moving the medium until the appropriate position is found.

The medium has two physical attributes called beginning-of-medium (BOM) and end-of-medium (EOM). Beginning-of-medium is at the end of the medium that is attached to the take-up reel. End-of-medium is at the end of the medium that is attached to the supply reel. In some cases, the medium is permanently affixed to one or both of the reel hubs. BOM or EOM is not required to be related to BOP or EOP of any partition.

A volume is composed of the medium, its physical carrier (e.g., reel, cartridge, cassette), and MAM if present. Volumes have an attribute of being mounted or demounted on a suitable transport mechanism.

A volume is defined as mounted when the device is physically capable of processing commands that cause the medium to be moved. A volume is defined as demounted when it is being loaded, threaded, unloaded, unthreaded, or when not attached to the device.

A logical unit is defined as ready when medium access commands are able to be processed. A logical unit is defined as not ready when no volume is mounted or whenever any medium access command reports CHECK CONDITION status and a NOT READY sense key. The logical unit is not ready during the transition from mounted to demounted, or demounted to mounted. Devices may have a physical control that places the device in a not ready state even when a volume is mounted.

As shown in figure 3, a portion of the physical length of medium is not usable for recording data. For most volumes, a length of the medium is reserved between the take-up reel and the beginning-of-medium, and between the end-of-medium position and the supply reel.

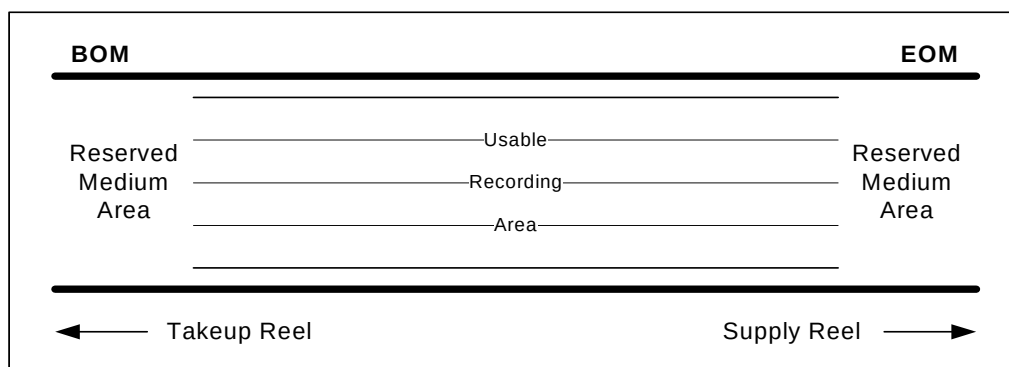


Figure 3 — Typical volume layout

The position on the medium where a pattern of recorded signals may be written by one write component is called a track (see figure 4). A device may write or read from one or more tracks at a time, depending on the format.

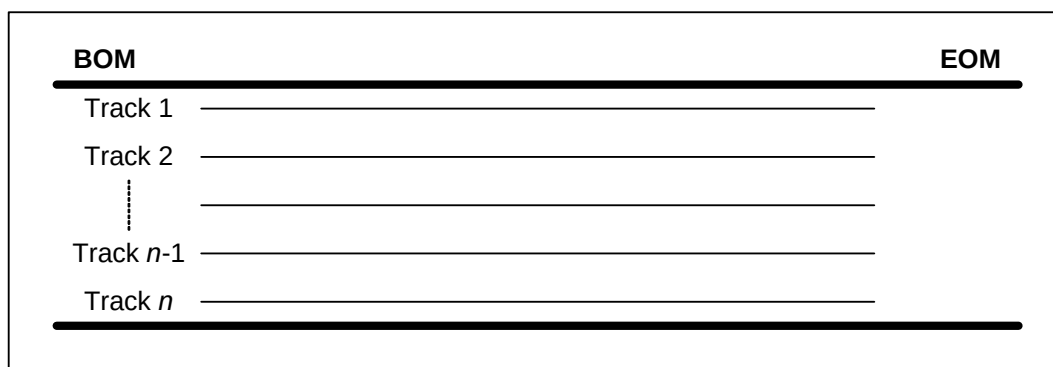


Figure 4 — Typical medium track layout

On a new volume, recording of one or more tracks begins after mounting the volume and moves from beginning-of-medium toward end-of-medium. The number of tracks written at one time is called a track group (TrkGrp). Track groups may be used by any recording format. For recorded volumes, reading in the forward direction follows the same course of tracks when writing.

In serpentine recording, not all tracks are recorded at the same time. At the end-of-medium or beginning-of-medium, the device reverses direction and begins recording the next track group. The process of reversing direction and recording the next track group may be repeated until all track groups are recorded. For serpentine devices that record only one track at a time, each physical track represents one track group (see figure 5).

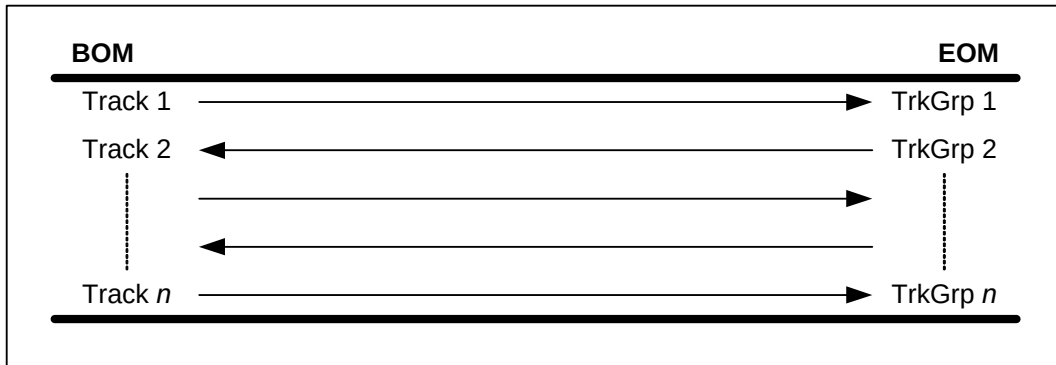


Figure 5 — Serpentine recording example

Some multi-track devices have only one track group, using a parallel storage format that supports the simultaneous recording of all available tracks (see figure 6).

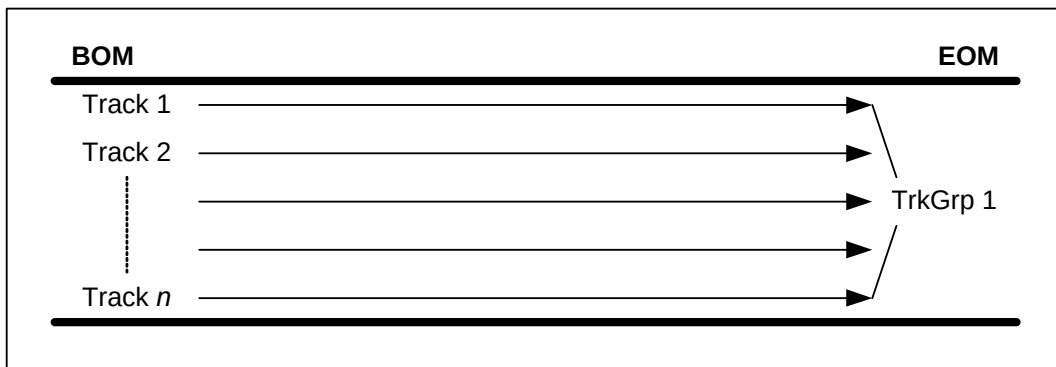


Figure 6 — Parallel recording example

The serpentine and parallel recording formats shown in the previous examples define tracks as longitudinal patterns of recorded information. One other storage format used by some devices records tracks diagonally across the medium. One or more tracks may be recorded at the same time. This recording technique is known as helical scan (see figure 7).

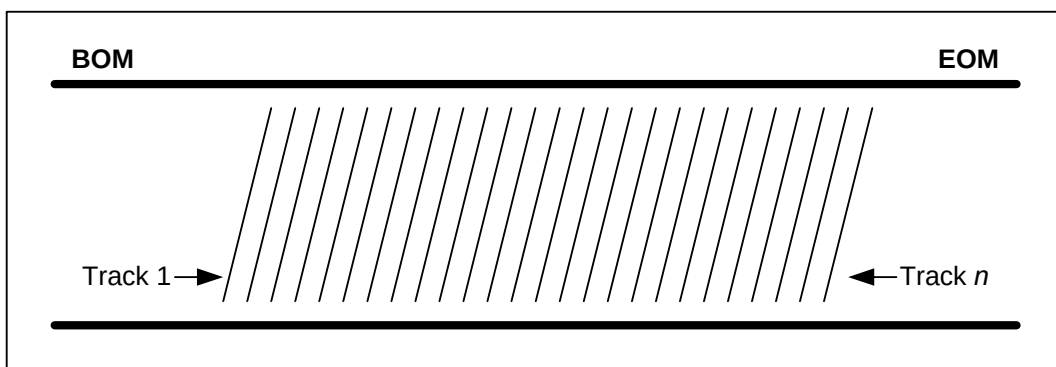


Figure 7 — Helical scan recording example

For most recording formats, a format identification in the form of a tone burst or some other recognizable pattern is recorded outside the user data area. The format identification is an attribute of a volume used for interchange purposes and is defined in applicable standards.

4.2.3 Removable volumes

A sequential-access device may support removable volumes (see 4.2.2). The sequential access device indicates that volumes are removable by setting the RMB bit to one in its standard INQUIRY data (SPC-4).

A removable volume has an attribute of being mounted or demounted on a suitable transport mechanism in a sequential-access device (see 4.2.2).

If the volume in a sequential-access device is removable, and a volume transitions from demounted to mounted, then the device server shall establish a unit attention condition with the additional sense code set to NOT READY TO READY CHANGE, MEDIUM MAY HAVE CHANGED.

The LOAD UNLOAD command (see 7.2) is used to mount or demount the volume.

The PREVENT ALLOW MEDIUM REMOVAL command (see 7.4) allows an application client to restrict the demounting of the removable volume.

4.2.4 Device entity

A sequential-access device contains one or more device entities. A device entity provides storage for values that are shared between multiple device servers and performs operations upon the volume (e.g., loading, unloading, positioning, writing and reading the medium, and reading and writing medium auxiliary memory).

The device entity is controlled by various other entities, which may include:

- a) one or more SCSI device servers (e.g., where a device entity is associated with an automation device that performs medium movement, both a device server that implement the command set defined in this standard and a device server that implements another command set such as ADC-2 may control the device);
- b) an operator interface;
- c) a management interface; and
- d) a media changer.

A media changer may control the device entity by inserting a volume into or removing a volume from the device entity. When inserting a volume into or removing a volume from the device entity, the operator acts in the role of a media changer.

These entities perform operations that change various attributes of the device entity. These attributes affect the operations on a volume. Figure 8 shows in UML notation an example of the entities in a SCSI target device, and shows the attributes of the device entity.

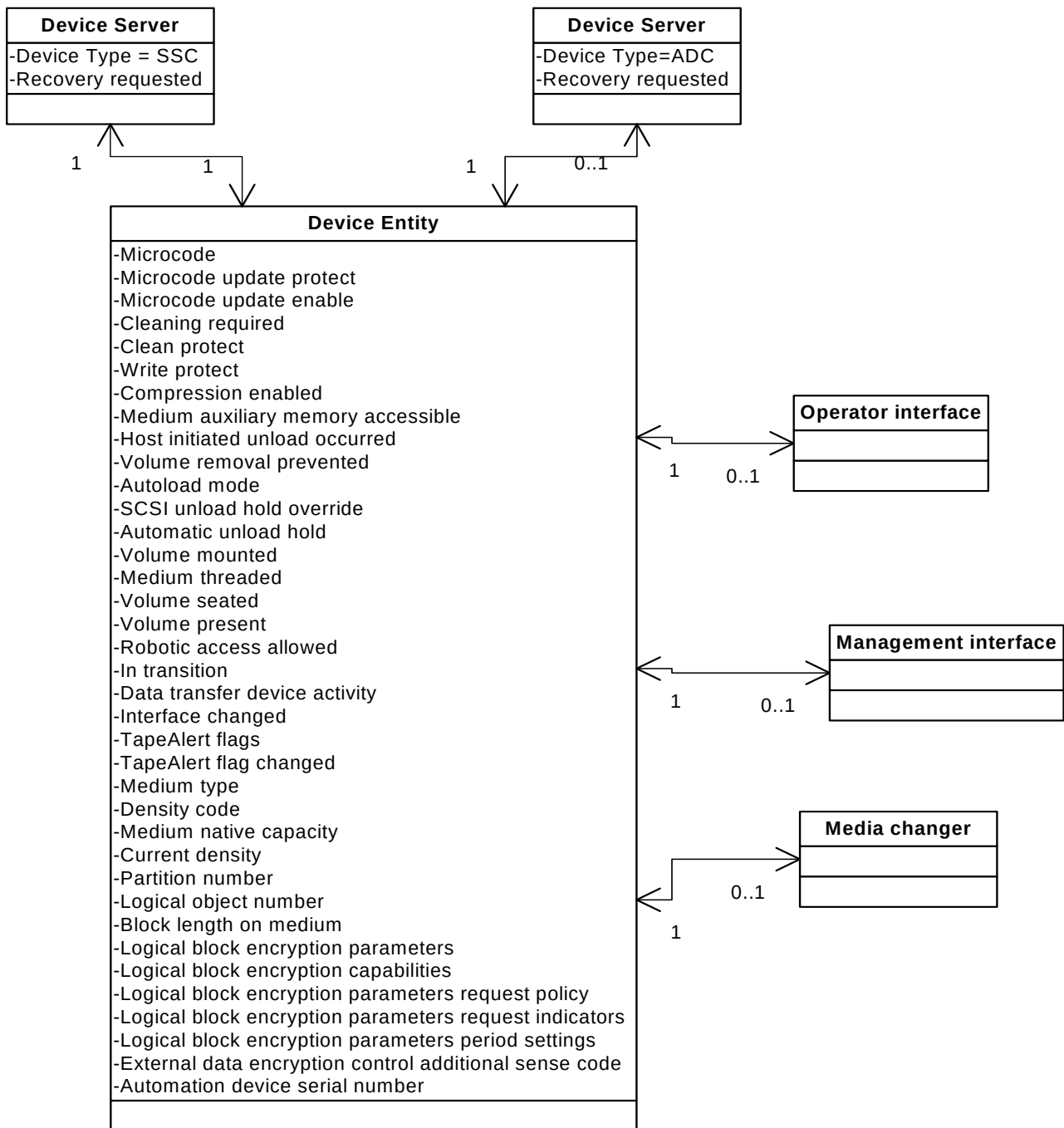


Figure 8 — UML example of SCSI target device and device entity

Table 2 specifies the standard that defines each attribute shown in figure 8.

Table 2 — Device entity attributes

Attribute	Reference
Microcode	SPC-4
Microcode update protect	ADC-3
Microcode update enable	ADC-3
a) Medium native capacity is the value reported in the CAPACITY field of the density support data block descriptor when the MEDIA bit is one, and a SET CAPACITY command has not been used to affect the capacity of the medium.	

Table 2 — Device entity attributes (Continued)

Attribute	Reference
Cleaning required	ADC-3
Clean protect	ADC-3
Write protect	ADC-3
Compression enabled	ADC-3
Medium auxiliary memory accessible	ADC-3
Host initiated unload occurred	ADC-3
Volume removal prevented	ADC-3
Autoload mode	SPC-4
SCSI unload hold override	ADC-3
Automatic unload hold	ADC-3
Volume mounted	ADC-3
Medium threaded	ADC-3
Volume seated	ADC-3
Volume present	ADC-3
Robotic access allowed	ADC-3
In transition	ADC-3
Data transfer device activity	ADC-3
Interface changed	ADC-3
TapeAlert flags	Table 10
TapeAlert flag changed	ADC-3
Medium type	7.8.4
Density code	8.2.4.3
Medium native capacity ^a	7.8.3
Current density	ADC-3
Partition number	7.6.3
Logical object number	7.6.3
Block length on medium	SPC-4
Logical block encryption parameters	4.2.22.14
Logical block encryption capabilities	4.2.22.7
Logical block encryption parameters request policy	4.2.23.3.2
Logical block encryption parameters request indicators	4.2.23.3.3
Logical block encryption parameters period settings	4.2.23.3.4
External data encryption control additional sense code	4.2.23.5
Automation device serial number	8.4.5
a) Medium native capacity is the value reported in the CAPACITY field of the density support data block descriptor when the MEDIA bit is one, and a SET CAPACITY command has not been used to affect the capacity of the medium.	

4.2.5 Early-warning

When writing, the application client needs an indication that it is approaching the end of the permissible recording area when moving in a direction toward the end of the partition (see 4.2.7). This position, called early-warning (EW), is typically reported to the application client at a position early enough for the device to write any buffered logical objects to the medium while still leaving enough room for additional recorded logical objects (see figure 9). Some American National Standards include physical requirements for a marker placed on the medium to be detected by the device as early-warning.

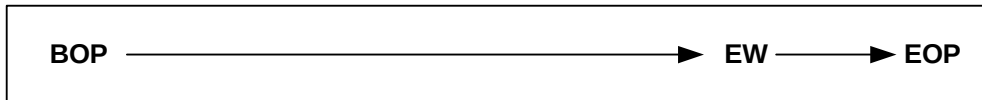


Figure 9 — Early-warning example

Devices are expected to report early warning to the application client when sufficient recording space is nominally available before EOP to record logical objects in the object buffer(s) and some additional logical objects. A logical concept of early-warning may be required to signal the application client at an appropriate location prior to the physical marker, particularly for devices that implement object buffers.

4.2.6 Programmable early warning

When writing, the application client may need an indication prior to early warning (see 4.2.5) to allow for the application client to prepare to be ready for early warning (e.g., flush buffers in the application client).

Application clients that need this indication may set the PEWS field (see 8.3.8) to a value that creates a PEWZ of the specified size.

Application clients that need this indication may request the device server to create a zone called the programmable-early-warning zone (PEWZ) (see figure 10) by setting the PEWS field (see 8.3.8) to the requested size of the PEWZ. The EOP side of PEWZ is established at early-warning and extends towards BOP for a distance indicated by the PEWS field.

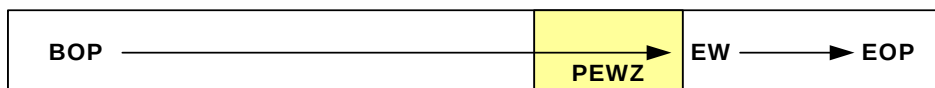


Figure 10 — Programmable early warning example

The REW bit in the Device Configuration mode page (see 8.3.3) shall have no effect on the device server behavior in the PEWZ.

The device server shall return CHECK CONDITION status, with the sense key set to NO SENSE, the EOM bit set to one and the additional sense code set to PROGRAMMABLE EARLY WARNING DETECTED at the completion of a command that caused the medium to transition into the PEWZ if that command is:

- a) WRITE(6);
- b) WRITE(16);
- c) WRITE FILEMARKS(6); or
- d) WRITE FILEMARKS(16).

Encountering the PEWZ shall not cause the device server to perform a synchronize operation or terminate the command. If processing this command results in any other exception condition except early-warning, the CHECK CONDITION status associated with that exception condition shall be reported instead. If early-warning is crossed prior to the PROGRAMMABLE EARLY WARNING DETECTED additional sense code

being reported, the PROGRAMMABLE EARLY WARNING DETECTED additional sense shall be reported before the early-warning CHECK CONDITION.

If the PROGRAMMABLE EARLY WARNING DETECTED additional sense code was not reported, the next write in PEWZ or beyond early-warning that would otherwise complete with GOOD status, shall return the programmable-early-warning CHECK CONDITION instead.

If the PEWZ is entered and exited on the BOP side before the PROGRAMMABLE EARLY WARNING DETECTED additional sense code is returned, the device server shall not report CHECK CONDITION status with the additional sense code set to PROGRAMMABLE EARLY WARNING DETECTED.

4.2.7 Partitions within a volume

Partitions consist of one or more non-overlapped logical volumes, each with its own beginning and ending points, contained within a single physical volume. Each partition (n) within a volume has a defined beginning-of-partition (BOP n), an early-warning position (EW n), and an end-of-partition (EOP n).

All volumes have a minimum of one partition called partition 0, the default data partition. For devices that support only one partition, the beginning-of-partition zero (BOP 0) may be equivalent to the beginning-of-medium and the end-of-partition zero (EOP 0) may be equivalent to the end-of-medium. For devices that support more than one partition, they shall be numbered sequentially starting with zero (i.e., beginning-of-partition 0).

When a volume is mounted, it is logically positioned to the beginning of the default data partition (BOP 0). When a REWIND command is received in any partition (x), the device positions to the beginning-of-partition of the current partition (BOP x).

Partitions on a volume may be recorded in any order and use any partition number unique to the physical volume. It is sufficient for a device to be able to locate a partition, given its partition number, or to determine that it does or does not exist on the volume. For interchange, information about which partitions are present on a volume may be stored on the volume in a format specified area, possibly unavailable to the application client, or the information may be an intrinsic attribute of the device implementation.

Figure 11 shows a possible partition implementation for a four-track serpentine recording device, assuming that each track group defines a partition.

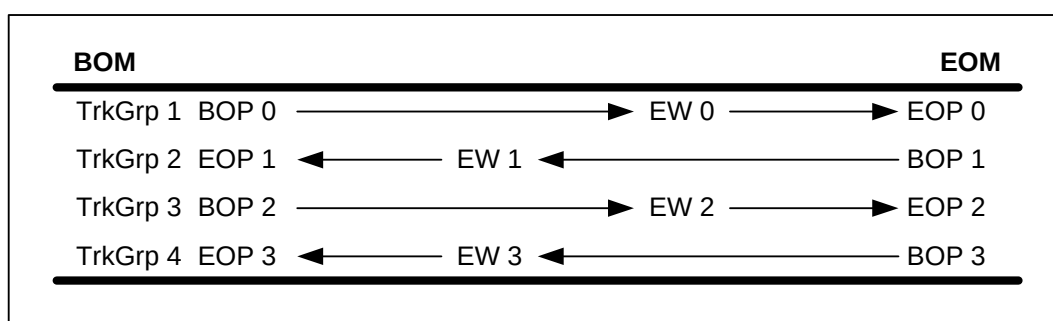


Figure 11 — Partitioning example - one partition per track group

Another possible partition implementation for this four-track serpentine recording device is shown in figure 12, using two track groups to define each partition.

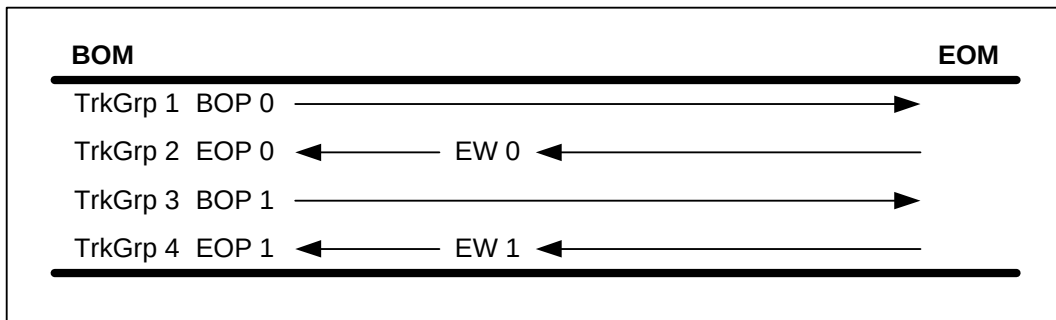


Figure 12 — Partitioning example - one partition per two track groups

The previous examples show the beginning and ending points for a partition aligned with physical bounds of the medium. This is not required for partitioning. It is sufficient for a device to be able to locate to and stay in any partition bounded by a BOP x and EOP x . In this case, a device-recognizable attribute could be used to delineate the partitions. Figure 13 shows a possible two-partition implementation for a device with only one track group.

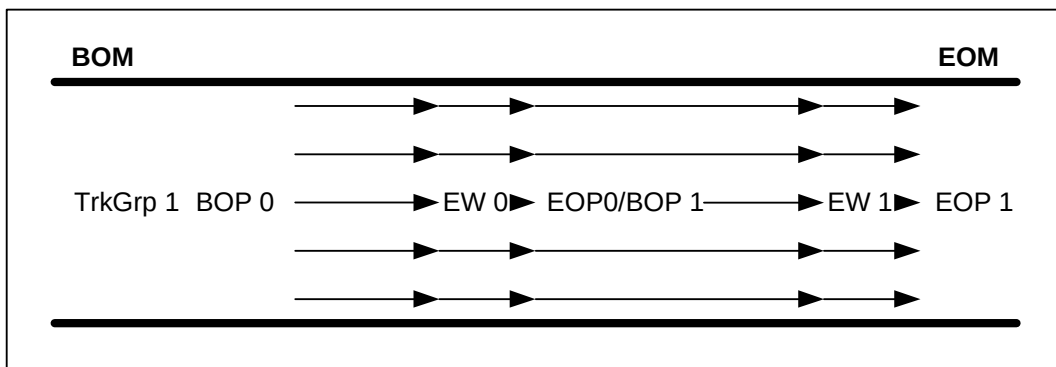


Figure 13 — Partitioning example - two partitions per track group

Three methods are defined in the MODE SENSE and MODE SELECT commands for managing partitions:

- device-defined fixed locations;
- device-defined based on an application client supplied number of partitions and a vendor-specific allocation algorithm; and
- definition by partition number and capacity by an application client.

4.2.8 Logical objects

4.2.8.1 Logical objects within a partition

The area between BOP n and EOP n on a recorded volume contains application client accessible logical objects. Logical objects are controlled and transferred between the application client and the medium using commands defined in this standard. Each logical object shall have a logical object identifier that is unique within a partition.

The basic unit of data transferred by an application client is called a logical block. Logical blocks are stored according to the specifications of the format for the volume and may be recorded as portions of one or more physical blocks on the medium. The mapping between physical and logical blocks is the responsibility of the device server.

Filemarks are special recorded logical objects not containing user data. Proper recording and detection of filemarks is the responsibility of the device server. Application clients traditionally use filemarks to separate data.

groups of user data from each other. Since some format standards do not define an explicit EOD, operating system software has often used conventions with filemarks to represent an EOD indication. In some implementations, the device's EOD definition may be specified by the application client using the Device Configuration mode page (see 8.3.3).

After writing data from BOP n , the medium is considered to contain a contiguous grouping of logical objects. Depending on the format, blank medium may be treated as an end-of-data indication, an error recovery area, or an unrecoverable medium error causing an interchange error. Unrecorded volumes, new or erased, may exhibit blank medium characteristics if an attempt is made to read or space the volume before data has been written.

A sequential-access device may be capable of supporting fixed-block transfers or variable-block transfers. The concept of fixed or variable mode for writing and reading logical blocks only specifies the method by which the application client specifies the size of a logical block for transfer, and not the method of recording physical blocks on the medium. However, a device that supports only fixed-length physical blocks may only be capable of supporting logical blocks of the same length. The length of a logical block is always described in bytes. Refer to the READ BLOCK LIMITS command (see 7.5) for additional information about fixed-block transfers and variable-block transfers.

4.2.8.2 Logical object identifier

The logical object identifier value shall be a sequentially increasing number assigned to each logical object recorded in the partition starting with zero for the recorded logical object at BOP.

The READ POSITION command (see 7.6) may be used to determine a logical object identifier and the application client may use this value with a LOCATE command (see 6.3 and 7.3) or an explicit command (see clause 5) to position to the same location at some future time.

4.2.9 Logical files

4.2.9.1 Logical files within a partition

Application clients may use filemarks to separate groups of user data into logical files. A logical file shall contain zero or more logical blocks and shall begin with the next logical object following a filemark or BOP. Each logical file shall have a logical file identifier that is unique within the partition.

4.2.9.2 Logical file identifier

The logical file identifier value shall be a sequentially increasing number assigned to each logical file recorded in the partition starting with zero for the recorded logical file beginning at BOP.

The READ POSITION command may be used to determine a logical file identifier and the application client may use this value with a LOCATE(16) command to position to the BOP side of the same logical file at some future time.

4.2.10 Object buffering

A device may contain a temporary storage area capable of holding one or more logical objects - an object buffer. A device object buffer may include any combination of logical objects in the process of being written to the medium, or it may contain read-ahead logical objects transferred from the medium.

A device with an object buffer may be capable of operating in either a buffered mode or an unbuffered mode. A device with no object buffer operates only in unbuffered mode. Either term is only applicable to the manner in which the device manages information to be written to the medium. Buffered mode is not applicable during read commands, regardless of whether read data passes through an object buffer.

A device operating in buffered mode may return GOOD status for write operations when all logical objects have been successfully transferred from the application client into the device object buffer. For devices operating in unbuffered mode, GOOD status is not returned until all requested logical objects are successfully recorded on the medium.

When issuing a buffered WRITE FILEMARKS command with the immediate bit set to one, GOOD status shall be returned as soon as the command is validated. For a WRITE FILEMARKS command with the immediate bit set to zero, the device server shall perform a synchronize operation (see 4.2.11).

If an unrecoverable write error occurs while in buffered mode, the device generates an error condition to the current active command. If no command is active, the error may be reported on the next applicable operation as a deferred error (see SPC-4). For some implementations auto contingent allegiance may be required. Refer to SAM-4 for a description auto contingent allegiance.

The READ POSITION command may be used to determine the number and storage space of buffered logical objects not written before the unrecoverable error was encountered.

A device that encounters an unrecoverable error during a read-ahead operation shall not report the error unless the logical object in error is accessed by an application client.

Prior to performing some commands, the device server shall perform a synchronize operation (see 4.2.11) as stated in table 22 and table 30.

4.2.11 Synchronize operation behavior

As stated in table 22 and table 30, some commands may require the device server to perform a synchronize operation (see 3.1.73). If a command requires a synchronize operation, the synchronize operation shall be performed prior to initiating any command-specific operations. Upon successful completion of the synchronize operation, no logical objects shall remain in the object buffer that have not been written to the medium. A synchronize operation shall have no effect on an object buffer that contains only read-ahead logical objects, or logical objects that have already been successfully written to the medium.

For a WRITE BUFFER command specifying modes 4, 5, 6, or 7 (download microcode operations), the device server shall perform a synchronize operation before performing the download operation.

For a MODE SELECT command specifying the Medium Partition mode page, the device server shall perform a synchronize operation before the logical unit partitions the volume.

For a SEND DIAGNOSTICS command, the device server shall perform a synchronization operation before any diagnostic tests that may affect the buffered logical objects, medium, or logical position, are initiated.

4.2.12 Direction and position definitions

For sequential-access devices, positioning has the connotation of logically being in, at, before, or after some defined place within a volume. Positioning requires that the position is capable of being repeated under the same circumstances. The orientation of usage for the four words (in, at, before, or after) is in one direction, from BOP x toward EOP x . All positioning defined below is worded from this perspective. Devices without object buffers have some physical position that relates to these logical positions. However, these definitions do not require the medium to have a physical position equivalent to the logical position unless explicitly stated.

The forward direction is defined as logically progressing from BOP x toward EOP x . The reverse direction is defined as logically progressing from EOP x toward BOP x . In serpentine devices, the logical forward or reverse direction has an alternating relationship to the physical motion of the medium.

The concept of being in some position means not being outside a defined region. The definition allows the position to be on the boundary of a defined region. When a volume is first mounted, the logical position is always at the beginning of the default data partition (BOP 0). Whenever a volume is mounted and the medium motion is stopped, the position is in some partition. While moving between partitions, there is no stable position.

The concept of being at some position specifies being positioned to a logical or physical extremity of a partition. A sequential-access device may be positioned at BOM, at BOP x , at EOD, at EOP x , or EOM, since these are stable positions at extremities of a partition.

The concept of being before some position specifies that there is some logical object or other defined point that may be encountered when moving toward EOP x , if the proper commands are issued. Being positioned before a particular logical block means that if the device receives a valid READ command, the logical block is transferred to the application client. This position may also be before EW x and EOP x , since these are defined points within any partition. However, if data has not been written to the end-of-partition, these points may not be accessible by the SCSI initiator device.

The concept of being after some position specifies that there is some logical object or other defined point on the BOP x side of the current position that may be encountered if the proper commands are issued. When a READ command for a single logical block has been successfully processed, the logical position is after the transferred logical block.

4.2.13 Error reporting

4.2.13.1 Overview

Sequential-access devices compliant with this standard shall support both the fixed and descriptor sense data formats (see SPC-4). If fixed format sense data is specified, but the INFORMATION field value exceeds the maximum value allowed in the fixed format sense data, the VALID bit shall be set to zero. Refer to SPC-4 for a description of the sense data VALID bit and INFORMATION field contained in the REQUEST SENSE sense data. In addition, this standard describes the use of the INFORMATION field specific to the sequential-access device type.

4.2.13.2 Stream commands sense data descriptor

Table 3 defines the stream commands sense data descriptor for sequential-access devices.

Table 3 — Stream commands sense data descriptor

Bit Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE (04h)							
1	ADDITIONAL LENGTH (02h)							
2	Reserved							
3	FILEMARK	EOM	ILI	Reserved				

See SPC-4 for a description of the DESCRIPTOR TYPE and ADDITIONAL LENGTH fields.

See the READ(16) (see 5.3), READ(6) (see 6.4), SPACE(6) (see 6.6), RECOVER BUFFERED data (see 7.7), and SPACE(16) (see 7.11) commands for a description of FILEMARK bit usage.

See the READ(16) (see 5.3), READ REVERSE(16) (see 5.4), WRITE(16) (see 5.6), WRITE FILEMARKS(16) (see 5.7), LOCATE(10) (see 6.3), READ(6) (see 6.4), READ REVERSE(6) (see 6.5), SPACE(6) (see 6.6), WRITE(6) (see 6.8), WRITE FILEMARKS(6) (see 6.9), LOCATE(16) (see 7.3), RECOVER BUFFERED data (see 7.7), and SPACE(16) (see 7.11) commands, and the Device Configuration mode page (see 8.3.3) for end-of-medium (EOM) bit usage.

See the READ(16) (see 5.3) and READ(6) (see 6.4) commands and the Data Compression mode page (see 8.3.2) for incorrect length indicator (ILI) bit usage.

4.2.13.3 Information sense data descriptor

If reporting descriptor format sense data, the device server shall include an information sense data descriptor (see table 4) for a unit attention condition established:

- by a TapeAlert specific informational exception condition; or
- by a TapeAlert flag meeting its threshold criteria if the corresponding ETC bit in the TapeAlert log parameter is set to one.

The information sense data descriptor format is specified in table 4.

Table 4 — Information sense data descriptor

Bit Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE (00h)							
1	ADDITIONAL LENGTH (0Ah)							
2	VALID (1b)	Reserved						
3	Reserved							
4	FLAG01h	FLAG02h	FLAG03h	FLAG04h	FLAG05h	FLAG06h	FLAG07h	FLAG08h
5	FLAG09h	FLAG0Ah	FLAG0Bh	FLAG0Ch	FLAG0Dh	FLAG0Eh	FLAG0Fh	FLAG10h
6	FLAG11h	FLAG12h	FLAG13h	FLAG14h	FLAG15h	FLAG16h	FLAG17h	FLAG18h
7	FLAG19h	FLAG1Ah	FLAG1Bh	FLAG1Ch	FLAG1Dh	FLAG1Eh	FLAG1Fh	FLAG20h
8	FLAG21h	FLAG22h	FLAG23h	FLAG24h	FLAG25h	FLAG26h	FLAG27h	FLAG28h
9	FLAG29h	FLAG2Ah	FLAG2Bh	FLAG2Ch	FLAG2Dh	FLAG2Eh	FLAG2Fh	FLAG30h
10	FLAG31h	FLAG32h	FLAG33h	FLAG34h	FLAG35h	FLAG36h	FLAG37h	FLAG38h
11	FLAG39h	FLAG3Ah	FLAG3Bh	FLAG3Ch	FLAG3Dh	FLAG3Eh	FLAG3Fh	FLAG40h

The VALID bit shall be set to one.

An active TapeAlert flag has the corresponding FLAGXX bit set to one. An inactive TapeAlert flag has the corresponding FLAGXX bit set to zero.

4.2.13.4 Error conditions

If any of the conditions specified in table 5 occur during the processing of a command or if a deferred error prevented the command from processing, the device server shall terminate the command 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.

Table 5 specifies some error conditions and the applicable sense keys. Table 5 does not provide a complete list of all conditions that may cause the CHECK CONDITION status.

Table 5 — Error conditions and sense keys

Condition	Sense key
Unsupported option requested.	ILLEGAL REQUEST
Logical unit reset, I_T nexus loss, or volume change since last command from this I_T nexus.	UNIT ATTENTION
Self diagnostic failed.	HARDWARE ERROR
Unrecovered read error.	MEDIUM ERROR HARDWARE ERROR
Recovered read or write error.	RECOVERED ERROR
Attempt a WRITE, READ, READ REVERSE, VERIFY, or RECOVER BUFFERED data command with the FIXED ^a bit set to zero and variable-block transfers are not supported.	ILLEGAL REQUEST
a) See the READ(16) (see 5.3), READ REVERSE(16) (see 5.4), VERIFY(16) (see 5.5), WRITE(16) (see 5.6), READ(6) (see 6.4), READ REVERSE(6) (see 6.5), VERIFY(6) (see 6.7), WRITE(6) (see 6.8), and RECOVER BUFFERED data (see 7.7) commands for a description of the FIXED bit.	

Table 5 — Error conditions and sense keys (Continued)

Condition	Sense key
Attempt a WRITE, READ, READ REVERSE, VERIFY, or RECOVER BUFFERED data command with the FIXED ^a bit set to zero and requested block length is not supported.	ILLEGAL REQUEST
Attempt a WRITE, READ, READ REVERSE, VERIFY, or RECOVER BUFFERED data command with the FIXED ^a bit set to one and MODE SENSE block length set to zero.	ILLEGAL REQUEST
Attempt to perform an erase, format, partition, set capacity, or write-type operation on write protected medium.	DATA PROTECT
Deferred write error.	MEDIUM ERROR VOLUME OVERFLOW HARDWARE ERROR
Medium failed to thread or unthread during the process of mounting or demounting.	MEDIUM ERROR HARDWARE ERROR
a) See the READ(16) (see 5.3), READ REVERSE(16) (see 5.4), VERIFY(16) (see 5.5), WRITE(16) (see 5.6), READ(6) (see 6.4), READ REVERSE(6) (see 6.5), VERIFY(6) (see 6.7), WRITE(6) (see 6.8), and RECOVER BUFFERED data (see 7.7) commands for a description of the FIXED bit.	

The Read-Write Error Recovery mode page (see 8.3.5) current values specify behavior when an unrecoverable read or write error is encountered. If the Read-Write Error Recovery mode page is not implemented, the behavior is vendor specific.

In the case of a deferred write error, the sense data VALID bit shall be set to zero.

In the case of an unrecovered write error or a deferred write error, if buffered mode 1h is selected, the error shall be reported to the first application client issuing a command (other than INQUIRY or REQUEST SENSE). If buffered mode 2h is selected, the error shall be reported to the SCSI initiator device with unwritten data in the object buffer.

In the case of a write attempt to write protected medium, the additional sense code specifies the cause of the DATA PROTECT sense key (see 4.2.14).

In the case of a medium thread or unthread failure, the additional sense code shall be set to MEDIUM THREAD OR UNTHREAD FAILURE. The sense key shall be set to MEDIUM ERROR or HARDWARE ERROR (see SPC-4).

4.2.14 Write protection

4.2.14.1 Write protection introduction

Write protection of the medium prevents the alteration of logical objects on the medium, and any change to the accessibility of logical objects on the medium, by commands issued to the device server. Write protection is usually controlled by the user of the volume through manual intervention (e.g., mechanical lock) or may result from hardware controls (e.g., tabs on the volume housing), conditions such as positioning within unrecoverable data, or software write protections. All sources of write protection are independent. When present, any write protection shall cause otherwise valid commands that request alteration of logical objects on the medium, or affect the accessibility of logical objects on the medium, to be rejected with a CHECK CONDITION status with the sense key set to DATA PROTECT (see 4.2.14.2). Only when all write protections are disabled shall the device server process commands that request alteration of logical objects on the medium, or commands that may affect the accessibility of logical objects on the medium.

Hardware write protection results when a physical attribute of the drive or volume is changed to specify that writing shall be prohibited. Changing the state of the hardware write protection requires physical intervention, either with the drive or the volume. If allowed by the drive, changing the hardware write protection while the

volume is mounted results in vendor-specific behavior that may include the writing of previously buffered logical objects.

Conditions such as positioning within unrecoverable data may result in a temporary write protection condition. To preserve future data integrity, the device server may reject any command that requires writing data to the medium when the recovery of the data is uncertain. A temporary write protection condition may be released by the device server at any time. Buffered logical objects may not be written to the medium (e.g., the application client unloads the volume before the temporary write protection condition is removed). The exact behavior of the device server during a temporary write protection condition is vendor specific.

Software write protection results when either the device server, volume, or medium is marked as write protected by a command from the application client. Four optional means of setting a software write protection state are available to an application client through the Device Configuration and Control mode pages:

- a) software write protection for the device server across mounts;
- b) associated write protection for the currently mounted volume;
- c) persistent write protection of a volume across mounts; and
- d) permanent write protection of a volume across mounts.

The application client may control these write protections using the Control mode page (see SPC-4) and the Device Configuration mode page (see 8.3.3). All of the software write protection methods are optional. Changing the state of any software write protection shall not prevent previously buffered logical objects from transferring to the medium.

4.2.14.2 Write protection additional sense code use

The additional sense code associated with the DATA PROTECT sense key depends on the write protection in effect at the time. Table 6 specifies the preferred additional sense code for the given write protection. Alternatively, the generic additional sense code of WRITE PROTECTED may be returned by the device server.

Table 6 — Write protect additional sense code combinations

Cause of DATA PROTECT error	Additional sense code
Hardware Write Protection	HARDWARE WRITE PROTECTED
Permanent Write Protection	PERMANENT WRITE PROTECT
Persistent Write Protection	PERSISTENT WRITE PROTECT
Associated Write Protection	ASSOCIATED WRITE PROTECT
Software Write Protection	LOGICAL UNIT SOFTWARE WRITE PROTECTED

If more than one condition exists, the device server shall:

- a) report the generic additional sense code of WRITE PROTECTED; or
- b) report the applicable condition in order of:
 - 1) HARDWARE WRITE PROTECTED;
 - 2) PERMANENT WRITE PROTECT;
 - 3) PERSISTENT WRITE PROTECT;
 - 4) ASSOCIATED WRITE PROTECT; and
 - 5) LOGICAL UNIT SOFTWARE WRITE PROTECTED.

Other conditions that may cause a command to be rejected with a DATA PROTECT sense key include:

- a) the current volume is maintained as read-only by the device server;
- b) the device server is only able to write from BOP or EOD and the current logical position is neither;
- c) the volume is a WORM volume (see 4.2.21) and an attempt to write in an unrecordable location is detected;
- d) the set of logical block encryption parameters in the device entity is not correct for the operation requested; and
- e) vendor-specific conditions.

4.2.14.3 Software write protection for the device server

Software write protection controls write protection for the device server. This method of write protection is optionally controlled from the Control mode page (see SPC-4) or the SWP bit in the Device Configuration mode page (see 8.3.3). Either or both methods may be implemented by the device server. If both methods are implemented, each control bit is independently set. Software write protection exists if either bit is non-zero. The state of software write protection for the device server shall not be recorded on the medium. The value of the SWP bit may be altered by the application client (if the SWP bit is changeable). The state of each control bit shall be set to its default state after a logical unit reset.

4.2.14.4 Associated write protection

Associated write protection controls write protection for the currently mounted volume as long as the current volume is mounted. The associated write protection state is controlled by the ASOCWP bit in the Device Configuration mode page (see 8.3.3). Associated write protection exists if the ASOCWP bit is non-zero. Associated write protection may be altered by the application client (if the ASOCWP bit is changeable) if a volume is mounted. If a volume is demounted or after a logical unit reset occurs, associated write protection shall be removed.

4.2.14.5 Persistent write protection

Persistent write protection controls write protection for the currently mounted volume. The persistent write protection state is controlled by the PERSWP bit in the Device Configuration mode page (see 8.3.3). If enabled, persistent write protection shall exist for the mounted volume until disabled by the application client. The state of persistent write protection shall be recorded with the volume and the persistent write protection shall only affect the application client accessible medium. The device server shall report the PERSWP bit as one when a mounted volume is marked with persistent write protection. If a volume is demounted or after a logical unit reset occurs, the device server shall report the PERSWP bit as zero prior to the mounting of a volume. The means for recording the state of persistent write protection for the volume may be specified in the applicable recording format standard or be vendor specific.

4.2.14.6 Permanent write protection

Permanent write protection controls write protection for the currently mounted volume. The permanent write protection state is controlled by the PRMWP bit in the Device Configuration mode page (see 8.3.3). If enabled, permanent write protection shall exist for the mounted volume until disabled by a vendor-specific method. The state of permanent write protection shall be recorded with the volume and the permanent write protection shall only affect the application client accessible medium. The device server shall report the PRMWP bit as one when a mounted volume is marked with permanent write protection. If a volume is demounted or after a logical unit reset occurs, the device server shall report the PRMWP bit as zero prior to the mounting of a volume. The means for recording the state of permanent write protection for the volume may be specified in the applicable recording format standard or be vendor specific. Permanent write protection shall not be removed by a MODE SELECT command using the PRMWP bit. Methods to remove this protection may exist and are vendor specific.

4.2.15 Progress indication

For the following immediate operations where the device server remains ready, an application client may test the progress of the operation (see table 7).

Table 7 — Commands providing progress indication without changing ready state

Operation	Options	Subclause	Additional Sense Code
ERASE	LONG = 1	5.2, 6.2	ERASE OPERATION IN PROGRESS
LOCATE		5.2, 6.3	LOCATE OPERATION IN PROGRESS
REWIND		7.9	REWIND OPERATION IN PROGRESS
SET CAPACITY		7.10	SET CAPACITY OPERATION IN PROGRESS
VERIFY		5.5, 6.7	VERIFY OPERATION IN PROGRESS

While the device server is performing the operation, an application client may test the progress of the operation by interpreting the progress indication information in the sense-key specific field of the sense data. During the operation, the device server may report a sense key value of NO SENSE and additional sense code as specified in table 7. The device server should use the sense key specific function for progress indication to provide information on the completion of the operation.

For the following immediate operations where the device server is ready. An application client may follow the progress of the operations specified in table 8.

Table 8 — Commands changing ready state and providing progress indication

Operation	Options	Subclause	Additional Sense Code
FORMAT MEDIUM		7.1	LOGICAL UNIT NOT READY, FORMAT IN PROGRESS
LOAD UNLOAD	LOAD = 1, EOT = 0	7.2	LOGICAL UNIT IS IN PROCESS OF BECOMING READY
LOAD UNLOAD	LOAD = 0, EOT = 1	7.2	LOGICAL UNIT NOT READY, OPERATION IN PROGRESS

While the device server is performing the operation, an application client may test the progress of the operation by interpreting the progress indication information in the sense-key specific field of the sense data. During the operation, the device server may report a sense key value of NOT READY and an additional sense code as specified in table 8. The sense key specific function for progress indication may be used by the device server to provide information on the completion of the operation.

NOTE 1 - A REQUEST SENSE command following a TEST UNIT READY command that results in CHECK CONDITION status may provide information that, if acted upon, may lead to unexpected conditions. For example, progress indication reporting is useful when a media changer is used to service a sequential-access device following an unload operation with IMMED=1b. A TEST UNIT READY command may receive CHECK CONDITION status and a NOT READY sense key reported in the subsequent sense data. This may imply that the unload operation is finished. If the application client ignores the progress indication information in the sense data, an EXCHANGE MEDIUM or MOVE MEDIUM command (see SMC-2) to move the demounted volume from the SCSI device may fail to grab the volume if the unload operation is still in progress.

4.2.16 Tagged command queuing

4.2.16.1 Tagged command queuing overview

A device server may choose to implement support for tagged tasks (i.e., command queuing). Issuing tagged write commands with object buffering disabled facilitates streaming operations up to the limit of the number of outstanding tagged commands supported by the application client and the device server. This limit may effectively reduce the usable portion of the object buffer which may affect device server performance.

For proper operation when performing tagged command queuing operations, an application client should wait for status to be returned for any MODE SELECT command before issuing the next command.

NOTE 2 - Since the EXTENDED COPY command does not specify an immediate bit, the use of tagged command queuing may be required to prevent an overlapped command condition (e.g., when querying the progress of the extended copy operation using the RECEIVE COPY RESULTS command).

4.2.16.2 Explicit address mode tagged write sequences

When operating in explicit address mode, tagged write sequences (see 3.1.74) are used to support tagged command queuing for write operations.

For explicit address mode tagged write sequences, the following rules apply:

- for a tagged write sequence consisting of more than one command, the FCS bit (see 5.2, 5.6, 5.7) shall be set to one in the first command of the tagged write sequence. For all other commands within the tagged write sequence, the FCS bit shall be set to zero;

- b) for a tagged write sequence consisting of more than one command, the LCS bit (see 5.2, 5.6, 5.7) shall be set to one in the last command of the tagged write sequence. For all other commands within the tagged write sequence, the LCS bit shall be set to zero;
- c) for a tagged write sequence consisting of only one command, the FCS bit and LCS bit shall be set to one;
- d) for a tagged write sequence consisting of more than one command, the application client shall issue the commands in sequentially-increasing logical object identifier order;
- e) the application client shall not issue a tagged write sequence prior to receiving status for all outstanding read type commands; and
- f) the application client shall specify a Command Reference Number (see SAM-4) for each command in a tagged write sequence.

4.2.17 Block address mode

4.2.17.1 Block address mode overview

When operating in implicit address mode, spacing operations and commands to read from and write on the medium do not contain positioning information (i.e., a command is processed based on the medium position relative to the last command that was processed). As such, the application client does not contain explicit knowledge of the medium position. If an error occurs, to maintain data integrity, the application client is required to determine the medium position before re-issuing a command that affects the medium position.

When operating in explicit address mode, commands to read from and write on the medium contain positioning information fields (i.e., a command is processed based on the medium position information specified in the command). As such, the application client contains explicit knowledge of the medium position. This results in enhanced error detection and recovery functionality that allows the application client to maintain data integrity while performing various operations without first determining the medium position. Some example operations include:

- a) the re-issuing of a command that affects the medium position;
- b) multi-path I/O; and
- c) tagged command queuing.

The device server shall support explicit address mode only, implicit address mode only, or both explicit address mode and implicit address mode. At any instance, the device server shall be in or transitioning between one of the following block address mode states (see 4.2.17.3):

- a) A0:Idle;
- b) E0:Explicit Address Mode - Neutral;
- c) E1:Explicit Address Mode - Write Capable; or
- d) F0:Implicit Address Mode.

4.2.17.2 Block address mode selection

The block address mode shall be selected as follows:

- a) if the BAML bit in the Device Configuration mode page (see 8.3.3) is set to zero, then the setting of the BAM bit in the Device Configuration mode page (see 8.3.3) shall be ignored and the block address mode shall be determined based on the first block address mode unique command that is received after a successful load operation or successful completion of a command that positions the medium to BOP;
- b) if the BAML bit in the Device Configuration mode page is set to one and the BAM bit in the Device Configuration mode page is set to zero, the logical unit shall support implicit address mode; or
- c) if the BAML bit in the Device Configuration mode page is set to one and the BAM bit in the Device Configuration mode page is set to one, the logical unit shall support explicit address mode.

Prior to performing a block address mode change, the logical unit shall perform a synchronize operation (see 4.2.11).

If the device server receives a command that is not supported by the currently selected mode, the device server shall return CHECK CONDITION, the sense key shall be set to ILLEGAL REQUEST, and the additional sense code shall be set to:

- a) ILLEGAL COMMAND WHILE IN EXPLICIT ADDRESS MODE if the currently selected mode is the explicit address mode;
- b) ILLEGAL COMMAND WHILE IN IMPLICIT ADDRESS MODE if the currently selected mode is the implicit address mode; or
- c) ILLEGAL COMMAND WHILE IN WRITE CAPABLE STATE if the device server is in explicit address mode write capable state.

4.2.17.3 Block address mode state diagrams

For the block address mode state diagrams (see figure 15, figure 16, figure 17, and figure 18), the following terminology shall apply:

- a) explicit command: a command contained only in the explicit address command set (see table 22);
- b) implicit command: a command contained only in the implicit address command set (see table 30); and
- c) generic command: an explicit command that is not a read type or write type command (see table 22).

A common command containing a BAM bit (e.g., LOCATE(16)) shall be processed as either an explicit or implicit command based on the setting of the BAM bit contained in the common command.

The SPACE(16) command shall be processed as either an explicit or implicit command based on the setting of the PARAMETER LENGTH field in the command descriptor block.

Figure 14 provides an overview of the block address model state diagram. Refer to figure 15, figure 16, figure 17, and figure 18 for detailed descriptions of the block address model state diagram.

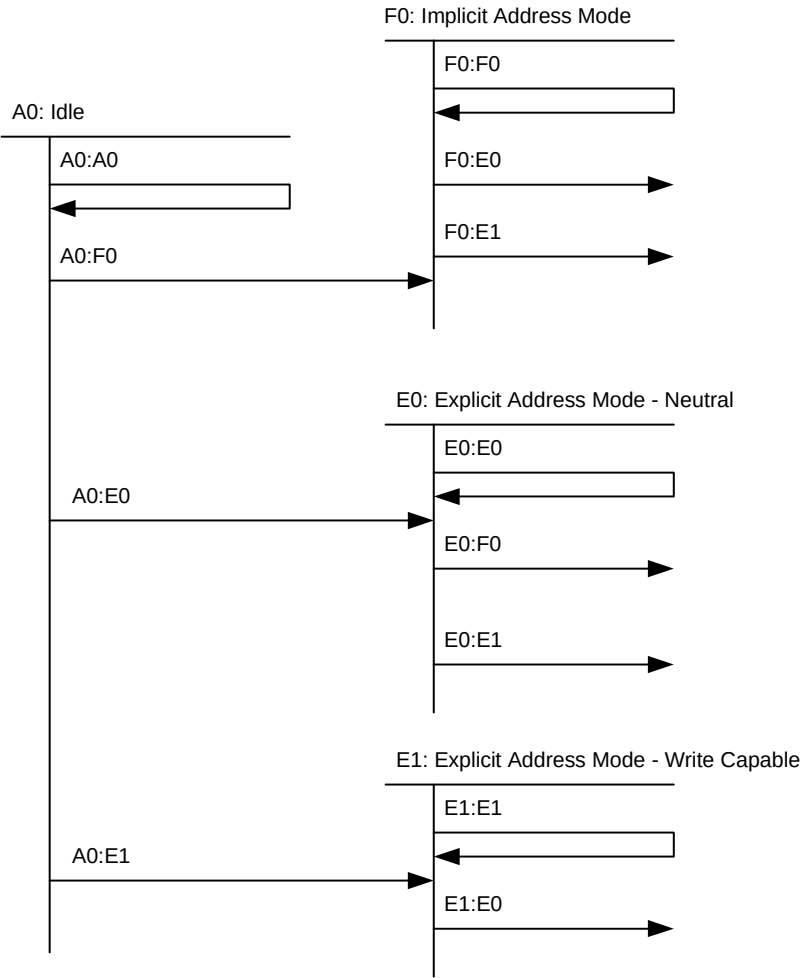


Figure 14 — Block address mode state diagram, overview

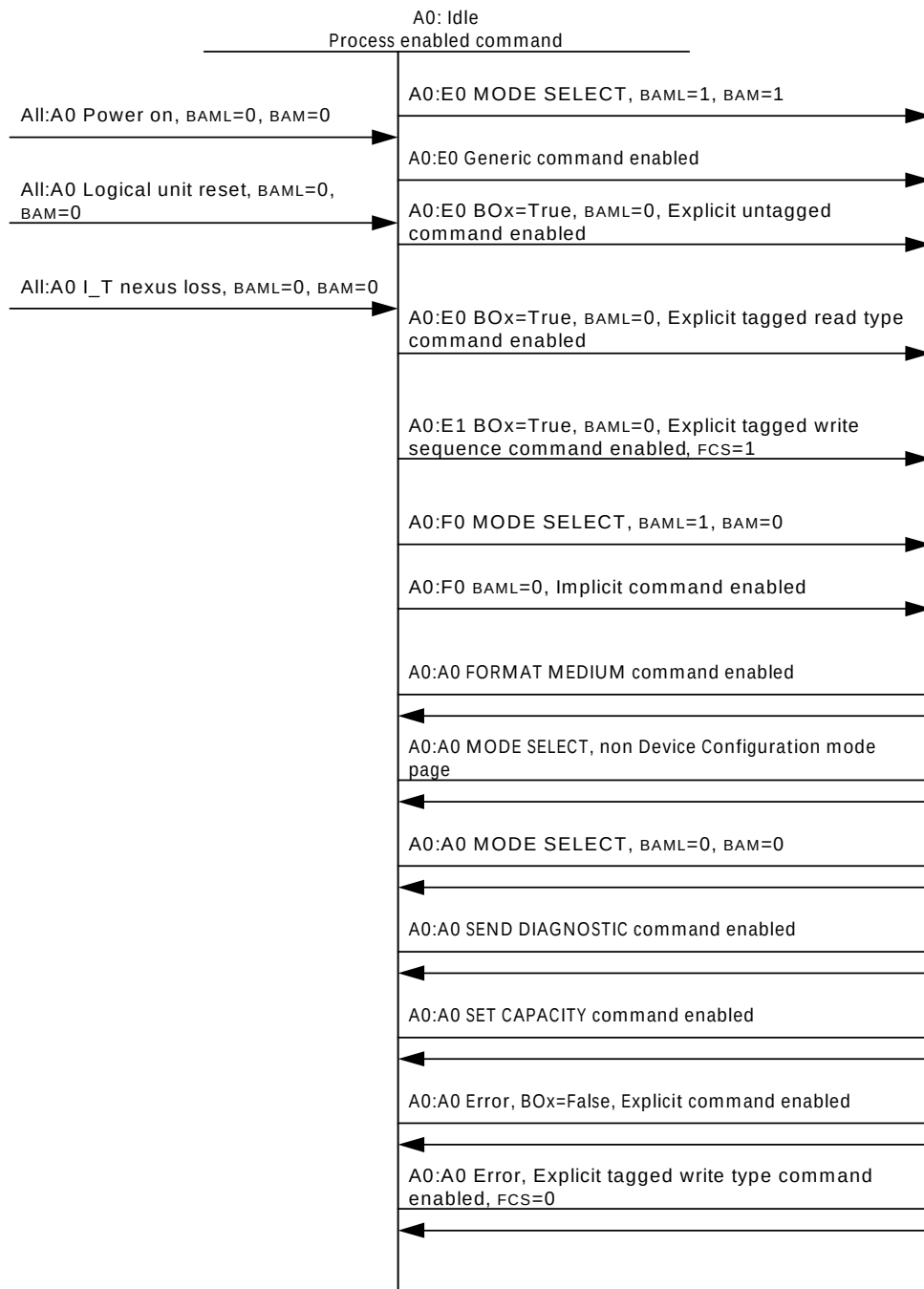


Figure 15 — Block address mode state diagram, Idle state

State A0:Idle: This is the idle state.

Transition All:A0: This transition shall occur when a power-on, logical unit reset, or I_T nexus loss event occurs and the BAML bit is set to zero and the BAM bit is set to zero.

Transition A0:E0: This transition shall occur when:

- a MODE SELECT command specifying a Device Configuration mode page with the BAML bit set to one and the BAM bit set to one completes with GOOD status;
- a generic command is enabled;
- an explicit untagged command is enabled, the medium position is at BOx, and the BAML bit is set to zero; or

- d) an explicit tagged read type command is enabled, the medium position is at BOx, and the BAML bit is set to zero.

Transition A0:E1: This transition shall occur when an explicit tagged write sequence command is enabled with the FCS bit set to one, the medium position is at BOx, and the BAML bit is set to zero.

Transition A0:F0: This transition shall occur when:

- a) a MODE SELECT command specifying a Device Configuration mode page with the BAML bit set to one and the BAM bit set to zero completes with GOOD status; or
- b) an implicit command is enabled, the medium position is at BOx, and the BAML bit is set to zero.

Transition A0:A0: This transition shall occur when:

- a) a FORMAT MEDIUM command is enabled;
- b) a MODE SELECT command specifying a mode page other than the Device Configuration mode page is enabled;
- c) a MODE SELECT command specifying a Device Configuration mode page with the BAML bit set to zero and the BAM bit set to zero is enabled;
- d) a SEND DIAGNOSTIC command is enabled;
- e) a SET CAPACITY command is enabled;
- f) an explicit command is enabled and the medium position is not at BOx. In this case the device server shall return CHECK CONDITION status, the sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to SEQUENTIAL POSITIONING ERROR; or
- g) an explicit tagged write type command is enabled with the FCS bit set to zero. In this case the device server shall return CHECK CONDITION status, the sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to INVALID FIELD IN CDB.

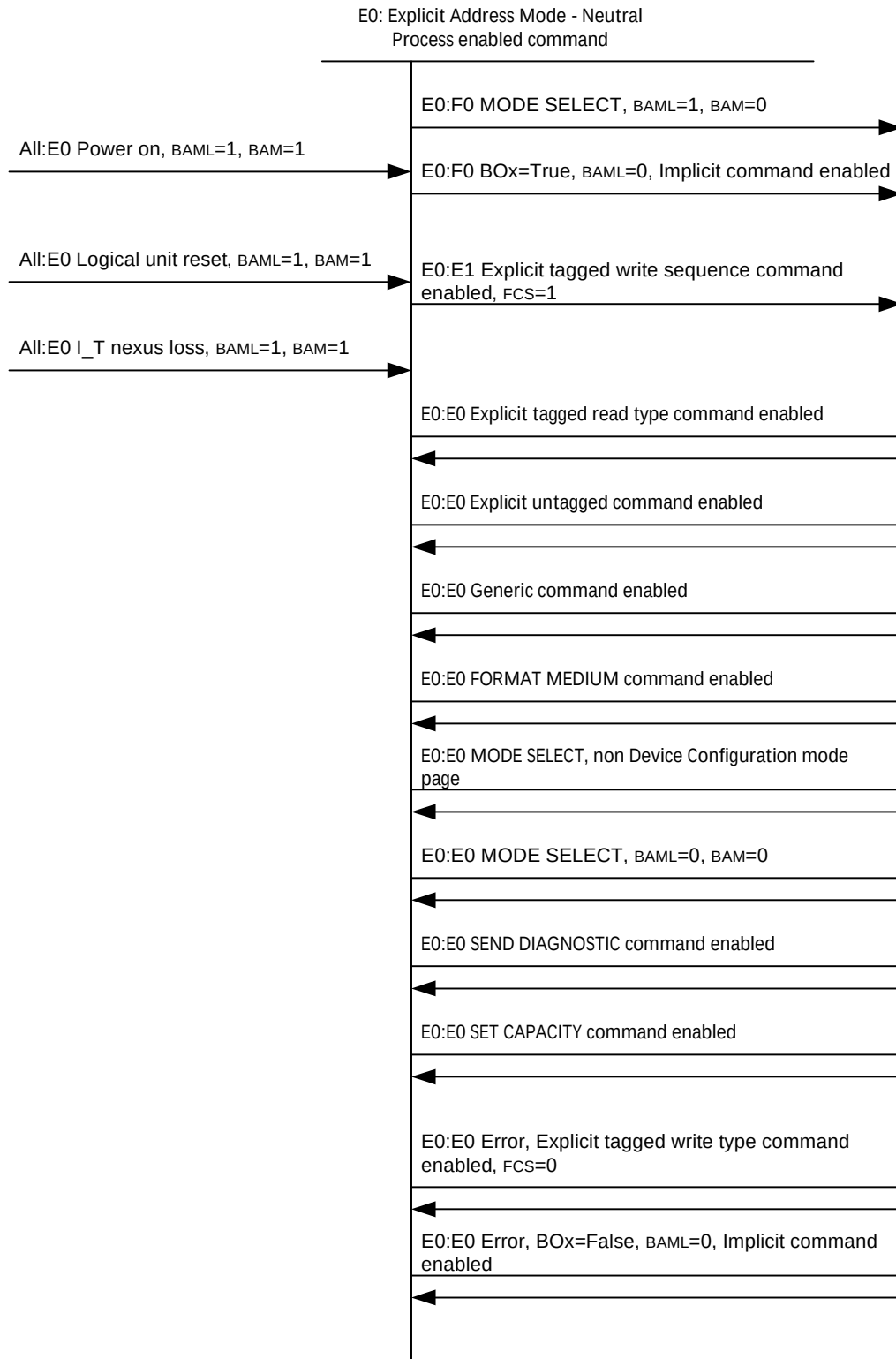


Figure 16 — Block address mode state diagram, Explicit Address Mode - Neutral

State E0:Explicit Address Mode - Neutral: This is the neutral state for explicit address mode.

Transition All:E0: This transition shall occur when a power-on, logical unit reset, or I_T nexus loss event occurs and the BAML bit is set to one and the BAM bit is set to one.