ISO/IEC 14776-453

Edition 1.0 2009-12

INTERNATIONAL STANDARD

Information technology – Small computer system interface (SCSI) – Part 453: Primary commands-3 (SPC-3)

ISO/IEC 14776-453:2009(E)

IEC Central Office
3, rue de Varembé
CH-1211 Geneva 20
Switzerland
Email: inmail@iec.ch
Web: 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.

- Catalogue of IEC publications: www.iec.ch/searchpub

The IEC on-line Catalogue enables you to search by a variety of criteria (reference number, text, technical committee,…). It also gives information on projects, withdrawn and replaced publications.

- IEC Just Published: www.iec.ch/online_news/justpub

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

- Electropedia: www.electropedia.org

The world's leading online dictionary of electronic and electrical terms containing more than 20 000 terms and definitions in English and French, with equivalent terms in additional languages. Also known as the International Electrotechnical Vocabulary online.

- Customer Service Centre: www.iec.ch/webstore/custserv

If you wish to give us your feedback on this publication or need further assistance, please visit the Customer Service Centre FAQ or contact us:

Email: csc@iec.ch
Tel.: +41 22 919 02 11
Fax: +41 22 919 03 00

ISO/IEC 14776-453

Edition 1.0 2009-12

INTERNATIONAL STANDARD

Information technology – Small computer system interface (SCSI) – Part 453: Primary commands-3 (SPC-3)

INTERNATIONAL ELECTROTECHNICAL COMMISSION

PRICE CODE XK

ICS 35.200

ISBN 2-8318-1070-9

Contents

Tables

Figures

INFORMATION TECHNOLOGY — SMALL COMPUTER SYSTEM INTERFACE SCSI —

Part 453: SCSI Primary Commands - 3 (SPC-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/IECpublication 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 14776-453 was prepared by subcommittee 25: Interconnection of Information technology equipment, of ISO/IEC joint technical committee 1: Information technology.

The 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.

Introduction

ISO/IEC 14776-453: SCSI Primary Commands - 3 (SPC-3) standard is divided into the following clauses and annexes:

Clause 1 is the scope.
Clause 2 enumerates the normative references that apply to this standard.
Clause 3 describes the definitions, symbols and abbreviations used in this standard.
Clause 4 describes the conceptual relationship between this document and the SCSI-3 architecture model.
Clause 5 describes the command model for all SCSI devices.
Clause 6 defines the commands that may be implemented by any SCSI device.
Clause 7 defines the parameter data formats that may be implemented by any SCSI device.
Clause 8 defines the well known logical units that may be implemented by any SCSI device.
Annex A identifies differences between the terminology used in this standard and previous versions of this standard (informative).
Annex B describes the PERSISTENT RESERVE OUT command features necessary to replace the reserve/release management method and provides guidance on how to perform a third party reservation using persistent reservations (informative).
Annex C elaborates on the procedures for logging operations (informative).
Annex D lists code values in numeric order (informative).
Annex E lists assigned vendor identifiers (informative).

The annexes provide information to assist with implementation of this standard. The information in the annexes applies to all the SCSI command standards. See 3.1.18 for more information about other SCSI command standards.

Information Technology - Small Computer System Interface (SCSI) -

Part 453: SCSI Primary Commands-3 (SPC-3)

1 Scope

The SCSI family of standards provides for many different types of SCSI devices (e.g., disks, tapes, printers, scanners). This standard defines a device model that is applicable to all SCSI devices. Other SCSI command standards (see 3.1.18) expand on the general SCSI device model in ways appropriate to specific types of SCSI devices.

The set of SCSI standards specifies the interfaces, functions, and operations necessary to ensure interoperability between conforming SCSI implementations. This standard is a functional description. Conforming implementations may employ any design technique that does not violate interoperability.

This standard defines the SCSI commands that are mandatory and optional for all SCSI devices. Support for any feature defined in this standard is optional unless otherwise stated. This standard also defines the SCSI commands that may apply to any device model.

The following commands, parameter data, and features defined in previous versions of this standard are made obsolete by this standard:

a) Contingent Allegiance;
b) Untagged tasks;
c) The RESERVE(6) and RESERVE(10) commands;
d) The RELEASE(6) and RELEASE(10) commands;
e) The ELEMENT_SCOPE for Persistent Reservations;
f) The command support data (CMDDT) feature of the INQUIRY command;
g) The relative addressing (RELADR) bit in the standard INQUIRY data;
h) The Medium Partition mode pages (2), (3), and (4);
i) The Control mode page DISABLE QUEUEING bit;
j) Discussion of the SBC REBUILD, REGENERATE and XDWRITE EXTENDED commands; and
k) The ASCII Implemented Operating Definition VPD page.

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

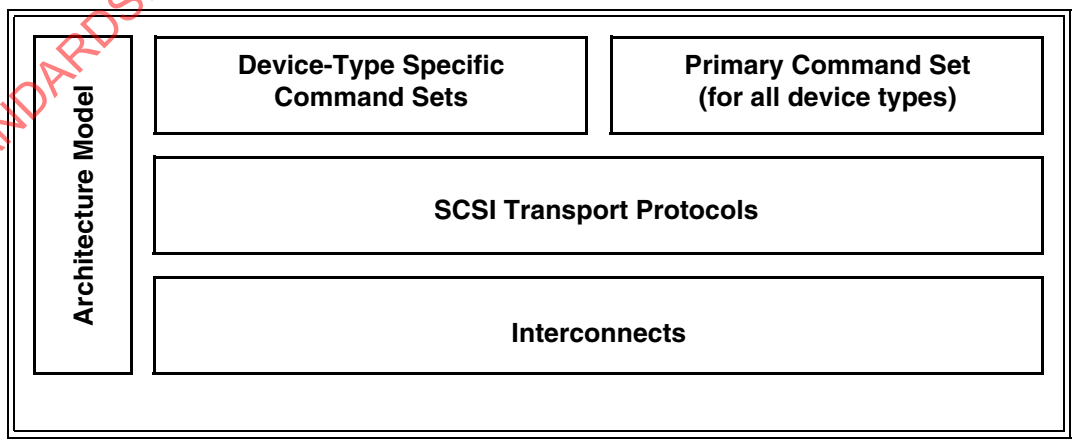

Figure 1 — SCSI document relationships

Figure 1 is intended to show the general relationship of the documents to one another. Figure 1 is not intended to imply a relationship such as a hierarchy, protocol stack, or system architecture. It indicates the applicability of a standard to the implementation of a given transport.

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

2 Normative references

2.1 General

The following standards contain provisions that, by reference in the text, constitute provisions of this standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this standard are encouraged to investigate the possibility of applying the most recent editions of the standards listed below.

2.2 Approved references

Copies of the following documents may be obtained from ANSI: approved ANSI standards, approved and draft international and regional standards (ISO, IEC, CEN/CENELEC, ITUT), and approved and draft foreign standards (including BSI, JIS, and DIN). For further information, contact ANSI Customer Service Department at 212-642-4900 (phone), 212-302-1286 (fax) or via the World Wide Web at http://www.ansi.org.

ISO/IEC 646:1991, *Information technology – ISO 7-bit coded character set for information interchange*

ISO/IEC 8859-1:1998, *Information technology – 8-bit single-byte coded graphic character sets – Part 1: Latin alphabet No. 1*

ISO/IEC 8859-2:1999, *Information technology – 8-bit single-byte coded graphic character sets – Part 2: Latin alphabet No. 2*

ISO/IEC 8859-3:1999, *Information technology – 8-bit single-byte coded graphic character sets – Part 3: Latin alphabet No. 3*

ISO/IEC 8859-4:1998, *Information technology – 8-bit single-byte coded graphic character sets – Part 4: Latin alphabet No. 4*

ISO/IEC 8859-5:1999, *Information technology – 8-bit single-byte coded graphic character sets – Part 5: Latin/Cyrillic alphabet*

ISO/IEC 8859-6:1999, *Information technology – 8-bit single-byte coded graphic character sets – Part 6: Latin/Arabic alphabet*

ISO/IEC 8859-7:1987, *Information processing – 8-bit single-byte coded graphic character sets – Part 7: Latin/Greek alphabet*

ISO/IEC 8859-8:1999, *Information technology – 8-bit single-byte coded graphic character sets – Part 8: Latin/Hebrew alphabet*

ISO/IEC 8859-9:1999, *Information technology – 8-bit single-byte coded graphic character sets – Part 9: Latin alphabet No. 5*

ISO/IEC 8859-10:1998, *Information technology – 8-bit single-byte coded graphic character sets – Part 10: Latin alphabet No. 6*

ISO/IEC 10646:2003, *Information technology – Universal Multiple-Octet Coded Character Set (UCS)*

ISO/IEC 13213:1994, *Information technology – Microprocessor systems – Control and Status Registers Architecture for microcomputer buses* *[ANSI/IEEE 1212, 1994 Edition]*

ISO/IEC 14165-251, *Information technology – Fibre Channel – Part 251: Framing and Signaling Interface (FC-FS)* *[ANSI INCITS 373-2003]*

ISO/IEC 14776-115, *Information technology – Small Computer System Interface (SCSI) – Part 115: SCSI Parallel Interface - 5 (SPI-5)* *[ANSI INCITS 367-2003]*

ISO/IEC 14776-150, *Information technology – Small Computer System Interface (SCSI) – Part 150: Serial Attached SCSI (SAS)* *[ANSI INCITS 376-2003]*

ISO/IEC 14776-222, *Information technology – Small Computer System Interface (SCSI) – Part 222: Fibre Channel Protocol for SCSI, second version (FCP-2)* *[ANSI INCITS 350:2003]*

ISO/IEC 14776-322, *Information technology – Small Computer System Interface (SCSI) – Part 322: SCSI Block Commands - 2 (SBC-2)* *[ANSI INCITS 405-2005]*

ISO/IEC 14776-381: 2000, *Information technology – Small computer system interface (SCSI) – Part 381: Optical Memory Card Device Commands (OMC)*

ISO/IEC 14776-412, *Information technology – Small Computer System Interface (SCSI) – Part 412: SCSI Architecture Model - 2 (SAM-2)* *[ANSI INCITS 366-2003]*

ISO/IEC 14776-413, *Information technology – Small Computer System Interface (SCSI) – Part 413: SCSI Architecture Model - 3 (SAM-3)* *[ANSI INCITS 402-2005]*

ISO/IEC 14776-452, *Information technology – Small Computer System Interface (SCSI) – Part 452: SCSI Primary Commands - 2 (SPC-2)* *[ANSI INCITS 351-2001]*

ISO/IEC 24739 (all parts), *Information technology – AT Attachment with Packet Interface - 7 (ATA/ATAPI-7) V1*

IEC 60027-2:2000, *Letter symbols to be used in electrical technology – Part 2: Telecommunications and electronics*

ANSI/IEEE 1394-1995, *High Performance Serial Bus*

ANSI/IEEE 1394a-2000, *High Performance Serial Bus (supplement to ANSI/IEEE 1394-1995)*

ANSI INCITS 4-1986 (R2002), *Information Systems – Coded Character Sets – 7-Bit American National Standard Code for Information Interchange (7-Bit ASCII)*

ANSI INCITS 309-1998, *Serial Storage Architecture SCSI-3 Protocol (SSA-S3P)*

ANSI INCITS 365-2002, *Information technology – Small Computer System Interface (SCSI) – Part 241: SCSI RDMA Protocol (SRP)* *[ISO/IEC 14776-241, under consideration]*

ANSI INCITS 375-2004, *Serial Bus Protocol - 3 (SBP-3)*

ANSI INCITS 382-2004, *Information technology – Small Computer System Interface (SCSI) – SCSI Media Changer Commands - 2 (SMC-2)* *[ISO/IEC 14776-352],*

ANSI INCITS 405-2005, *Automation/Drive Interface – Transport Protocol (ADT)*

ANSI INCITS 467, *Information technology – Small Computer System Interface (SCSI) – Stream commands-3 (SSC-3)* (to be published)

2.3 IETF References

Copies of the following approved IETF standards may be obtained through the Internet Engineering Task Force (IETF) at www.ietf.org.

RFC 791, *Internet Protocol - DARPA Internet Program - Protocol Specification*

RFC 793, *Transmission Control Protocol - DARPA Internet Program - Protocol Specification*

RFC 1035, *Domain Names - Implementation and Specification*

RFC 1321, *The MD5 Message-Digest Algorithm*

RFC 1591, *Domain Name System Structure and Delegation*

RFC 2279, *UTF-8, a transformation format of ISO 10646*

RFC 2373, *IP Version 6 Addressing Architecture*

RFC 2396, *Uniform Resource Identifiers (URI): Generic Syntax*

RFC 2616, *Hypertext Transfer Protocol -- HTTP/1.1*

RFC 3305, *Report from the Joint W3C/IETF URI Planning Interest Group: Uniform Resource Identifiers (URIs), URLs, and Uniform Resource Names (URNs): Clarifications and Recommendations*

RFC 3720, *Internet Small Computer Systems Interface (iSCSI)*

3 Terms and definitions, symbols, abbreviations and conventions

3.1 Terms and definitions

3.1.1
access control list (ACL):
The data used by a SCSI target device to configure access rights for initiator ports according to the access controls state of the SCSI target device (see 8.3.1.3).

3.1.2
access control list entry (ACE):
One entry in the access control list (see 3.1.1).

3.1.3
access controls:
An optional SCSI target device feature that restricts initiator port access to specific logical units and modifies the information about logical units in the parameter data of the INQUIRY and REPORT LUNS commands (see 8.3.1).

3.1.4
access controls coordinator:
The entity within a SCSI target device that coordinates the management and enforcement of access controls (see 8.3.1) for all logical units within the SCSI target device. The access controls coordinator is always addressable through the ACCESS CONTROLS well known logical unit (see 8.3) and LUN 0.

3.1.5
active power condition:
When a device server is capable of responding to all of its supported commands, including media access requests, without delay. See 5.9.

3.1.6
additional sense code:
A combination of the ADDITIONAL SENSE CODE and ADDITIONAL SENSE CODE QUALIFIER fields (see 4.5) in the sense data.

3.1.7
alias list:
A list of alias values (see 3.1.8) and their associated designations (see 3.1.27) maintained by the device server and managed by the CHANGE ALIASES command (see 6.2) and REPORT ALIASES command (see 6.19).

3.1.8
alias value:
A numeric value associated to a designation (see 3.1.27) in the alias list (see 3.1.7) and used in command or parameter data to reference a SCSI target device or SCSI target port. See 6.2.2.

3.1.9
application client:
An object that is the source of SCSI commands. Further definition of an application client may be found in SAM-3.

3.1.10
attached medium changer:
A medium changer that is attached to and accessed through some other type of SCSI device. See 5.10.

3.1.11
attribute:
A single unit of MAM (see 3.1.64) information.

3.1.12
auto contingent allegiance (ACA):
The task set condition established following the return of a CHECK CONDITION status when the NACA bit is set to one in the CONTROL byte. A detailed definition of ACA may be found in SAM-3.

3.1.13
blocked task:
A task that is in the blocked state. Tasks become blocked when an ACA condition occurs. The blocked state ends when the ACA condition is cleared. A detailed definition of the blocked task state may be found in SAM-3.

3.1.14
byte:
A sequence of eight contiguous bits considered as a unit.

3.1.15
cache memory:
A temporary and often volatile data storage area outside the area accessible by application clients that may contain a subset of the data stored in the non-volatile data storage area.

3.1.16
command:
A request describing a unit of work to be performed by a device server. A detailed definition of a command may be found in SAM-3.

3.1.17
command descriptor block (CDB):
The structure used to communicate commands from an application client to a device server (see 4.3). A CDB may have a fixed length of up to 16 bytes or a variable length of between 12 and 260 bytes.

3.1.18
command standard:
A SCSI standard that defines the model, commands, and parameter data for a device type (e.g., SBC-2, SSC-2, SMC-2, MMC-4, or SES-2). See clause 2.

3.1.19
company_id:
Synonym for OUI (see 3.1.74).

3.1.20
Control mode page:
A mode page that provides controls over SCSI features (e.g., task set management and error logging) that are applicable to all device types. See 7.4.6.

3.1.21
Control Extension mode page:
A mode page that provides controls over SCSI features that are applicable to all device types. See 7.4.7.

3.1.22
copy manager:
The device server that receives an EXTENDED COPY command and performs the operation requested.

3.1.23
copy target device:
The name given by the EXTENDED COPY command (see 6.3) to a source or destination logical unit (i.e., a copy target device is a logical unit (see 3.1.58), not a SCSI target device (see 3.1.100)).

3.1.24
data-in buffer:
The buffer specified by the application client to receive data from the device server during the processing of a command (see 4.2 and SAM-3).

3.1.25
data-out buffer:
The buffer specified by the application client to supply data that is sent from the application client to the device server during the processing of a command (see 4.2 and SAM-3).

3.1.26
deferred error:
A CHECK CONDITION status and sense data that is returned as the result of an error or exception condition that occurred during processing of a previous command for which GOOD, CONDITION MET, INTERMEDIATE, and INTERMEDIATE-CONDITION MET status has already been returned. See 4.5.5.

3.1.27
designation:
A name and optional identifier information that specifies a SCSI target device or SCSI target port for association with an alias value (see 3.1.8) in the alias list (see 3.1.7). See 6.2.2.

3.1.28
Device Identification VPD page:
A VPD page that provides the means to retrieve identification information about the SCSI device, logical unit, and SCSI port. See 7.6.3.

3.1.29
device server:
An object within a logical unit that processes SCSI tasks according to the rules of task management. A detailed definition of a device server may be found in SAM-3.

3.1.30
device service request:
A request, submitted by an application client, conveying a SCSI command to a device server. A detailed definition of a device service request may be found in SAM-3.

3.1.31
device service response:
The response returned to an application client by a device server on completion of a SCSI command. A detailed definition of a device service response may be found in SAM-3.

3.1.32
device type:
The type of peripheral device (i.e., device model) implemented by the device server and indicated by the contents of the PERIPHERAL DEVICE TYPE field in the standard INQUIRY data (see 6.4.2).

3.1.33
Disconnect-Reconnect mode page:
A mode page that provides the application client the means to tune the performance of the service delivery subsystem. See 7.4.8.

3.1.34
element:
An addressable physical component of a medium changer SCSI device that may serve as the location of a removable unit of data storage medium. A detailed definition of an element may be found in SMC-2.

3.1.35
enabled task state:
The only task state in which a task may make progress towards completion. A detailed definition of the enabled task state may be found in SAM-3.

3.1.36
Extended Unique Identifier, a 48-bit globally unique identifier (EUI-48):
The IEEE maintains a tutorial describing EUI-48 at http://standards.ieee.org/regauth/oui/tutorials/EUI48.html.

3.1.37
Extended Unique Identifier, a 64-bit globally unique identifier (EUI-64):
The IEEE maintains a tutorial describing EUI-64 at http://standards.ieee.org/regauth/oui/tutorials/EUI64.html.

3.1.38
faulted I_T nexus:
The I_T nexus on which a CHECK CONDITION status was returned that resulted in the establishment of an ACA. The faulted I_T nexus condition is cleared when the ACA condition is cleared. See SAM-3.

3.1.39
field:
A group of one or more contiguous bits, a part of a larger structure such as a CDB (see 3.1.17) or sense data (see 3.1.103).

3.1.40
hard reset:
A condition resulting from the events defined by SAM-3 in which the SCSI device performs the hard reset operations described in SAM-3, this standard, and the applicable command standards.

3.1.41
host:
A SCSI device with the characteristics of a primary computing device, typically a personal computer, workstation, server, minicomputer, mainframe computer, or auxiliary computing device. A host includes one or more SCSI initiator devices.

3.1.42
IEEE company_id:
Synonym for OUI (see 3.1.74).

3.1.43
I_T nexus:
A nexus between a SCSI initiator port and a SCSI target port. See SAM-3.

3.1.44
I_T nexus loss:
A condition resulting from the events defined by SAM-3 in which the SCSI device performs the I_T nexus loss operations described in SAM-3, this standard, and the applicable command standards.

3.1.45
I_T_L nexus:
A nexus between a SCSI initiator port, a SCSI target port, and a logical unit. See SAM-3.

3.1.46
I_T_L_Q nexus transaction:
The information transferred between SCSI ports in a single data structure with defined boundaries (e.g., an information unit).

3.1.47
idle power condition:
When a device server is capable of responding to all of its supported commands, including media access requests, but commands may take longer to complete than when in the active power condition. See 5.9.

3.1.48
implicit head of queue:
An optional processing model for specified commands wherein the specified commands may be treated as if they had been received with a HEAD OF QUEUE task attribute (see SAM-3 and 5.3).

3.1.49
initiator device name:
A SCSI device name (see 3.1.92) of a SCSI initiator device or of a SCSI target/initiator device when operating as a SCSI initiator device (see SAM-3).

3.1.50
initiator port:
Synonymous with SCSI initiator port (see 3.1.95).

3.1.51
initiator port identifier:
A value by which a SCSI initiator port (see 3.1.95) is referenced within a SCSI domain (see SAM-3).

3.1.52
initiator port name:
A SCSI port name (see 3.1.98) of a SCSI initiator port (see 3.1.95) or of a SCSI target/initiator port when operating as a SCSI initiator port (see SAM-3).

3.1.53
Internet protocol domain name:
The name of a computer or hierarchy of computers within the domain name system defined by the IETF (see RFC 1035 and RFC 1591). The Internet Assigned Numbers Authority maintains a list of domain name assignments at http://www.iana.org/assignments/domain-names.

3.1.54
Internet protocol number:
A coded value assigned to identify protocols that layer on the Internet protocol (see RFC 791). The Internet protocol number assigned to the transmission control protocol (TCP, see RFC 793) is six. The Internet Assigned Numbers Authority maintains a list of Internet protocol number assignments at http://www.iana.org/assignments/protocol-numbers.

3.1.55
linked command:
One in a series of SCSI commands processed by a single task that collectively make up a discrete I/O operation. A detailed definition of a linked command may be found in SAM-3.

3.1.56
least significant bit (LSB):
In a binary code, the bit or bit position with the smallest numerical weighting in a group of bits that, when taken as a whole, represent a numerical value (e.g., in the number 0001b, the bit that is set to one).

3.1.57
left-aligned:
A type of field containing ASCII data in which unused bytes are placed at the end of the field (highest offset) and are filled with ASCII space (20h) characters. See 4.4.1.

3.1.58
logical unit:
An externally addressable entity within a SCSI target device that implements a SCSI device model and contains a device server. A detailed definition of a logical unit may be found in SAM-3.

3.1.59
logical unit access control descriptor (LUACD):
The structure within an ACE (see 3.1.2) that identifies a logical unit to which access is allowed and specifies the LUN by which the logical unit is to be accessed (see 8.3.1.3.3).

3.1.60
logical unit inventory:
The list of the logical unit numbers reported by a REPORT LUNS command (see 6.21).

3.1.61
logical unit number (LUN):
An encoded 64-bit identifier for a logical unit. A detailed definition of a logical unit number may be found in SAM-3.

3.1.62
logical unit reset:
A condition resulting from the events defined by SAM-3 in which the logical unit performs the logical unit reset operations described in SAM-3, this standard, and the applicable command standards.

3.1.63
medium:
A physical entity that stores data in a nonvolatile manner (i.e., retained through a power cycle) in accordance with commands processed by the device server.

3.1.64
medium auxiliary memory (MAM):
An auxiliary memory residing on a medium that is accessible to the device server (e.g., a tape cartridge). Medium auxiliary memory may be nonvolatile and independent of the main function of the device server.

3.1.65
medium changer:
A device that mechanizes the movement of media to and from the SCSI device that records on or reads from the media. A detailed definition of a medium changer may be found in SMC-2.

3.1.66
most significant bit (MSB):
In a binary code, the bit or bit position with the largest numerical weighting in a group of bits that, when taken as a whole, represent a numerical value (e.g., in the number 1000b, the bit that is set to one).

3.1.67
name:
A label of an object that is unique within a specified context and should never change (e.g., the term name and worldwide identifier (WWID) may be interchangeable).

3.1.68
network address authority (NAA):
A field within a name (see 3.1.67) that specifies the format and length of that name. See 7.6.3.6 and FC-FS.

3.1.69
nexus:
A relationship between two SCSI devices, and the SCSI initiator port and SCSI target port objects within those SCSI devices. See SAM-3.

3.1.70
non-volatile cache memory:
Cache memory (see 3.1.15) that retains data through power cycles.

3.1.71
null-padded:
A type of field in which unused bytes are placed at the end of the field (i.e., highest offset) and are filled with ASCII null (00h) characters. See 4.4.2.

3.1.72
null-terminated:
A type of field in which the last used byte (i.e., highest offset) is required to contain an ASCII null (00h) character. See 4.4.2.

3.1.73
one:
The logical true condition of a variable.

3.1.74
organizationally unique identifier (OUI):
A numeric identifier that is assigned by the IEEE such that no assigned identifiers are identical. OUI is equivalent to company_id or IEEE company_id. The IEEE prefers OUI for EUI-48 identifiers (see 3.1.36) and company_id for EUI-64 identifiers (see 3.1.37). However, the numeric identifier is called an OUI when it is assigned by the IEEE. The IEEE maintains a tutorial describing the OUI at http://standards.ieee.org/regauth/oui/.

3.1.75
page:
A regular parameter structure (or format) used by several commands. These pages are identified with a value known as a page code.

3.1.76
persist through power loss:
An optional capability associated with some features that allows an application client to request that a device server maintain information regarding that feature across power failures.

3.1.77
persistent reservation holder:
The I_T nexus(es) that are allowed to release or change a persistent reservation without preempting it. See 5.6.9.

3.1.78
power cycle:
Power being removed from and later applied to a SCSI device.

3.1.79
power on:
A condition resulting from the events defined by SAM-3 in which the SCSI device performs the power on operations described in SAM-3, this standard, and the applicable command standards.

3.1.80
protocol identifier:
A coded value used in various fields to identify the protocol to which other fields apply. See 7.5.1.

3.1.81
protocol specific:
A requirement that is defined by a SCSI transport protocol standard (see clause 1). A detailed definition of protocol specific may be found in SAM-3.

3.1.82
protocol standard:
A SCSI standard that defines SCSI transport protocol (e.g., SAS, SPI-5, SBP-3, or FCP-2). See clause 2.

3.1.83
proxy token:
An identifier for a logical unit that may be used to gain temporary access to that logical unit in the presence of access controls (see 8.3.1.6.2).

3.1.84
request for comment (RFC):
The name given to standards developed by the Internet Engineering Task Force (see 2.3).

3.1.85
registered:
The condition that exists for an I_T nexus following the successful completion of a PERSISTENT RESERVE OUT command with a REGISTER service action, REGISTER AND IGNORE EXISTING KEY service action (see 5.6.6), or REGISTER AND MOVE service action (see 5.6.7) and lasting until the registration is removed (see 5.6.10).

3.1.86
registrant:
An I_T nexus that is registered (see 3.1.85).

3.1.87
right-aligned:
A type of field containing ASCII data in which unused bytes are placed at the start of the field (i.e., lowest offset) and are filled with ASCII space (20h) characters. See 4.4.1.

3.1.88
relative port identifier:
An identifier for a SCSI port (see 3.1.96) that is unique within a SCSI device (see 3.1.91). See SAM-3. Application clients may use the SCSI Ports VPD page (see 7.6.7) to determine relative port identifier values.

3.1.89
relative initiator port identifier:
A relative port identifier (see 3.1.88) for a SCSI initiator port (see 3.1.95).

3.1.90
relative target port identifier:
A relative port identifier (see 3.1.88) for a SCSI target port (see 3.1.101).

3.1.91
SCSI device:
A device that contains one or more SCSI ports that are connected to a service delivery subsystem and supports a SCSI application protocol (see SAM-3).

3.1.92
SCSI device name:
A name (see 3.1.67) of a SCSI device that is world wide unique within the protocol of a SCSI domain (see 3.1.93) in which the SCSI device has SCSI ports (see 3.1.96). The SCSI device name may be made available to other SCSI devices or SCSI ports in protocol specific ways. See SAM-3.

3.1.93
SCSI domain:
The interconnection of two or more SCSI devices and a service delivery subsystem. A detailed definition of a SCSI Domain may be found in SAM-3.

3.1.94
SCSI initiator device:
A SCSI device (see 3.1.91) containing application clients (see 3.1.9) and SCSI initiator ports (see 3.1.95) that originate device service and task management requests (see SAM-3) to be processed by a SCSI target device (see 3.1.100) and receives device service and task management responses from SCSI target devices.

3.1.95
SCSI initiator port:
A SCSI initiator device (see SAM-3) object acts as the connection between application clients and the service delivery subsystem through which requests and responses are routed.

3.1.96
SCSI port:
A port of a SCSI device that connects the application client, device server or task manager to the service delivery subsystem (see SAM-3).

3.1.97
SCSI port identifier:
A value by which a SCSI port is referenced within a domain. The SCSI port identifier is either an initiator port identifier (see 3.1.51) or a target port identifier (see 3.1.115).

3.1.98
SCSI port name:
A name (see 3.1.67) of a SCSI port that is world wide unique within the protocol of the SCSI domain of that SCSI port (see 3.1.96). The name may be made available to other SCSI devices or SCSI ports in that SCSI domain in protocol specific ways. See SAM-3.

3.1.99
SCSI Ports VPD page:
A VPD page that allows retrieval of information about all the SCSI ports in a SCSI target device or SCSI target/initiator device. See 7.6.7.

3.1.100
SCSI target device:
A SCSI device (see 3.1.91) containing logical units (see 3.1.58) and SCSI target ports (see 3.1.101) that receives device service and task management requests (see SAM-3) for processing and sends device service and task management responses to SCSI initiator devices.

3.1.101
SCSI target port:
A SCSI target device (see SAM-3) object that acts as the connection between device servers and task managers and the service delivery subsystem through which requests and responses are routed.

3.1.102
SCSI transport protocol standard:
A SCSI standard that defines a SCSI transport protocol (e.g., FCP-2, SAS, SRP, or SBP-3). See clause 2.

3.1.103
sense data:
Data describing an error or exceptional condition that a device server delivers to an application client in the same I_T_L_Q nexus transaction (see 3.1.46) as a CHECK CONDITION status or in response to a REQUEST SENSE command (see 6.27). The format of sense data is defined in 4.5.

3.1.104
sense key:
The contents of the SENSE KEY field (see 4.5) in the sense data.

3.1.105
service action:
A request describing a unit of work to be performed by a device server. A service action is an extension of a command. See SAM-3.

3.1.106
service delivery subsystem:
That part of a 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. See SAM-3.

3.1.107
standby power condition:
When a device server is capable of accepting commands, but not capable of processing media access commands. See 5.9.

3.1.108
status:
One byte of response information sent from a device server to an application client upon completion of each command. See SAM-3.

3.1.109
system:
One or more SCSI domains operating as a single configuration.

3.1.110
target device name:
A SCSI device name (see 3.1.92) of a SCSI target device or of a SCSI target/initiator device when operating as a SCSI target device (see SAM-3).

3.1.111
target port:
Synonymous with SCSI target port (see 3.1.101).

3.1.112
target port asymmetric access state:
The characteristic that defines the behavior of a target port and the allowable command set for a logical unit when commands and task management functions are routed through the target port maintaining that state (see 5.8.2.1).

3.1.113
target port group:
A set of target ports that are in the same target port asymmetric access state (see 3.1.112) at all times (see 5.8.2.1).

3.1.114
target port group asymmetric access state:
The target port asymmetric access state (see 3.1.112) common to the set of target ports in a target port group (see 3.1.113).

3.1.115
target port identifier:
A value by which a SCSI target port (see 3.1.101) is referenced within a SCSI domain (see SAM-3).

3.1.116
target port name:
A SCSI port name (see 3.1.98) of a SCSI target port or of a SCSI target/initiator port when operating as a SCSI target port (see SAM-3).

3.1.117
task:
An object within a logical unit that represents the work associated with a command or a group of linked commands. A detailed definition of a task may be found in SAM-3.

3.1.118
task set:
A group of tasks within a logical unit, whose interaction is dependent on the task management (queuing) and ACA rules. See SAM-3 and the Control mode page (see 7.4.6).

3.1.119
TCP port numbers:
One of the data needed to establish a TCP connection. TCP port numbers may be assigned to protocols that layer on TCP by the Internet Assigned Numbers Authority. The Internet Assigned Numbers Authority maintains a list of TCP port number assignments at http://www.iana.org/assignments/port-numbers.

3.1.120
third-party command:
A command sent to one SCSI device requesting than an operation be performed involving two other SCSI devices (e.g., the EXTENDED COPY command may perform copy operations between two or more SCSI devices none of which are the SCSI device to which the EXTENDED COPY command was sent).

3.1.121
unit attention condition:
A state that a logical unit maintains while it has asynchronous status information to report to the initiator ports associated with one or more I_T nexuses. See SAM-3.

3.1.122
universal time (UT):
The time at longitude zero, colloquially known as Greenwich Mean Time. See http://aa.usno.navy.mil/faq/docs/UT.html.

3.1.123
URI Schemes:
The Internet Assigned Numbers Authority maintains a list of schemes for URI and URL names at http://www.iana.org/assignments/uri-schemes.

3.1.124
UTF-8:
A character set that is a transformation format of the character set defined by ISO 10646. See RFC 2279.

3.1.125
vendor specific (VS):
Something (e.g., a bit, field, or code value) that is not defined by this standard and may be vendor defined.

3.1.126
volatile cache memory:
Cache memory (see 3.1.15) that does not retain data through power cycles.

3.1.127
well known logical unit:
A logical unit that only does specific functions (see clause 8). Well known logical units allow an application client to issue requests to receive and manage specific information usually relating to a SCSI target device.

3.1.128
well known logical unit number (W-LUN):
The logical unit number that identifies a well known logical unit.

3.1.129
zero:
The logical false condition of a variable.

3.1.130
zero-padded:
A type of field in which unused bytes are placed at the end of the field (i.e., highest offset) and are filled with zeros. See 4.4.2.

3.2 Acronyms

÷	divided by
×	multiplied by
<	less than
>	greater than
ACE	Access Control list Entry (see 3.1.2)
ACL	Access Control List (see 3.1.1)
ACA	Auto Contingent Allegiance (see 3.1.12)
ADC	Automation/Drive Interface - Commands (see clause 2)
ADT	Automation/Drive Interface - Transport Protocol (see clause 2)
ASC	Additional Sense Code (see 4.5)
ASCII	American Standard Code for Information Interchange (see 2.2)
ASCQ	Additional Sense Code Qualifier (see 4.5)
ATA	AT Attachment (see www.t13.org)
ATAPI	AT Attachment with Packet Interface (see www.t13.org)
CDB	Command Descriptor Block (see 3.1.17)
CRC	Cyclic Redundancy Check
D_ID	Destination Identifier (defined in FC-FS, see clause 2)
EUI-48	Extended Unique Identifier, a 48-bit globally unique identifier (see 3.1.36)
EUI-64	Extended Unique Identifier, a 64-bit globally unique identifier (see 3.1.37)
FC-FS	Fibre Channel Framing and Signaling Interface (see clause 2)
FCP-2	Fibre Channel Protocol for SCSI -2 (see clause 2)
HTTP	Hypertext Transfer Protocol (see RFC 2616)
ID	Identifier or Identification
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force (see 2.3)
IP	Internet Protocol (see 2.3)
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
iSCSI	Internet SCSI (see 2.3)
ISO	Organization for International Standards
LBA	Logical Block Address
LSB	Least Significant Bit (see 3.1.56)
LUACD	Logical Unit Access Control Descriptor (see 3.1.59)
LUN	Logical Unit Number (see 3.1.61)
MAM	Medium Auxiliary Memory (see 3.1.64)
MMC-4	SCSI Multi-Media Commands -4 (see clause 2)
MSB	Most Significant Bit (see 3.1.66)
NAA	Network Address Authority (see 3.1.68)
n/a	not applicable
INCITS	InterNational Committee for Information Technology Standards
OCRW	SCSI Specification for Optical Card Reader/Writer (see clause 2)
OSD	Object-based Storage Devices Commands (see clause 2)
OUI	Organizationally Unique Identifier (see 3.1.74)
RAID	Redundant Array of Independent Disks

RBC	SCSI Reduced Block Commands (see clause 2)
RDMA	Remote Direct Memory Access (see SRP)
RFC	Request For Comments (see 3.1.84)
RMC	SCSI Reduced Multi-Media Commands (see clause 2)
SAM-2	SCSI Architecture Model -2 (see clause 2)
SAM-3	SCSI Architecture Model -3 (see clause 2)
SAT	SCSI / ATA Translation (see clause 2)
SBC-2	SCSI Block Commands -2 (see clause 2)
SBP-3	Serial Bus Protocol -3 (see clause 2)
SCC-2	SCSI Controller Commands -2 (see clause 2)
SCC-3	SCSI Controller Commands -3 (see clause 2)
SCSI	The architecture defined by the family of standards described in clause 1
SES	SCSI-3 Enclosure Services (see clause 2)
SES-2	SCSI Enclosure Services -2 (see clause 2)
SMC-2	SCSI Media Changer Commands -2 (see clause 2)
SPC	SCSI-3 Primary Commands (ANSI INCITS 301-1997, see clause 2)
SPC-2	SCSI Primary Commands -2 (see clause 2)
SPC-3	SCSI Primary Commands -3 (this standard, see clause 2)
SPI-5	SCSI Parallel Interface -5 (see clause 2)
SRP	SCSI RDMA Protocol (see clause 2)
SSC-2	SCSI Stream Commands -2 (see clause 2)
TCP	Transmission Control Protocol (see RFC 793)
URI	Uniform Resource Identifier (see RFC 2396, RFC 3305, and 3.1.123)
URL	Uniform Resource Locator (see RFC 2396, RFC 3305, and 3.1.123)
UT	Universal time (see 3.1.122)
USB	Universal Serial Bus (see www.usb.org)
VPD	Vital Product Data (see 7.6)
VS	Vendor Specific (see 3.1.125)
W-LUN	Well known logical unit number (see 3.1.128)

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
ignored:
A keyword used to describe an unused bit, byte, word, field or code value. The contents or value of an ignored bit, byte, word, 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.

3.3.3
invalid:
A keyword used to describe an illegal or unsupported bit, byte, word, field or code value. Receipt of an invalid bit, byte, word, field or code value shall be reported as an error.

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 indicates flexibility of choice with no implied preference (equivalent to "may or may not").

3.3.6
may not:
A keyword that indicates 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 bits, bytes, words, fields and code values that are set aside for future standardization. A reserved bit, byte, word or field shall be set to zero, or in accordance with a future extension to this standard. Recipients are not required to check reserved bits, bytes, words or fields for zero values. Receipt of reserved code values in defined fields shall be reported as an error.

3.3.10
restricted:
A keyword referring to bits, bytes, words, and fields that are set aside for use in other SCSI standards. A restricted bit, byte, word, or field shall be treated as a reserved bit, byte, word or field for the purposes of the requirements defined in this standard.

3.3.11
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.

3.3.12
should:
A keyword indicating flexibility of choice with a strongly preferred alternative; equivalent to the phrase "it is strongly recommended".

3.3.13
x or **xx:**
The value of the bit or field is not relevant.

3.4 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 3.1 or in the text where they first appear. Names of commands, statuses, sense keys, and additional sense codes are in all uppercase (e.g., REQUEST SENSE). Lowercase is used for words having the normal English meaning.

If there is more than one CDB length for a particular command (e.g., MODE SENSE(6) and MODE SENSE(10)) and the name of the command is used in a sentence without any CDB length descriptor (e.g., MODE SENSE), then the condition specified in the sentence applies to all CDB lengths for that command.

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.

A binary number is represented in this standard by any sequence of digits comprised of only the Western-Arabic numerals 0 and 1 immediately followed by a lower-case b (e.g., 0101b). Underscores or spaces may be included in binary number representations to increase readability or delineate field boundaries (e.g., 0 0101 1010b or 0_0101_1010b).

A hexadecimal number is represented in this standard by any sequence of digits comprised of only the Western-Arabic numerals 0 through 9 and/or the upper-case English letters A through F immediately followed by a lower-case h (e.g., FA23h). Underscores or spaces may be included in hexadecimal number representations to increase readability or delineate field boundaries (e.g., B FD8C FA23h or B_FD8C_FA23h).

A decimal number is represented in this standard by any sequence of digits comprised of only the Western-Arabic numerals 0 through 9 not immediately followed by a lower-case b or lower-case h (e.g., 25).

When the value of the bit or field is not relevant, x or xx appears in place of a specific value.

This standard uses the ISO convention for representing decimal numbers (e.g., the thousands and higher multiples are separated by a space and a comma is used as the decimal point). Table 1 shows some examples of decimal numbers represented using the ISO and American conventions.

Table 1 — ISO and American numbering conventions examples

ISO	American
0,6	0.6
3,141 592 65	3.14159265
1 000	1,000
1 323 462,95	1,323,462.95

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

3.5 Bit and byte ordering

This subclause describes the representation of fields in a table that defines the format of a SCSI structure (e.g., the format of a CDB).

If a field consists of more than one bit and contains a single value (e.g., a number), the least significant bit (LSB) is shown on the right and the most significant bit (MSB) is shown on the left (e.g., in a byte, bit 7 is the MSB and is shown on the left, and bit 0 is the LSB and is shown on the right). The MSB and LSB are not labeled if the field consists of 8 or fewer bits.

If a field consists of more than one byte and contains a single value, the byte containing the MSB is stored at the lowest address and the byte containing the LSB is stored at the highest address (i.e., big-endian byte ordering). The MSB and LSB are labeled.

If a field consists of more than one byte and contains multiple fields each with their own values (e.g., a descriptor), there is no MSB and LSB of the field itself and thus there are no MSB and LSB labels. Each individual field has an MSB and LSB that are labeled as appropriate in the table, if any, that describes the format of the sub-structure having multiple fields.

If a field contains a text string (e.g., ASCII or UTF-8), the MSB label is the MSB of the first character and the LSB label is the LSB of the last character.

When required for clarity, multiple byte fields may be represented with only two rows in a table. This condition is represented by values in the byte number column not increasing by one in each subsequent table row, thus indicating the presence of additional bytes.

3.6 Notation conventions

3.6.1 Notation for byte encoded character strings

When this standard requires one or more bytes to contain specific encoded characters, the specific characters are enclosed in double quotation marks. The double quotation marks identify the start and end of the characters that are required to be encoded but are not themselves to be encoded. The characters that are to be encoded are shown in the case that is to be encoded.

The encoded characters and the double quotation marks that enclose them are preceded by text that specifies the character encoding methodology and the number of characters required to be encoded.

Using the notation described in this subclause, stating that eleven ASCII characters “SCSI device” are to be encoded would be the same writing out the following sequence of byte values: 53h 43h 53h 49h 20h 64h 65h 76h 69h 63h 65h.

3.6.2 Notation for procedure calls

In this standard, the model for functional interfaces between objects is a procedure call. Such interfaces are specified using the following notation:

[Result =] Procedure Name (IN ([input-1] [,input-2] …]), OUT ([output-1] [,output-2] …))

Where:

Result: A single value representing the outcome of the procedure call.

Procedure Name: A descriptive name for the function modeled by the procedure call. When the procedure call model is used to describe a SCSI transport protocol service, the procedure name is the same as the service name.

Input-1, Input-2, …: A comma-separated list of names identifying caller-supplied input arguments.

Output-1, Output-2, …: A comma-separated list of names identifying output arguments to be returned by the procedure call.

"[…]": Brackets enclosing optional or conditional arguments.

This notation allows arguments to be specified as inputs and outputs. The following is an example of a procedure call specification:

Found = Search (IN (Pattern, Item List), OUT ([Item Found]))

Where:

Found = Flag
Flag: if set to one, indicates that a matching item was located.

Input Arguments:

Pattern = … /* Definition of Pattern argument */
Argument containing the search pattern.

Item List = Item<NN> /* Definition of Item List as an array of NN Item arguments*/
Contains the items to be searched for a match.

Output Arguments:

Item Found = Item … /* Item located by the search procedure call */
This argument is only returned if the search succeeds.

3.6.3 Notation for state diagrams

All state diagrams use the notation shown in figure 2.

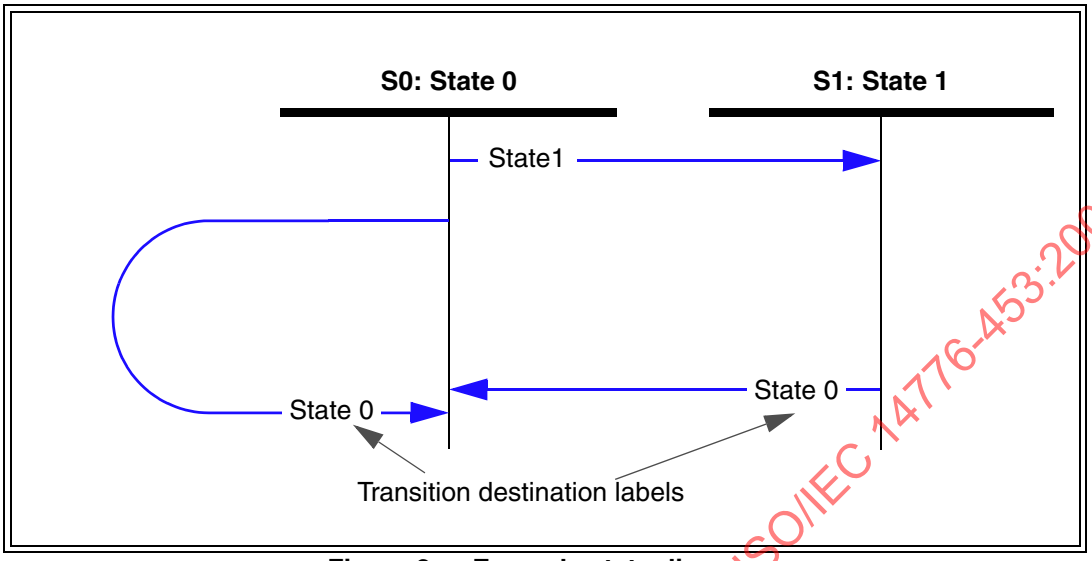

Figure 2 — Example state diagram

The state diagram is followed by subclauses describing the states and state transitions.

Each state and state 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.

A system specified in this manner has the following properties:

a) Time elapses only within discrete states; and
b) State transitions are logically instantaneous.

3.6.4 Notation for binary power multipliers

The nomenclature used for binary power multiplier values in this standard is based on IEC 60027-2:2000 (see table 2).

Table 2 — Binary power multiplier nomenclature

Prefix	Abbreviation	Power of two	Example
kibi	Ki	2^{10} or 1 024	one kibibit is 1 Kib is 1 024 bits
mebi	Mi	2^{20}	one mebibyte is 1 MiB is 1 048 576 bytes
gebi	Gi	2^{30}	one gebibyte is 1 GiB is 1 073 741 824 bytes
tebi	Ti	2^{40}	
pebi	Pi	2^{50}	
exbi	Ei	2^{60}	

4 General Concepts

4.1 Introduction

This standard defines behaviors that are common to all SCSI device models (see clause 5). This standard defines the SCSI commands that are basic to more than one device model and the SCSI commands that may apply to any device model (see clause 6). This standard defines the parameters that are basic to more than one device model (see clause 7).

4.2 The request-response model

The SCSI command set assumes an underlying request-response protocol. The fundamental properties of the request-response protocol are defined in SAM-3. Action on SCSI commands shall not be deemed completed until a response is received. The response shall include a status that indicates the final disposition of the command. As per SAM-3, the request-response protocol may be modeled as a procedure call, specifically:

Service response = Execute Command (IN (I_T_L_Q Nexus, CDB, Task Attribute, [Data-In Buffer Size], [Data-Out Buffer], [Data-Out Buffer Size], [Command Reference Number], [Task Priority]), OUT ([Data-In Buffer], [Sense Data], [Sense Data Length], Status))

SAM-3 defines all of the inputs and outputs in the procedure call above. As they may apply to any SCSI device, this standard defines the contents of the following procedure inputs and outputs; CDB, Data-Out Buffer, Data-In Buffer, and Sense Data. This standard does not define all possible instances of these procedure inputs and outputs. This standard defines only those instances that may apply to any SCSI device. Instances of the procedure inputs and outputs that apply to specific SCSI device models are defined in the applicable SCSI command standards (see 3.1.18).

This standard references values returned via the Status output parameter (e.g., CHECK CONDITION, RESERVATION CONFLICT). Status values are not defined by this standard. SAM-3 defines all Status values.

The entity that makes the procedure call is an application client (see SAM-3). The procedure call's representation arrives at the SCSI target device in the form of a device service request. The entity that performs the work of the procedure call is a device server (see SAM-3).

4.3 The Command Descriptor Block (CDB)

4.3.1 CDB usage and structure

A command is communicated by sending a command descriptor block (CDB) to the device server. For several commands, the CDB is accompanied by a list of parameters in the Data-Out Buffer. See the specific commands for detailed information.

If a logical unit validates reserved CDB fields and receives a reserved field within the CDB that is not zero, then the logical unit shall terminate the command with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

If a logical unit receives a reserved CDB code value in a field other than the OPERATION CODE field, then the logical unit shall terminate the command with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

The fixed length CDB formats are described in 4.3.2. The variable length CDB formats are described in 4.3.3. The CDB fields that are common to most commands are described in 4.3.4. The fields shown in 4.3.2 and 4.3.3 and described in 4.3.4 are used consistently by most commands. However, the actual usage of any field (except OPERATION CODE and CONTROL) is described in the subclause defining that command. If a device server receives a

CDB containing an operation code that is invalid or not supported, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID COMMAND OPERATION CODE.

For all commands, if there is an invalid parameter in the CDB, the device server shall terminate the command without altering the medium.

4.3.2 The fixed length CDB formats

All fixed length CDBs shall have an OPERATION CODE field as their first byte and a CONTROL byte as their last byte. Table 3 shows the typical format of a 6-byte CDB. Table 4 shows the typical format of a 10-byte CDB. Table 5 shows the typical format of a 12-byte CDB. Table 6 shows the typical format of a 16-byte CDB. Table 7 shows the format of a 16-byte CDB for commands that provide for a long LBA.

Table 3 — Typical CDB for 6-byte commands

Bit Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE							
1	miscellaneous CDB information			(MSB)				
2	LOGICAL BLOCK ADDRESS (if required)							
3								(LSB)
4	TRANSFER LENGTH (if required) PARAMETER LIST LENGTH (if required) ALLOCATION LENGTH (if required)							
5	CONTROL							

Table 4 — Typical CDB for 10-byte commands

Bit Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE							
1	miscellaneous CDB information			SERVICE ACTION (if required)				
2	(MSB)	LOGICAL BLOCK ADDRESS (if required)						
3								
4								
5								(LSB)
6	miscellaneous CDB information							
7	(MSB)	TRANSFER LENGTH (if required) PARAMETER LIST LENGTH (if required) ALLOCATION LENGTH (if required)						
8								(LSB)
9	CONTROL							

Table 5 — Typical CDB for 12-byte commands

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE							
1	miscellaneous CDB information			SERVICE ACTION (if required)				
2	(MSB)							
3				LOGICAL BLOCK ADDRESS (if required)				
4								
5								(LSB)
6	(MSB)							
7				TRANSFER LENGTH (if required) PARAMETER LIST LENGTH (if required) ALLOCATION LENGTH (if required)				
8								
9								(LSB)
10	miscellaneous CDB information							
11	CONTROL							

Table 6 — Typical CDB for 16-byte commands

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE							
1	miscellaneous CDB information			SERVICE ACTION (if required)				
2	(MSB)							
3				LOGICAL BLOCK ADDRESS (if required)				
4								
5								(LSB)
6								
7				Additional CDB data (if required)				
8								
9								
10	(MSB)							
11				TRANSFER LENGTH (if required) PARAMETER LIST LENGTH (if required) ALLOCATION LENGTH (if required)				
12								
13								(LSB)
14	miscellaneous CDB information							
15	CONTROL							

Table 7 — Typical CDB for long LBA 16-byte commands

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE							
1	miscellaneous CDB information							
2	(MSB)							
3								
4								
5								
6				LOGICAL BLOCK ADDRESS				
7								
8								
9								(LSB)
10	(MSB)							
11				TRANSFER LENGTH (if required)				
12				PARAMETER LIST LENGTH (if required) ALLOCATION LENGTH (if required)				
13								(LSB)
14	miscellaneous CDB information							
15	CONTROL							

4.3.3 The variable length CDB formats

The first byte of a variable length CDB shall contain the operation code 7Fh. The CONTROL byte is the second byte in the variable length CDB (see table 8).

Table 8 — Typical variable length CDB

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (7Fh)							
1	CONTROL							
2	miscellaneous CDB information							
3	miscellaneous CDB information							
4	miscellaneous CDB information							
5	miscellaneous CDB information							
6	miscellaneous CDB information							
7	ADDITIONAL CDB LENGTH (n-7)							
8	(MSB)							
9				SERVICE ACTION				(LSB)
10								
n	Service action specific fields							

The ADDITIONAL CDB LENGTH field specifies the number of additional CDB bytes. This value in the ADDITIONAL CDB LENGTH field shall be a multiple of 4. If the number of CDB bytes delivered by the service delivery subsystem is not sufficient to contain the number of bytes specified by the ADDITIONAL CDB LENGTH field, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

The SERVICE ACTION field specifies the action being requested by the application client. The SERVICE ACTION field is required in the variable length CDB format and is described in 4.3.4.2. Each service action code description defines a number of service action specific fields that are needed for that service action.

A 32-byte variable length CDB format is defined for long LBA operations (see table 9).

Table 9 — Typical variable length CDB for long LBA 32-byte commands

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (7Fh)							
1	CONTROL							
2	miscellaneous CDB information							
3	miscellaneous CDB information							
4	miscellaneous CDB information							
5	miscellaneous CDB information							
6	miscellaneous CDB information							
7	ADDITIONAL CDB LENGTH (18h)							
8	(MSB)			SERVICE ACTION				
9								(LSB)
10	miscellaneous CDB information			DPO	FUA	miscellaneous CDB information		
11	miscellaneous CDB information							
12	(MSB)			LOGICAL BLOCK ADDRESS				
19								(LSB)
20				miscellaneous CDB information				
27								
28	(MSB)			TRANSFER LENGTH (if required) PARAMETER LIST LENGTH (if required) ALLOCATION LENGTH (if required)				
31								(LSB)

4.3.4 Common CDB fields

4.3.4.1 Operation code

The first byte of a SCSI CDB shall contain an operation code identifying the operation being requested by the CDB. Some operation codes provide for modification of their operation based on a service action (see 4.3.4.2). In such cases, the operation code and service action code combine to identify the operation being requested. The location of the SERVICE ACTION field in the CDB varies depending on the operation code value.

The OPERATION CODE (see table 10) of the CDB has a GROUP CODE field and a COMMAND CODE field. The three-bit GROUP CODE field provides for eight groups of command codes. The five-bit COMMAND CODE field provides for thirty-two command codes in each group. A total of 256 possible operation codes exist. Operation codes are defined in this standard and other command standards (see 3.1.18). The group code value shall determine the length of the CDB (see table 11).

Table 10 — OPERATION CODE **byte**

Bit	7	6	5	4	3	2	1	0
	GROUP CODE			COMMAND CODE				

The value in the GROUP CODE field specifies one of the groups shown in table 11.

Table 11 — Group Code values

Group Code	Meaning	Typical CDB format
000b	6 byte commands	see table 3 in 4.3.2
001b	10 byte commands	see table 4 in 4.3.2
010b	10 byte commands	see table 4 in 4.3.2
011b	reserved [a]	
100b	16 byte commands	see table 6 and table 7 in 4.3.2
101b	12 byte commands	see table 5 in 4.3.2
110b	vendor specific	
111b	vendor specific	

[a] The format of the commands using the group code 011b and operation code 7Fh is described in 4.3.3. With the exception of operation code 7Fh, all group code 011b operation codes are reserved.

4.3.4.2 Service action

All CDB formats except the 6-byte format provide for a SERVICE ACTION field containing a coded value identifying a function to be performed under the more general command function specified in the OPERATION CODE field. While the SERVICE ACTION field is defined for CDB formats, it is used as described in this subclause only in those CDB formats that contain a SERVICE ACTION field. When the specific field SERVICE ACTION is not defined in a CDB format, the bits identified as the SERVICE ACTION field in a CDB shall be used or reserved as specified by the particular CDB format.

4.3.4.3 Logical block address

The logical block addresses on a logical unit or within a volume or partition shall begin with block zero and be contiguous up to the last logical block of that logical unit or within that volume or partition.

A six-byte CDB may contain a 21-bit LOGICAL BLOCK ADDRESS field. The ten-byte and the twelve-byte CDBs may contain 32-bit LOGICAL BLOCK ADDRESS fields. The sixteen-byte CDB has two formats: one allows a 32-bit LOGICAL BLOCK ADDRESS field (see table 6) and the other allows a 64-bit LOGICAL BLOCK ADDRESS field (see table 7). LOGICAL BLOCK ADDRESS fields in additional parameter data have their length specified for each occurrence. See the specific command descriptions.

4.3.4.4 Transfer length

The TRANSFER LENGTH field specifies the amount of data to be transferred, usually the number of blocks. Some commands use transfer length to specify the requested number of bytes to be sent as defined in the command description.

Commands that use one byte for the TRANSFER LENGTH field may allow up to 256 blocks or 256 bytes of data to be transferred by one command.

In commands that use multiple bytes for the TRANSFER LENGTH field, a transfer length of zero specifies that no data transfer shall take place. A value of one or greater specifies the number of blocks or bytes that shall be transferred.

Refer to the specific command description for further information.

4.3.4.5 Parameter list length

The PARAMETER LIST LENGTH field is used to specify the number of bytes sent from the Data-Out Buffer. This field is typically used in CDBs for parameters that are sent to a device server (e.g., mode parameters, diagnostic parameters, log parameters). A parameter length of zero specifies that no data shall be transferred. This condition shall not be considered as an error, unless otherwise specified.

4.3.4.6 Allocation length

The ALLOCATION LENGTH field specifies the maximum number of bytes that an application client has allocated in the Data-In Buffer. An allocation length of zero specifies that no data shall be transferred. This condition shall not be considered as an error. The device server shall terminate transfers to the Data-In Buffer when the number of bytes specified by the ALLOCATION LENGTH field have been transferred or when all available data have been transferred, whichever is less. The allocation length is used to limit the maximum amount of variable length data (e.g., mode data, log data, diagnostic data) returned to an application client. If the information being transferred to the Data-In Buffer includes fields containing counts of the number of bytes in some or all of the data, then the contents of these fields shall not be altered to reflect the truncation, if any, that results from an insufficient ALLOCATION LENGTH value, unless the standard that describes the Data-In Buffer format states otherwise.

If the amount of information to be transferred exceeds the maximum value that the ALLOCATION LENGTH field is capable of specifying, the device server shall transfer no data and terminate the command with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

4.3.4.7 Control

The contents of the CONTROL byte are defined in SAM-3. The CONTROL byte has the same definition for all commands.

4.4 Data field requirements

4.4.1 ASCII data field requirements

ASCII data fields shall contain only ASCII printable characters (i.e., code values 20h through 7Eh) and may be terminated with one or more ASCII null (00h) characters.

ASCII data fields described as being left-aligned shall have any unused bytes at the end of the field (i.e., highest offset) and the unused bytes shall be filled with ASCII space characters (20h).

ASCII data fields described as being right-aligned shall have any unused bytes at the start of the field (i.e., lowest offset) and the unused bytes shall be filled with ASCII space characters (20h).

4.4.2 Null data field termination and zero padding requirements

A data field that is described as being null-terminated shall have one byte containing an ASCII or UTF-8 null (00h) character in the last used byte (i.e., highest offset) of the field and no other bytes in the field shall contain the ASCII/UTF-8 null character.

A data field may be specified to be a fixed length. The length specified for a data field may be greater than the length required to contain the contents of the field. A data field may be specified to have a length that is a multiple of a given value (e.g., a multiple of four bytes). When such fields are described as being null-padded, the bytes at the end of the field that are not needed to contain the field data shall contain ASCII or UTF-8 null (00h) characters. When such fields are described as being zero-padded, the bytes at the end of the field that are not needed to contain the field data shall contain zeros.

NOTE 1 - There is no difference between the pad byte contents in null-padded and zero-padded fields. The difference is in the format of the other bytes in the field.

A data field that is described as being both null-terminated and null-padded shall have at least one byte containing an ASCII or UTF-8 null (00h) character in the end of the field (i.e., highest offset) and may have more than one byte containing ASCII or UTF-8 null characters, if needed, to meet the specified field length requirements. If more than one byte in a null-terminated, null-padded field contains the ASCII or UTF-8 null character, then all the bytes containing the ASCII or UTF-8 null character shall be at the end of the field (i.e., only the highest offsets).

4.5 Sense data

4.5.1 Sense data introduction

Sense data shall be returned in the same I_T_L_Q nexus transaction (see 3.1.46) as a CHECK CONDITION status and as parameter data in response to the REQUEST SENSE command (see 6.27). Sense data returned in the same I_T_L_Q nexus transaction as a CHECK CONDITION status shall be either fixed or descriptor format sense data format based on the value of the D_SENSE bit in the Control mode page (see 7.4.6). The REQUEST SENSE command may be used to request either the fixed format sense data or the descriptor format sense data.

The first byte of all sense data contains the RESPONSE CODE field that indicates the error type and format of the sense data (see table 12).

Table 12 — Sense data response codes

Response Code	Error type		Sense data format	
	Description	Reference	Description	Reference
00h - 6Fh	Reserved			
70h	Current	4.5.4	Fixed	4.5.3
71h	Deferred	4.5.5	Fixed	4.5.3
72h	Current	4.5.4	Descriptor	4.5.2
73h	Deferred	4.5.5	Descriptor	4.5.2
74h - 7Eh	Reserved			
7Fh	Vendor specific			

The RESPONSE CODE field shall be set to 70h in all unit attention sense data in which:

a) The ADDITIONAL SENSE CODE field is set to 29h; or
b) The additional sense code is set to MODE PARAMETERS CHANGED.

4.5.2 Descriptor format sense data

4.5.2.1 Descriptor format sense data overview

The descriptor format sense data for response codes 72h (current errors) and 73h (deferred errors) is defined in table 13.

Table 13 — Descriptor format sense data

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved	RESPONSE CODE (72h or 73h)						
1	Reserved				SENSE KEY			
2	ADDITIONAL SENSE CODE							
3	ADDITIONAL SENSE CODE QUALIFIER							
4	Reserved							
5								
6								
7	ADDITIONAL SENSE LENGTH (n-7)							
	Sense data descriptor(s)							
8	Sense data descriptor 0 (see table 14)							
	. . .							
n	Sense data descriptor x (see table 14)							

The contents of the RESPONSE CODE field indicate the error type and format of the sense data (see 4.5.1). For descriptor format sense data, the RESPONSE CODE field shall be set to 72h or 73h.

The SENSE KEY, ADDITIONAL SENSE CODE and ADDITIONAL SENSE CODE QUALIFIER fields provide a hierarchy of information. The hierarchy provides a top-down approach for an application client to determine information relating to the error and exception conditions.

The SENSE KEY field indicates generic information describing an error or exception condition. The sense keys are defined in 4.5.6.

The ADDITIONAL SENSE CODE (ASC) field indicates further information related to the error or exception condition reported in the SENSE KEY field. Support of the additional sense codes not required by this standard is optional. A list of additional sense codes is in 4.5.6. If the device server does not have further information related to the error or exception condition, the additional sense code shall be set to zero.

The ADDITIONAL SENSE CODE QUALIFIER (ASCQ) field indicates detailed information related to the additional sense code. If the error or exception condition is reported by the device server, the value returned shall be as specified in 4.5.6. If the device server does not have detailed information related to the error or exception condition, the additional sense code qualifier shall be set to zero.

The ADDITIONAL SENSE LENGTH field indicates the number of additional sense bytes that follow. The additional sense length shall be less than or equal to 244 (i.e., limiting the total length of the sense data to 252 bytes). If the sense data is being returned as parameter data by a REQUEST SENSE command, then the relationship between the ADDITIONAL SENSE LENGTH field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6.

Sense data descriptors (see table 14) provide specific sense information. A given type of sense data descriptor shall be included in the sense data only when the information it contains is valid.

Table 14 — Sense data descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE							
1	ADDITIONAL LENGTH (n-1)							
2	Sense data descriptor specific							
n								

The DESCRIPTOR TYPE field contains a type code (see table 15) that identifies the type of sense data descriptor. No more than one sense data descriptor of each type shall be included in the descriptor format sense data.

Table 15 — Sense data descriptor types

Type	Description	Reference
00h	Information	4.5.2.2
01h	Command specific information	4.5.2.3
02h	Sense key specific	4.5.2.4
03h	Field replaceable unit	4.5.2.5
04h	Stream commands	SSC-3
05h	Block commands	SBC-2
06h	OSD object identification	OSD
07h	OSD response integrity check value	OSD
08h	OSD attribute identification	OSD
09h	ATA Return	SAT
0Ah - 7Fh	Reserved	
80h - FFh	Vendor specific	4.5.2.6

The ADDITIONAL LENGTH field indicates the number of sense data descriptor specific bytes that follow in the sense data descriptor.

4.5.2.2 Information sense data descriptor

The information sense data descriptor (see table 16) provides information that is device-type or command specific and is defined in a command standard (see 3.1.18).

Table 16 — Information sense data descriptor format

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	INFORMATION							
11								

The DESCRIPTOR TYPE and ADDITIONAL LENGTH fields are described in 4.5.2.1. For the information sense data descriptor, the DESCRIPTOR TYPE field shall be set to 00h and the ADDITIONAL LENGTH field shall be set to 0Ah.

The VALID bit shall be set to one.

NOTE 2 - In previous versions of this standard and in the fixed format sense data, the VALID bit indicates whether the contents of the INFORMATION field is valid as defined by a command standard. Since the contents of the INFORMATION field are valid whenever an information sense data descriptor is included in the sense data, the only legal value for the VALID bit is set to one.

The contents of the INFORMATION field are device-type or command specific and are defined in a command standard (see 3.1.18). When a four byte quantity is stored in the INFORMATION field, the first four bytes shall be zero.

4.5.2.3 Command-specific information sense data descriptor

The command-specific information sense data descriptor (see table 17) provides information that depends on the command on which the exception condition occurred.

Table 17 — Command-specific information sense data descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE (01h)							
1	ADDITIONAL LENGTH (0Ah)							
2	Reserved							
3	Reserved							
4	COMMAND-SPECIFIC INFORMATION							
11								

The DESCRIPTOR TYPE and ADDITIONAL LENGTH fields are described in 4.5.2.1. For the command-specific information sense data descriptor, the DESCRIPTOR TYPE field shall be set to 01h and the ADDITIONAL LENGTH field shall be set to 0Ah.

The COMMAND-SPECIFIC INFORMATION field contains information that depends on the command on which the exception condition occurred. When a four byte quantity is stored in the COMMAND-SPECIFIC INFORMATION field, the first four bytes shall be zero.

Further meaning for the COMMAND-SPECIFIC INFORMATION field is defined within the command description in the appropriate command standard (e.g., see SBC-2 for the REASSIGN BLOCKS commands, or see 6.3 for the EXTENDED COPY command).

4.5.2.4 Sense key specific sense data descriptor

4.5.2.4.1 Sense key specific sense data descriptor introduction

The sense key specific sense data descriptor (see table 18) provides additional information about the exception condition. The format and content of the sense-key specific data depends on the value in the SENSE KEY field (see 4.5.2.1).

Table 18 — Sense key specific sense data descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE (02h)							
1	ADDITIONAL LENGTH (06h)							
2	Reserved							
3	Reserved							
4	SKSV (1b)	SENSE KEY SPECIFIC						
5								
6								
7	Reserved							

The DESCRIPTOR TYPE and ADDITIONAL LENGTH fields are described in 4.5.2.1. For the sense-key specific sense data descriptor, the DESCRIPTOR TYPE field shall be set to 01h and the ADDITIONAL LENGTH field shall be set to 06h.

The sense-key specific valid (SKSV) bit shall be set to one.

NOTE 3 - In previous versions of this standard and in the fixed format sense data, the SKSV bit indicates whether the contents of the SENSE KEY SPECIFIC field are valid as defined by a command standard. Since the contents of the SENSE KEY SPECIFIC field are valid whenever a sense key specific sense data descriptor is included in the sense data, the only legal value for the SKSV bit is set to one.

The definition of the SENSE KEY SPECIFIC field (see table 19) is determined by the value of the SENSE KEY field (see 4.5.2.1).

Table 19 — Sense key specific field definitions

Sense Key	Sense Key Specific Field Definition	Reference
ILLEGAL REQUEST	Field pointer	4.5.2.4.2
HARDWARE ERROR, MEDIUM ERROR, or RECOVERED ERROR	Actual retry count	4.5.2.4.3
NO SENSE or NOT READY	Progress indication	4.5.2.4.4
COPY ABORTED	Segment pointer	4.5.2.4.5
All other sense keys	The sense key specific sense data descriptor shall not appear in the descriptor format sense data and the SKSV bit (see 4.5.3) shall be set to zero in the fixed format sense data.	

4.5.2.4.2 Field pointer sense key specific data

If the sense key is ILLEGAL REQUEST, then the SENSE KEY SPECIFIC field shall be as shown in table 20.

Table 20 — Field pointer sense key specific data

Bit Byte	7	6	5	4	3	2	1	0
0	SKSV (1b)	C/D	Reserved		BPV	BIT POINTER		
1	(MSB)	FIELD POINTER						
2								(LSB)

The SKSV bit is described in 4.5.2.4.1 for descriptor format sense data and in 4.5.3 for fixed format sense data.

A command data (C/D) bit set to one indicates that the illegal parameter is in the CDB. A C/D bit set to zero indicates that the illegal parameter is in the data parameters sent by the application client in the Data-Out Buffer.

A bit pointer valid (BPV) bit set to zero indicates that the value in the BIT POINTER field is not valid. A BPV bit set to one indicates that the BIT POINTER field specifies which bit of the byte designated by the FIELD POINTER field is in error. When a multiple-bit field is in error, the BIT POINTER field shall point to the first bit (i.e., the left-most bit) of the field.

The FIELD POINTER field indicates which byte of the CDB or of the parameter data was in error. Bytes are numbered starting from zero, as shown in the tables describing the commands and parameters. When a multiple-byte field is in error, the field pointer shall point to the first byte (i.e., the left-most byte) of the field. If several consecutive bytes are reserved, each shall be treated as a single-byte field.

NOTE 4 - The bytes identified as being in error are not necessarily the bytes that need to be changed to correct the problem.

4.5.2.4.3 Actual retry count sense key specific data

If the sense key is HARDWARE ERROR, MEDIUM ERROR, or RECOVERED ERROR, then the SENSE KEY SPECIFIC field shall be as shown in table 21.

Table 21 — Actual retry count sense key specific data

Bit Byte	7	6	5	4	3	2	1	0
0	SKSV (1b)	Reserved						
1	(MSB)	ACTUAL RETRY COUNT						
2								(LSB)

The SKSV bit is described in 4.5.2.4.1 for descriptor format sense data and in 4.5.3 for fixed format sense data.

The ACTUAL RETRY COUNT field returns vendor specific information on the number of retries of the recovery algorithm used in attempting to recover an error or exception condition.

NOTE 5 - This field should be computed in the same way as the retry count fields within the Read-Write Error Recovery mode page.

4.5.2.4.4 Progress indication sense key specific data

If the sense key is NO SENSE or NOT READY, the SENSE KEY SPECIFIC field shall be as shown in table 22.

Table 22 — Progress indication sense key specific data

Bit / Byte	7	6	5	4	3	2	1	0
0	SKSV (1b)	Reserved						
1	(MSB)	PROGRESS INDICATION						
2								(LSB)

The SKSV bit is described in 4.5.2.4.1 for descriptor format sense data and in 4.5.3 for fixed format sense data.

The PROGRESS INDICATION field is a percent complete indication in which the returned value is a numerator that has 65 536 (10000h) as its denominator. The progress indication shall be based upon the total operation.

NOTE 6 - The progress indication should be time related, however this is not an absolute requirement. (E.g., since format time varies with the number of defects encountered, etc., it is reasonable for the device server to assign values to various steps within the process. The granularity of these steps should be small enough to provide reasonable assurances to the application client that progress is being made.)

4.5.2.4.5 Segment pointer sense key specific data

If the sense key is COPY ABORTED, the SENSE KEY SPECIFIC field shall be as shown in table 23.

Table 23 — Segment pointer sense key specific data

Bit / Byte	7	6	5	4	3	2	1	0
0	SKSV (1b)	Reserved	SD	Reserved	BPV	BIT POINTER		
1	(MSB)	FIELD POINTER						
2								(LSB)

The SKSV bit is described in 4.5.2.4.1 for descriptor format sense data and in 4.5.3 for fixed format sense data.

The segment descriptor (SD) bit indicates whether the field pointer is relative to the start of the parameter list or to the start of a segment descriptor. An SD bit set to zero indicates that the field pointer is relative to the start of the parameter list. An SD bit set to one indicates that the field pointer is relative to the start of the segment descriptor indicated by the third and fourth bytes of the COMMAND-SPECIFIC INFORMATION field (see 6.3.3).

A bit pointer valid (BPV) bit set to zero indicates that the value in the BIT POINTER field is not valid. A BPV bit set to one indicates that the BIT POINTER field specifies which bit of the byte designated by the FIELD POINTER field is in error. When a multiple-bit field is in error, the BIT POINTER field shall point to the most-significant (i.e., left-most) bit of the field.

The FIELD POINTER field indicates which byte of the parameter list or segment descriptor was in error.

If the parameter list is in excess of 65 528 bytes in length and SD is set to zero, the FIELD POINTER value may not fit in two bytes provided by the sense key specific sense data descriptor.

4.5.2.5 Field replaceable unit sense data descriptor

The field replaceable unit sense data descriptor (see table 24) provides information about a component that has failed.

Table 24 — Field replaceable unit sense data descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE (03h)							
1	ADDITIONAL LENGTH (02h)							
2	Reserved							
3	FIELD REPLACEABLE UNIT CODE							

The DESCRIPTOR TYPE and ADDITIONAL LENGTH fields are described in 4.5.2.1. For the field replaceable unit sense data descriptor, the DESCRIPTOR TYPE field shall be set to 03h and the ADDITIONAL LENGTH field shall be set to 02h.

Non-zero values in the FIELD REPLACEABLE UNIT CODE field are used to identify a component that has failed. A value of zero in this field indicates that no specific component has been identified to have failed or that the data is not available. The format of this information is not specified by this standard. Additional information about the field replaceable unit may be available in the ASCII Information VPD page (see 7.6.2), if supported by the device server.

4.5.2.6 Vendor specific sense data descriptors

Vendor specific sense data descriptors (see table 25) contain vendor specific data that further defines the nature of the exception condition.

Table 25 — Vendor specific sense data descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE (80h - FFh)							
1	ADDITIONAL LENGTH (n-1)							
2	Vendor specific							
n								

The DESCRIPTOR TYPE and ADDITIONAL LENGTH fields are described in 4.5.2.1. For the vendor specific sense data descriptor, the DESCRIPTOR TYPE field shall be set to a value between 80h and FFh, inclusive.

4.5.3 Fixed format sense data

The fixed format sense data for response codes 70h (current errors) and 71h (deferred errors) is defined in table 26.

Table 26 — Fixed format sense data

Bit / Byte	7	6	5	4	3	2	1	0
0	VALID	RESPONSE CODE (70h or 71h)						
1	Obsolete							
2	FILEMARK	EOM	ILI	Reserved	SENSE KEY			
3	INFORMATION							
6								
7	ADDITIONAL SENSE LENGTH (n-7)							
8	COMMAND-SPECIFIC INFORMATION							
11								
12	ADDITIONAL SENSE CODE							
13	ADDITIONAL SENSE CODE QUALIFIER							
14	FIELD REPLACEABLE UNIT CODE							
15	SKSV	SENSE KEY SPECIFIC						
17								
18	Additional sense bytes							
n								

A VALID bit set to zero indicates that the INFORMATION field is not defined in this standard or any other command standard (see 3.1.18). A VALID bit set to one indicates the INFORMATION field contains valid information as defined in this standard or a command standard.

The contents of the RESPONSE CODE field indicate the error type and format of the sense data (see 4.5.1). For fixed format sense data, the RESPONSE CODE field shall be set to 70h or 71h.

See the SSC-2 READ and SPACE commands for examples of FILEMARK bit usage.

See the SSC-2 READ, SPACE, and WRITE commands for examples of end-of-medium (EOM) bit usage.

See the SBC-2 READ LONG, SBC-2 WRITE LONG, and SSC-2 READ commands and for examples of incorrect length indicator (ILI) bit usage.

The SENSE KEY, ADDITIONAL SENSE CODE, and ADDITIONAL SENSE CODE QUALIFIER fields are described in 4.5.2.1.

The contents of the INFORMATION field are device-type or command specific and are defined in a command standard (see 3.1.18).

The ADDITIONAL SENSE LENGTH field indicates the number of additional sense bytes that follow. The additional sense length shall be less than or equal to 244 (i.e., limiting the total length of the sense data to 252 bytes). If the sense data is being returned as parameter data by a REQUEST SENSE command, then the relationship between the ADDITIONAL SENSE LENGTH field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6.

The COMMAND-SPECIFIC INFORMATION field contains information that depends on the command on which the exception condition occurred.

The FIELD REPLACEABLE UNIT CODE field is described in 4.5.2.5.

A sense-key specific valid (SKSV) bit set to one indicates the SENSE KEY SPECIFIC field contains valid information as defined in this standard. An SKSV bit set to zero indicates that the SENSE KEY SPECIFIC field is not as defined by this standard.

The SENSE KEY SPECIFIC field is described in 4.5.2.4.

The additional sense bytes may contain vendor specific data that further defines the nature of the exception condition.

4.5.4 Current errors

Response codes 70h and 72h (current error) indicate that the sense data returned is the result of an error or exception condition on the task that returned the CHECK CONDITION status or a protocol specific failure condition. This includes errors generated during processing of the command. It also includes errors not related to any command that are detected during processing of a command (e.g., disk servo-mechanism failure, off-track errors, or power-up test errors).

4.5.5 Deferred errors

Response codes 71h and 73h (deferred error) indicate that the sense data returned is the result of an error or exception condition that occurred during processing of a previous command for which GOOD, CONDITION MET, INTERMEDIATE, and INTERMEDIATE-CONDITION MET status has already been returned. Such commands are associated with the use of the immediate bit and with some forms of caching. Device servers that implement these features shall implement deferred error reporting.

The deferred error may be indicated by returning CHECK CONDITION status to an application client accessed through a defined I_T nexus as described in this subclause.

If the task terminates with CHECK CONDITION status and the sense data describes a deferred error, the command for the terminated task shall not have been processed. After the device server detects a deferred error condition, it shall return a deferred error according to the following rules:

a) If no external intervention is necessary to recover a deferred error, a deferred error indication shall not be returned unless required by the error handling parameters of a MODE SELECT command. The occurrence of the error may be logged;
b) If it is possible to associate a deferred error with an I_T nexus and with a particular function or a particular subset of data, and the error is either unrecovered or required to be reported by the mode parameters, then a deferred error indication shall be returned for a command received on the I_T nexus associated with the deferred error. If an application client request received on an I_T nexus other than the I_T nexus associated with the deferred error attempts to access the particular function or subset of data associated with the deferred error and the TST field equals 000b (see 7.4.6), then the device server shall respond to the command with a BUSY or ACA ACTIVE status according to the requirements in SAM-3. If an application client request received on an I_T nexus other than the I_T nexus associated with the deferred error attempts to access the particular function or subset of data associated with the deferred error and the TST field equals 001b, then the command attempting the access shall not be blocked by the deferred error and the cause of the deferred error may result in an error being reported for the command attempting the access;
c) If the device server is unable to associate a deferred error with an I_T nexus or with a particular subset of data, the device server shall return a deferred error for one command received on each I_T nexus. If multiple deferred errors have accumulated for an I_T nexus, only the last error shall be returned;
d) If the SCSI target device is unable to associate a deferred error with a particular logical unit, it shall establish a deferred error for every logical unit and shall return the deferred error for one command for each logical unit received on each appropriate I_T nexus; or
e) If a task has never entered the enabled task state, and a deferred error occurs, the task shall be terminated with CHECK CONDITION status and deferred error information returned in the sense data. If a deferred error occurs after a task has entered the enabled task state and the task is affected by the error, the task

shall be terminated with CHECK CONDITION status and the current error information shall be returned in the sense data. In this case, if the current error information does not adequately define the deferred error condition, a deferred error may be returned after the current error information has been returned. If a deferred error occurs after a task has entered the enabled task state and the task completes successfully, the device server may choose to return the deferred error information after the completion of the current command in conjunction with a subsequent command that has not begun processing.

NOTE 7 - A deferred error may indicate that an operation was unsuccessful long after GOOD status was returned. If the application client is unable to replicate or recover from other sources the data that is being written using cached or buffered write operations, then synchronization commands should be performed before the critical data is destroyed. This is necessary for actions taken when deferred errors occur in the storing of the data. The synchronizing process should provide the necessary commands to allow returning CHECK CONDITION status and subsequent returning of deferred error sense information after all cached or buffered operations are completed.

4.5.6 Sense key and sense code definitions

The sense keys are defined in table 27.

Table 27 — Sense key descriptions (part 1 of 2)

Sense Key	Description
0h	**NO SENSE:** Indicates that there is no specific sense key information to be reported. This may occur for a successful command or for a command that receives CHECK CONDITION status because one of the FILEMARK, EOM, or ILI bits is set to one.
1h	**RECOVERED ERROR:** Indicates that the command completed successfully, with some recovery action performed by the device server. Details may be determined by examining the additional sense bytes and the INFORMATION field. When multiple recovered errors occur during one command, the choice of which error to report (e.g., first, last, most severe) is vendor specific.
2h	**NOT READY:** Indicates that the logical unit is not accessible. Operator intervention may be required to correct this condition.
3h	**MEDIUM ERROR:** Indicates that the command terminated with a non-recovered error condition that may have been caused by a flaw in the medium or an error in the recorded data. This sense key may also be returned if the device server is unable to distinguish between a flaw in the medium and a specific hardware failure (i.e., sense key 4h).
4h	**HARDWARE ERROR:** Indicates that the device server detected a non-recoverable hardware failure (e.g., controller failure, device failure, or parity error) while performing the command or during a self test.
5h	**ILLEGAL REQUEST:** Indicates that: a) The command was addressed to an incorrect logical unit number (see SAM-3); b) The command had an invalid task attribute (see SAM-3); c) The command was addressed to a logical unit whose current configuration prohibits processing the command; d) There was an illegal parameter in the CDB; or e) There was an illegal parameter in the additional parameters supplied as data for some commands (e.g., PERSISTENT RESERVE OUT). If the device server detects an invalid parameter in the CDB, it shall terminate the command without altering the medium. If the device server detects an invalid parameter in the additional parameters supplied as data, the device server may have already altered the medium.
6h	**UNIT ATTENTION:** Indicates that a unit attention condition has been established (e.g., the removable medium may have been changed, a logical unit reset occurred). See SAM-3.
7h	**DATA PROTECT:** Indicates that a command that reads or writes the medium was attempted on a block that is protected. The read or write operation is not performed.

Table 27 — Sense key descriptions (part 2 of 2)

Sense Key	Description
8h	**BLANK CHECK:** Indicates that a write-once device or a sequential-access device encountered blank medium or format-defined end-of-data indication while reading or that a write-once device encountered a non-blank medium while writing.
9h	**VENDOR SPECIFIC:** This sense key is available for reporting vendor specific conditions.
Ah	**COPY ABORTED:** Indicates an EXTENDED COPY command was aborted due to an error condition on the source device, the destination device, or both (see 6.3.3).
Bh	**ABORTED COMMAND:** Indicates that the device server aborted the command. The application client may be able to recover by trying the command again.
Ch	Obsolete
Dh	**VOLUME OVERFLOW:** Indicates that a buffered SCSI device has reached the end-of-partition and data may remain in the buffer that has not been written to the medium. One or more RECOVER BUFFERED DATA command(s) may be issued to read the unwritten data from the buffer. (See SSC-2.)
Eh	**MISCOMPARE:** Indicates that the source data did not match the data read from the medium.
Fh	Reserved

The additional sense codes (i.e., the additional sense code field and additional sense code qualifier field values returned in sense data) are defined in table 28.

Table 28 — ASC and ASCQ assignments (part 1 of 14)

D - DIRECT ACCESS BLOCK DEVICE (SBC-2)
. T - SEQUENTIAL ACCESS DEVICE (SSC-2)
. L - PRINTER DEVICE (SSC)
. P - PROCESSOR DEVICE (SPC-2)
. . W - WRITE ONCE BLOCK DEVICE (SBC)
. . R - CD/DVD DEVICE (MMC-4)
. . O - OPTICAL MEMORY BLOCK DEVICE (SBC)
. . . M - MEDIA CHANGER DEVICE (SMC-2)
. . . A - STORAGE ARRAY DEVICE (SCC-2)
. . . E - ENCLOSURE SERVICES DEVICE (SES)
. . . . B - SIMPLIFIED DIRECT-ACCESS DEVICE (RBC)
. . . . K - OPTICAL CARD READER/WRITER DEVICE (OCRW)
. . . . V - AUTOMATION/DRIVE INTERFACE (ADC)
. . . . . F - OBJECT-BASED STORAGE (OSD)

Device Column key
blank = code not used
not blank = code used

ASC	ASCQ	D	T	L	P	W	R	O	M	A	E	B	K	V	F	Description
20h	0Bh	D	T		P	W	R	O	M	A	E	B	K			ACCESS DENIED - ACL LUN CONFLICT
20h	08h	D	T		P	W	R	O	M	A	E	B	K			ACCESS DENIED - ENROLLMENT CONFLICT
20h	01h	D	T		P	W	R	O	M	A	E	B	K			ACCESS DENIED - INITIATOR PENDING-ENROLLED
20h	09h	D	T		P	W	R	O	M	A	E	B	K			ACCESS DENIED - INVALID LU IDENTIFIER
20h	03h	D	T		P	W	R	O	M	A	E	B	K			ACCESS DENIED - INVALID MGMT ID KEY
20h	0Ah	D	T		P	W	R	O	M	A	E	B	K			ACCESS DENIED - INVALID PROXY TOKEN
20h	02h	D	T		P	W	R	O	M	A	E	B	K			ACCESS DENIED - NO ACCESS RIGHTS
4Bh	03h	D	T		P	W	R	O	M	A	E	B	K			ACK/NAK TIMEOUT
67h	02h									A						ADD LOGICAL UNIT FAILED
13h	00h	D				W		O				B	K			ADDRESS MARK NOT FOUND FOR DATA FIELD
12h	00h	D				W		O				B	K			ADDRESS MARK NOT FOUND FOR ID FIELD
67h	08h									A						ASSIGN FAILURE OCCURRED
27h	03h		T				R									ASSOCIATED WRITE PROTECT
2Ah	06h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	ASYMMETRIC ACCESS STATE CHANGED
47h	04h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	ASYNCHRONOUS INFORMATION PROTECTION ERROR DETECTED

Annex D contains the ASC and ASCQ assignments in numeric order.

Table 28 — ASC and ASCQ assignments (part 2 of 14)

D - DIRECT ACCESS BLOCK DEVICE (SBC-2)
. T - SEQUENTIAL ACCESS DEVICE (SSC-2)
. L - PRINTER DEVICE (SSC)
. P - PROCESSOR DEVICE (SPC-2)
. . W - WRITE ONCE BLOCK DEVICE (SBC)
. . R - CD/DVD DEVICE (MMC-4)
. . O - OPTICAL MEMORY BLOCK DEVICE (SBC)
. . . M - MEDIA CHANGER DEVICE (SMC-2)
. . . A - STORAGE ARRAY DEVICE (SCC-2)
. . . E - ENCLOSURE SERVICES DEVICE (SES)
. . . . B - SIMPLIFIED DIRECT-ACCESS DEVICE (RBC)
. . . . K - OPTICAL CARD READER/WRITER DEVICE (OCRW)
. . . . V - AUTOMATION/DRIVE INTERFACE (ADC)
. . . . . F - OBJECT-BASED STORAGE (OSD)

Device Column key
blank = code not used
not blank = code used

ASC	ASCQ	D	T	L	P	W	R	O	M	A	E	B	K	V	F	Description
67h	06h									A						ATTACHMENT OF LOGICAL UNIT FAILED
00h	11h						R									AUDIO PLAY OPERATION IN PROGRESS
00h	12h						R									AUDIO PLAY OPERATION PAUSED
00h	14h						R									AUDIO PLAY OPERATION STOPPED DUE TO ERROR
00h	13h						R									AUDIO PLAY OPERATION SUCCESSFULLY COMPLETED
66h	00h															AUTOMATIC DOCUMENT FEEDER COVER UP
66h	01h															AUTOMATIC DOCUMENT FEEDER LIFT UP
55h	06h	D	T			W	R	O	M			B				AUXILIARY MEMORY OUT OF SPACE
11h	12h	D	T			W	R	O	M			B				AUXILIARY MEMORY READ ERROR
0Ch	0Bh	D	T			W	R	O	M			B				AUXILIARY MEMORY WRITE ERROR
00h	04h		T													BEGINNING-OF-PARTITION/MEDIUM DETECTED
0Ch	06h	D	T			W		O				B				BLOCK NOT COMPRESSIBLE
14h	04h		T													BLOCK SEQUENCE ERROR
29h	03h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	BUS DEVICE RESET FUNCTION OCCURRED
11h	0Eh	D	T			W	R	O				B				CANNOT DECOMPRESS USING DECLARED ALGORITHM
30h	06h	D	T			W	R	O				B				CANNOT FORMAT MEDIUM - INCOMPATIBLE MEDIUM
30h	02h	D	T			W	R	O				B	K			CANNOT READ MEDIUM - INCOMPATIBLE FORMAT
30h	01h	D	T			W	R	O				B	K			CANNOT READ MEDIUM - UNKNOWN FORMAT
30h	08h						R									CANNOT WRITE - APPLICATION CODE MISMATCH
30h	05h	D	T			W	R	O				B	K			CANNOT WRITE MEDIUM - INCOMPATIBLE FORMAT
30h	04h	D	T			W	R	O				B	K			CANNOT WRITE MEDIUM - UNKNOWN FORMAT
2Ah	09h	D														CAPACITY DATA HAS CHANGED
52h	00h		T													CARTRIDGE FAULT
73h	00h						R									CD CONTROL ERROR
24h	01h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	CDB DECRYPTION ERROR
3Fh	02h	D	T	L	P	W	R	O	M			B	K			CHANGED OPERATING DEFINITION
11h	06h					W	R	O				B				CIRC UNRECOVERED ERROR
30h	03h	D	T				R						K			CLEANING CARTRIDGE INSTALLED
30h	07h	D	T	L		W	R	O	M	A	E	B	K	V	F	CLEANING FAILURE
30h	0Ah	D	T			W	R	O	M	A	E	B	K			CLEANING REQUEST REJECTED
00h	17h	D	T	L		W	R	O	M	A	E	B	K	V	F	CLEANING REQUESTED
4Ah	00h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	COMMAND PHASE ERROR
2Ch	00h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	COMMAND SEQUENCE ERROR
6Eh	00h									A						COMMAND TO LOGICAL UNIT FAILED
2Fh	00h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	COMMANDS CLEARED BY ANOTHER INITIATOR
3Fh	04h	D	T			W	R	O	M	A	E	B	K			COMPONENT DEVICE ATTACHED
0Ch	04h	D	T			W		O				B				COMPRESSION CHECK MISCOMPARE ERROR
27h	06h						R									CONDITIONAL WRITE PROTECT
67h	00h									A						CONFIGURATION FAILURE
67h	01h									A						CONFIGURATION OF INCAPABLE LOGICAL UNITS FAILED
5Dh	25h	D										B				CONTROLLER IMPENDING FAILURE ACCESS TIMES TOO HIGH
5Dh	27h	D										B				CONTROLLER IMPENDING FAILURE CHANNEL PARAMETRICS
5Dh	28h	D										B				CONTROLLER IMPENDING FAILURE CONTROLLER DETECTED
5Dh	22h	D										B				CONTROLLER IMPENDING FAILURE DATA ERROR RATE TOO HIGH

Annex D contains the ASC and ASCQ assignments in numeric order.

Table 28 — ASC and ASCQ assignments (part 3 of 14)

D - DIRECT ACCESS BLOCK DEVICE (SBC-2)
. T - SEQUENTIAL ACCESS DEVICE (SSC-2)
. L - PRINTER DEVICE (SSC)
. P - PROCESSOR DEVICE (SPC-2)
. . W - WRITE ONCE BLOCK DEVICE (SBC)
. . R - CD/DVD DEVICE (MMC-4)
. . O - OPTICAL MEMORY BLOCK DEVICE (SBC)
. . . M - MEDIA CHANGER DEVICE (SMC-2)
. . . A - STORAGE ARRAY DEVICE (SCC-2)
. . . E - ENCLOSURE SERVICES DEVICE (SES)
. . . . B - SIMPLIFIED DIRECT-ACCESS DEVICE (RBC)
. . . . K - OPTICAL CARD READER/WRITER DEVICE (OCRW)
. . . . V - AUTOMATION/DRIVE INTERFACE (ADC)
. . . . . F - OBJECT-BASED STORAGE (OSD)

Device Column key
blank = code not used
not blank = code used

ASC	ASCQ	D	T	L	P	W	R	O	M	A	E	B	K	V	F	Description
5Dh	2Ch	D										B				CONTROLLER IMPENDING FAILURE DRIVE CALIBRATION RETRY COUNT
5Dh	21h	D										B				CONTROLLER IMPENDING FAILURE DRIVE ERROR RATE TOO HIGH
5Dh	20h	D										B				CONTROLLER IMPENDING FAILURE GENERAL HARD DRIVE FAILURE
5Dh	23h	D										B				CONTROLLER IMPENDING FAILURE SEEK ERROR RATE TOO HIGH
5Dh	2Ah	D										B				CONTROLLER IMPENDING FAILURE SEEK TIME PERFORMANCE
5Dh	2Bh	D										B				CONTROLLER IMPENDING FAILURE SPIN-UP RETRY COUNT
5Dh	26h	D										B				CONTROLLER IMPENDING FAILURE START UNIT TIMES TOO HIGH
5Dh	29h	D										B				CONTROLLER IMPENDING FAILURE THROUGHPUT PERFORMANCE
5Dh	24h	D										B				CONTROLLER IMPENDING FAILURE TOO MANY BLOCK REASSIGNS
2Bh	00h	D	T	L	P	W	R	O					K			COPY CANNOT EXECUTE SINCE HOST CANNOT DISCONNECT
6Fh	00h						R									COPY PROTECTION KEY EXCHANGE FAILURE - AUTHENTICATION FAILURE
6Fh	02h						R									COPY PROTECTION KEY EXCHANGE FAILURE - KEY NOT ESTABLISHED
6Fh	01h						R									COPY PROTECTION KEY EXCHANGE FAILURE - KEY NOT PRESENT
26h	0Dh	D	T	L	P	W	R	O					K			COPY SEGMENT GRANULARITY VIOLATION
0Dh	05h	D	T	L	P	W	R	O		A			K			COPY TARGET DEVICE DATA OVERRUN
0Dh	04h	D	T	L	P	W	R	O		A			K			COPY TARGET DEVICE DATA UNDERRUN
0Dh	02h	D	T	L	P	W	R	O		A			K			COPY TARGET DEVICE NOT REACHABLE
67h	07h									A						CREATION OF LOGICAL UNIT FAILED
2Ch	04h						R									CURRENT PROGRAM AREA IS EMPTY
2Ch	03h						R									CURRENT PROGRAM AREA IS NOT EMPTY
30h	09h						R									CURRENT SESSION NOT FIXATED FOR APPEND
10h	02h	D	T			W		O								DATA BLOCK APPLICATION TAG CHECK FAILED
10h	01h	D	T			W		O								DATA BLOCK GUARD CHECK FAILED
10h	03h	D	T			W		O								DATA BLOCK REFERENCE TAG CHECK FAILED
5Dh	35h	D										B				DATA CHANNEL IMPENDING FAILURE ACCESS TIMES TOO HIGH
5Dh	37h	D										B				DATA CHANNEL IMPENDING FAILURE CHANNEL PARAMETRICS
5Dh	38h	D										B				DATA CHANNEL IMPENDING FAILURE CONTROLLER DETECTED
5Dh	32h	D										B				DATA CHANNEL IMPENDING FAILURE DATA ERROR RATE TOO HIGH
5Dh	3Ch	D										B				DATA CHANNEL IMPENDING FAILURE DRIVE CALIBRATION RETRY COUNT
5Dh	31h	D										B				DATA CHANNEL IMPENDING FAILURE DRIVE ERROR RATE TOO HIGH
5Dh	30h	D										B				DATA CHANNEL IMPENDING FAILURE GENERAL HARD DRIVE FAILURE
5Dh	33h	D										B				DATA CHANNEL IMPENDING FAILURE SEEK ERROR RATE TOO HIGH
5Dh	3Ah	D										B				DATA CHANNEL IMPENDING FAILURE SEEK TIME PERFORMANCE
5Dh	3Bh	D										B				DATA CHANNEL IMPENDING FAILURE SPIN-UP RETRY COUNT
5Dh	36h	D										B				DATA CHANNEL IMPENDING FAILURE START UNIT TIMES TOO HIGH
5Dh	39h	D										B				DATA CHANNEL IMPENDING FAILURE THROUGHPUT PERFORMANCE
5Dh	34h	D										B				DATA CHANNEL IMPENDING FAILURE TOO MANY BLOCK REASSIGNS
26h	05h	D	T	L	P	W	R	O	M	A		B	K			DATA DECRYPTION ERROR
0Ch	05h	D	T			W		O				B				DATA EXPANSION OCCURRED DURING COMPRESSION

Annex D contains the ASC and ASCQ assignments in numeric order.

Table 28 — ASC and ASCQ assignments (part 4 of 14)

D - DIRECT ACCESS BLOCK DEVICE (SBC-2)
. T - SEQUENTIAL ACCESS DEVICE (SSC-2)
. L - PRINTER DEVICE (SSC)
. P - PROCESSOR DEVICE (SPC-2)
. . W - WRITE ONCE BLOCK DEVICE (SBC)
. . R - CD/DVD DEVICE (MMC-4)
. . O - OPTICAL MEMORY BLOCK DEVICE (SBC)
. . . M - MEDIA CHANGER DEVICE (SMC-2)
. . . A - STORAGE ARRAY DEVICE (SCC-2)
. . . E - ENCLOSURE SERVICES DEVICE (SES)
. . . . B - SIMPLIFIED DIRECT-ACCESS DEVICE (RBC)
. . . . K - OPTICAL CARD READER/WRITER DEVICE (OCRW)
. . . . V - AUTOMATION/DRIVE INTERFACE (ADC)
. . . . . F - OBJECT-BASED STORAGE (OSD)

Device Column key
blank = code not used
not blank = code used

ASC	ASCQ	D	T	L	P	W	R	O	M	A	E	B	K	V	F	Description
69h	00h									A						DATA LOSS ON LOGICAL UNIT
4Bh	05h	D	T		P	W	R	O	M	A	E	B	K			DATA OFFSET ERROR
41h	00h	D														DATA PATH FAILURE (SHOULD USE 40 NN)
47h	01h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	DATA PHASE CRC ERROR DETECTED
4Bh	00h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	DATA PHASE ERROR
11h	07h					W		O				B				DATA RE-SYNCHRONIZATION ERROR
16h	03h	D				W		O				B	K			DATA SYNC ERROR - DATA AUTO-REALLOCATED
16h	01h	D				W		O				B	K			DATA SYNC ERROR - DATA REWRITTEN
16h	04h	D				W		O				B	K			DATA SYNC ERROR - RECOMMEND REASSIGNMENT
16h	02h	D				W		O				B	K			DATA SYNC ERROR - RECOMMEND REWRITE
16h	00h	D				W		O				B	K			DATA SYNCHRONIZATION MARK ERROR
11h	0Dh	D	T			W	R	O				B				DE-COMPRESSION CRC ERROR
71h	00h		T													DECOMPRESSION EXCEPTION LONG ALGORITHM ID
70h	NNh		T													DECOMPRESSION EXCEPTION SHORT ALGORITHM ID OF NN
19h	00h	D						O					K			DEFECT LIST ERROR
19h	03h	D						O					K			DEFECT LIST ERROR IN GROWN LIST
19h	02h	D						O					K			DEFECT LIST ERROR IN PRIMARY LIST
19h	01h	D						O					K			DEFECT LIST NOT AVAILABLE
1Ch	00h	D						O				B	K			DEFECT LIST NOT FOUND
32h	01h	D				W		O				B	K			DEFECT LIST UPDATE FAILURE
3Fh	05h	D	T			W	R	O	M	A	E	B	K			DEVICE IDENTIFIER CHANGED
29h	04h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	DEVICE INTERNAL RESET
40h	NNh	D	T	L	P	W	R	O	M	A	E	B	K	V	F	DIAGNOSTIC FAILURE ON COMPONENT NN (80H-FFH)
66h	02h															DOCUMENT JAM IN AUTOMATIC DOCUMENT FEEDER
66h	03h															DOCUMENT MISS FEED AUTOMATIC IN DOCUMENT FEEDER
6Fh	05h							R								DRIVE REGION MUST BE PERMANENT/REGION RESET COUNT ERROR
3Fh	0Fh	D	T	L	P	W	R	O	M	A	E	B	K	V	F	ECHO BUFFER OVERWRITTEN
72h	04h						R									EMPTY OR PARTIALLY WRITTEN RESERVED TRACK
34h	00h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	ENCLOSURE FAILURE
35h	05h	D	T	L		W	R	O	M	A	E	B	K	V	F	ENCLOSURE SERVICES CHECKSUM ERROR
35h	00h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	ENCLOSURE SERVICES FAILURE
35h	03h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	ENCLOSURE SERVICES TRANSFER FAILURE
35h	04h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	ENCLOSURE SERVICES TRANSFER REFUSED
35h	02h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	ENCLOSURE SERVICES UNAVAILABLE
3Bh	0Fh						R									END OF MEDIUM REACHED
63h	00h						R									END OF USER AREA ENCOUNTERED ON THIS TRACK
00h	05h		T	L												END-OF-DATA DETECTED
14h	03h		T													END-OF-DATA NOT FOUND
00h	02h		T													END-OF-PARTITION/MEDIUM DETECTED
51h	00h		T				R	O								ERASE FAILURE
51h	01h						R									ERASE FAILURE - INCOMPLETE ERASE OPERATION DETECTED
00h	18h		T													ERASE OPERATION IN PROGRESS
0Dh	00h	D	T	L	P	W	R	O		A			K			ERROR DETECTED BY THIRD PARTY TEMPORARY INITIATOR

Annex D contains the ASC and ASCQ assignments in numeric order.

Table 28 — ASC and ASCQ assignments (part 5 of 14)

D - DIRECT ACCESS BLOCK DEVICE (SBC-2)
. T - SEQUENTIAL ACCESS DEVICE (SSC-2)
. L - PRINTER DEVICE (SSC)
. P - PROCESSOR DEVICE (SPC-2)
. . W - WRITE ONCE BLOCK DEVICE (SBC)
. . R - CD/DVD DEVICE (MMC-4)
. . O - OPTICAL MEMORY BLOCK DEVICE (SBC)
. . . M - MEDIA CHANGER DEVICE (SMC-2)
. . . A - STORAGE ARRAY DEVICE (SCC-2)
. . . E - ENCLOSURE SERVICES DEVICE (SES)
. . . . B - SIMPLIFIED DIRECT-ACCESS DEVICE (RBC)
. . . . K - OPTICAL CARD READER/WRITER DEVICE (OCRW)
. . . . V - AUTOMATION/DRIVE INTERFACE (ADC)
. . . . . F - OBJECT-BASED STORAGE (OSD)

Device Column key
blank = code not used
not blank = code used

ASC	ASCQ	D	T	L	P	W	R	O	M	A	E	B	K	V	F	Description
0Ah	00h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	ERROR LOG OVERFLOW
11h	10h						R									ERROR READING ISRC NUMBER
11h	0Fh						R									ERROR READING UPC/EAN NUMBER
11h	02h	D	T			W	R	O				B	K			ERROR TOO LONG TO CORRECT
38h	06h											B				ESN - DEVICE BUSY CLASS EVENT
38h	04h											B				ESN - MEDIA CLASS EVENT
38h	02h											B				ESN - POWER MANAGEMENT CLASS EVENT
38h	00h											B				EVENT STATUS NOTIFICATION
03h	02h		T													EXCESSIVE WRITE ERRORS
67h	04h									A						EXCHANGE OF LOGICAL UNIT FAILED
3Bh	07h			L												FAILED TO SENSE BOTTOM-OF-FORM
3Bh	06h			L												FAILED TO SENSE TOP-OF-FORM
5Dh	00h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	FAILURE PREDICTION THRESHOLD EXCEEDED
5Dh	FFh	D	T	L	P	W	R	O	M	A	E	B	K	V	F	FAILURE PREDICTION THRESHOLD EXCEEDED (FALSE)
00h	01h		T													FILEMARK DETECTED
14h	02h		T													FILEMARK OR SETMARK NOT FOUND
5Dh	65h	D										B				FIRMWARE IMPENDING FAILURE ACCESS TIMES TOO HIGH
5Dh	67h	D										B				FIRMWARE IMPENDING FAILURE CHANNEL PARAMETRICS
5Dh	68h	D										B				FIRMWARE IMPENDING FAILURE CONTROLLER DETECTED
5Dh	62h	D										B				FIRMWARE IMPENDING FAILURE DATA ERROR RATE TOO HIGH
5Dh	6Ch	D										B				FIRMWARE IMPENDING FAILURE DRIVE CALIBRATION RETRY COUNT
5Dh	61h	D										B				FIRMWARE IMPENDING FAILURE DRIVE ERROR RATE TOO HIGH
5Dh	60h	D										B				FIRMWARE IMPENDING FAILURE GENERAL HARD DRIVE FAILURE
5Dh	63h	D										B				FIRMWARE IMPENDING FAILURE SEEK ERROR RATE TOO HIGH
5Dh	6Ah	D										B				FIRMWARE IMPENDING FAILURE SEEK TIME PERFORMANCE
5Dh	6Bh	D										B				FIRMWARE IMPENDING FAILURE SPIN-UP RETRY COUNT
5Dh	66h	D										B				FIRMWARE IMPENDING FAILURE START UNIT TIMES TOO HIGH
5Dh	69h	D										B				FIRMWARE IMPENDING FAILURE THROUGHPUT PERFORMANCE
5Dh	64h	D										B				FIRMWARE IMPENDING FAILURE TOO MANY BLOCK REASSIGNS
09h	02h					W	R	O					K			FOCUS SERVO FAILURE
31h	01h	D		L			R	O				B				FORMAT COMMAND FAILED
58h	00h							O								GENERATION DOES NOT EXIST
1Ch	02h	D						O				B	K			GROWN DEFECT LIST NOT FOUND
5Dh	15h	D										B				HARDWARE IMPENDING FAILURE ACCESS TIMES TOO HIGH
5Dh	17h	D										B				HARDWARE IMPENDING FAILURE CHANNEL PARAMETRICS
5Dh	18h	D										B				HARDWARE IMPENDING FAILURE CONTROLLER DETECTED
5Dh	12h	D										B				HARDWARE IMPENDING FAILURE DATA ERROR RATE TOO HIGH
5Dh	1Ch	D										B				HARDWARE IMPENDING FAILURE DRIVE CALIBRATION RETRY COUNT
5Dh	11h	D										B				HARDWARE IMPENDING FAILURE DRIVE ERROR RATE TOO HIGH
5Dh	10h	D										B				HARDWARE IMPENDING FAILURE GENERAL HARD DRIVE FAILURE
5Dh	13h	D										B				HARDWARE IMPENDING FAILURE SEEK ERROR RATE TOO HIGH
5Dh	1Ah	D										B				HARDWARE IMPENDING FAILURE SEEK TIME PERFORMANCE
5Dh	1Bh	D										B				HARDWARE IMPENDING FAILURE SPIN-UP RETRY COUNT

Annex D contains the ASC and ASCQ assignments in numeric order.

Table 28 — ASC and ASCQ assignments (part 6 of 14)

D - DIRECT ACCESS BLOCK DEVICE (SBC-2)
. T - SEQUENTIAL ACCESS DEVICE (SSC-2)
. L - PRINTER DEVICE (SSC)
. P - PROCESSOR DEVICE (SPC-2)
. . W - WRITE ONCE BLOCK DEVICE (SBC)
. . R - CD/DVD DEVICE (MMC-4)
. . O - OPTICAL MEMORY BLOCK DEVICE (SBC)
. . . M - MEDIA CHANGER DEVICE (SMC-2)
. . . A - STORAGE ARRAY DEVICE (SCC-2)
. . . E - ENCLOSURE SERVICES DEVICE (SES)
. . . . B - SIMPLIFIED DIRECT-ACCESS DEVICE (RBC)
. . . . K - OPTICAL CARD READER/WRITER DEVICE (OCRW)
. . . . V - AUTOMATION/DRIVE INTERFACE (ADC)
. . . . . F - OBJECT-BASED STORAGE (OSD)

Device Column key
blank = code not used
not blank = code used

ASC	ASCQ	D	T	L	P	W	R	O	M	A	E	B	K	V	F	Description
5Dh	16h	D										B				HARDWARE IMPENDING FAILURE START UNIT TIMES TOO HIGH
5Dh	19h	D										B				HARDWARE IMPENDING FAILURE THROUGHPUT PERFORMANCE
5Dh	14h	D										B				HARDWARE IMPENDING FAILURE TOO MANY BLOCK REASSIGNS
27h	01h	D	T			W	R	O				B	K			HARDWARE WRITE PROTECTED
09h	04h	D	T			W	R	O				B				HEAD SELECT FAULT
00h	06h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	I/O PROCESS TERMINATED
10h	00h	D				W		O				B	K			ID CRC OR ECC ERROR
5Eh	03h	D	T	L	P	W	R	O		A			K			IDLE CONDITION ACTIVATED BY COMMAND
5Eh	01h	D	T	L	P	W	R	O		A			K			IDLE CONDITION ACTIVATED BY TIMER
20h	06h		T													ILLEGAL COMMAND WHILE IN EXPLICIT ADDRESS MODE
20h	07h		T													ILLEGAL COMMAND WHILE IN IMPLICIT ADDRESS MODE
20h	04h		T													ILLEGAL COMMAND WHILE IN WRITE CAPABLE STATE
22h	00h	D														ILLEGAL FUNCTION (USE 20 00, 24 00, OR 26 00)
64h	00h						R									ILLEGAL MODE FOR THIS TRACK
2Ch	05h											B				ILLEGAL POWER CONDITION REQUEST
2Ah	07h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	IMPLICIT ASYMMETRIC ACCESS STATE TRANSITION FAILED
28h	01h	D	T			W	R	O	M			B				IMPORT OR EXPORT ELEMENT ACCESSED
30h	00h	D	T			W	R	O	M			B	K			INCOMPATIBLE MEDIUM INSTALLED
11h	08h		T													INCOMPLETE BLOCK READ
0Dh	03h	D	T	L	P	W	R	O		A			K			INCORRECT COPY TARGET DEVICE TYPE
0Eh	02h	D	T		P	W	R	O	M	A	E	B	K		F	INFORMATION UNIT TOO LONG
0Eh	01h	D	T		P	W	R	O	M	A	E	B	K		F	INFORMATION UNIT TOO SHORT
47h	03h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	INFORMATION UNIT iuCRC ERROR DETECTED
6Ah	00h									A						INFORMATIONAL, REFER TO LOG
48h	00h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	INITIATOR DETECTED ERROR MESSAGE RECEIVED
4Bh	06h	D	T		P	W	R	O	M	A	E	B	K			INITIATOR RESPONSE TIMEOUT
26h	0Bh	D	T	L	P	W	R	O					K			INLINE DATA LENGTH EXCEEDED
3Fh	03h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	INQUIRY DATA HAS CHANGED
55h	05h	D	T		P	W	R	O	M	A	E	B	K			INSUFFICIENT ACCESS CONTROL RESOURCES
55h	04h	D	T	L	P	W	R	O	M	A	E		K			INSUFFICIENT REGISTRATION RESOURCES
55h	02h	D	T	L	P	W	R	O	M	A	E		K			INSUFFICIENT RESERVATION RESOURCES
55h	03h	D	T	L	P	W	R	O	M	A	E		K			INSUFFICIENT RESOURCES
2Eh	00h						R									INSUFFICIENT TIME FOR OPERATION
44h	00h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	INTERNAL TARGET FAILURE
21h	02h						R									INVALID ADDRESS FOR WRITE
3Dh	00h	D	T	L	P	W	R	O	M	A	E		K			INVALID BITS IN IDENTIFY MESSAGE
2Ch	02h															INVALID COMBINATION OF WINDOWS SPECIFIED
20h	00h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	INVALID COMMAND OPERATION CODE
26h	0Fh														F	INVALID DATA-OUT BUFFER INTEGRITY CHECK VALUE
21h	01h	D	T			W	R	O	M			B	K			INVALID ELEMENT ADDRESS
24h	00h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	INVALID FIELD IN CDB
0Eh	03h	D	T		P		R		M	A	E	B	K		F	INVALID FIELD IN COMMAND INFORMATION UNIT
26h	00h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	INVALID FIELD IN PARAMETER LIST
0Eh	00h	D	T		P	W	R	O	M	A	E	B	K		F	INVALID INFORMATION UNIT

Annex D contains the ASC and ASCQ assignments in numeric order.

Table 28 — ASC and ASCQ assignments (part 7 of 14)

D - DIRECT ACCESS BLOCK DEVICE (SBC-2)
. T - SEQUENTIAL ACCESS DEVICE (SSC-2)
. L - PRINTER DEVICE (SSC)
. P - PROCESSOR DEVICE (SPC-2)
. . W - WRITE ONCE BLOCK DEVICE (SBC)
. . R - CD/DVD DEVICE (MMC-4)
. . O - OPTICAL MEMORY BLOCK DEVICE (SBC)
. . . M - MEDIA CHANGER DEVICE (SMC-2)
. . . A - STORAGE ARRAY DEVICE (SCC-2)
. . . E - ENCLOSURE SERVICES DEVICE (SES)
. . . . B - SIMPLIFIED DIRECT-ACCESS DEVICE (RBC)
. . . . K - OPTICAL CARD READER/WRITER DEVICE (OCRW)
. . . . V - AUTOMATION/DRIVE INTERFACE (ADC)
. . . . . F - OBJECT-BASED STORAGE (OSD)

Device Column key
blank = code not used
not blank = code used

ASC	ASCQ	D	T	L	P	W	R	O	M	A	E	B	K	V	F	Description
49h	00h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	INVALID MESSAGE ERROR
26h	0Ch	D	T	L	P	W	R	O					K			INVALID OPERATION FOR COPY SOURCE OR DESTINATION
64h	01h						R									INVALID PACKET SIZE
26h	0Eh	D	T		P	W	R	O	M	A	E	B	K			INVALID PARAMETER WHILE PORT IS ENABLED
26h	04h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	INVALID RELEASE OF PERSISTENT RESERVATION
4Bh	01h	D	T		P	W	R	O	M	A	E	B	K			INVALID TARGET PORT TRANSFER TAG RECEIVED
29h	07h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	I_T NEXUS LOSS OCCURRED
11h	05h					W	R	O				B				L-EC UNCORRECTABLE ERROR
60h	00h															LAMP FAILURE
14h	07h		T													LOCATE OPERATION FAILURE
00h	19h		T													LOCATE OPERATION IN PROGRESS
5Bh	02h	D	T	L	P	W	R	O	M				K			LOG COUNTER AT MAXIMUM
5Bh	00h	D	T	L	P	W	R	O	M				K			LOG EXCEPTION
5Bh	03h	D	T	L	P	W	R	O	M				K			LOG LIST CODES EXHAUSTED
2Ah	02h	D	T	L		W	R	O	M	A	E		K			LOG PARAMETERS CHANGED
21h	00h	D	T			W	R	O	M			B	K			LOGICAL BLOCK ADDRESS OUT OF RANGE
08h	03h	D	T				R	O	M			B	K			LOGICAL UNIT COMMUNICATION CRC ERROR (ULTRA-DMA/32)
08h	00h	D	T	L		W	R	O	M	A	E	B	K	V	F	LOGICAL UNIT COMMUNICATION FAILURE
08h	02h	D	T	L		W	R	O	M	A	E	B	K	V	F	LOGICAL UNIT COMMUNICATION PARITY ERROR
08h	01h	D	T	L		W	R	O	M	A	E	B	K	V	F	LOGICAL UNIT COMMUNICATION TIME-OUT
05h	00h	D	T	L		W	R	O	M	A	E	B	K	V	F	LOGICAL UNIT DOES NOT RESPOND TO SELECTION
4Ch	00h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	LOGICAL UNIT FAILED SELF-CONFIGURATION
3Eh	03h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	LOGICAL UNIT FAILED SELF-TEST
3Eh	01h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	LOGICAL UNIT FAILURE
5Dh	02h						R									LOGICAL UNIT FAILURE PREDICTION THRESHOLD EXCEEDED
3Eh	00h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	LOGICAL UNIT HAS NOT SELF-CONFIGURED YET
04h	01h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	LOGICAL UNIT IS IN PROCESS OF BECOMING READY
04h	0Ah	D	T	L	P	W	R	O	M	A	E	B	K	V	F	LOGICAL UNIT NOT ACCESSIBLE, ASYMMETRIC ACCESS STATE TRANSITION
04h	0Bh	D	T	L	P	W	R	O	M	A	E	B	K	V	F	LOGICAL UNIT NOT ACCESSIBLE, TARGET PORT IN STANDBY STATE
04h	0Ch	D	T	L	P	W	R	O	M	A	E	B	K	V	F	LOGICAL UNIT NOT ACCESSIBLE, TARGET PORT IN UNAVAILABLE STATE
68h	00h									A						LOGICAL UNIT NOT CONFIGURED
04h	10h	D	T			W	R	O	M			B				LOGICAL UNIT NOT READY, AUXILIARY MEMORY NOT ACCESSIBLE
04h	00h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	LOGICAL UNIT NOT READY, CAUSE NOT REPORTABLE
04h	04h	D	T	L			R	O				B				LOGICAL UNIT NOT READY, FORMAT IN PROGRESS
04h	02h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	LOGICAL UNIT NOT READY, INITIALIZING COMMAND REQUIRED
04h	08h						R									LOGICAL UNIT NOT READY, LONG WRITE IN PROGRESS
04h	03h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	LOGICAL UNIT NOT READY, MANUAL INTERVENTION REQUIRED
04h	11h	D	T			W	R	O	M	A	E	B		V	F	LOGICAL UNIT NOT READY, NOTIFY (ENABLE SPINUP) REQUIRED
04h	12h													V		LOGICAL UNIT NOT READY, OFFLINE
04h	07h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	LOGICAL UNIT NOT READY, OPERATION IN PROGRESS
04h	05h	D	T			W		O	M	A		B	K			LOGICAL UNIT NOT READY, REBUILD IN PROGRESS
04h	06h	D	T			W		O	M	A		B	K			LOGICAL UNIT NOT READY, RECALCULATION IN PROGRESS

Annex D contains the ASC and ASCQ assignments in numeric order.

Table 28 — ASC and ASCQ assignments (part 8 of 14)

D - DIRECT ACCESS BLOCK DEVICE (SBC-2)
. T - SEQUENTIAL ACCESS DEVICE (SSC-2)
. L - PRINTER DEVICE (SSC)
. P - PROCESSOR DEVICE (SPC-2)
. . W - WRITE ONCE BLOCK DEVICE (SBC)
. . R - CD/DVD DEVICE (MMC-4)
. . O - OPTICAL MEMORY BLOCK DEVICE (SBC)
. . . M - MEDIA CHANGER DEVICE (SMC-2)
. . . A - STORAGE ARRAY DEVICE (SCC-2)
. . . E - ENCLOSURE SERVICES DEVICE (SES)
. . . . B - SIMPLIFIED DIRECT-ACCESS DEVICE (RBC)
. . . . K - OPTICAL CARD READER/WRITER DEVICE (OCRW)
. . . . V - AUTOMATION/DRIVE INTERFACE (ADC)
. . . . . F - OBJECT-BASED STORAGE (OSD)

Device Column key
blank = code not used
not blank = code used

ASC	ASCQ	D	T	L	P	W	R	O	M	A	E	B	K	V	F	Description
04h	09h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	LOGICAL UNIT NOT READY, SELF-TEST IN PROGRESS
25h	00h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	LOGICAL UNIT NOT SUPPORTED
27h	02h	D	T			W	R	O				B	K			LOGICAL UNIT SOFTWARE WRITE PROTECTED
3Eh	04h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	LOGICAL UNIT UNABLE TO UPDATE SELF-TEST LOG
5Eh	00h	D	T	L	P	W	R	O		A			K			LOW POWER CONDITION ON
15h	01h	D	T	L		W	R	O	M			B	K			MECHANICAL POSITIONING ERROR
3Bh	16h						R									MECHANICAL POSITIONING OR CHANGER ERROR
5Dh	01h						R					B				MEDIA FAILURE PREDICTION THRESHOLD EXCEEDED
53h	00h	D	T	L		W	R	O	M			B	K			MEDIA LOAD OR EJECT FAILED
6Fh	04h						R									MEDIA REGION CODE IS MISMATCHED TO LOGICAL UNIT REGION
3Fh	11h	D	T			W	R	O	M			B				MEDIUM AUXILIARY MEMORY ACCESSIBLE
3Bh	0Dh	D	T			W	R	O	M			B	K			MEDIUM DESTINATION ELEMENT FULL
31h	00h	D	T			W	R	O				B	K			MEDIUM FORMAT CORRUPTED
3Fh	10h	D	T			W	R	O	M			B				MEDIUM LOADABLE
3Bh	13h	D	T			W	R	O	M			B	K			MEDIUM MAGAZINE INSERTED
3Bh	14h	D	T			W	R	O	M			B	K			MEDIUM MAGAZINE LOCKED
3Bh	11h	D	T			W	R	O	M			B	K			MEDIUM MAGAZINE NOT ACCESSIBLE
3Bh	12h	D	T			W	R	O	M			B	K			MEDIUM MAGAZINE REMOVED
3Bh	15h	D	T			W	R	O	M			B	K			MEDIUM MAGAZINE UNLOCKED
30h	10h						R									MEDIUM NOT FORMATTED
3Ah	00h	D	T	L		W	R	O	M			B	K			MEDIUM NOT PRESENT
3Ah	03h	D	T			W	R	O	M			B				MEDIUM NOT PRESENT - LOADABLE
3Ah	04h	D	T			W	R	O	M			B				MEDIUM NOT PRESENT - MEDIUM AUXILIARY MEMORY ACCESSIBLE
3Ah	01h	D	T			W	R	O	M			B	K			MEDIUM NOT PRESENT - TRAY CLOSED
3Ah	02h	D	T			W	R	O	M			B	K			MEDIUM NOT PRESENT - TRAY OPEN
53h	02h	D	T			W	R	O	M			B	K			MEDIUM REMOVAL PREVENTED
3Bh	0Eh	D	T			W	R	O	M			B	K			MEDIUM SOURCE ELEMENT EMPTY
43h	00h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	MESSAGE ERROR
3Fh	01h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	MICROCODE HAS BEEN CHANGED
1Dh	00h	D	T			W	R	O				B	K			MISCOMPARE DURING VERIFY OPERATION
11h	0Ah	D	T					O				B	K			MISCORRECTED ERROR
2Ah	01h	D	T	L		W	R	O	M	A	E	B	K	V	F	MODE PARAMETERS CHANGED
67h	03h									A						MODIFICATION OF LOGICAL UNIT FAILED
69h	01h									A						MULTIPLE LOGICAL UNIT FAILURES
07h	00h	D	T	L		W	R	O	M			B	K			MULTIPLE PERIPHERAL DEVICES SELECTED
11h	03h	D	T			W		O				B	K			MULTIPLE READ ERRORS
67h	09h									A						MULTIPLY ASSIGNED LOGICAL UNIT
4Bh	04h	D	T		P	W	R	O	M	A	E	B	K			NAK RECEIVED
00h	00h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	NO ADDITIONAL SENSE INFORMATION
00h	15h						R									NO CURRENT AUDIO STATUS TO RETURN
32h	00h	D				W		O				B	K			NO DEFECT SPARE LOCATION AVAILABLE
11h	09h		T													NO GAP FOUND
01h	00h	D				W		O				B	K			NO INDEX/SECTOR SIGNAL
72h	05h						R									NO MORE TRACK RESERVATIONS ALLOWED

Annex D contains the ASC and ASCQ assignments in numeric order.

Table 28 — ASC and ASCQ assignments (part 9 of 14)

D - DIRECT ACCESS BLOCK DEVICE (SBC-2)
. T - SEQUENTIAL ACCESS DEVICE (SSC-2)
. L - PRINTER DEVICE (SSC)
. P - PROCESSOR DEVICE (SPC-2)
. . W - WRITE ONCE BLOCK DEVICE (SBC)
. . R - CD/DVD DEVICE (MMC-4)
. . O - OPTICAL MEMORY BLOCK DEVICE (SBC)
. . . M - MEDIA CHANGER DEVICE (SMC-2)
. . . A - STORAGE ARRAY DEVICE (SCC-2)
. . . E - ENCLOSURE SERVICES DEVICE (SES)
. . . . B - SIMPLIFIED DIRECT-ACCESS DEVICE (RBC)
. . . . K - OPTICAL CARD READER/WRITER DEVICE (OCRW)
. . . . V - AUTOMATION/DRIVE INTERFACE (ADC)
. . . . . F - OBJECT-BASED STORAGE (OSD)

Device Column key
blank = code not used
not blank = code used

ASC	ASCQ	D	T	L	P	W	R	O	M	A	E	B	K	V	F	Description
06h	00h	D				W	R	O	M			B	K			NO REFERENCE POSITION FOUND
02h	00h	D				W	R	O	M			B	K			NO SEEK COMPLETE
03h	01h		T													NO WRITE CURRENT
24h	06h														F	NONCE NOT UNIQUE
24h	07h														F	NONCE TIMESTAMP OUT OF RANGE
28h	00h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	NOT READY TO READY CHANGE, MEDIUM MAY HAVE CHANGED
2Ch	0Bh		T													NOT RESERVED
00h	16h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	OPERATION IN PROGRESS
5Ah	01h	D	T			W	R	O	M			B	K			OPERATOR MEDIUM REMOVAL REQUEST
5Ah	00h	D	T	L	P	W	R	O	M			B	K			OPERATOR REQUEST OR STATE CHANGE INPUT
5Ah	03h	D	T			W	R	O		A		B	K			OPERATOR SELECTED WRITE PERMIT
5Ah	02h	D	T			W	R	O		A		B	K			OPERATOR SELECTED WRITE PROTECT
61h	02h															OUT OF FOCUS
4Eh	00h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	OVERLAPPED COMMANDS ATTEMPTED
2Dh	00h		T													OVERWRITE ERROR ON UPDATE IN PLACE
20h	05h		T													Obsolete
24h	02h		T													Obsolete
24h	03h		T													Obsolete
63h	01h						R									PACKET DOES NOT FIT IN AVAILABLE SPACE
3Bh	05h			L												PAPER JAM
1Ah	00h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	PARAMETER LIST LENGTH ERROR
26h	01h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	PARAMETER NOT SUPPORTED
26h	02h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	PARAMETER VALUE INVALID
2Ah	00h	D	T	L		W	R	O	M	A	E	B	K	V	F	PARAMETERS CHANGED
69h	02h									A						PARITY/DATA MISMATCH
1Fh	00h	D						O					K			PARTIAL DEFECT LIST TRANSFER
2Ch	0Ah														F	PARTITION OR COLLECTION CONTAINS USER OBJECTS
03h	00h	D	T	L		W		O				B	K			PERIPHERAL DEVICE WRITE FAULT
27h	05h		T				R									PERMANENT WRITE PROTECT
2Ch	06h						R									PERSISTENT PREVENT CONFLICT
27h	04h		T				R									PERSISTENT WRITE PROTECT
47h	06h	D	T						M	A	E	B	K	V	F	PHY TEST FUNCTION IN PROGRESS
50h	02h		T													POSITION ERROR RELATED TO TIMING
3Bh	0Ch		T													POSITION PAST BEGINNING OF MEDIUM
3Bh	0Bh															POSITION PAST END OF MEDIUM
15h	02h	D	T			W	R	O				B	K			POSITIONING ERROR DETECTED BY READ OF MEDIUM
73h	01h						R									POWER CALIBRATION AREA ALMOST FULL
73h	03h						R									POWER CALIBRATION AREA ERROR
73h	02h						R									POWER CALIBRATION AREA IS FULL
29h	01h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	POWER ON OCCURRED
29h	00h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	POWER ON, RESET, OR BUS DEVICE RESET OCCURRED
5Eh	41h											B				POWER STATE CHANGE TO ACTIVE
5Eh	47h											B	K			POWER STATE CHANGE TO DEVICE CONTROL
5Eh	42h											B				POWER STATE CHANGE TO IDLE

Annex D contains the ASC and ASCQ assignments in numeric order.

Table 28 — ASC and ASCQ assignments (part 10 of 14)

D - DIRECT ACCESS BLOCK DEVICE (SBC-2)
. T - SEQUENTIAL ACCESS DEVICE (SSC-2)
. L - PRINTER DEVICE (SSC)
. P - PROCESSOR DEVICE (SPC-2)
. . W - WRITE ONCE BLOCK DEVICE (SBC)
. . R - CD/DVD DEVICE (MMC-4)
. . O - OPTICAL MEMORY BLOCK DEVICE (SBC)
. . . M - MEDIA CHANGER DEVICE (SMC-2)
. . . A - STORAGE ARRAY DEVICE (SCC-2)
. . . E - ENCLOSURE SERVICES DEVICE (SES)
. . . . B - SIMPLIFIED DIRECT-ACCESS DEVICE (RBC)
. . . . K - OPTICAL CARD READER/WRITER DEVICE (OCRW)
. . . . V - AUTOMATION/DRIVE INTERFACE (ADC)
. . . . . F - OBJECT-BASED STORAGE (OSD)

Device Column key
blank = code not used
not blank = code used

ASC	ASCQ	D	T	L	P	W	R	O	M	A	E	B	K	V	F	Description
5Eh	45h											B				POWER STATE CHANGE TO SLEEP
5Eh	43h											B				POWER STATE CHANGE TO STANDBY
42h	00h	D														POWER-ON OR SELF-TEST FAILURE (SHOULD USE 40 NN)
2Ch	07h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	PREVIOUS BUSY STATUS
2Ch	09h	D	T	L	P	W	R	O	M		E	B	K	V	F	PREVIOUS RESERVATION CONFLICT STATUS
2Ch	08h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	PREVIOUS TASK SET FULL STATUS
1Ch	01h	D						O				B	K			PRIMARY DEFECT LIST NOT FOUND
2Ah	08h	D	T			W	R	O	M	A	E	B	K	V	F	PRIORITY CHANGED
73h	05h						R									PROGRAM MEMORY AREA IS FULL
73h	04h						R									PROGRAM MEMORY AREA UPDATE FAILURE
47h	05h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	PROTOCOL SERVICE CRC ERROR
55h	07h														F	QUOTA ERROR
40h	00h	D														RAM FAILURE (SHOULD USE 40 NN)
15h	00h	D	T	L		W	R	O	M			B	K			RANDOM POSITIONING ERROR
11h	13h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	READ ERROR - FAILED RETRANSMISSION REQUEST
11h	11h						R									READ ERROR - LOSS OF STREAMING
6Fh	03h						R									READ OF SCRAMBLED SECTOR WITHOUT AUTHENTICATION
3Bh	0Ah															READ PAST BEGINNING OF MEDIUM
3Bh	09h															READ PAST END OF MEDIUM
3Bh	17h														F	READ PAST END OF USER OBJECT
11h	01h	D	T			W	R	O				B	K			READ RETRIES EXHAUSTED
6Ch	00h									A						REBUILD FAILURE OCCURRED
6Dh	00h									A						RECALCULATE FAILURE OCCURRED
14h	01h	D	T			W	R	O				B	K			RECORD NOT FOUND
14h	06h	D	T			W		O				B	K			RECORD NOT FOUND - DATA AUTO-REALLOCATED
14h	05h	D	T			W		O				B	K			RECORD NOT FOUND - RECOMMEND REASSIGNMENT
14h	00h	D	T	L		W	R	O				B	K			RECORDED ENTITY NOT FOUND
18h	02h	D				W	R	O				B	K			RECOVERED DATA - DATA AUTO-REALLOCATED
18h	05h	D				W	R	O				B	K			RECOVERED DATA - RECOMMEND REASSIGNMENT
18h	06h	D				W	R	O				B	K			RECOVERED DATA - RECOMMEND REWRITE
17h	05h	D				W	R	O				B	K			RECOVERED DATA USING PREVIOUS SECTOR ID
18h	03h						R									RECOVERED DATA WITH CIRC
18h	07h	D				W		O				B	K			RECOVERED DATA WITH ECC - DATA REWRITTEN
18h	01h	D				W	R	O				B	K			RECOVERED DATA WITH ERROR CORR. & RETRIES APPLIED
18h	00h	D	T			W	R	O				B	K			RECOVERED DATA WITH ERROR CORRECTION APPLIED
18h	04h						R									RECOVERED DATA WITH L-EC
18h	08h						R									RECOVERED DATA WITH LINKING
17h	03h	D	T			W	R	O				B	K			RECOVERED DATA WITH NEGATIVE HEAD OFFSET
17h	00h	D	T			W	R	O				B	K			RECOVERED DATA WITH NO ERROR CORRECTION APPLIED
17h	02h	D	T			W	R	O				B	K			RECOVERED DATA WITH POSITIVE HEAD OFFSET
17h	01h	D	T			W	R	O				B	K			RECOVERED DATA WITH RETRIES
17h	04h					W	R	O				B				RECOVERED DATA WITH RETRIES AND/OR CIRC APPLIED
17h	06h	D				W		O				B	K			RECOVERED DATA WITHOUT ECC - DATA AUTO-REALLOCATED
17h	09h	D				W	R	O				B	K			RECOVERED DATA WITHOUT ECC - DATA REWRITTEN

Annex D contains the ASC and ASCQ assignments in numeric order.

Table 28 — ASC and ASCQ assignments (part 11 of 14)

D - DIRECT ACCESS BLOCK DEVICE (SBC-2)
. T - SEQUENTIAL ACCESS DEVICE (SSC-2)
. L - PRINTER DEVICE (SSC)
. P - PROCESSOR DEVICE (SPC-2)
. . W - WRITE ONCE BLOCK DEVICE (SBC)
. . R - CD/DVD DEVICE (MMC-4)
. . O - OPTICAL MEMORY BLOCK DEVICE (SBC)
. . . M - MEDIA CHANGER DEVICE (SMC-2)
. . . A - STORAGE ARRAY DEVICE (SCC-2)
. . . E - ENCLOSURE SERVICES DEVICE (SES)
. . . . B - SIMPLIFIED DIRECT-ACCESS DEVICE (RBC)
. . . . K - OPTICAL CARD READER/WRITER DEVICE (OCRW)
. . . . V - AUTOMATION/DRIVE INTERFACE (ADC)
. . . . . F - OBJECT-BASED STORAGE (OSD)

Device Column key
blank = code not used
not blank = code used

ASC	ASCQ	D	T	L	P	W	R	O	M	A	E	B	K	V	F	Description
17h	07h	D				W	R	O				B	K			RECOVERED DATA WITHOUT ECC - RECOMMEND REASSIGNMENT
17h	08h	D				W	R	O				B	K			RECOVERED DATA WITHOUT ECC - RECOMMEND REWRITE
1Eh	00h	D				W		O				B	K			RECOVERED ID WITH ECC CORRECTION
3Fh	06h	D	T			W	R	O	M	A	E	B				REDUNDANCY GROUP CREATED OR MODIFIED
3Fh	07h	D	T			W	R	O	M	A	E	B				REDUNDANCY GROUP DELETED
6Bh	01h									A						REDUNDANCY LEVEL GOT BETTER
6Bh	02h									A						REDUNDANCY LEVEL GOT WORSE
2Ah	05h	D	T	L	P	W	R	O	M	A	E					REGISTRATIONS PREEMPTED
67h	05h									A						REMOVE OF LOGICAL UNIT FAILED
3Fh	0Eh	D	T	L	P	W	R	O	M	A	E					REPORTED LUNS DATA HAS CHANGED
3Bh	08h		T													REPOSITION ERROR
2Ah	03h	D	T	L	P	W	R	O	M	A	E		K			RESERVATIONS PREEMPTED
2Ah	04h	D	T	L	P	W	R	O	M	A	E					RESERVATIONS RELEASED
00h	1Ah		T													REWIND OPERATION IN PROGRESS
36h	00h			L												RIBBON, INK, OR TONER FAILURE
73h	06h						R									RMA/PMA IS ALMOST FULL
37h	00h	D	T	L		W	R	O	M	A	E	B	K	V	F	ROUNDED PARAMETER
5Ch	00h	D						O								RPL STATUS CHANGE
39h	00h	D	T	L		W	R	O	M	A	E		K			SAVING PARAMETERS NOT SUPPORTED
62h	00h															SCAN HEAD POSITIONING ERROR
29h	02h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	SCSI BUS RESET OCCURRED
47h	00h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	SCSI PARITY ERROR
47h	02h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	SCSI PARITY ERROR DETECTED DURING ST DATA PHASE
54h	00h				P											SCSI TO HOST SYSTEM INTERFACE FAILURE
24h	04h														F	SECURITY AUDIT VALUE FROZEN
24h	05h														F	SECURITY WORKING KEY FROZEN
45h	00h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	SELECT OR RESELECT FAILURE
3Bh	00h		T	L												SEQUENTIAL POSITIONING ERROR
5Dh	45h	D										B				SERVO IMPENDING FAILURE ACCESS TIMES TOO HIGH
5Dh	47h	D										B				SERVO IMPENDING FAILURE CHANNEL PARAMETRICS
5Dh	48h	D										B				SERVO IMPENDING FAILURE CONTROLLER DETECTED
5Dh	42h	D										B				SERVO IMPENDING FAILURE DATA ERROR RATE TOO HIGH
5Dh	4Ch	D										B				SERVO IMPENDING FAILURE DRIVE CALIBRATION RETRY COUNT
5Dh	41h	D										B				SERVO IMPENDING FAILURE DRIVE ERROR RATE TOO HIGH
5Dh	40h	D										B				SERVO IMPENDING FAILURE GENERAL HARD DRIVE FAILURE
5Dh	43h	D										B				SERVO IMPENDING FAILURE SEEK ERROR RATE TOO HIGH
5Dh	4Ah	D										B				SERVO IMPENDING FAILURE SEEK TIME PERFORMANCE
5Dh	4Bh	D										B				SERVO IMPENDING FAILURE SPIN-UP RETRY COUNT
5Dh	46h	D										B				SERVO IMPENDING FAILURE START UNIT TIMES TOO HIGH
5Dh	49h	D										B				SERVO IMPENDING FAILURE THROUGHPUT PERFORMANCE
5Dh	44h	D										B				SERVO IMPENDING FAILURE TOO MANY BLOCK REASSIGNS
72h	00h						R									SESSION FIXATION ERROR
72h	03h						R									SESSION FIXATION ERROR - INCOMPLETE TRACK IN SESSION
72h	01h						R									SESSION FIXATION ERROR WRITING LEAD-IN

Annex D contains the ASC and ASCQ assignments in numeric order.

Table 28 — ASC and ASCQ assignments (part 12 of 14)

D - DIRECT ACCESS BLOCK DEVICE (SBC-2)
. T - SEQUENTIAL ACCESS DEVICE (SSC-2)
. L - PRINTER DEVICE (SSC)
. P - PROCESSOR DEVICE (SPC-2)
. . W - WRITE ONCE BLOCK DEVICE (SBC)
. . R - CD/DVD DEVICE (MMC-4)
. . O - OPTICAL MEMORY BLOCK DEVICE (SBC)
. . . M - MEDIA CHANGER DEVICE (SMC-2)
. . . A - STORAGE ARRAY DEVICE (SCC-2)
. . . E - ENCLOSURE SERVICES DEVICE (SES)
. . . . B - SIMPLIFIED DIRECT-ACCESS DEVICE (RBC)
. . . . K - OPTICAL CARD READER/WRITER DEVICE (OCRW)
. . . . V - AUTOMATION/DRIVE INTERFACE (ADC)
. . . . . F - OBJECT-BASED STORAGE (OSD)

Device Column key
blank = code not used
not blank = code used

ASC	ASCQ	D	T	L	P	W	R	O	M	A	E	B	K	V	F	Description
72h	02h						R									SESSION FIXATION ERROR WRITING LEAD-OUT
00h	1Bh		T													SET CAPACITY OPERATION IN PROGRESS
67h	0Ah	D	T	L	P	W	R	O	M	A	E	B	K	V	F	SET TARGET PORT GROUPS COMMAND FAILED
00h	03h		T													SETMARK DETECTED
3Bh	04h			L												SLEW FAILURE
47h	7Fh	D	T		P	W	R	O	M	A	E	B	K			SOME COMMANDS CLEARED BY ISCSI PROTOCOL EVENT
5Dh	03h						R									SPARE AREA EXHAUSTION PREDICTION THRESHOLD EXCEEDED
3Fh	08h	D	T			W	R	O	M	A	E	B				SPARE CREATED OR MODIFIED
3Fh	09h	D	T			W	R	O	M	A	E	B				SPARE DELETED
5Dh	55h	D										B				SPINDLE IMPENDING FAILURE ACCESS TIMES TOO HIGH
5Dh	57h	D										B				SPINDLE IMPENDING FAILURE CHANNEL PARAMETRICS
5Dh	58h	D										B				SPINDLE IMPENDING FAILURE CONTROLLER DETECTED
5Dh	52h	D										B				SPINDLE IMPENDING FAILURE DATA ERROR RATE TOO HIGH
5Dh	5Ch	D										B				SPINDLE IMPENDING FAILURE DRIVE CALIBRATION RETRY COUNT
5Dh	51h	D										B				SPINDLE IMPENDING FAILURE DRIVE ERROR RATE TOO HIGH
5Dh	50h	D										B				SPINDLE IMPENDING FAILURE GENERAL HARD DRIVE FAILURE
5Dh	53h	D										B				SPINDLE IMPENDING FAILURE SEEK ERROR RATE TOO HIGH
5Dh	5Ah	D										B				SPINDLE IMPENDING FAILURE SEEK TIME PERFORMANCE
5Dh	5Bh	D										B				SPINDLE IMPENDING FAILURE SPIN-UP RETRY COUNT
5Dh	56h	D										B				SPINDLE IMPENDING FAILURE START UNIT TIMES TOO HIGH
5Dh	59h	D										B				SPINDLE IMPENDING FAILURE THROUGHPUT PERFORMANCE
5Dh	54h	D										B				SPINDLE IMPENDING FAILURE TOO MANY BLOCK REASSIGNS
09h	03h					W	R	O								SPINDLE SERVO FAILURE
5Ch	02h	D						O								SPINDLES NOT SYNCHRONIZED
5Ch	01h	D						O								SPINDLES SYNCHRONIZED
5Eh	04h	D	T	L	P	W	R	O		A			K			STANDBY CONDITION ACTIVATED BY COMMAND
5Eh	02h	D	T	L	P	W	R	O		A			K			STANDBY CONDITION ACTIVATED BY TIMER
6Bh	00h									A						STATE CHANGE HAS OCCURRED
1Bh	00h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	SYNCHRONOUS DATA TRANSFER ERROR
55h	01h	D						O				B	K			SYSTEM BUFFER FULL
55h	00h				P											SYSTEM RESOURCE FAILURE
4Dh	NNh	D	T	L	P	W	R	O	M	A	E	B	K	V	F	TAGGED OVERLAPPED COMMANDS (NN = TASK TAG)
33h	00h		T													TAPE LENGTH ERROR
3Bh	03h			L												TAPE OR ELECTRONIC VERTICAL FORMS UNIT NOT READY
3Bh	01h		T													TAPE POSITION ERROR AT BEGINNING-OF-MEDIUM
3Bh	02h		T													TAPE POSITION ERROR AT END-OF-MEDIUM
3Fh	00h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	TARGET OPERATING CONDITIONS HAVE CHANGED
0Dh	01h	D	T	L	P	W	R	O		A			K			THIRD PARTY DEVICE FAILURE
5Bh	01h	D	T	L	P	W	R	O	M				K			THRESHOLD CONDITION MET
26h	03h	D	T	L	P	W	R	O	M	A	E		K			THRESHOLD PARAMETERS NOT SUPPORTED
3Eh	02h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	TIMEOUT ON LOGICAL UNIT
2Ah	10h	D	T						M		E			V		TIMESTAMP CHANGED
26h	08h	D	T	L	P	W	R	O					K			TOO MANY SEGMENT DESCRIPTORS
26h	06h	D	T	L	P	W	R	O					K			TOO MANY TARGET DESCRIPTORS

Annex D contains the ASC and ASCQ assignments in numeric order.

Table 28 — ASC and ASCQ assignments (part 13 of 14)

D - DIRECT ACCESS BLOCK DEVICE (SBC-2)
. T - SEQUENTIAL ACCESS DEVICE (SSC-2)
. L - PRINTER DEVICE (SSC)
. P - PROCESSOR DEVICE (SPC-2)
. . W - WRITE ONCE BLOCK DEVICE (SBC)
. . R - CD/DVD DEVICE (MMC-4)
. . O - OPTICAL MEMORY BLOCK DEVICE (SBC)
. . . M - MEDIA CHANGER DEVICE (SMC-2)
. . . A - STORAGE ARRAY DEVICE (SCC-2)
. . . E - ENCLOSURE SERVICES DEVICE (SES)
. . . . B - SIMPLIFIED DIRECT-ACCESS DEVICE (RBC)
. . . . K - OPTICAL CARD READER/WRITER DEVICE (OCRW)
. . . . V - AUTOMATION/DRIVE INTERFACE (ADC)
. . . . . F - OBJECT-BASED STORAGE (OSD)

Device Column key
blank = code not used
not blank = code used

ASC	ASCQ	D	T	L	P	W	R	O	M	A	E	B	K	V	F	Description
2Ch	01h															TOO MANY WINDOWS SPECIFIED
4Bh	02h	D	T		P	W	R	O	M	A	E	B	K			TOO MUCH WRITE DATA
09h	00h	D	T			W	R	O				B				TRACK FOLLOWING ERROR
09h	01h					W	R	O					K			TRACKING SERVO FAILURE
29h	06h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	TRANSCEIVER MODE CHANGED TO LVD
29h	05h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	TRANSCEIVER MODE CHANGED TO SINGLE-ENDED
61h	01h															UNABLE TO ACQUIRE VIDEO
57h	00h						R									UNABLE TO RECOVER TABLE-OF-CONTENTS
26h	0Ah	D	T	L	P	W	R	O					K			UNEXPECTED INEXACT SEGMENT
53h	01h		T													UNLOAD TAPE FAILURE
08h	04h	D	T	L	P	W	R	O					K			UNREACHABLE COPY TARGET
11h	00h	D	T			W	R	O				B	K			UNRECOVERED READ ERROR
11h	04h	D				W		O				B	K			UNRECOVERED READ ERROR - AUTO REALLOCATE FAILED
11h	0Bh	D				W		O				B	K			UNRECOVERED READ ERROR - RECOMMEND REASSIGNMENT
11h	0Ch	D				W		O				B	K			UNRECOVERED READ ERROR - RECOMMEND REWRITE THE DATA
46h	00h	D	T	L	P	W	R	O	M			B	K			UNSUCCESSFUL SOFT RESET
35h	01h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	UNSUPPORTED ENCLOSURE FUNCTION
26h	09h	D	T	L	P	W	R	O					K			UNSUPPORTED SEGMENT DESCRIPTOR TYPE CODE
26h	07h	D	T	L	P	W	R	O					K			UNSUPPORTED TARGET DESCRIPTOR TYPE CODE
59h	00h							O								UPDATED BLOCK READ
00h	1Ch		T													VERIFY OPERATION IN PROGRESS
61h	00h															VIDEO ACQUISITION ERROR
65h	00h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	VOLTAGE FAULT
3Fh	0Ah	D	T			W	R	O	M	A	E	B	K			VOLUME SET CREATED OR MODIFIED
3Fh	0Ch	D	T			W	R	O	M	A	E	B	K			VOLUME SET DEASSIGNED
3Fh	0Bh	D	T			W	R	O	M	A	E	B	K			VOLUME SET DELETED
3Fh	0Dh	D	T			W	R	O	M	A	E	B	K			VOLUME SET REASSIGNED
0Bh	00h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	WARNING
0Bh	02h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	WARNING - ENCLOSURE DEGRADED
0Bh	01h	D	T	L	P	W	R	O	M	A	E	B	K	V	F	WARNING - SPECIFIED TEMPERATURE EXCEEDED
30h	0Ch		T													WORM MEDIUM - OVERWRITE ATTEMPTED
50h	00h		T													WRITE APPEND ERROR
50h	01h		T													WRITE APPEND POSITION ERROR
0Ch	00h		T				R									WRITE ERROR
0Ch	02h	D				W		O				B	K			WRITE ERROR - AUTO REALLOCATION FAILED
0Ch	09h						R									WRITE ERROR - LOSS OF STREAMING
0Ch	0Dh	D	T	L	P	W	R	O	M	A	E	B	K	V	F	WRITE ERROR - NOT ENOUGH UNSOLICITED DATA
0Ch	0Ah						R									WRITE ERROR - PADDING BLOCKS ADDED
0Ch	03h	D				W		O				B	K			WRITE ERROR - RECOMMEND REASSIGNMENT
0Ch	01h												K			WRITE ERROR - RECOVERED WITH AUTO REALLOCATION
0Ch	08h						R									WRITE ERROR - RECOVERY FAILED
0Ch	07h						R									WRITE ERROR - RECOVERY NEEDED
0Ch	0Ch	D	T	L	P	W	R	O	M	A	E	B	K	V	F	WRITE ERROR - UNEXPECTED UNSOLICITED DATA
27h	00h	D	T			W	R	O				B	K			WRITE PROTECTED

Annex D contains the ASC and ASCQ assignments in numeric order.

Table 28 — ASC and ASCQ assignments (part 14 of 14)

D - DIRECT ACCESS BLOCK DEVICE (SBC-2)
. T - SEQUENTIAL ACCESS DEVICE (SSC-2)
. L - PRINTER DEVICE (SSC)
. P - PROCESSOR DEVICE (SPC-2)
. . W - WRITE ONCE BLOCK DEVICE (SBC)
. . R - CD/DVD DEVICE (MMC-4)
. . O - OPTICAL MEMORY BLOCK DEVICE (SBC)
. . . M - MEDIA CHANGER DEVICE (SMC-2)
. . . A - STORAGE ARRAY DEVICE (SCC-2)
. . . E - ENCLOSURE SERVICES DEVICE (SES)
. . . . B - SIMPLIFIED DIRECT-ACCESS DEVICE (RBC)
. . . . K - OPTICAL CARD READER/WRITER DEVICE (OCRW)
. . . . V - AUTOMATION/DRIVE INTERFACE (ADC)
. . . . . F - OBJECT-BASED STORAGE (OSD)
. . . . .

Device Column key
blank = code not used
not blank = code used

ASC	ASCQ	D	T	L	P	W	R	O	M	A	E	B	K	V	F	Description
31h	02h						R									ZONED FORMATTING FAILED DUE TO SPARE LINKING
80h	xxh						\									
Through							>									Vendor specific
FFh	xxh						/									
xxh	80h						\									
Through							>									Vendor specific qualification of standard ASC
xxh	FFh						/									
																All codes not shown are reserved.

Annex D contains the ASC and ASCQ assignments in numeric order.

5 Model common to all device types

5.1 Introduction to the model common to all device types

This model describes some of the general characteristics expected of most SCSI devices. It is not intended to alter any requirements defined elsewhere in SCSI. Devices conforming to this standard also shall conform to SAM-3.

5.2 Important commands for all SCSI device servers

5.2.1 Commands implemented by all SCSI device servers

This standard defines three commands that all SCSI device servers shall implement - INQUIRY, REPORT LUNS, and TEST UNIT READY. These commands are used to discover a logical unit's capabilities, to discover the system configuration, and to determine whether a logical unit is ready.

5.2.2 Commands recommended for all SCSI device servers

Support for the REQUEST SENSE command is recommended to provide compatibility with application clients designed to use previous versions of this standard or status polling features defined by command standards (see 3.1.18).

5.2.3 Using the INQUIRY command

The INQUIRY command (see 6.4) may be used by an application client to determine the configuration of a logical unit. Device servers respond with information that includes their device type and standard version and may include the vendor's identification, model number and other information.

The Device Identification VPD page (see 7.6.3) returned in response to an INQUIRY command with the EVPD bit set to one and the PAGE CODE field set to 83h contains identifying information for the logical unit, the target port, and the SCSI target device.

It is recommended that device servers be capable of returning this information, or whatever part of it that is available, upon completing power-on initialization. A device server may take longer to get certain portions of this information, especially if it retrieves the information from the medium.

5.2.4 Using the REPORT LUNS command

The REPORT LUNS command (see 6.21) may be used by an application client to discover the logical unit inventory (see 3.1.60) that is accessible to the I_T nexus on which the command is sent.

5.2.5 Using the TEST UNIT READY command

The TEST UNIT READY command (see 6.33) allows an application client to poll a logical unit until it is ready without the need to allocate space for returned data. The TEST UNIT READY command may be used to check the media status of logical units with removable media. Device servers should respond promptly to indicate the current status of the SCSI device.

NOTE 8 - Delays to achieve GOOD status from a TEST UNIT READY command may adversely affect initiator device performance.

5.2.6 Using the REQUEST SENSE command

The REQUEST SENSE command (see 6.27) may be used by an application client to poll the status of some background operations and to clear interlocked unit attention conditions (see 7.4.6).

5.3 Implicit head of queue

Each of the following commands may be processed by the task manager as if it has a task attribute of HEAD OF QUEUE (see SAM-3) if it is received with a SIMPLE task attribute, an ORDERED task attribute, or no task attribute:

a) INQUIRY; and
b) REPORT LUNS.

An application client should not send a command with the ORDERED task attribute if the command may be processed as if it has a task attribute of HEAD OF QUEUE because whether the ORDERED task attribute is honored for these commands is vendor specific.

5.4 Parameter rounding

Certain parameters sent to a device server with various commands contain a range of values. Device servers may choose to implement only selected values from this range. When the device server receives a value that it does not support, it either rejects the command (i.e., CHECK CONDITION status with ILLEGAL REQUEST sense key) or it rounds the value received to a supported value.

When parameter rounding is implemented, a device server that receives a parameter value that is not an exact supported value shall adjust the value to one that it supports and shall return CHECK CONDITION status, with the sense key set to RECOVERED ERROR, and the additional sense code set to ROUNDED PARAMETER. The application client should issue an appropriate command to learn what value the device server has selected.

The device server shall reject unsupported values unless rounding is permitted in the description of the parameter. When the description of a parameter states that rounding is permitted, the device server should adjust maximum-value fields down to the next lower supported value than the one specified by the application client. Minimum-value fields should be rounded up to the next higher supported value than the one specified by the application client. In some cases, the type of rounding (i.e., up or down) is specified in the description of the parameter.

5.5 Self-test operations

5.5.1 Default self-test

The SEND DIAGNOSTIC command provides a means to request that a SCSI device perform a self test. While the test is vendor specific, the means of requesting the test is standardized.

The default self-test is mandatory for all device types that support the SEND DIAGNOSTIC command. The response is GOOD status if the test detects no exceptions, or CHECK CONDITION status if the test detects exceptions.

5.5.2 The short and extended self-tests

There are two optional types of self-test aside from the mandatory default self-test that may be invoked using the SELF-TEST CODE field in the SEND DIAGNOSTIC command; a short self-test and an extended self-test. The goal of the short self-test is to quickly identify if the logical unit determines that it is faulty. A goal of the extended self-test routine is to simplify factory testing during integration by having logical units perform more comprehensive testing without application client intervention. A second goal of the extended self-test is to provide a more comprehensive test to validate the results of a short self-test, if its results are judged by the application client to be inconclusive.

The criteria for the short self-test are that it has one or more segments and completes in two minutes or less. The criteria for the extended self-test are that it is has one or more segments and that the completion time is vendor specific. Any tests performed in the segments are vendor specific.

The following are examples of segments:

a) An electrical segment wherein the logical unit tests its own electronics. The tests in this segment are vendor specific, but some examples of tests that may be included are:
   A) A buffer RAM test;
   B) A read/write circuitry test; and/or
   C) A test of the read/write heads;
b) A seek/servo segment wherein a device tests it capability to find and servo on data tracks; and
c) A read/verify scan segment wherein a device performs read scanning of some or all of the medium surface.

The tests performed in the segments may be the same for the short and extended self-tests. The time required by a logical unit to complete its extended self-test is reported in the EXTENDED SELF-TEST COMPLETION TIME field in the Control mode page (see 7.4.6).

5.5.3 Self-test modes

5.5.3.1 Self-test modes overview

Both a foreground mode (see 5.5.3.2) and a background mode (see 5.5.3.3) is defined for both short and extended self-tests.

5.5.3.2 Foreground mode

When a device server receives a SEND DIAGNOSTIC command specifying a self-test to be performed in the foreground mode, the device server shall return status for that command after the self-test has been completed.

While performing a self-test in the foreground mode, the device server shall respond to all commands except INQUIRY, REPORT LUNS, and REQUEST SENSE with CHECK CONDITION status, with the sense key set to NOT READY, and the additional sense code set to LOGICAL UNIT NOT READY, SELF-TEST IN PROGRESS.

If a device server is performing a self-test in the foreground mode and a test segment error occurs during the test, the device server shall update the Self-Test Results log page (see 7.2.10) and terminate the SEND DIAGNOSTIC command with CHECK CONDITION status, with the sense key set to HARDWARE ERROR, and the additional sense code set to LOGICAL UNIT FAILED SELF-TEST. The application client may obtain additional information about the failure by reading the Self-Test Results log page. If the device server is unable to update the Self-Test Results log page, it shall terminate the SEND DIAGNOSTIC command with CHECK CONDITION status, with the sense key set to HARDWARE ERROR, and the additional sense code set to LOGICAL UNIT UNABLE TO UPDATE SELF-TEST LOG.

An application client should reserve the logical unit before initiating a self-test in the foreground mode. An application client may terminate a self-test that is being performed in the foreground mode using commands (see clause 6) or task management functions (see SAM-3) (e.g., a PERSISTENT RESERVE OUT command with PREEMPT AND ABORT service action, an ABORT TASK task management function, a CLEAR TASK SET task management function). In addition, a foreground mode self-test shall be terminated by an I_T nexus loss (see SAM-3). If a SEND DIAGNOSTIC command that requested a self-test in the foreground mode is terminated while the SCSI target device is performing the self-test, the device server shall abort the self-test and update the Self-Test Results log page (see 7.2.10).

5.5.3.3 Background mode

When a device server receives a SEND DIAGNOSTIC command specifying a self-test to be performed in the background mode, the device server shall return status for that command as soon as the CDB has been validated.

After returning status for the SEND DIAGNOSTIC command specifying a self-test to be performed in the background mode, the device server shall initialize the Self-Test Results log page (see 7.2.10) as follows. The self-test code from the SEND DIAGNOSTIC command shall be placed in the SELF-TEST CODE field in the log page.

The SELF-TEST RESULTS field shall be set to Fh. After the Self-Test Results log page is initialized, the device server shall begin the first self-test segment.

While the device server is performing a self-test in the background mode, it shall terminate with CHECK CONDITION status any SEND DIAGNOSTIC command it receives that meets one of the following criteria:

a) The SELFTEST bit is set to one; or
b) The SELF-TEST CODE field contains a value other than 000b or 100b.

When terminating the SEND DIAGNOSTIC command, the sense key shall be set to NOT READY and the additional sense code shall be set to LOGICAL UNIT NOT READY, SELF-TEST IN PROGRESS.

While performing a self-test in the background mode, the device server shall suspend the self-test to service any other commands received with the exceptions listed in table 29. Suspension of the self-test to service the command shall occur as soon as practical and shall not take longer than two seconds.

Table 29 — Exception commands for background self-tests

Device type	Command	Reference
All device types	SEND DIAGNOSTIC (with SELF-TEST CODE field set to 100b) WRITE BUFFER (with the mode set to any download microcode option)	6.28 6.35
Direct access block	FORMAT UNIT START STOP UNIT	SBC-2
Sequential access	ERASE FORMAT MEDIUM LOAD UNLOAD LOCATE READ READ POSITION READ REVERSE REWIND SPACE VERIFY WRITE WRITE BUFFER WRITE FILEMARKS	SSC-2
Medium changer	EXCHANGE MEDIUM INITIALIZE ELEMENT STATUS MOVE MEDIUM POSITION TO ELEMENT READ ELEMENT STATUS (if CURDATA=0 and device motion is required) WRITE BUFFER	SMC-2
Object-based storage	Any command with operation code 7Fh (i.e., all commands defined by the OSD standard)	OSD

NOTE Device types not listed in this table do not have commands that are exceptions for background self-tests, other than those listed above for all device types.

If one of the exception commands listed in table 29 is received, the device server shall abort the self-test, update the self-test log, and service the command as soon as practical but not longer than two seconds after the CDB has been validated.

An application client may terminate a self-test that is being performed in the background mode by issuing a SEND DIAGNOSTIC command with the SELF-TEST CODE field set to 100b (i.e., abort background self-test function). A background mode self-test shall not be terminated by an I_T nexus loss (see SAM-3).

5.5.3.4 Features common to foreground and background self-test modes

The PROGRESS INDICATION field in parameter data returned in response to a REQUEST SENSE command (see 6.27) may be used by the application client at any time during a self-test operation to poll the logical unit's progress. While a self-test operation is in progress unless an error has occurred, a device server shall respond to a REQUEST SENSE command by returning parameter data containing sense data with the sense key set to NOT

READY and the additional sense code set to LOGICAL UNIT NOT READY, SELF-TEST IN PROGRESS with the sense key specific bytes set for progress indication.

The application client may obtain information about the 20 most recently completed self-tests by reading the Self-Test Results log page (see 7.2.10). This is the only method for an application client to obtain information about self-tests performed in the background mode.

Table 30 summarizes when a logical unit returns status after receipt of a self-test command, how an application client may abort a self-test, how a logical unit handles commands that are entered into the task set while a self-test is in progress, and how a logical unit reports a self-test failure.

Table 30 — Self-test mode summary

Mode	When Status is Returned	How to abort the self-test	Processing of subsequent commands while self-test is being processed	Self-test failure reporting
Fore-ground	After the self-test is complete	One of the commands (see 5.5.3.2) and task management functions (see SAM-3) that cause tasks to be aborted	If the command is INQUIRY, REPORT LUNS or REQUEST SENSE, process normally. Otherwise, terminate with CHECK CONDITION status, with the sense key set to NOT READY, and the additional sense code set to LOGICAL UNIT NOT READY, SELF-TEST IN PROGRESS.	Terminate with CHECK CONDITION status, with the sense key set to HARDWARE ERROR, and the additional sense code set to LOGICAL UNIT FAILED SELF-TEST or LOGICAL UNIT UNABLE TO UPDATE SELF-TEST LOG.
Back-ground	After the CDB is validated	SEND DIAGNOSTIC command with SELF-TEST CODE field set to 100b	Process the command, except as described in 5.5.3.3.	Application client checks Self-Test Results log page (see 7.2.10) after the PROGRESS INDICATION field returned from REQUEST SENSE indicates the self-test is complete.

5.6 Reservations

5.6.1 Persistent Reservations overview

Reservations may be used to allow a device server to process commands from a selected set of I_T nexuses (i.e., combinations of initiator ports accessing target ports) and reject commands from I_T nexuses outside the selected set. The device server uniquely identifies I_T nexuses using protocol specific mechanisms.

Application clients may add or remove I_T nexuses from the selected set using reservation commands. If the application clients do not cooperate in the reservation protocol, data may be unexpectedly modified and deadlock conditions may occur.

The persistent reservations mechanism allows multiple application clients communicating through multiple I_T nexuses to preserve reservation operations across SCSI initiator device failures, which usually involve logical unit resets and involve I_T nexus losses. Persistent reservations persist across recovery actions. Persistent reservations are not reset by hard reset, logical unit reset, or I_T nexus loss.

The persistent reservation held by a failing I_T nexus may be preempted by another I_T nexus as part of its recovery process. Persistent reservations shall be retained by the device server until released, preempted, or cleared by mechanisms specified in this standard. Optionally, persistent reservations may be retained when power to the SCSI target device is removed.

The PERSISTENT RESERVE OUT and PERSISTENT RESERVE IN commands provide the basic mechanism for dynamic contention resolution in systems with multiple initiator ports accessing a logical unit.

Before a persistent reservation may be established, the application client shall register a reservation key for each I_T nexus with the device server. Reservation keys are necessary to allow:

a) Authentication of subsequent PERSISTENT RESERVE OUT commands;
b) Identification of other I_T nexuses that are registered;
c) Identification of the reservation key(s) that have an associated persistent reservation;
d) Preemption of a persistent reservation from a failing or uncooperative I_T nexus; and
e) Multiple I_T nexuses to participate in a persistent reservation.

The reservation key provides a method for the application client to associate a protocol-independent identifier with a registered I_T nexus. The reservation key is used in the PERSISTENT RESERVE IN command to identify which I_T nexuses are registered and which I_T nexus, if any, holds the persistent reservation. The reservation key is used in the PERSISTENT RESERVE OUT command to register an I_T nexus, to verify the I_T nexus being used for the PERSISTENT RESERVE OUT command is registered, and to specify which registrations or persistent reservation to preempt.

Reservation key values may be used by application clients to identify registered I_T nexuses, using application specific methods that are outside the scope of this standard. This standard provides the ability to register no more than one reservation key per I_T nexus. Multiple initiator ports may use the same reservation key value for a logical unit accessed through the same target ports. An initiator port may use the same reservation key value for a logical unit accessed through different target ports. The logical unit shall maintain a separate reservation key for each I_T nexus, regardless of the reservation key's value.

An application client may register an I_T nexus with multiple logical units in a SCSI target device using any combination of unique or duplicate reservation keys. These rules provide the ability for an application client to preempt multiple I_T nexuses with a single PERSISTENT RESERVE OUT command, but they do not provide the ability for the application client to uniquely identify the I_T nexuses using the PERSISTENT RESERVE commands.

See table 112 in 6.12.2 for a list of PERSISTENT RESERVE OUT service actions. See table 101 in 6.11.1 for a list of PERSISTENT RESERVE IN service actions.

The scope (see 6.11.3.3) of a persistent reservation shall be the entire logical unit.

The type (see 6.11.3.4) of a persistent reservation defines the selected set of I_T nexuses for which the persistent reservation places restrictions on commands.

The details of which commands are allowed under what types of reservations are described in table 31.

In table 31 and table 32 the following key words are used:

allowed: Commands received from I_T nexuses not holding the reservation or from I_T nexuses not registered when a registrants only or all registrants type persistent reservation is present should complete normally.

conflict: Commands received from I_T nexuses not holding the reservation or from I_T nexuses not registered when a registrants only or all registrants type persistent reservation is present shall not be performed and the device server shall terminate the command with a RESERVATION CONFLICT status.

Commands from I_T nexuses holding a reservation should complete normally. The behavior of commands from registered I_T nexuses when a registrants only or all registrants type persistent reservation is present is specified in table 31 and table 32.

An unlinked command shall be checked for reservation conflicts before the task containing that command enters the enabled task state. The reservation state as it exists when the first command in a group of linked commands enters the enabled task state shall be used in checking for reservation conflicts for all the commands in the task. Once a task has entered the enabled task state, the command or commands comprising that task shall not be

terminated with a RESERVATION CONFLICT due to a subsequent reservation. Any command in a group of linked commands that changes the reservation state shall be the last command in the group.

For each command, this standard or a command standard (see 3.1.18) defines the conditions that result in RESERVATION CONFLICT. Command standards define the conditions either in the device model or in the descriptions each of specific command.

Table 31 — SPC commands that are allowed in the presence of various reservations (part 1 of 2)

Command	Addressed logical unit has this type of persistent reservation held by another I_T nexus				
	From any I_T nexus		From registered I_T nexus (RR all types)	From not registered I_T nexus	
	Write Excl	Excl Access		Write Excl RR	Excl Access – RR
ACCESS CONTROL IN	Allowed	Allowed	Allowed	Allowed	Allowed
ACCESS CONTROL OUT	Allowed	Allowed	Allowed	Allowed	Allowed
CHANGE ALIASES	Conflict	Conflict	Allowed	Conflict	Conflict
EXTENDED COPY	Conflict	Conflict	Allowed	Conflict	Conflict
INQUIRY	Allowed	Allowed	Allowed	Allowed	Allowed
LOG SELECT	Conflict	Conflict	Allowed	Conflict	Conflict
LOG SENSE	Allowed	Allowed	Allowed	Allowed	Allowed
MODE SELECT(6) / MODE SELECT(10)	Conflict	Conflict	Allowed	Conflict	Conflict
MODE SENSE(6) / MODE SENSE(10)	Conflict	Conflict	Allowed	Conflict	Conflict
PERSISTENT RESERVE IN	Allowed	Allowed	Allowed	Allowed	Allowed
PERSISTENT RESERVE OUT	see table 32				
PREVENT ALLOW MEDIUM REMOVAL (Prevent=0)	Allowed	Allowed	Allowed	Allowed	Allowed
PREVENT ALLOW MEDIUM REMOVAL (Prevent<>0)	Conflict	Conflict	Allowed	Conflict	Conflict
READ ATTRIBUTE	Conflict	Conflict	Allowed	Conflict	Conflict
READ BUFFER	Conflict	Conflict	Allowed	Conflict	Conflict
READ MEDIA SERIAL NUMBER	Allowed	Allowed	Allowed	Allowed	Allowed
RECEIVE COPY RESULTS	Conflict	Conflict	Allowed	Conflict	Conflict
RECEIVE DIAGNOSTIC RESULTS	Conflict	Conflict	Allowed	Conflict	Conflict
RELEASE(6)/ RELEASE(10)	As defined in SPC-2 [a]				
REPORT ALIASES	Allowed	Allowed	Allowed	Allowed	Allowed
REPORT DEVICE IDENTIFIER	Allowed	Allowed	Allowed	Allowed	Allowed
REPORT LUNS	Allowed	Allowed	Allowed	Allowed	Allowed

Key: **Excl**=Exclusive, **RR**=Registrants Only or All Registrants, **<>** Not Equal

[a] Exceptions to the behavior of the RESERVE and RELEASE commands described in SPC-2 are defined in 5.6.3.

[b] Logical units claiming compliance with previous versions of this standard (e.g., SPC-2) may return RESERVATION CONFLICT in this case.

Table 31 — SPC commands that are allowed in the presence of various reservations (part 2 of 2)

Command	Addressed logical unit has this type of persistent reservation held by another I_T nexus				
	From any I_T nexus		From registered I_T nexus (RR all types)	From not registered I_T nexus	
	Write Excl	Excl Access		Write Excl RR	Excl Access – RR
REPORT PRIORITY	Allowed	Allowed	Allowed	Allowed	Allowed
REPORT SUPPORTED OPERATION CODES	Conflict	Conflict	Allowed	Conflict	Conflict
REPORT SUPPORTED TASK MANAGEMENT FUNCTIONS	Conflict	Conflict	Allowed	Conflict	Conflict
REPORT TARGET PORT GROUPS	Allowed	Allowed	Allowed	Allowed	Allowed
REPORT TIMESTAMP	Allowed	Allowed	Allowed	Allowed	Allowed
REQUEST SENSE	Allowed	Allowed	Allowed	Allowed	Allowed
RESERVE(6) / RESERVE(10)	As defined in SPC-2 [a]				
SEND DIAGNOSTIC	Conflict	Conflict	Allowed	Conflict	Conflict
SET DEVICE IDENTIFIER	Conflict	Conflict	Allowed	Conflict	Conflict
SET PRIORITY	Conflict	Conflict	Allowed	Conflict	Conflict
SET TARGET PORT GROUPS	Conflict	Conflict	Allowed	Conflict	Conflict
SET TIMESTAMP	Conflict	Conflict	Allowed	Conflict	Conflict
TEST UNIT READY	Allowed[b]	Allowed[b]	Allowed	Allowed [b]	Allowed [b]
WRITE ATTRIBUTE	Conflict	Conflict	Allowed	Conflict	Conflict
WRITE BUFFER	Conflict	Conflict	Allowed	Conflict	Conflict

Key: **Excl**=Exclusive, **RR**=Registrants Only or All Registrants, **<>** Not Equal

[a] Exceptions to the behavior of the RESERVE and RELEASE commands described in SPC-2 are defined in 5.6.3.

[b] Logical units claiming compliance with previous versions of this standard (e.g., SPC-2) may return RESERVATION CONFLICT in this case.

Table 32 — PERSISTENT RESERVE OUT service actions that are allowed in the presence of various reservations

Command Service Action	Addressed logical unit has a persistent reservation held by another I_T nexus	
	Command is from a registered I_T nexus	Command is from a not registered I_T nexus
CLEAR	Allowed	Conflict
PREEMPT	Allowed	Conflict
PREEMPT AND ABORT	Allowed	Conflict
REGISTER	Allowed	Allowed
REGISTER AND IGNORE EXISTING KEY	Allowed	Allowed
REGISTER AND MOVE	Conflict	Conflict
RELEASE	Allowed [a]	Conflict
RESERVE	Conflict	Conflict

[a] The reservation is not released (see 5.6.10.2).

The time at which a reservation is established with respect to other tasks being managed by the device server is vendor specific. Successful completion of a reservation command indicates that the new reservation is established. A reservation may apply to some or all of the tasks in the task set before the completion of the reservation command. The reservation shall apply to all tasks received by the device server after successful completion of the reservation command. Any persistent reserve service action shall be performed as a single indivisible event.

Multiple persistent reserve service actions may be present in the task set at the same time. The order of processing of such service actions is defined by the task set management requirements defined in SAM-3, but each is processed as a single indivisible command without any interleaving of actions that may be required by other reservation commands.

5.6.2 Third party persistent reservations

Except for all registrants type reservations, a reservation holder (see 5.6.9) may move the persistent reservation to a third party (e.g., a copy manager supporting the EXTENDED COPY command) using the REGISTER AND MOVE service action (see 5.6.7). A copy manager supporting the EXTENDED COPY command may be instructed to move the persistent reservation to a specified I_T nexus using the third party persistent reservations source I_T nexus segment descriptor (see 6.3.7.19).

5.6.3 Exceptions to SPC-2 RESERVE and RELEASE behavior

This subclause defines exceptions to the behavior of the RESERVE and RELEASE commands defined in SPC-2. The RESERVE and RELEASE commands are obsolete in this standard, except for the behavior defined in this subclause. Device servers that operate using the exceptions described in this subclause shall set the CRH bit to one in the parameter data returned by the REPORT CAPABILITIES service action of the PERSISTENT RESERVE IN command (see 6.11.4).

A RELEASE(6) or RELEASE(10) command shall complete with GOOD status, but the persistent reservation shall not be released, if the command is received from:

a) An I_T nexus that is a persistent reservation holder (see 5.6.9); or
b) An I_T nexus that is registered if a registrants only or all registrants type persistent reservation is present.

A RESERVE(6) or RESERVE(10) command shall complete with GOOD status, but no reservation shall be established and the persistent reservation shall not be changed, if the command is received from:

a) An I_T nexus that is a persistent reservation holder; or
b) An I_T nexus that is registered if a registrants only or all registrants type persistent reservation is present.

In all other cases, a RESERVE(6) command, RESERVE(10) command, RELEASE(6) command, or RELEASE(10) command shall be processed as defined in SPC-2.

5.6.4 Preserving persistent reservations and registrations

5.6.4.1 Preserving persistent reservations and registrations through power loss

The application client may request activation of the persist through power loss device server capability to preserve the persistent reservation and registrations across power cycles by setting the APTPL bit to one in the PERSISTENT RESERVE OUT parameter data sent with a REGISTER service action, REGISTER AND IGNORE EXISTING KEY service action, or REGISTER AND MOVE service action.

After the application client enables the persist through power loss capability the device server shall preserve the persistent reservation, if any, and all current and future registrations associated with the logical unit to which the REGISTER service action, the REGISTER AND IGNORE EXISTING KEY service action, or REGISTER AND MOVE service action was addressed until an application client disables the persist through power loss capability. The APTPL value from the most recent successfully completed REGISTER service action, REGISTER AND IGNORE EXISTING KEY service action, or REGISTER AND MOVE service action from any application client shall determine the logical unit's behavior in the event of a power loss.

The device server shall preserve the following information for each existing registration across any hard reset, logical unit reset, or I_T nexus loss, and if the persist through power loss capability is enabled, across any power cycle:

a) For SCSI transport protocols where initiator port names (see 3.1.52) are required, the initiator port name; otherwise, the initiator port identifier (see 3.1.51);
b) Reservation key; and
c) Indication of the target port to which the registration was applied.

The device server shall preserve the following information about the existing persistent reservation across any hard reset, logical unit reset, or I_T nexus loss, and if the persist through power loss capability is enabled, across any power cycle:

a) For SCSI transport protocols where initiator port names are required, the initiator port name; otherwise, the initiator port identifier;
b) Reservation key;
c) Scope;
d) Type; and
e) Indication of the target port through which the reservation was established.

NOTE 9 - The scope of a persistent reservation is always LU_SCOPE (see 6.11.3.3). For an all registrants type persistent reservation, only the scope and type need to be preserved.

5.6.4.2 Nonvolatile memory considerations for preserving persistent reservations and registrations

The capability of preserving persistent reservations and registrations across power cycles requires logical units to use nonvolatile memory within the SCSI device. Any logical unit that supports the persist through power loss capability of persistent reservation and has nonvolatile memory that is not ready shall allow the following commands into the task set:

a) INQUIRY;
b) LOG SENSE;

c) READ BUFFER;
d) REPORT LUNS;
e) REQUEST SENSE;
f) START STOP UNIT (with the START bit set to one and POWER CONDITIONS field value of 0h); and
g) WRITE BUFFER.

When nonvolatile memory has not become ready since a power cycle, commands other than those listed in this subclause shall be terminated with CHECK CONDITION status, with the sense key set to NOT READY, and the additional sense code set as described in table 184 (see 6.33).

5.6.5 Finding persistent reservations and reservation keys

5.6.5.1 Summary of commands for finding persistent reservations and reservation keys

The application client may obtain information about the persistent reservation and the reservation keys (i.e., registrations) that are present within a device server by issuing a PERSISTENT RESERVE IN command with a READ RESERVATION service action, a READ KEYS service action, or a READ FULL STATUS service action.

5.6.5.2 Reporting reservation keys

An application client may issue a PERSISTENT RESERVE IN command with READ KEYS service action to determine if any I_T nexuses have been registered with a logical unit through any target port.

In response to a PERSISTENT RESERVE IN with READ KEYS service action the device server shall report the following:

a) The current PRgeneration value (see 6.11.2); and
b) The reservation key for every I_T nexus that is currently registered regardless of the target port through which the registration occurred.

The PRgeneration value allows the application client to verify that the configuration of the I_T nexuses registered with a logical unit has not been modified.

Duplicate reservation keys shall be reported if multiple I_T nexuses are registered using the same reservation key.

If an application client uses a different reservation key for each I_T nexus, the application client may use the reservation key to uniquely identify an I_T nexus.

5.6.5.3 Reporting the persistent reservation

An application client may issue a PERSISTENT RESERVE IN command with READ RESERVATION service action to receive the persistent reservation information.

In response to a PERSISTENT RESERVE IN command with READ RESERVATION service action the device server shall report the following information for the persistent reservation, if any:

a) The current PRgeneration value (see 6.11.2);
b) The registered reservation key, if any, associated with the I_T nexus that holds the persistent reservation (see 5.6.9). If the persistent reservation is an all registrants type, the registered reservation key reported shall be zero; and
c) The scope and type of the persistent reservation, if any.

If an application client uses a different reservation key for each I_T nexus, the application client may use the reservation key to associate the persistent reservation with the I_T nexus that holds the persistent reservation. This association is done using techniques that are outside the scope of this standard.

5.6.5.4 Reporting full status

An application client may issue a PERSISTENT RESERVE IN command with READ FULL STATUS service action to receive all information about registrations and the persistent reservation, if any.

In response to a PERSISTENT RESERVE IN command with READ FULL STATUS service action the device server shall report the current PRgeneration value (see 6.11.2) and, for every I_T nexus that is currently registered, the following information:

a) The registered reservation key;
b) Whether the I_T nexus is a persistent reservation holder;
c) If the I_T nexus is a persistent reservation holder, the scope and type of the persistent reservation;
d) The relative target port identifier identifying the target port of the I_T nexus; and
e) A TransportID identifying the initiator port of the I_T nexus.

5.6.6 Registering

To establish a persistent reservation the application client shall first register an I_T nexus with the device server. An application client registers with a logical unit by issuing a PERSISTENT RESERVE OUT command with REGISTER service action or REGISTER AND IGNORE EXISTING KEY service action.

If the I_T nexus has an established registration, an application client may remove the reservation key (see 5.6.10.3). This is accomplished by issuing a PERSISTENT RESERVE OUT command with a REGISTER service action or a REGISTER AND IGNORE EXISTING KEY service action as shown in table 33 and table 34, respectively.

If an I_T nexus has not yet established a reservation key or the reservation key and registration have been removed, an application client may register that I_T nexus and zero or more specified unregistered I_T nexuses by issuing a PERSISTENT RESERVE OUT command with REGISTER service action as defined in table 33.

If the I_T nexus has an established registration, the application client may change the reservation key by issuing a PERSISTENT RESERVE OUT command with REGISTER service action as defined in table 33.

Table 33 — Register behaviors for a REGISTER service action

Command I_T nexus status	Parameter list fields [a]			Results
	RESERVATION KEY	SERVICE ACTION RESERVATION KEY	SPEC_I_PT	
received on an unregistered I_T nexus	zero	zero	ignore	Do nothing except return GOOD status.
		non-zero	zero	Register the I_T nexus on which the command was received with the value specified in the SERVICE ACTION RESERVATION KEY field.
			one	Register the I_T nexus on which the command was received and each unregistered I_T nexus specified in the parameter list with the value specified in the SERVICE ACTION RESERVATION KEY field. [b]
	non-zero	ignore	ignore	Return RESERVATION CONFLICT status.
received on a registered I_T nexus	Not equal to I_T nexus reservation key	ignore	ignore	Return RESERVATION CONFLICT status.
	Equal to I_T nexus reservation key	zero	zero	Unregister the I_T nexus on which the command was received (see 5.6.10.3).
			one	Return CHECK CONDITION status. [c]
		non-zero	zero	Change the reservation key of the I_T nexus on which the command was received to the value specified in the SERVICE ACTION RESERVATION KEY field.
			one	Return CHECK CONDITION status. [c]

[a] For requirements regarding the parameter list fields not shown in this table see 6.12.3.

[b] If any I_T nexus specified in the parameter list is registered, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

[c] The sense key shall be set to ILLEGAL REQUEST, and the additional sense code shall be set to INVALID FIELD IN CDB.

Alternatively, an application client may establish a reservation key for an I_T nexus without regard for whether one has previously been established by issuing a PERSISTENT RESERVE OUT command with REGISTER AND IGNORE EXISTING KEY service action as defined in table 34.

Table 34 — Register behaviors for a REGISTER AND IGNORE EXISTING KEY service action

Command I_T nexus status	Parameter list fields [a]		Results
	SERVICE ACTION RESERVATION KEY	SPEC_I_PT	
received on an unregistered I_T nexus	zero	ignore	Do nothing except return GOOD status.
	non-zero	zero	Register the I_T nexus on which the command was received with the value specified in the SERVICE ACTION RESERVATION KEY field.
		one	Return CHECK CONDITION status. [b]
received on a registered I_T nexus	zero	zero	Unregister the I_T nexus on which the command was received (see 5.6.10.3).
		one	Return CHECK CONDITION status. [b]
	non-zero	zero	Change the reservation key of the I_T nexus on which the command was received to the value specified in the SERVICE ACTION RESERVATION KEY field.
		one	Return CHECK CONDITION status. [b]

[a] The RESERVATION KEY field is ignored when processing a REGISTER AND IGNORE EXISTING KEY service action. For requirements regarding other parameter list fields not shown in this table see 6.12.3.

[b] The sense key shall be set to ILLEGAL REQUEST, and the additional sense code shall be set to INVALID FIELD IN CDB.

If a PERSISTENT RESERVE OUT command with a REGISTER service action or a REGISTER AND IGNORE EXISTING KEY service action is attempted, but there are insufficient device server resources to complete the operation, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INSUFFICIENT REGISTRATION RESOURCES.

In response to a PERSISTENT RESERVE OUT command with a REGISTER service action or a REGISTER AND IGNORE EXISTING KEY service action the device server shall perform a registration for each specified I_T nexus by doing the following as an uninterrupted series of actions:

a) Process the registration request regardless of any persistent reservations;
b) Process the APTPL bit;
c) Ignore the contents of the SCOPE and TYPE fields;
d) Associate the reservation key specified in the SERVICE ACTION RESERVATION KEY field with the I_T nexus being registered, where:
   A) The I_T nexus(es) being registered are shown in table 35; and
   B) Regardless of how the I_T nexus initiator port is specified, the association for the initiator port is based on either the initiator port name (see 3.1.52) on SCSI transport protocols where port names are required or the initiator port identifier (see 3.1.51) on SCSI transport protocols where port names are not required;
e) Register the reservation key specified in the SERVICE ACTION RESERVATION KEY field without changing any persistent reservation that may exist; and
f) Retain the reservation key specified in the SERVICE ACTION RESERVATION KEY field and associated information.

Table 35 — I_T Nexuses being registered

SPEC_I_PT	ALL_TG_PT	I_T nexus(es) being registered	
		Initiator port	Target port
0	0	The port's names or identifiers to be registered are determined from the I_T nexus on which the PERSISTENT RESERVE OUT command was received	
0	1	The port's name or identifier to be registered is determined from the I_T nexus on which the PERSISTENT RESERVE OUT command was received	Register all of the target ports in the SCSI target device
1	0	a) The port's name or identifier to be registered is determined from the I_T nexus on which the PERSISTENT RESERVE OUT command was received; and b) Specified by each TransportID in the additional parameter data (see 6.12.3)	The port's name or identifier to be registered is determined from the I_T nexus on which the PERSISTENT RESERVE OUT command was received
1	1	a) The port's name or identifier to be registered is determined from the I_T nexus on which the PERSISTENT RESERVE OUT command was received; and b) Specified by each TransportID in the additional parameter data	Register all of the target ports in the SCSI target device

After the registration request has been processed, the device server shall then allow other PERSISTENT RESERVE OUT commands from the registered I_T nexus to be processed. The device server shall retain the reservation key until the key is changed as described in this subclause or removed as described in 5.6.10.

Any PERSISTENT RESERVE OUT command service action received from an unregistered I_T nexus, other than the REGISTER or the REGISTER AND IGNORE EXISTING KEY service action, shall be rejected with a RESERVATION CONFLICT status.

It is not an error for an I_T nexus that is registered to be registered again with the same reservation key or a new reservation key. A registration shall have no effect on any other registrations (e.g., when more than one I_T nexus is registered with the same reservation key and one of those I_T nexuses registers again it has no effect on the other I_T nexus' registrations). A registration that contains a non-zero value in the SERVICE ACTION RESERVATION KEY field shall have no effect on any persistent reservations (i.e., the reservation key for an I_T nexus may be changed without affecting any previously created persistent reservation).

Multiple I_T nexuses may be registered with the same reservation key. An application client may use the same reservation key for other I_T nexuses and logical units.

5.6.7 Registering and moving the reservation

The PERSISTENT RESERVE OUT command REGISTER AND MOVE service action is used to register a specified I_T nexus (see table 36) and move the reservation to that I_T nexus.

Table 36 — Register behaviors for a REGISTER AND MOVE service action

Command I_T nexus status	Parameter list fields [a]			Results
	RESERVATION KEY	SERVICE ACTION RESERVATION KEY	UNREG	
received on an unregistered I_T nexus	ignore	ignore	ignore	If there is an existing persistent reservation, return RESERVATION CONFLICT status. If there is not an existing persistent reservation, return CHECK CONDITION status. [b]
received on the registered I_T nexus of reservation holder	Not equal to I_T nexus reservation key	ignore	ignore	Return RESERVATION CONFLICT status.
	Equal to I_T nexus reservation key	zero	ignore	Return CHECK CONDITION status. [b]
		non-zero [c]	zero	The I_T nexus on which PERSISTENT RESERVE OUT command was received shall remain registered. See this subclause for the registration and the move specifications.
			one	The I_T nexus on which PERSISTENT RESERVE OUT command was received shall be unregistered (see 5.6.10.3) upon completion of command processing. See this subclause for the registration and the move specifications.
received on a registered I_T nexus that is not the reservation holder	ignore	ignore	ignore	Return RESERVATION CONFLICT status.

[a] For requirements regarding other parameter list fields not shown in this table see 6.12.4.
[b] The sense key shall be set to ILLEGAL REQUEST, and the additional sense code shall be set to INVALID FIELD IN CDB.
[c] The application client and backup application should use the same reservation key.

If a PERSISTENT RESERVE OUT command with a REGISTER AND MOVE service action is attempted, but there are insufficient device server resources to complete the operation, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INSUFFICIENT REGISTRATION RESOURCES.

If a PERSISTENT RESERVE OUT command with a REGISTER AND MOVE service action is received and the established persistent reservation is a Write Exclusive - All Registrants type or Exclusive Access - All Registrants type reservation, then the command shall be terminated with RESERVATION CONFLICT status.

If a PERSISTENT RESERVE OUT command with a REGISTER AND MOVE service action is received and there is no persistent reservation established, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

If a PERSISTENT RESERVE OUT command with a REGISTER AND MOVE service action specifies a TransportID that is the same as the initiator port of the I_T nexus on which the command received, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

In response to a PERSISTENT RESERVE OUT command with a REGISTER AND MOVE service action the device server shall perform a register and move by doing the following as an uninterrupted series of actions:

a) Process the APTPL bit;
b) Ignore the contents of the SCOPE and TYPE fields;
c) Associate the reservation key specified in the SERVICE ACTION RESERVATION KEY field with the I_T nexus specified as the destination of the register and move, where:
   A) The I_T nexus is specified by the TransportID and the RELATIVE TARGET PORT IDENTIFIER field (see 6.12.4); and
   B) Regardless of the TransportID format used, the association for the initiator port is based on either the initiator port name (see 3.1.52) on SCSI transport protocols where port names are required or the initiator port identifier (see 3.1.51) on SCSI transport protocols where port names are not required;
d) Register the reservation key specified in the SERVICE ACTION RESERVATION KEY field;
e) Retain the reservation key specified in the SERVICE ACTION RESERVATION KEY field and associated information;
f) Release the persistent reservation for the persistent reservation holder (i.e., the I_T nexus on which the command was received);
g) Move the persistent reservation to the specified I_T nexus using the same scope and type as the persistent reservation released in item f); and
h) If the UNREG bit is set to one, unregister (see 5.6.10.3) the I_T nexus on which PERSISTENT RESERVE OUT command was received.

It is not an error for a REGISTER AND MOVE service action to register an I_T nexus that is already registered with the same reservation key or a different reservation key.

5.6.8 Reserving

An application client creates a persistent reservation by issuing a PERSISTENT RESERVE OUT command with RESERVE service action through a registered I_T nexus with the following parameters:

a) RESERVATION KEY set to the value of the reservation key that is registered with the logical unit for the I_T nexus; and
b) TYPE and SCOPE fields set to the persistent reservation being created.

Only one persistent reservation is allowed at a time per logical unit and that persistent reservation has a scope of LU_SCOPE.

If the device server receives a PERSISTENT RESERVE OUT command from an I_T nexus other than a persistent reservation holder (see 5.6.9) that attempts to create a persistent reservation when a persistent reservation already exists for the logical unit, then the command shall be rejected with a RESERVATION CONFLICT status.

If a persistent reservation holder attempts to modify the TYPE or SCOPE of an existing persistent reservation, the command shall be rejected with a RESERVATION CONFLICT status.

If the device server receives a PERSISTENT RESERVE OUT command with RESERVE service action where the TYPE and SCOPE are the same as the existing TYPE and SCOPE from a persistent reservation holder, it shall not make any change to the existing persistent reservation and shall return a GOOD status.

See 5.6.1 for information on when a persistent reservation takes effect.

5.6.9 Persistent reservation holder

The persistent reservation holder is determined by the type of the persistent reservation as follows:

a) For a persistent reservation of the type Write Exclusive – All Registrants or Exclusive Access – All Registrants, the persistent reservation holder is any registered I_T nexus; or
b) For all other persistent reservation types, the persistent reservation holder is the I_T nexus:
   A) For which the reservation was established with a PERSISTENT RESERVE OUT command with REGISTER service action, REGISTER AND IGNORE EXISTING KEY service action, PREEMPT service action, or PREEMPT AND ABORT service action; or
   B) To which the reservation was moved by a PERSISTENT RESERVE OUT command with REGISTER AND MOVE service action.

A persistent reservation holder has its reservation key returned in the parameter data from a PERSISTENT RESERVE IN command with READ RESERVATION service action as follows:

a) For a persistent reservation of the type Write Exclusive – All Registrants or Exclusive Access – All Registrants, the reservation key shall be set to zero; or
b) For all other persistent reservation types, the reservation key shall be set to the registered reservation key for the I_T nexus that holds the persistent reservation.

It is not an error for a persistent reservation holder to send a PERSISTENT RESERVE OUT command with RESERVE service action to the reserved logical unit with TYPE and SCOPE fields that match those of the persistent reservation (see 5.6.8).

A persistent reservation holder is allowed to release the persistent reservation using the PERSISTENT RESERVE OUT command with RELEASE service action (see 5.6.10.2).

If the registration of the persistent reservation holder is removed (see 5.6.10.1.1), the reservation shall be released. If the persistent reservation holder is more than one I_T nexus, the reservation shall not be released until the registrations for all persistent reservation holder I_T nexuses are removed.

5.6.10 Releasing persistent reservations and removing registrations

5.6.10.1 Overview

5.6.10.1.1 Summary of service actions that release persistent reservations and remove registrations

An application client may release or preempt the persistent reservation by issuing one of the following commands through a registered I_T nexus with the RESERVATION KEY field set to the reservation key value that is registered with the logical unit for that I_T nexus:

a) A PERSISTENT RESERVE OUT command with RELEASE service action from a persistent reservation holder (see 5.6.10.2);
b) A PERSISTENT RESERVE OUT command with PREEMPT service action specifying the reservation key of the persistent reservation holder or holders (see 5.6.10.4);
c) A PERSISTENT RESERVE OUT command with PREEMPT AND ABORT service action specifying the reservation key of the persistent reservation holder or holders (see 5.6.10.5);
d) A PERSISTENT RESERVE OUT command with CLEAR service action (see 5.6.10.6); or
e) If the I_T nexus is the persistent reservation holder and the persistent reservation is not an all registrants type, then a PERSISTENT RESERVE OUT command with REGISTER service action or REGISTER AND IGNORE EXISTING KEY service action with the SERVICE ACTION RESERVATION KEY field set to zero (see 5.6.10.3).

An application client may remove registrations by issuing one of the following commands through a registered I_T nexus with the RESERVATION KEY field set to the reservation key value that is registered with the logical unit for that I_T nexus:

a) A PERSISTENT RESERVE OUT command with PREEMPT service action with the SERVICE ACTION RESERVATION KEY field set to the reservation key (see 5.6.10.4) to be removed;
b) A PERSISTENT RESERVE OUT command with PREEMPT AND ABORT service action with the SERVICE ACTION RESERVATION KEY field set to the reservation key (see 5.6.10.5) to be removed;
c) A PERSISTENT RESERVE OUT command with CLEAR service action (see 5.6.10.6); or
d) A PERSISTENT RESERVE OUT command with REGISTER service action or REGISTER AND IGNORE EXISTING KEY service action with the SERVICE ACTION RESERVATION KEY field set to zero (see 5.6.10.3).

When a reservation key (i.e., registration) has been removed, no information shall be reported for that unregistered I_T nexus in subsequent READ KEYS service actions until the I_T nexus is registered again (see 5.6.6). As shown in table 37, the processing of any persistent reservation whose persistent reservation holder or holders become unregistered depends on the reservation type.

Table 37 — Processing for released persistent reservations

Reservation Type	Reference
Write Exclusive – Registrants Only or Exclusive Access – Registrants Only	5.6.10.1.2
Write Exclusive – All Registrants or Exclusive Access – All Registrants	5.6.10.1.3
Write Exclusive or Exclusive Access	5.6.10.1.4

Registrations and persistent reservations may also be released by a loss of power, if the persist through power loss capability is not enabled. When the most recent APTPL value received by the device server is zero (see 6.12.3), a power cycle:

a) Releases all persistent reservations; and
b) Removes all registered reservation keys (see 5.6.6).

5.6.10.1.2 Processing for released Registrants Only persistent reservations

When the persistent reservation holder (see 5.6.9) of a Write Exclusive – Registrants Only or Exclusive Access – Registrants Only type reservation becomes unregistered the persistent reservation shall be released.

For every I_T nexus whose reservation key is removed, the device server shall establish a unit attention condition for the initiator port associated with that I_T nexus and the additional sense code shall be based on the PERSISTENT RESERVE OUT command service action as follows:

a) If the service action was CLEAR, the additional sense code shall be set to RESERVATIONS PREEMPTED; or
b) If the service action was PREEMPT or PREEMPT AND ABORT, the additional sense code shall be set to REGISTRATIONS PREEMPTED.

If the TYPE or SCOPE have changed, then for every I_T nexus whose reservation key was not removed except for the I_T nexus on which the PERSISTENT RESERVE OUT command was received, the device server shall establish a unit attention condition for the initiator port associated with that I_T nexus, with the additional sense code set to RESERVATIONS RELEASED. If the TYPE or SCOPE have not changed, then no unit attention condition(s) shall be established for this reason.

If the reservation was released, then for every I_T nexus whose reservation key was not removed except for the I_T nexus on which the PERSISTENT RESERVE OUT command was received, the device server shall establish a unit attention condition for the initiator port associated with that I_T nexus, with the additional sense code set to

RESERVATIONS RELEASED. If the reservation was not released, then no unit attention condition(s) shall be established for this reason.

5.6.10.1.3 Processing for released All Registrants persistent reservations

A Write Exclusive – All Registrants or Exclusive Access – All Registrants type persistent reservation shall be released when the registration for the last registered I_T nexus is removed or when the TYPE or SCOPE is changed.

The device server shall establish a unit attention condition for the initiator port associated with every registered I_T nexus whose reservation key was removed, with the additional sense code set as follows:

a) If the service action was CLEAR, the additional sense code shall be set to RESERVATIONS PREEMPTED; or
b) If the service action was PREEMPT or PREEMPT AND ABORT, the additional sense code shall be set to REGISTRATIONS PREEMPTED.

If a persistent reservation was released using a RELEASE service action, see 5.6.10.2.

5.6.10.1.4 Processing for other released persistent reservations

When the persistent reservation holder (see 5.6.9) of a Write Exclusive or Exclusive Access type reservation becomes unregistered the persistent reservation shall be released.

5.6.10.2 Releasing

Only the persistent reservation holder (see 5.6.9) is allowed to release a persistent reservation.

An application client releases the persistent reservation by issuing a PERSISTENT RESERVE OUT command with RELEASE service action through an I_T nexus that is a persistent reservation holder with the following parameters:

a) RESERVATION KEY field set to the value of the reservation key that is registered with the logical unit for the I_T nexus; and
b) TYPE and SCOPE fields set to match the persistent reservation being released.

In response to a persistent reservation release request from the persistent reservation holder the device server shall perform a release by doing the following as an uninterrupted series of actions:

a) Release the persistent reservation;
b) Not remove any registration(s);
c) If the released persistent reservation is a registrants only type or all registrants type persistent reservation, the device server shall establish a unit attention condition for the initiator port associated with every registered I_T nexus other than I_T nexus on which the PERSISTENT RESERVE OUT command with RELEASE service action was received, with the additional sense code set to RESERVATIONS RELEASED; and
d) If the persistent reservation is of any other type, the device server shall not establish a unit attention condition.

The established persistent reservation shall not be altered and the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID RELEASE OF PERSISTENT RESERVATION, for a PERSISTENT RESERVE OUT command that specifies the release of a persistent reservation if:

a) The requesting I_T nexus is a persistent reservation holder (see 5.6.9); and
b) The SCOPE and TYPE fields do not match the scope and type of the established persistent reservation.

If there is no persistent reservation or in response to a persistent reservation release request from a registered I_T nexus that is not a persistent reservation holder (see 5.6.9), the device server shall do the following:

a) Not release the persistent reservation, if any;
b) Not remove any registrations; and
c) Return GOOD status.

5.6.10.3 Unregistering

An application client may remove a registration for an I_T nexus by issuing a PERSISTENT RESERVE OUT command with REGISTER service action or a REGISTER AND IGNORE EXISTING KEY service action with the SERVICE ACTION RESERVATION KEY field set to zero through that I_T nexus.

If the I_T nexus is a reservation holder, the persistent reservation is of an all registrants type, and the I_T nexus is the last remaining registered I_T nexus, then the device server shall also release the persistent reservation.

If the I_T nexus is the reservation holder and the persistent reservation is of a type other than all registrants, the device server shall also release the persistent reservation. If the persistent reservation is a registrants only type, the device server shall establish a unit attention condition for the initiator port associated with every registered I_T nexus, with the additional sense code set to RESERVATIONS RELEASED.

5.6.10.4 Preempting

5.6.10.4.1 Overview

A PERSISTENT RESERVE OUT command with PREEMPT service action or PREEMPT AND ABORT service action is used to:

a) Preempt (i.e., replace) the persistent reservation and remove registrations; or
b) Remove registrations.

Table 38 lists the actions taken based on the current persistent reservation type and the SERVICE ACTION RESERVATION KEY field in the PERSISTENT RESERVE OUT command.

Table 38 — Preempting actions

Reservation Type	Service Action Reservation Key	Action	Reference
All Registrants	Zero	Preempt the persistent reservation and remove registrations.	5.6.10.4.3
	Not Zero	Remove registrations.	5.6.10.4.4
All other types	Zero	Terminate the command with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.	
	Reservation holder's reservation key	Preempt the persistent reservation and remove registrations.	5.6.10.4.3
	Any other, non-zero reservation key	Remove registrations.	5.6.10.4.4

See figure 3 for a description of how a device server interprets a PREEMPT service action to determine its actions (e.g., preempt the persistent reservation, remove registration, or both preempt the persistent reservation and remove registration).

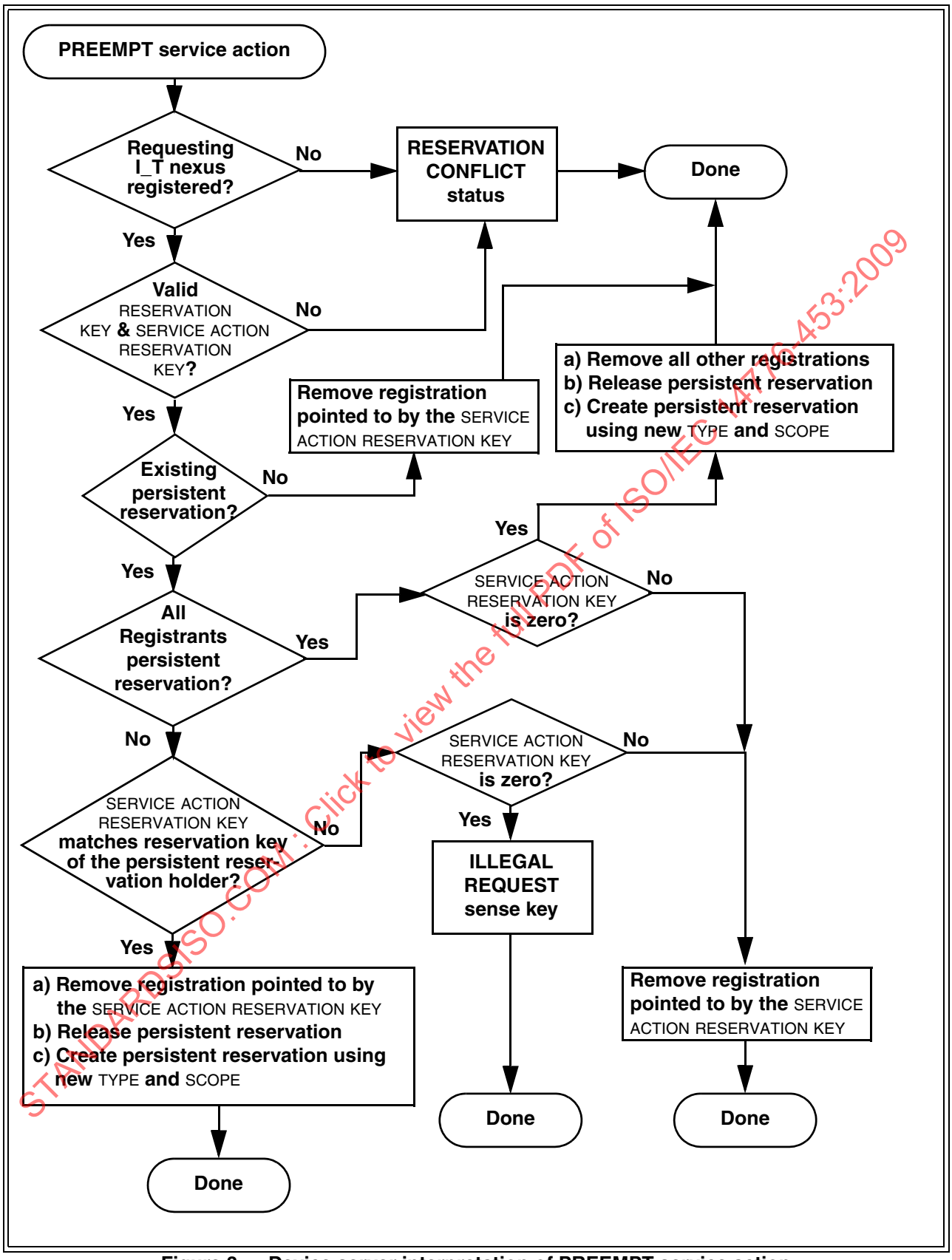

Figure 3 — Device server interpretation of PREEMPT service action

5.6.10.4.2 Failed persistent reservation preempt

If the preempting I_T nexus' PREEMPT service action or PREEMPT AND ABORT service action fails (e.g., repeated TASK SET FULL status, repeated BUSY status, SCSI transport protocol time-out, or time-out due to the task set being blocked due to failed initiator port or failed SCSI initiator device), the application client may send a LOGICAL UNIT RESET task management function to the failing logical unit to remove blocking tasks and then reissue the preempting service action.

5.6.10.4.3 Preempting persistent reservations and registration handling

An application client may preempt the persistent reservation with another persistent reservation by issuing a PERSISTENT RESERVE OUT command with PREEMPT service action or PREEMPT AND ABORT service action through a registered I_T nexus with the following parameters:

a) RESERVATION KEY field set to the value of the reservation key that is registered with the logical unit for the I_T nexus;
b) SERVICE ACTION RESERVATION KEY field set to the value of the reservation key of the persistent reservation to be preempted; and
c) TYPE and SCOPE fields set to define a new persistent reservation. The SCOPE and TYPE of the persistent reservation created by the preempting I_T nexus may be different than those of the persistent reservation being preempted.

If the SERVICE ACTION RESERVATION KEY field identifies a persistent reservation holder (see 5.6.9), the device server shall perform a preempt by doing the following as an uninterrupted series of actions:

a) Release the persistent reservation for the holder identified by the SERVICE ACTION RESERVATION KEY field;
b) Remove the registrations for all I_T nexuses identified by the SERVICE ACTION RESERVATION KEY field, except the I_T nexus that is being used for the PERSISTENT RESERVE OUT command. If an all registrants persistent reservation is present and the SERVICE ACTION RESERVATION KEY field is set to zero, then all registrations shall be removed except for that of the I_T nexus that is being used for the PERSISTENT RESERVE OUT command;
c) Establish a persistent reservation for the preempting I_T nexus using the contents of the SCOPE and TYPE fields;
d) Process tasks as defined in 5.6.11; and
e) Establish a unit attention condition for the initiator port associated with every I_T nexus that lost its persistent reservation and/or registration, with the additional sense code set to REGISTRATIONS PREEMPTED.

After GOOD status has been returned for the PERSISTENT RESERVE OUT command, new tasks are subject to the persistent reservation restrictions established by the preempting I_T nexus.

The following tasks shall be subjected in a vendor specific manner either to the restrictions established by the persistent reservation being preempted or to the restrictions established by the preempting I_T nexus:

a) A task received after the arrival, but before the completion of the PERSISTENT RESERVE OUT command with the PREEMPT service action or the PREEMPT AND ABORT service action; or
b) A task in the dormant, blocked, or enabled state (see SAM-3) at the time the PERSISTENT RESERVE OUT command with the PREEMPT service action or the PREEMPT AND ABORT service action is received.

Completion status shall be returned for each task unless it was aborted by a PERSISTENT RESERVE OUT command with the PREEMPT AND ABORT service action and TAS bit set to zero in the Control mode page (see 7.4.6).

If an all registrants persistent reservation is not present, it is not an error for the persistent reservation holder to preempt itself (i.e., a PERSISTENT RESERVE OUT with a PREEMPT service action or a PREEMPT AND ABORT service action with the SERVICE ACTION RESERVATION KEY value equal to the persistent reservation holder's reser-

vation key that is received from the persistent reservation holder). In that case, the device server shall establish the new persistent reservation and maintain the registration.

5.6.10.4.4 Removing registrations

When a registered reservation key does not identify a persistent reservation holder (see 5.6.9), an application client may remove the registration(s) without affecting any persistent reservations by issuing a PERSISTENT RESERVE OUT command with PREEMPT service action through a registered I_T nexus with the following parameters:

a) RESERVATION KEY field set to the value of the reservation key that is registered for the I_T nexus; and
b) SERVICE ACTION RESERVATION KEY field set to match the reservation key of the registration or registrations being removed.

If the SERVICE ACTION RESERVATION KEY field does not identify a persistent reservation holder or there is no persistent reservation holder (i.e., there is no persistent reservation), then the device server shall perform a preempt by doing the following in an uninterrupted series of actions:

a) Remove the registrations for all I_T nexuses specified by the SERVICE ACTION RESERVATION KEY field;
b) Ignore the contents of the SCOPE and TYPE fields;
c) Process tasks as defined in 5.6.1; and
d) Establish a unit attention condition for the initiator port associated with every I_T nexus that lost its registration other than the I_T nexus on which the PERSISTENT RESERVE OUT command was received, with the additional sense code set to REGISTRATIONS PREEMPTED.

If a PERSISTENT RESERVE OUT with a PREEMPT service action or a PREEMPT AND ABORT service action sets the SERVICE ACTION RESERVATION KEY field to a value that does not match any registered reservation key, then the device server shall return a RESERVATION CONFLICT status.

It is not an error for a PERSISTENT RESERVE OUT with a PREEMPT service action or a PREEMPT AND ABORT service action to set the RESERVATION KEY and the SERVICE ACTION RESERVATION KEY to the same value, however, no unit attention condition is established for the I_T nexus on which the PERSISTENT RESERVE OUT command was received. The registration is removed.

5.6.10.5 Preempting and aborting

The application client's request for and the device server's responses to a PERSISTENT RESERVE OUT command PREEMPT AND ABORT service action are identical to the responses to a PREEMPT service action (see 5.6.10.4) except for the following additions. If no reservation conflict occurred, the device server shall perform the following uninterrupted series of actions:

a) If the persistent reservation is not an all registrants type then:
   A) If the TST field is 000b (see 7.4.6) and the faulted I_T nexus (see 3.1.38), if any, is not the I_T nexus associated with the persistent reservation or registration being preempted, then the task set ACA condition shall be processed as defined in SAM-3;
   B) If the TST field contains 000b and the faulted I_T nexus, if any, is the I_T nexus associated with the persistent reservation or registration being preempted, then the PERSISTENT RESERVE OUT command shall be processed without regard for the task set ACA condition; or
   C) If the TST field contains 001b, then the ACA condition shall be processed as defined in SAM-3;
b) Perform the uninterrupted series of actions described for the PREEMPT service action (see 5.6.10.4);
c) All tasks from the I_T nexus(es) associated with the persistent reservations or registrations being preempted (i.e., preempted tasks) except the task containing the PERSISTENT RESERVE OUT command itself shall be aborted as defined in SAM-3. If an aborted task is a command that causes the device server to generate additional commands and data transfers (e.g., EXTENDED COPY), then all commands and data transfers generated by the command shall be aborted before the ABORT TASK SET task management function is considered completed. After the ABORT TASK SET function has completed, all new tasks are subject to the persistent reservation restrictions established by the preempting I_T nexus;

d) If the persistent reservation is not an all registrants type, then the device server shall clear any ACA condition associated with an I_T nexus being preempted and shall abort any tasks with an ACA attribute received on that I_T nexus;
e) If the persistent reservation is an all registrants type, then:
   A) If the service action reservation key is set to zero, the device server shall clear any ACA condition and shall abort any tasks with an ACA attribute; or
   B) If the service action reservation key is not set to zero, the device server shall do the following for any I_T nexus registered using the specified reservation key:
      a) Clear any ACA condition; and
      b) Abort any tasks with an ACA attribute;

   and
f) For logical units that implement the PREVENT ALLOW MEDIUM REMOVAL command, the device server shall perform an action equivalent to the processing of a PREVENT ALLOW MEDIUM REMOVAL command with the PREVENT field equal to zero received on the I_T nexuses associated with the persistent reservation being preempted (see 6.13).

The actions described in this subclause shall be performed for all I_T nexuses that are registered with the non-zero SERVICE ACTION RESERVATION KEY value, without regard for whether the preempted I_T nexuses hold the persistent reservation. If the SERVICE ACTION RESERVATION KEY value is zero and an all registrants persistent reservation is present, the device server shall abort all tasks for all registered I_T nexuses.

5.6.10.6 Clearing

Any application client may release the persistent reservation and remove all registrations from a device server by issuing a PERSISTENT RESERVE OUT command with CLEAR service action through a registered I_T nexus with the following parameter:

a) RESERVATION KEY field set to the value of the reservation key that is registered with the logical unit for the I_T nexus.

In response to this request the device server shall perform a clear by doing the following as part of an uninterrupted series of actions:

a) Release the persistent reservation, if any;
b) Remove all registration(s) (see 5.6.6);
c) Ignore the contents of the SCOPE and TYPE fields;
d) Continue normal processing of any tasks from any I_T nexus that have been accepted by the device server as allowed (i.e., nonconflicting); and
e) Establish a unit attention condition for the initiator port associated with every registered I_T nexus other than the I_T nexus on which the PERSISTENT RESERVE OUT command with CLEAR service action was received, with the additional sense code set to RESERVATIONS PREEMPTED.

NOTE 10 - Application clients should not use the CLEAR service action except during recovery operations that are associated with a specific initiator port, since the effect of the CLEAR service action defeats the persistent reservations features that protect data integrity.

5.7 Multiple target port and initiator port behavior

SAM-3 specifies the behavior of logical units being accessed by application clients through more than one initiator port and/or through more than one target port. Additional initiator ports and target ports allow the definition of multiple I_T nexuses through which the device server may be reached. Multiple I_T nexuses may be used to improve the availability of logical units in the presence of certain types of failures and to improve the performance between an application client and logical unit when some I_T nexuses may be busy.

If one target port is being used by an initiator port, accesses attempted through other target port(s) may:

a) Receive a status of BUSY; or
b) Be accepted as if the other target port(s) were not in use.

The device server shall indicate the presence of multiple target ports by setting the MULTIP bit to one in its standard INQUIRY data.

Only the following operations allow one I_T nexus to interact with the tasks of other I_T nexuses:

a) The PERSISTENT RESERVE OUT with PREEMPT service action preempts persistent reservations (see 5.6.10.4);
b) The PERSISTENT RESERVE OUT with CLEAR service action releases persistent reservations for all I_T nexuses (see 5.6.10.6); and
c) Commands and task management functions that allow one I_T nexus to abort tasks received on a different I_T nexus (see SAM-3).

5.8 Target port group access states

5.8.1 Target port group access overview

Logical units may be connected to the service delivery subsystem via multiple target ports (see SAM-3). The access to logical units through the multiple target ports may be symmetrical (see 5.8.3) or asymmetrical (see 5.8.2).

5.8.2 Asymmetric logical unit access

5.8.2.1 Introduction to asymmetric logical unit access

Asymmetric logical unit access occurs when the access characteristics of one port may differ from those of another port. SCSI target devices with target ports implemented in separate physical units may need to designate differing levels of access for the target ports associated with each logical unit. While commands and task management functions (see SAM-3) may be routed to a logical unit through any target port, the performance may not be optimal, and the allowable command set may be less complete than when the same commands and task management functions are routed through a different target port. When a failure on the path to one target port is detected, the SCSI target device may perform automatic internal reconfiguration to make a logical unit accessible from a different set of target ports or may be instructed by the application client to make a logical unit accessible from a different set of target ports.

A target port characteristic called target port asymmetric access state (see 5.8.2.4) defines properties of a target port and the allowable command set for a logical unit when commands and task management functions are routed through the target port maintaining that state.

A target port group is defined as a set of target ports that are in the same target port asymmetric access state at all times. A target port group asymmetric access state is defined as the target port asymmetric access state common to the set of target ports in a target port group. The grouping of target ports is vendor specific.

A logical unit may have commands and task management functions routed through multiple target port groups. Logical units support asymmetric logical unit access if different target port groups may be in different target port group asymmetric access states.

An example of asymmetric logical unit access is a SCSI controller device with two separated controllers where all target ports on one controller are in the same asymmetric access state with respect to a logical unit and are members of the same target port group. Target ports on the other controller are members of another target port group. The behavior of each target port group may be different with respect to a logical unit, but all members of a single target port group are always in the same target port asymmetric access state with respect to a logical unit.

An example of target port groups is shown in figure 4.

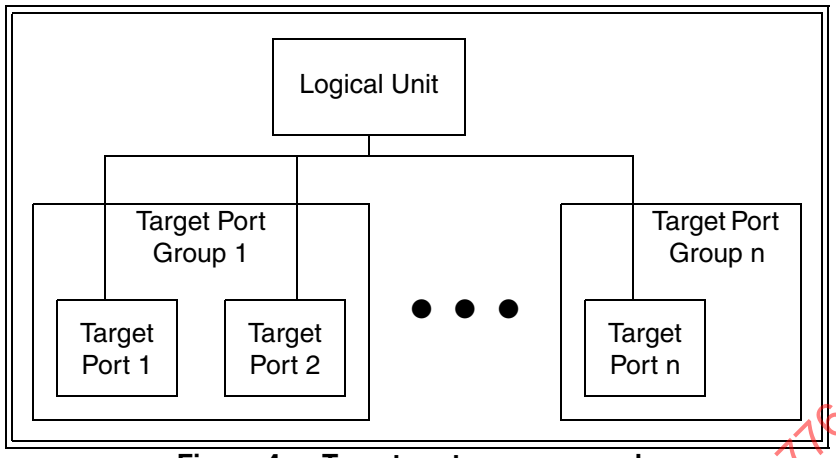

Figure 4 — Target port group example

5.8.2.2 Explicit and implicit asymmetric logical unit access

Asymmetric logical unit access may be managed explicitly by an application client using the REPORT TARGET PORT GROUPS (see 6.25) and SET TARGET PORT GROUPS (see 6.31) commands.

Alternatively, asymmetric logical unit access may be managed implicitly by the SCSI target device based on the type of transactions being routed through each target port and the internal configuration capabilities of the target port group(s) through which the logical unit may be accessed. The logical units may attempt to maintain full performance across the target port groups that are busiest and that show the most reliable performance, allowing other target port groups to select a lower performance target port asymmetric access state.

If both explicit and implicit asymmetric logical unit access management methods are implemented, the precedence of one over the other is vendor specific.

5.8.2.3 Discovery of asymmetric logical unit access behavior

SCSI logical units with asymmetric logical unit access may be identified using the INQUIRY command. The value in the target port group support (TPGS) field (see 6.4.2) indicates whether or not the logical unit supports asymmetric logical unit access and if so whether implicit or explicit management is supported. The asymmetric access states supported by a logical unit may be determined by the REPORT TARGET PORT GROUPS command parameter data (see 6.25).

5.8.2.4 Target port asymmetric access states

5.8.2.4.1 Target port asymmetric access states overview

For all SCSI target devices that report in the INQUIRY data that they support asymmetric logical unit access, all of the target ports in a target port group shall be in the same target port asymmetric access state with respect to the ability to route information to a logical unit. The target port asymmetric access states are:

a) Active/optimized;
b) Active/non-optimized;
c) Standby; and
d) Unavailable.

5.8.2.4.2 Active/optimized state

When commands and task management functions are being routed through a target port in the active/optimized target port asymmetric access state, the device server shall function (e.g., respond to commands) as specified in

the appropriate command standards (see 3.1.18). All target ports within a target port group should be capable of immediately accessing the logical unit.

The SCSI target device shall participate in all task management functions as defined in SAM-3 and modified by the applicable SCSI transport protocol standards (see 3.1.102).

5.8.2.4.3 Active/non-optimized state

When commands and task management functions are being routed through a target port in the active/non-optimized target port asymmetric access state, the device server shall function as specified in the appropriate command standards.

The processing of some task management functions and commands, especially those involving data transfer or caching, may operate with lower performance than they would if the target port were in the active/optimized target port asymmetric access state.

The SCSI target device shall participate in all task management functions as defined in SAM-3 and modified by the applicable SCSI transport protocol standards (see 3.1.102).

5.8.2.4.4 Standby state

When being accessed through a target port in the standby target port asymmetric access state, the device server shall support those of the following commands that it supports while in the active/optimized target port asymmetric access state:

a) INQUIRY;
b) LOG SELECT;
c) LOG SENSE;
d) MODE SELECT;
e) MODE SENSE;
f) REPORT LUNS (for LUN 0);
g) RECEIVE DIAGNOSTIC RESULTS;
h) SEND DIAGNOSTIC;
i) REPORT TARGET PORT GROUPS;
j) SET TARGET PORT GROUPS;
k) REQUEST SENSE;
l) PERSISTENT RESERVE IN;
m) PERSISTENT RESERVE OUT;
n) Echo buffer modes of READ BUFFER; and
o) Echo buffer modes of WRITE BUFFER.

The device server may support other commands.

For those commands that are not supported, the device server shall terminate the command with CHECK CONDITION status, with the sense key set to NOT READY, and the additional sense code set to LOGICAL UNIT NOT ACCESSIBLE, TARGET PORT IN STANDBY STATE.

The SCSI target device shall participate in all task management functions as defined in SAM-3 and modified by the applicable SCSI transport protocol standards (see 3.1.102).

5.8.2.4.5 Unavailable state

When being accessed through a target port in the unavailable target port asymmetric access state, the device server shall accept only a limited set of commands. The unavailable target port asymmetric access state is intended for situations when the target port accessibility to a logical unit may be severely restricted due to SCSI target device limitations (e.g., hardware errors). Therefore it may not be possible to transition from this state to either the active/optimized, active/non-optimized or standby states. The unavailable target port asymmetric access state is also intended for minimizing any disruption when using the downloading microcode mode of the WRITE BUFFER command.

While in the unavailable target port asymmetric access state, the device server shall support those of the following commands that it supports while in the active/optimized state:

a) INQUIRY (the peripheral qualifier (see 6.4.2) shall be set to 001b);
b) REPORT LUNS (for LUN 0);
c) REPORT TARGET PORT GROUPS;
d) SET TARGET PORT GROUPS;
e) REQUEST SENSE;
f) Echo buffer modes of READ BUFFER;
g) Echo buffer modes of WRITE BUFFER; and
h) Download microcode mode of WRITE BUFFER.

The device server may support other commands.

For those commands that are not supported, the device server shall terminate the command with CHECK CONDITION status, with the sense key set to NOT READY, and the additional sense code set to LOGICAL UNIT NOT ACCESSIBLE, TARGET PORT IN UNAVAILABLE STATE.

The SCSI target device is not required to participate in all task management functions (see SAM-3 and the applicable SCSI transport protocol standards).

5.8.2.5 Transitions between target port asymmetric access states

The movement from one target port asymmetric access state to another is called a transition.

During a transition between target port asymmetric access states the device server shall respond to a command in one of the following ways:

a) If during the transition the logical unit is inaccessible, then the transition is performed as a single indivisible event and the device server shall respond by either returning BUSY status, or returning CHECK CONDITION status, with the sense key set to NOT READY, and an the sense code set to LOGICAL UNIT NOT ACCESSIBLE, ASYMMETRIC ACCESS STATE TRANSITION; or
b) If during the transition the target ports in a target port group are able to access the requested logical unit, then the device server shall support those of the following commands that it supports while in the active/optimized asymmetric access state:
   A) INQUIRY;
   B) REPORT LUNS (for LUN 0);
   C) REPORT TARGET PORT GROUPS;
   D) REQUEST SENSE;
   E) Echo Buffer modes of READ BUFFER; and
   F) Echo Buffer modes of WRITE BUFFER.

   The device server may support other commands when those commands are routed though a target port that is transitioning between asymmetric access states.

   For those commands that are not supported during a transition, the device server shall terminate the command with CHECK CONDITION status, with the sense key set to NOT READY, and the additional sense code set to LOGICAL UNIT NOT ACCESSIBLE, ASYMMETRIC ACCESS STATE TRANSITION.

   The SCSI target device is not required to participate in all task management functions.

If the transition was explicit to a supported asymmetric access state and it failed, then the command shall be terminated with CHECK CONDITION status, with the sense key set to HARDWARE ERROR, and the additional sense code set to SET TARGET PORT GROUPS COMMAND FAILED. The target port group that encountered the error should complete a transition to the unavailable target port asymmetric access state. If a target port group asymmetric access state change occurred as a result of the failed transition, then the device server shall establish a unit attention condition for the initiator port associated with every I_T nexus other than the I_T nexus on which the SET TARGET PORT GROUPS command was received with the additional sense code set to ASYMMETRIC ACCESS STATE CHANGED.

If the transition was implicit and it failed, then the device server shall establish a unit attention condition for the initiator port associated with every I_T nexus with the additional sense code set to IMPLICIT ASYMMETRIC ACCESS STATE TRANSITION FAILED.

An implicit CLEAR TASK SET task management function may be performed following a transition failure.

Once a transition is completed, the new target port asymmetric access state may apply to some or all tasks entered into the task set before the completion of the transition. The new target port asymmetric access state shall apply to all tasks received by the device server after completion of a transition.

After an implicit target port asymmetric access state change, a device server shall establish a unit attention condition for the initiator port associated with every I_T nexus with the additional sense code set to ASYMMETRIC ACCESS STATE CHANGED.

After an explicit target port asymmetric access state change, a device server shall establish a unit attention condition with the additional sense code set to ASYMMETRIC ACCESS STATE CHANGED for the initiator port associated with every I_T nexus other than the I_T nexus on which the SET TARGET GROUPS command was received.

5.8.2.6 Preference Indicator

A device server may indicate one or more target port groups is a preferred target port group for accessing a logical unit by setting the PREF bit to one in the target port group descriptor (see 6.25). The preference indication is independent of the asymmetric access state.

An application client may use the PREF bit value in the target port group descriptor to influence the path selected to a logical unit (e.g., a target port group in the standby target port asymmetric access state with the PREF bit set to one may be chosen over a target port group in the active/optimized target port asymmetric access state with the PREF bit set to zero).

The value of the PREF bit for a target port group may change whenever an asymmetric access state changes.

5.8.2.7 Implicit asymmetric logical units access management

SCSI target devices with implicit asymmetric logical units access management are capable of setting the target port group asymmetric access state of each target port group using mechanisms other than the SET TARGET PORT GROUPS command.

All logical units that report in the standard INQUIRY data (see 6.4.2) that they support asymmetric logical units access and support implicit asymmetric logical unit access (i.e., the TPGS field contains 01b or 11b) shall:

a) Implement the INQUIRY command Device Identification VPD page identifier types 4h (see 7.6.3.7) and 5h (see 7.6.3.8); and
b) Support the REPORT TARGET PORT GROUPS command as described in 6.25.

Implicit logical unit access state changes may be disabled with the IALUAE bit in the Control Extension mode page (see 7.4.7).

5.8.2.8 Explicit asymmetric logical units access management

All logical units that report in the standard INQUIRY data (see 6.4.2) that they support asymmetric logical units access and support explicit asymmetric logical unit access (i.e., the TPGS field contains 10b or 11b) shall:

a) Implement the INQUIRY command Device Identification VPD page (see 7.6.3) identifier types 4h and 5h;
b) Support the REPORT TARGET PORT GROUPS command as described in 6.25; and
c) Support the SET TARGET PORT GROUPS command as described in 6.31.

5.8.2.9 Behavior after power on, hard reset, logical unit reset, and I_T nexus loss

For all SCSI target devices that report in the standard INQUIRY data (see 6.4.2) that they support only explicit asymmetric logical unit access (i.e., the TPGS field contains 10b), the target port shall preserve the target port asymmetric access state during any power cycle, hard reset, logical unit reset, and I_T nexus loss.

5.8.3 Symmetric logical unit access

A device server that provides symmetrical access to a logical unit may use a subset of the asymmetrical logical access features (see 5.8.2) to indicate this ability to an application client, providing an application client a common set of commands to determine how to manage target port access to a logical unit.

Symmetrical logical unit access should be represented as follows:

a) The TPGS field in the standard INQUIRY data (see 6.4.2) indicates that implicit asymmetric access is supported;
b) The REPORT TARGET PORT GROUPS command is supported; and
c) The REPORT TARGET PORT GROUPS parameter data indicates that the same state (e.g., active/optimized state) is in effect for all target port groups.

5.9 Power conditions

5.9.1 Power conditions overview

The optional Power Condition mode page (see 7.4.12) allows an application client to control the power condition of a logical unit in a manner that may reduce power consumption of the SCSI target device. This control is invoked by enabling and setting the idle condition timer and/or the standby condition timer using the mode page. A change in the power condition of any logical unit in a SCSI target device may result in a change in the SCSI target device's power consumption.

In addition to the Power Condition mode page, the power condition of a logical unit may be controlled by the START STOP UNIT command (see SBC-2 or RBC). If both the Power Condition mode page and the START STOP UNIT command methods are being used to control the power condition of the same logical unit, then any START STOP UNIT command's power condition specification shall override the Power Condition mode page's power control and may disable the idle condition and standby condition timers.

There shall be no notification to the application client that a logical unit has transitioned from one power condition to another. An application client may determine the current power condition of a logical unit by issuing a REQUEST SENSE command (see 6.27).

No power condition shall affect the supply of any power required for proper operation of the service delivery subsystem.

Logical units that contain cache memory shall write all cached data to the medium for the logical unit (e.g., as a logical unit would do in response to a SYNCHRONIZE CACHE command as described in SBC-2) prior to entering into any power condition that prevents accessing the media (e.g., before a hard drive stops its spindle motor during transition to the standby power condition).

The power conditions are described in table 39.

Table 39 — Power Conditions

Power Condition	Description
active	While in the active power condition (see 3.1.5): a) A device server is capable of responding to all of its supported commands including media access requests; b) A logical unit completes processing of operations in the shortest time when compared to the time required for completion while in the idle or standby power conditions; and c) The SCSI target device may consume more power than when the logical unit is in the idle power condition (e.g., a disk drive's spindle motor may be active).
idle	While in the idle power condition (see 3.1.47): a) A device server is capable of responding to all of its supported commands including media access requests; b) A logical unit may take longer to complete processing a command than it would while in the active power condition (e.g., the device may have to activate some circuitry before processing a command); and c) The power consumed by the SCSI target device should be less than or equal to the power consumed when the logical unit is in the active power condition and may be greater than the power consumed when the logical unit is in the standby power condition.
standby	While in the standby power condition (see 3.1.107): a) A device server is not capable of processing media access commands; and b) The power consumed by the SCSI target device should be less than or equal to the power consumed when the logical unit is in the idle power condition (e.g., a disk drive's spindle motor is stopped).

5.9.2 Power condition state machine

5.9.2.1 Power condition state machine overview

The PC (power condition) state machine describes the logical unit power states and transitions resulting from Power Condition mode page settings.

The PC states are as follows:

a) PC0:Powered_on (see 5.9.2.2) (initial state);
b) PC1:Active (see 5.9.2.3);
c) PC2:Idle (see 5.9.2.4); and
d) PC3:Standby (see 5.9.2.5).

The PC state machine stall start in the PC0:Powered_on state after power on.

Figure 5 describes the PC state machine.

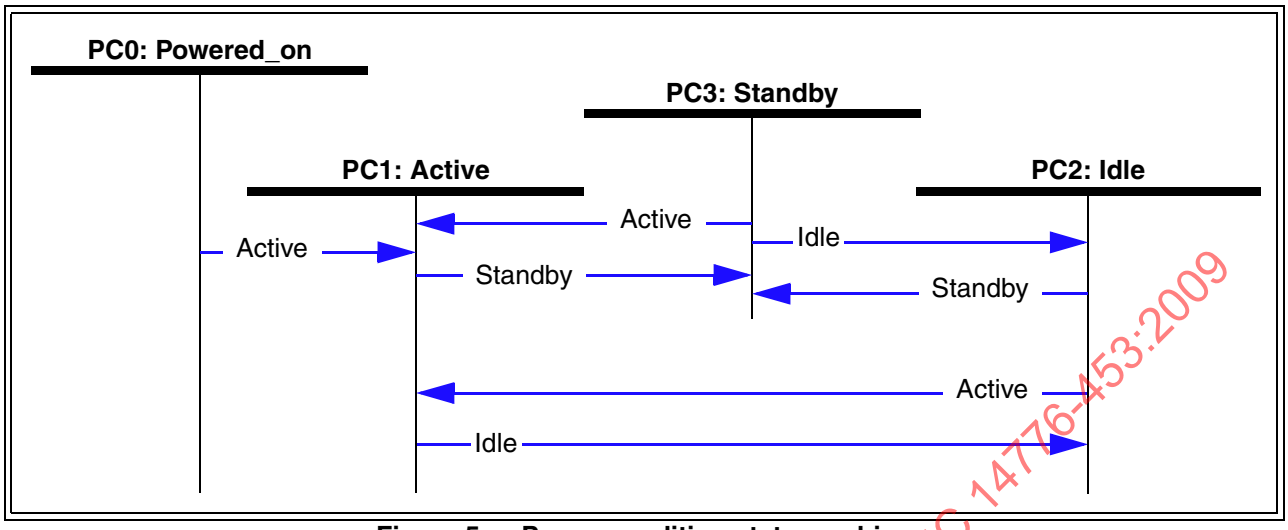

Figure 5 — Power condition state machine

5.9.2.2 PC0:Powered_on state

5.9.2.2.1 PC0:Powered_on state description

The logical unit shall enter this state upon power on. This state consumes zero time.

5.9.2.2.2 Transition PC0:Powered_on to PC1:Active

This transition shall occur after the logical unit is ready to begin its power on initialization.

5.9.2.3 PC1:Active state

5.9.2.3.1 PC1:Active state description

While in this state, if power on initialization is not complete, then the logical unit shall complete its power on initialization.

While in this state, if power on initialization is complete, then:

a) A logical unit is in the active power condition (see table 39);
b) If the idle condition timer is active, then the idle condition timer is running; and
c) If the standby condition timer is active, then the standby condition timer is running.

5.9.2.3.2 Transition PC1:Active to PC2:Idle

This transition shall occur after:

a) The idle condition timer is active; and
b) The idle condition timer is zero.

5.9.2.3.3 Transition PC1:Active to PC3:Standby

This transition shall occur after:

a) The standby condition timer is active; and
b) The standby condition timer is zero.

5.9.2.4 PC2:Idle state

5.9.2.4.1 PC2:Idle state description

While in this state:

a) A logical unit is in the idle power condition (see table 39); and
b) If the standby condition timer is active, then the standby condition timer is running.

5.9.2.4.2 Transition PC2:Idle to PC1:Active

This transition shall occur after the device server receives a command that requires the logical unit to be in the PC1:Active state to process the command.

5.9.2.4.3 Transition PC2:Idle to PC3:Standby

This transition shall occur after:

a) The standby condition timer is active; and
b) The standby condition timer is zero.

5.9.2.5 PC3:Standby state

5.9.2.5.1 PC3:Standby state description

While in this state, a logical unit is in the standby power condition (see table 39).

5.9.2.5.2 Transition PC3:Standby to PC1:Active

This transition shall occur after the device server receives a command that requires the logical unit to be in the PC1:Active state to process the command.

5.9.2.5.3 PC3:Standby to PC2:Idle

This transition shall occur after the device server receives a command that requires the logical unit to be in the PC2:Idle state to process the command.

5.10 Removable medium devices with an attached medium changer

When a logical unit is served by a medium changer, control over one medium transport element may be effected using medium changer commands sent to the device server within the logical unit. The level of control is not as complete as would be available if a fully functional medium-changer device server were implemented (see SMC-2). However, the amount of control is sufficient for paired device and medium changer configurations.

The device server shall indicate its ability to support medium changer commands by setting the MCHNGR bit to one in its standard INQUIRY data (see 6.4.2). An MCHNGR bit set to one shall indicate that the MOVE MEDIUM ATTACHED and READ ELEMENT STATUS ATTACHED commands are supported by the device server. Definitions of the MOVE MEDIUM ATTACHED and READ ELEMENT STATUS ATTACHED commands may be found in SMC-2.

5.11 Medium auxiliary memory

Some types of media, especially removable media, include a non-volatile memory referred to as MAM (medium auxiliary memory). Medium auxiliary memory is used to store data describing the media and its contents. This

standard supports medium auxiliary memory with the READ ATTRIBUTE command (see 6.14) and the WRITE ATTRIBUTE command (see 6.34). These commands are used to retrieve and store information in the medium auxiliary memory in the form of attributes.

A MAM attribute is represented in a format described in 7.3 and is composed of:

a) An attribute identifier;
b) An attribute format code;
c) A bit indicating whether the identified attribute is read only;
d) An attribute length specifying the number of bytes in the identified attribute value; and
e) The value of the identified attribute.

There are three types of attributes (see table 40).

Table 40 — Types of MAM attributes

Attribute Type	Attribute Source	Example	Readable with READ ATTRIBUTE	Writable with WRITE ATTRIBUTE
Medium	Permanently stored in the medium auxiliary memory during manufacture.	Media Serial Number	Yes	No
Device	Maintained by the device server.	Load Count	Yes	No
Host	Maintained by the application client.	Backup Date	Yes	Yes

Depending on that attribute type, attributes have the states shown in table 41.

Table 41 — MAM attribute states

Attribute Type	Attribute State	Description
Medium or Device	Read Only	An application client may read the contents of the attribute with the READ ATTRIBUTE command, but an attempt to clear or change the attribute using the WRITE ATTRIBUTE command shall result in the command being terminated with CHECK CONDITION status. When the READ ONLY bit (see 7.3.1) is one, the attribute is in the read only state.
	Unsupported	The device server does not support the attribute and shall not return it in response to a READ ATTRIBUTE command.
Host	Nonexistent	A host attribute does not exist in the medium auxiliary memory until a WRITE ATTRIBUTE command creates it.
	Read/Write	The attribute has been created using the WRITE ATTRIBUTE command. After the attribute has been created, the contents may be altered using subsequent WRITE ATTRIBUTE commands. A read/write attribute may be returned to the nonexistent state using a WRITE ATTRIBUTE command with the attribute length set to zero. When the READ ONLY bit (see 7.3.1) is zero, the attribute is in the read/write state.

5.12 Application client logging

Application client logging is a method the application client may use to store application client detected error information in a logical unit's non-volatile storage (see 6.35.12). The information the application client sends to the logical unit is appended to an application error log. The application client error information is recovered by means outside the scope of this standard and is not used for any logical unit related error recovery.

A log that contains a mix of application client error information and logical unit error information may be used to correlate an application client error with any errors internal to the logical unit. This does not replace the vendor specific methods for collecting and analyzing engineering data, but provides a vendor independent way of correlating error logs.

Application clients should minimize the amount of error information that is requested to be logged to prevent log overflows.

5.13 Device clocks

A timestamp may be included in data logged or recorded by a device server. There shall be one timestamp per logical unit.

The timestamp origin shall be one of those specified in table 42.

Table 42 — TIMESTAMP ORIGIN field

Code	Description
000b	Timestamp initialized to zero at power-on or as the result of a hard reset
001b	Reserved
010b	Timestamp initialized by the SET TIMESTAMP command (see 6.32)
011b	Timestamp initialized by methods outside the scope of this standard
100b - 111b	Reserved

The Timestamp shall not be affected by an I_T nexus loss or a logical unit reset.

Once a timestamp is initialized it shall begin counting from that time forward. Once the timestamp is initialized it shall remain in effect until one of the following occurs:

a) A hard reset occurs;
b) A SET TIMESTAMP command is processed; or
c) A method outside the scope of this standard affects the timestamp.

The methods by which a timestamp may be changed are indicated in the Control Extension mode page (see 7.4.7).

If the timestamp is changed by means other than the SET TIMESTAMP command then the device server shall generate a unit attention condition for the initiator port associated with every I_T nexus (see SAM-3), with the additional sense code set to TIMESTAMP CHANGED.

The TIMESTAMP field format is shown in table 43.

Table 43 — TIMESTAMP field format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
				TIMESTAMP				
5								(LSB)

The TIMESTAMP field contains the value established at the last action that set the timestamp incremented by one for every millisecond that has elapsed since the timestamp was set, within vendor-specific constraints.

6 Commands for all device types

6.1 Summary of commands for all device types

The operation codes for commands that apply to all device types when the MCHNGR bit is set to zero, the SSCS bit is set to zero, and the ENCSERV bit is set to zero in the standard INQUIRY data (see 6.4.2) are listed in table 44.

Table 44 — Commands for all device types (part 1 of 2)

Command name	Operation code	Type	Reference
ACCESS CONTROL IN	86h	O	8.3.2
ACCESS CONTROL OUT	87h	O	8.3.3
CHANGE ALIASES	A4h/0Bh [a]	O	6.2
EXTENDED COPY	83h	O	6.3
INQUIRY	12h	M	6.4
LOG SELECT	4Ch	O	6.5
LOG SENSE	4Dh	O	6.6
MODE SELECT(6)	15h	C	6.7
MODE SELECT(10)	55h	C	6.8
MODE SENSE(6)	1Ah	C	6.9
MODE SENSE(10)	5Ah	C	6.10
PERSISTENT RESERVE IN	5Eh	C	6.11
PERSISTENT RESERVE OUT	5Fh	C	6.12
PREVENT ALLOW MEDIUM REMOVAL	1Eh	C	6.13
READ ATTRIBUTE	8Ch	O	6.14
READ BUFFER	3Ch	O	6.15
READ MEDIA SERIAL NUMBER	ABh/01h [a]	C	6.16
RECEIVE COPY RESULTS	84h	O	6.17
RECEIVE DIAGNOSTIC RESULTS	1Ch	O	6.18
REPORT ALIASES	A3h/0Bh [a]	O	6.19
REPORT DEVICE IDENTIFIER	A3h/05h [a]	O	6.20
REPORT LUNS	A0h	M	6.21
REPORT PRIORITY	A3h/0Eh [a]	O	6.22
REPORT SUPPORTED OPERATION CODES	A3h/0Ch [a]	O	6.23
REPORT SUPPORTED TASK MANAGEMENT FUNCTIONS	A3h/0Dh [a]	O	6.24
REPORT TARGET PORT GROUPS	A3h/0Ah [a]	O	6.25
REPORT TIMESTAMP	A3h/0Fh [a]	O	6.26
REQUEST SENSE	03h	C	6.27
SEND DIAGNOSTIC	1Dh	C	6.28
SET DEVICE IDENTIFIER	A4h/06h [a]	O	6.29
SET PRIORITY	A4h/0Eh [a]	O	6.30

Type Key: C = Command implementation is defined in the applicable command standard (see 3.1.18).
M = Command implementation is mandatory.
O = Command implementation is optional.
Z = Command implementation is defined in a previous standard.

[a] This command is defined by a combination of operation code and service action. The operation code value is shown preceding the slash and the service action value is shown after the slash.

Table 44 — Commands for all device types (part 2 of 2)

Command name	Operation code	Type	Reference
SET TARGET PORT GROUPS	A4h/0Ah [a]	O	6.31
SET TIMESTAMP	A4h/0Fh [a]	O	6.32
TEST UNIT READY	00h	M	6.33
WRITE ATTRIBUTE	8Dh	O	6.34
WRITE BUFFER	3Bh	C	6.35
Obsolete	16h	Z	
Obsolete	17h	Z	
Obsolete	18h	Z	
Obsolete	39h	Z	
Obsolete	3Ah	Z	
Obsolete	40h	Z	
Obsolete	56h	Z	
Obsolete	57h	Z	

Type Key: C = Command implementation is defined in the applicable command standard (see 3.1.18).
M = Command implementation is mandatory.
O = Command implementation is optional.
Z = Command implementation is defined in a previous standard.

[a] This command is defined by a combination of operation code and service action. The operation code value is shown preceding the slash and the service action value is shown after the slash.

6.2 CHANGE ALIASES command

6.2.1 CHANGE ALIASES command introduction

The CHANGE ALIASES command (see table 45) requests that the device server maintain and make changes to a list of associations between eight byte alias values and SCSI target device or SCSI target port designations. A designation contains a name and optional identifier information that specifies a SCSI target device or SCSI target port (see 6.2.2). The alias list may be queried by the application client via the REPORT ALIASES command (see 6.19). If the REPORT ALIASES command is supported, the CHANGE ALIASES command shall also be supported.

The CHANGE ALIASES command is a service action of the MAINTENANCE OUT command. Additional MAINTENANCE OUT service actions are defined in SCC-2 and in this standard. The MAINTENANCE OUT service actions defined in SCC-2 apply only to logical units that return a device type of 0Ch or the SCCS bit set to one in their standard INQUIRY data (see 6.4.2).

Table 45 — CHANGE ALIASES command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (A4h)							
1	Reserved			SERVICE ACTION (0Bh)				
2	Reserved							
5								
6	(MSB)			PARAMETER LIST LENGTH				
9								(LSB)
10	Reserved							
11	CONTROL							

The PARAMETER LIST LENGTH field specifies the length in bytes of the parameter data that shall be transferred from the application client to the device server. A parameter list length value of zero specifies that no data shall be transferred and no changes shall be made in the alias list.

If the parameter list length results in the truncation of the header or any alias entry, then the device server shall make no changes to the alias list and terminate the command with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to PARAMETER LIST LENGTH ERROR.

On successful completion of a CHANGE ALIASES command, the device server shall maintain an association of each assigned eight byte alias value to the SCSI target device or SCSI target port designation. These associations shall be cleared by a logical unit reset or I_T nexus loss. The device server shall maintain a separate alias list for each I_T nexus.

A CHANGE ALIASES command may add, change or remove entries from the alias list. Alias list entries not referenced in the CHANGE ALIASES parameter data shall not be changed.

NOTE 11 - An application client may use alias values to reference SCSI target devices or SCSI target ports in third party commands (e.g., EXTENDED COPY). The alias list provides a mechanism for eight byte third party identifier fields to reference a third party device or port whose name or addressing information is longer than eight bytes. (E.g., an application may use the CHANGE ALIASES command to establish an association between an alias value and a SCSI target device or target port designation. Then, it may send an EXTENDED COPY command containing in the parameter data an alias target descriptor (see 6.3.6.3) that includes this alias value. At the completion of the EXTENDED COPY command the application should clear this entry from the device server's alias list by sending a CHANGE ALIASES command that requests association of the alias value to a NULL DESIGNATION (see 6.2.4.2) alias format.)

If the device server has insufficient resources to make all requested changes to the alias list, then the device server shall make no changes to the alias list and shall terminate the command with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INSUFFICIENT RESOURCES.

The parameter data for a CHANGE ALIASES command (see table 46) contains zero or more alias entries. If the device server processes a CHANGE ALIASES command that contains at least one alias entry while there exists any other enabled task that references an alias entry in the alias list, then the device server shall terminate the CHANGE ALIASES command with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to OPERATION IN PROGRESS.

Table 46 — CHANGE ALIASES parameter list

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
3				PARAMETER DATA LENGTH (n-3)				(LSB)
4								
7				Reserved				
	Alias entry (or entries)							
8								
				Alias entry 0 (see 6.2.2)				
				. . .				
n				Alias entry x (see 6.2.2)				

The PARAMETER DATA LENGTH field should contain the number of bytes of attribute data and shall be ignored by the device server.

The format of an alias entry is described in 6.2.2.

6.2.2 Alias entry format

One alias entry (see table 47) describes one alias reported via the REPORT ALIASES command (see 6.19) or to be changed via the CHANGE ALIASES command.

Table 47 — Alias entry format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
7				ALIAS VALUE				(LSB)
8				PROTOCOL IDENTIFIER				
9								
10				Reserved				
11				FORMAT CODE				
12								
13				Reserved				
14	(MSB)							
15				DESIGNATION LENGTH (n-15)				(LSB)
16								
n				DESIGNATION				

The ALIAS VALUE field contains the numeric alias value that the device server shall associate with the SCSI target device or target port specified by the values in the PROTOCOL IDENTIFIER, FORMAT CODE and DESIGNATION fields.

The PROTOCOL IDENTIFIER field (see table 48) specifies that the alias entry designation is independent of SCSI transport protocol or the SCSI transport protocol to which the alias entry applies.

Table 48 — Alias entry protocol identifiers

PROTOCOL IDENTIFIER	Description	Subclause
00h - 0Fh	Protocol specific designation	7.5.2
10h - 7Fh	Reserved	
80h	Protocol independent designation	6.2.4
81h - FFh	Reserved	

The FORMAT CODE field contents combined with the PROTOCOL IDENTIFIER field contents defines the format of the DESIGNATION field. The subclauses that describe each PROTOCOL IDENTIFIER field usage (see table 48) define the applicable FORMAT CODE field values.

The DESIGNATION LENGTH field specifies the number of bytes of the DESIGNATION field. The DESIGNATION LENGTH value shall be a multiple of four.

The zero-padded (see 4.4.2) DESIGNATION field should designate a unique SCSI target device or target port using the following:

a) A SCSI target device name or a target port name, and
b) Optionally, one or more target port identifiers or SCSI transport protocol specific identifiers.

6.2.3 Alias designation validation

The device server shall not validate any designation at the time of processing either the REPORT ALIASES or CHANGE ALIASES command. Such validation shall occur only when the device server consults the alias list to resolve an alias to a designation in the context of third-party commands (e.g., EXTENDED COPY) or any other command that requires reference to the alias list.

If a designation identifies a unique SCSI target device or target port that is within a SCSI domain accessible to the device server, the designation is considered valid.

Based on the SCSI transport protocol specific requirements for a given designation format, a designation that does not identify a unique SCSI target device or target port within the SCSI domains accessible to the device server is considered invalid.

NOTE 12 - For example, a designation may be considered invalid if the device server has no ports on the SCSI domain of the designated SCSI target device or target port.

A designation having both SCSI name and SCSI port identifier information may be inconsistent if the device server is not able to access the named SCSI target device or target port through one or more of the names or identifiers in the designation. In such cases, the designation shall be treated as valid or invalid according to the SCSI transport protocol specific requirements.

NOTE 13 - For example, in FCP-2 both an N_Port and World Wide Name for a SCSI port may be given in a designation. The designation definition may require that the N_Port be that of the named port. In that case, the designation is invalid. Alternatively, the designation definition may view the N_Port as a hint for the named FC Port accessible to the device server through a different D_ID. In that case, the designation is valid and designate the named FC Port.

NOTE 14 - When only name information is provided in a designation, it is assumed that the device server has access to a mechanism for resolving names to identifiers. Access to such a service is SCSI transport protocol specific and vendor specific.

6.2.4 Alias entry protocol independent designations

6.2.4.1 Alias entry protocol independent designations overview

The protocol independent alias entry designations have a protocol identifier of 80h and one of the format codes shown in table 49.

Table 49 — Protocol independent alias entry format codes

Format Code	Name	Designation Length (bytes)	Designation Contents	Subclause
00h	NULL DESIGNATION	0	none	6.2.4.2
01h - FFh	Reserved			

6.2.4.2 NULL DESIGNATION alias format

In response to an alias entry with the NULL DESIGNATION format, the device server shall remove the specified alias value from the alias list. Application clients should use the NULL DESIGNATION format in a CHANGE ALIASES command to remove an alias entry from the alias list when that alias entry is no longer needed. The NULL DESIGNATION format shall not appear in REPORT ALIASES parameter data.

6.3 EXTENDED COPY command

6.3.1 EXTENDED COPY command introduction

The EXTENDED COPY command (see table 50) provides a means to copy data from one set of logical units to another set of logical units or to the same set of logical units. The entity within a SCSI device that receives and performs the EXTENDED COPY command is called the copy manager. The copy manager is responsible for copying data from the source devices to the destination devices. The copy source and destination devices are logical units that may reside in different SCSI devices or the same SCSI device. It is possible that the copy source device, copy destination device, and the copy manager are the same logical unit.

Table 50 — EXTENDED COPY command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (83h)							
1	Reserved							
9								
10	(MSB)							
13	PARAMETER LIST LENGTH							(LSB)
14	Reserved							
15	CONTROL							

Before the copy manager is instructed to move data, the application controlling the data movement shall independently take any necessary actions required to prepare the copy source and destination devices for the EXTENDED COPY command (e.g. loading tapes, sending media changer commands, MODE SELECT commands, reservation commands, and/or tape positioning commands). After all preparatory actions have been accomplished, the EXTENDED COPY command should be issued to the copy manager to start the data transfer.

The PARAMETER LIST LENGTH field specifies the length in bytes of the parameter data that shall be contained in the Data-Out Buffer. A parameter list length of zero specifies that copy manager shall not transfer any data or alter any internal state; this shall not be considered an error. If the parameter list length causes truncation of the parameter list in a target descriptor or segment descriptor, then no data shall be transferred and the EXTENDED COPY command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to PARAMETER LIST LENGTH ERROR.

The EXTENDED COPY parameter list (see table 51) begins with a 16 byte header that contains the LIST IDENTIFIER field, the STR bit, the NRCR bit, the PRIORITY field, the length of the target descriptor list, the length of the segment descriptor list, and the length of the optional inline data. Immediately following the header is one or more target descriptors, followed by one or more segment descriptors, followed by any optional inline data.

Table 51 — EXTENDED COPY parameter list

Bit / Byte	7	6	5	4	3	2	1	0
0	LIST IDENTIFIER							
1	Reserved		STR	NRCR	Reserved	PRIORITY		
2	(MSB)							
3	TARGET DESCRIPTOR LIST LENGTH (n-15)							(LSB)
4	Reserved							
7								
8	(MSB)							
11	SEGMENT DESCRIPTOR LIST LENGTH (m-n)							(LSB)
12	(MSB)							
15	INLINE DATA LENGTH (k-m)							(LSB)
	Target descriptor(s)							
16	Target descriptor 0							
47								
	.							
n-31	Target descriptor x							
n								
	Segment descriptor(s)							
n+1	Segment descriptor 0 (See specific table for length.)							
n+1+l								
	.							
	Segment descriptor y (See specific table for length.)							
m								
m+1	Inline data							
k								

NOTE 15 - Unexpected results may occur when an application client fails to zero the reserved bytes in this parameter list. Copy managers should ensure that the reserved bytes 4 through 7 contain zeros.

The LIST IDENTIFIER field is a value selected by the application client to uniquely identify the extended copy operation to the copy manager. The list identifier also may be used in the RECEIVE COPY RESULTS command (see 6.17) to request status for a specific EXTENDED COPY command. The LIST IDENTIFIER value shall be unique for each concurrent EXTENDED COPY command sent via one I_T nexus. If the copy manager detects a duplicate LIST IDENTIFIER value, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to OPERATION IN PROGRESS.

The PRIORITY field establishes the priority of data transfer operations resulting from this EXTENDED COPY command relative to data transfer operations resulting from other commands being processed by the same device server. All commands other than copy commands have a priority of 1h. Priority 0h is the highest priority, with increasing PRIORITY values indicating lower priorities.

A Sequential Striped (STR) bit set to one specifies to the copy manager that the majority of the disk references in the parameter list represent sequential access of several striped disks. This may be used by the copy manager to perform read operations from a copy source disk at any time and in any order during processing of an EXTENDED COPY command as described in 6.3.6.4. A STR bit set to zero specifies to the copy manager that disk references may not be sequential.

If the No Receive Copy Results (NRCR) bit is set to zero, the copy manager shall hold data for retrieval by the application client using the RECEIVE COPY RESULTS command with the RECEIVE DATA service action (see 6.17.3) and specified by the segment descriptors. If NRCR is one, the copy manager may discard all data accessible to the application client via the RECEIVE COPY RESULTS command with the RECEIVE DATA service action. If the application client requests delivery of data that has been discarded as a result of NRCR being one, the copy manager shall respond as if the EXTENDED COPY command has not been processed.

The TARGET DESCRIPTOR LIST LENGTH contains the length in bytes of the target descriptor list that immediately follows the parameter list header. The number of target descriptors equals the length in bytes of the target descriptor list divided by 32.

An EXTENDED COPY command may reference one or more copy target devices (i.e., the name given by the EXTENDED COPY command description to copy source and/or the destination devices). Each copy target device is described by a target descriptor. All target descriptors have their formats specified by an EXTENDED COPY descriptor code. A copy manager may not support all target descriptor formats, however, the copy manager shall list all target descriptor formats supported in response to the RECEIVE COPY RESULTS command with OPERATING PARAMETERS service action (see 6.17.4). See 6.3.6 for a detailed description of the target descriptors.

Segment descriptors reference target descriptors by their position, or index, in the target descriptor list. The index for a target descriptor is computed by subtracting 16 from the starting byte number for the target descriptor in the parameter data and dividing the result by 32. The maximum number of target descriptors permitted within a parameter list is indicated by the MAXIMUM TARGET COUNT field in the copy manager's operating parameters (see 6.17.4). If the number of target descriptors exceeds the allowed number, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to TOO MANY TARGET DESCRIPTORS.

The SEGMENT DESCRIPTOR LIST LENGTH contains the length in bytes of the segment descriptor list that follows the target descriptors. See 6.3.7 for a detailed description of the segment descriptors. The maximum number of segment descriptors permitted within a parameter list is indicated by the MAXIMUM SEGMENT COUNT field in the copy manager's operating parameters (see 6.17.4). If the number of segment descriptors exceeds the allowed number, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to TOO MANY SEGMENT DESCRIPTORS.

The maximum length of the target and segment descriptors permitted within a parameter list is indicated by the MAXIMUM DESCRIPTOR LIST LENGTH field in the copy manager's operating parameters (see 6.17.4). If the combined length of the target and segment descriptors exceeds the allowed value, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to PARAMETER LIST LENGTH ERROR.

The INLINE DATA LENGTH field contains the number of bytes of inline data, after the last segment descriptor. A value of zero specifies that no inline data is present.

The copy manager shall move data from the copy source devices to the copy destination devices as specified by the segment descriptors. The specific commands issued by the copy manager to the copy source and destination devices while processing the segment descriptors is vendor specific. Upon completion of an EXTENDED COPY command that returns GOOD status, the copy source and destination devices, particularly stream devices, shall be

positioned at deterministic locations such that the device may be repositioned to the same location by the application client with appropriate commands.

6.3.2 Errors detected before starting processing of the segment descriptors

Errors may occur during processing of an EXTENDED COPY command before the first segment descriptor is processed. These errors include CRC or parity errors while transferring the EXTENDED COPY command, invalid parameters in the CDB or parameter data, invalid segment descriptors, and inability of the copy manager to continue operating. In the event of such an exception condition, the copy manager shall:

a) Terminate the EXTENDED COPY command with CHECK CONDITION status; and
b) Set the VALID bit in the sense data to zero. The sense key shall contain the sense key code describing the exception condition (i.e., not COPY ABORTED).

6.3.3 Errors detected during processing of segment descriptors

Errors may occur after the copy manager has begun processing segment descriptors. These errors include invalid parameters in segment descriptors, invalid segment descriptors, unavailable targets referenced by target descriptors, inability of the copy manager to continue operating, and errors reported by source or destination copy target devices. If the copy manager receives CHECK CONDITION status from one of the copy target devices, it shall recover the sense data associated with the exception condition and clear any ACA condition associated with the CHECK CONDITION status.

If it is not possible to complete processing of a segment because the copy manager is unable to establish communications with a copy target device, because the copy target device does not respond to INQUIRY, or because the data returned in response to INQUIRY indicates an unsupported logical unit, then the EXTENDED COPY command shall be terminated with CHECK CONDITION status, with the sense key set to COPY ABORTED, and the additional sense code set to COPY TARGET DEVICE NOT REACHABLE.

If it is not possible to complete processing of a segment because the data returned in response to an INQUIRY command indicates a device type that does not match the type in the target descriptor, then the EXTENDED COPY command shall be terminated with CHECK CONDITION status, with the sense key set to COPY ABORTED, and the additional sense code set to INCORRECT COPY TARGET DEVICE TYPE.

If the copy manager has issued a command other than INQUIRY to a copy target device while processing an EXTENDED COPY command and the copy target device either fails to respond with status or responds with status other than BUSY, TASK SET FULL, ACA ACTIVE, or RESERVATION CONFLICT, then the condition shall be considered a copy target device command failure. In response to a copy target device command failure the EXTENDED COPY command shall be terminated with CHECK CONDITION status, with the sense key set to COPY ABORTED, and the additional sense code set to THIRD PARTY DEVICE FAILURE.

If a copy target device responds to a command from the copy manager with a status of BUSY, TASK SET FULL, ACA ACTIVE, or RESERVATION CONFLICT, the copy manager shall either retry the command or terminate the EXTENDED COPY command as a copy target device command failure.

NOTE 16 - The copy manager is assumed to employ a vendor specific retry policy that minimizes time consuming and/or fruitless repetition of retries.

NOTE 17 - RESERVATION CONFLICT status is listed only to give the copy manager leeway in multi-port cases. The copy manager may have multiple initiator ports that are capable of reaching a copy target device, and a persistent reservation may restrict access to a single I_T nexus. The copy manager may need to try access from multiple initiator ports to find the correct I_T nexus.

If a copy target device responds to an input or output operation with a GOOD status but less data than expected is transferred, then the EXTENDED COPY command shall be terminated with CHECK CONDITION status, with the sense key set to COPY ABORTED, and the additional sense code set to COPY TARGET DEVICE DATA UNDERRUN. If an overrun is detected, then the EXTENDED COPY command shall be terminated with CHECK

CONDITION status, with the sense key set to COPY ABORTED, and the additional sense code set to COPY TARGET DEVICE DATA OVERRUN.

Following an exception condition detected during segment descriptor processing:

a) The copy manager shall terminate the EXTENDED COPY command with CHECK CONDITION status, with the sense key set to COPY ABORTED;
b) The copy manager shall indicate the segment that was being processed at the time of the exception by writing the segment number to the third and forth bytes of the COMMAND-SPECIFIC INFORMATION field. The segment number is based on the relative position of the segment descriptor in the EXTENDED COPY parameter list (i.e., the first segment descriptor in the parameter list is assigned descriptor number zero, the second is assigned one, etc.);
c) If any data has been written to the destination for the segment being processed at the time the error occurred, the residual for the segment shall be placed in the INFORMATION field, and the VALID bit shall be set to one. The residual count shall be reported in bytes if the peripheral device type in the destination target descriptor is 03h (i.e., processor device), and in destination device blocks for all other device type codes. The residual count shall be computed by subtracting the number of bytes or blocks successfully written during the processing of the current segment from the number of bytes or blocks which would have been written if all commands had completed with GOOD status and all READ commands had returned the full data length requested. When computing the residual count, the copy manager shall include only the results of commands successfully completed by a copy destination device (i.e., commands completed by a copy destination device with GOOD status or with CHECK CONDITION status and the EOM bit set to one in the sense data). If the copy manager has used out of order transfers, the residual count shall be based only on the contiguous successfully completed transfers starting at relative byte zero of the segment (i.e., any successfully completed transfers farther from relative byte zero than the first incomplete or unsuccessful transfer shall not contribute to the computation of the residual count). If no data has been written to the destination for the segment being processed at the time the error occurred, then the VALID bit shall be set to zero and the contents of the INFORMATION field are not defined. Segment descriptors that do not specify a transfer count shall not have a valid residual count returned;
d) If the exception condition is reported by the copy source device and fixed format sense data (see 4.5.3) is being returned, then the first byte of the COMMAND-SPECIFIC INFORMATION field shall be set to the starting byte number, relative to the first byte of sense data, of an area that contains the status byte and sense data delivered to the copy manager by the copy source device. The status byte and sense data shall not be modified by the copy manager or device server. A zero value indicates that no status byte and sense data is being returned for the copy source device;
e) If the exception condition is reported by the copy destination device and fixed format sense data is being returned, then the second byte of the COMMAND-SPECIFIC INFORMATION field shall be set to the starting byte number, relative to the first byte of sense data, of an area that contains the status byte and sense data delivered to the copy manager by the copy destination device. The status byte and sense data shall not be modified by the copy manager or device server. A zero value indicates that no status byte and sense data is being returned for the copy destination device;
f) If segment processing is terminated because a copy target device is unreachable or as the result of a failure in a command sent to a copy target device, then the SENSE-KEY SPECIFIC field shall be set as described in 4.5.2.4.5, with the FIELD POINTER field indicating the first byte of the target descriptor that identifies the copy target device;
g) If, during the processing of a segment descriptor, the copy manager detects an error in the segment descriptor, then the SENSE-KEY SPECIFIC field shall be set as described in 4.5.2.4.5, with the FIELD POINTER field indicating the byte in error. The FIELD POINTER field may be used to indicate an offset into either the parameter data or the segment descriptor. The SD bit is used to differentiate between these two cases. The SD bit shall be set to zero to indicate the FIELD POINTER field contains an offset from the start of the parameter data. The SD bit shall be set to one to indicate the FIELD POINTER field contains an offset from the start of the segment descriptor; and
h) The copy manager shall preserve information for the FAILED SEGMENT DETAILS service action of the RECEIVE COPY RESULTS command (see 6.17.5). The information shall be discarded as described in 6.17.5.

6.3.4 Abort task management functions

When a device server processes an ABORT TASK, ABORT TASK SET, or CLEAR TASK SET task management function that terminates an EXTENDED COPY command, the copy manager shall ensure that all commands and data transfers generated by the terminated EXENDED COPY command have been terminated and are no longer transferring data before allowing the task manager to complete the task management function. This requirement shall also apply to the processing the PREEMPT AND ABORT service action of the PERSISTENT RESERVE OUT command as described in 5.6.10.5.

6.3.5 Descriptor type codes

Target descriptors and segment descriptors share a single set of code values that identify the type of descriptor (see table 52). Segment descriptors use codes in the range 00h to BFh. The definitions of codes between C0h and DFh are vendor specific. Target descriptors use codes in the range E0h to FFh.

Table 52 — EXTENDED COPY descriptor type codes (part 1 of 2)

Descriptor type code	Reference	Description [a]	Shorthand [a]
00h	6.3.7.3	Copy from block device to stream device	block→ stream
01h	6.3.7.4	Copy from stream device to block device	stream→ block
02h	6.3.7.5	Copy from block device to block device	block→ block
03h	6.3.7.6	Copy from stream device to stream device	stream→ stream
04h	6.3.7.7	Copy inline data to stream device	inline→ stream
05h	6.3.7.8	Copy embedded data to stream device	embedded→ stream
06h	6.3.7.9	Read from stream device and discard	stream→ discard
07h	6.3.7.10	Verify block or stream device operation	
08h	6.3.7.11	Copy block device with offset to stream device	block<o>→ stream
09h	6.3.7.12	Copy stream device to block device with offset	stream→ block<o>
0Ah	6.3.7.13	Copy block device with offset to block device with offset	block<o>→ block<o>
0Bh	6.3.7.3	Copy from block device to stream device and hold a copy of processed data for the application client [b]	block→ stream +application client
0Ch	6.3.7.4	Copy from stream device to block device and hold a copy of processed data for the application client [b]	stream→ block +application client
0Dh	6.3.7.5	Copy from block device to block device and hold a copy of processed data for the application client [b]	block→ block +application client
0Eh	6.3.7.6	Copy from stream device to stream device and hold a copy of processed data for the application client [b]	stream→ stream +application client
0Fh	6.3.7.9	Read from stream device and hold a copy of processed data for the application client [b]	stream→ discard +application client

[a] Block devices are those with peripheral device type codes 0h (i.e, direct access block), 5h (i.e., CD/DVD), and Eh (i.e., simplified direct-access). Stream devices are those devices with peripheral device type codes 1h (i.e., sequential-access) and 3h (i.e., processor). Sequential-access stream (indicated by the term tape in the shorthand column) devices are those with peripheral device type code 1h. See 6.4.2 for peripheral device type code definitions.

[b] The application client shall use the RECEIVE COPY RESULTS with a RECEIVE DATA service action to retrieve data held for it by the copy manager (see 6.17.3).

Table 52 — EXTENDED COPY descriptor type codes (part 2 of 2)

Descriptor type code	Reference	Description [a]	Shorthand [a]
10h	6.3.7.14	Write filemarks to sequential-access device	filemark→ tape
11h	6.3.7.15	Space records or filemarks on sequential-access device	space→ tape
12h	6.3.7.16	Locate on sequential-access device	locate→ tape
13h	6.3.7.17	Image copy from sequential-access device to sequential-access device	<i>tape→ <i>tape
14h	6.3.7.18	Register persistent reservation key	
15h	6.3.7.19	Third party persistent reservations source I_T nexus	
16h - BFh		Reserved for segment descriptors	
C0h - DFh		Vendor specific descriptors	
E0h	7.5.3.2	Fibre Channel N_Port_Name target descriptor	
E1h	7.5.3.3	Fibre Channel N_Port_ID target descriptor	
E2h	7.5.3.4	Fibre Channel N_Port_ID with N_Port_Name checking target descriptor	
E3h	7.5.3.5	Parallel Interface T_L target descriptor	
E4h	6.3.6.2	Identification descriptor target descriptor	
E5h	7.5.3.8	IPv4 target descriptor	
E6h	6.3.6.3	Alias target descriptor	
E7h	7.5.3.7	RDMA target descriptor	
E8h	7.5.3.6	IEEE 1394 EUI-64 target descriptor	
E9h	7.5.3.9	SAS Serial SCSI Protocol target descriptor	
EAh - FFh		Reserved for target descriptors	

[a] Block devices are those with peripheral device type codes 0h (i.e, direct access block), 5h (i.e., CD/DVD), and Eh (i.e., simplified direct-access). Stream devices are those devices with peripheral device type codes 1h (i.e., sequential-access) and 3h (i.e., processor). Sequential-access stream (indicated by the term tape in the shorthand column) devices are those with peripheral device type code 1h. See 6.4.2 for peripheral device type code definitions.

[b] The application client shall use the RECEIVE COPY RESULTS with a RECEIVE DATA service action to retrieve data held for it by the copy manager (see 6.17.3).

6.3.6 Target descriptors

6.3.6.1 Target descriptors introduction

All target descriptors are 32 bytes in length and begin with a four-byte header (see table 53) containing the DESCRIPTOR TYPE CODE field that identifies the format of the descriptor. The assigned descriptor type code values are shown in table 52. Support for each target descriptor format is optional. If a copy manager receives an unsupported descriptor type code in a target descriptor, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to UNSUPPORTED TARGET DESCRIPTOR TYPE CODE.

Table 53 — Target descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE CODE (E0h - FFh)							
1	LU ID TYPE		NUL	PERIPHERAL DEVICE TYPE				
2	(MSB)			RELATIVE INITIATOR PORT IDENTIFIER				
3								(LSB)
4	Target descriptor parameters							
27								
28	Device type specific parameters							
31								

The DESCRIPTOR TYPE CODE field is described in 6.3.5.

The LU ID TYPE field (see table 54) specifies the interpretation of the LU IDENTIFIER field in target descriptors that contain a LU IDENTIFIER field.

Table 54 — LU ID TYPE field

Code	LU IDENTIFIER field contents	Reference
00b	Logical Unit Number	SAM-3
01b	Proxy Token	8.3.1.6.2
10b - 11b	Reserved	

Support for LU ID type codes other than 00b is optional. If a copy manager receives an unsupported LU ID type code, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

If the LU ID TYPE field specifies that the LU IDENTIFIER field contains a logical unit number, then the LU IDENTIFIER field specifies the logical unit within the SCSI device specified by other fields in the target descriptor that shall be the source or destination for EXTENDED COPY operations.

If the LU ID TYPE field specifies that the LU IDENTIFIER field contains a proxy token (see 8.3.1.6.2), then the copy manager shall use the LU IDENTIFIER field contents to obtain proxy access rights to the logical unit associated with the proxy token. The logical unit number that represents the proxy access rights shall be the source or destination for EXTENDED COPY operations.

The copy manager should obtain a LUN value for addressing this logical unit by sending an ACCESS CONTROL OUT command with ASSIGN PROXY LUN service action (see 8.3.3.11) to the access controls coordinator of the SCSI target device that is identified by other fields in the target descriptor. The copy manager shall use a LUN assigned on the basis of a proxy token only for those commands that are necessary for the processing of the EXTENDED COPY command whose parameter data contains the proxy token. When the copy manager has

completed EXTENDED COPY commands involving a proxy token, the copy manager should release the LUN value using an ACCESS CONTROL OUT command with RELEASE PROXY LUN service action (see 8.3.3.12).

EXTENDED COPY access to proxy logical units is to be accomplished only via LU ID type 01b. If the copy manager receives a target descriptor containing LU ID type 00b and a logical unit number matching a LUN value that the copy manager has obtained using an ACCESS CONTROL OUT command with ASSIGN PROXY LUN service action, then the EXTENDED COPY command shall be terminated with CHECK CONDITION status, with the sense key set to COPY ABORTED, and the additional sense code set to COPY TARGET DEVICE NOT REACHABLE.

A null device (NUL) bit set to zero specifies that the target descriptor identifies a SCSI device that is expected to respond to an INQUIRY command and to which data movement commands may be sent. A NUL bit set to one specifies that the descriptor identifies a null device that is not expected to be the recipient of any SCSI commands. If NUL is one, bytes 4-27 of the target descriptor shall be ignored. If the processing required by a segment descriptor necessitates sending a command to a copy target device whose target descriptor has the NUL bit set to one, then the EXTENDED COPY command shall be terminated as if an unreachable copy target device had been encountered (see 6.3.3).

NOTE 18 - Target descriptors with the NUL bit set to one are useful for processing the residual data from previous segment descriptors without affecting any media. (E.g., a segment descriptor of type 06h (stream device to discard) with a byte count of zero, CAT equal to zero, and a null source target descriptor with PAD equal to one may be used to discard all residual data.)

The PERIPHERAL DEVICE TYPE field is described in 6.4.2. The value in the DESCRIPTOR TYPE CODE field determines the format of the target descriptor parameters that follow the four-byte header and precede the device type specific parameters. The values in the DESCRIPTOR TYPE CODE field are listed in table 52.

The value in the PERIPHERAL DEVICE TYPE field determines the format of the device type specific parameters that follow the target descriptor parameters. The device type specific parameters convey information specific to the type of device identified by the target descriptor.

Table 55 lists the peripheral device type code values having formats defined for the device type specific parameters in a target descriptor. Peripheral device types with code values not listed in table 55 are reserved in the EXTENDED COPY parameter list.

Table 55 — Device type specific parameters in target descriptors

Peripheral Device Type	Reference	Description	Shorthand
00h, 04h, 05h, 07h, and 0Eh	6.3.6.4	Block devices	Block
01h	6.3.6.5	Sequential-access devices	Stream or Tape
03h	6.3.6.6	Processor devices	Stream

The RELATIVE INITIATOR PORT IDENTIFIER field specifies the relative port identifier (see 3.1.88) of the initiator port within the SCSI device that the copy manager shall use to access the logical unit described by the target descriptor, if such access requires use of an initiator port (i.e., if the logical unit is in the same SCSI device as the copy manager, the RELATIVE INITIATOR PORT IDENTIFIER field is ignored). A RELATIVE INITIATOR PORT IDENTIFIER field set to zero specifies that the copy manager may use any initiator port or ports within the SCSI device.

The copy manager may, as part of processing a segment descriptor, verify the information in a target descriptor's device specific fields. However, when verifying the information, the copy manager shall not issue any commands that change the position of read/write media on the copy target device without returning the media to its original position. Any errors encountered while verifying the information shall be handled as described in 6.3.3.

6.3.6.2 Identification descriptor target descriptor format

The target descriptor format shown in table 56 instructs the copy manager to locate a SCSI target device and logical unit that returns a Device Identification VPD page (see 7.6.3) containing an Identification descriptor having the specified CODE SET, ASSOCIATION, IDENTIFIER TYPE, IDENTIFIER LENGTH, and IDENTIFIER field values. The copy manager may use any N_Port, target port identifier and logical unit number values that result in matching VPD field values to address the logical unit. If multiple target port identifiers and logical unit number combinations access matching VPD field values, the copy manager may use any combination to address the logical unit and shall try other combinations in the event that one combination becomes non-operational during the processing of an EXTENDED COPY command.

Table 56 — Identification descriptor target descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE CODE (E4h)							
1	LU ID TYPE		NUL	PERIPHERAL DEVICE TYPE				
2	(MSB)			RELATIVE INITIATOR PORT IDENTIFIER				
3								(LSB)
4	Reserved				CODE SET			
5	Reserved		ASSOCIATION		IDENTIFIER TYPE			
6	Reserved							
7	IDENTIFIER LENGTH (n-7)							
8	IDENTIFIER							
n								
n+1	Reserved							
27								
28	Device type specific parameters							
31								

The DESCRIPTOR TYPE CODE field, PERIPHERAL DEVICE TYPE field, NUL bit, RELATIVE INITIATOR PORT IDENTIFIER field, and the device type specific parameters are described in 6.3.6.1.

The LU ID TYPE field is reserved for this target descriptor.

The contents of the CODE SET, ASSOCIATION, IDENTIFIER TYPE, IDENTIFIER LENGTH, and IDENTIFIER fields are specified in 7.6.3.

The identifier length shall be 20 or less. If the identifier length is 20, there shall be no reserved bytes between the target descriptor parameters and the device type specific parameters.

Some combinations of code set, association, identifier type, identifier length and identifier do not uniquely identify a logical unit to serve as a copy target device. The behavior of the copy manager when such combinations are received is unpredictable.

6.3.6.3 Alias target descriptor format

The target descriptor format shown in table 57 instructs the copy manager to locate a SCSI target port and logical unit using the alias list (see 3.1.7) designation associated with the specified alias value. The alias list is maintained using the CHANGE ALIASES command (see 6.2).

Table 57 — Alias target descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE CODE (E6h)							
1	LU ID TYPE		NUL	PERIPHERAL DEVICE TYPE				
2	(MSB)			RELATIVE INITIATOR PORT IDENTIFIER				
3								(LSB)
4				LU IDENTIFIER				
11								
12				ALIAS VALUE				
19								
20				Reserved				
27								
28				Device type specific parameters				
31								

The DESCRIPTOR TYPE CODE field, PERIPHERAL DEVICE TYPE field, NUL bit, RELATIVE INITIATOR PORT IDENTIFIER field, and the device type specific parameters are described in 6.3.6.1.

The ALIAS VALUE field specifies an alias value in the alias list as managed by the CHANGE ALIASES command (see 6.2) and maintained by the device server.

When the device server first processes an alias target descriptor, it shall check the value of the ALIAS VALUE field for a corresponding entry in the alias list. If the value is not in the alias list or the device server is unable to validate the designation (see 6.2.3) associated with the alias value, the command shall be terminated because the copy target device is unavailable (see 6.3.3). An application client generating EXTENDED COPY commands that include alias target descriptors in the parameter data is responsible for providing a valid entry in the alias list using the CHANGE ALIASES command (see 6.2) prior to sending the EXTENDED COPY command.

6.3.6.4 Device type specific target descriptor parameters for block device types

The format for the device type specific target descriptor parameters for block device types (i.e., device type code values 00h, 04h, 05h, 07h, and 0Eh) is shown in table 58.

Table 58 — Device type specific target descriptor parameters for block device types

Bit / Byte	7	6	5	4	3	2	1	0
28	Reserved					PAD	Reserved	
29	(MSB)			DISK BLOCK LENGTH				
30								
31								(LSB)

The PAD bit is used in conjunction with the CAT bit (see 6.3.7.1) in the segment descriptor to determine what action should be taken when a segment of the copy does not fit exactly into an integer number of destination blocks (see 6.3.7.2).

The DISK BLOCK LENGTH field contains the number of bytes in a disk block, excluding any protection information (see SBC-2), for the logical device being addressed.

The copy manager may read ahead from sources of block device type (i.e., the copy manager may perform read operations from a source disk at any time and in any order during processing of an EXTENDED COPY command, provided that the relative order of writes and reads on the same blocks within the same target descriptor does not differ from their order in the segment descriptor list).

6.3.6.5 Device type specific target descriptor parameters for sequential-access device types

The format for the device type specific target descriptor parameters for the sequential-access device type (i.e., device type code value 01h) is shown in table 59.

Table 59 — Device type specific target descriptor parameters for sequential-access device types

Bit Byte	7	6	5	4	3	2	1	0
28	Reserved					PAD	Reserved	FIXED
29	(MSB)							
30				STREAM BLOCK LENGTH				
31								(LSB)

The contents of the FIXED bit and STREAM BLOCK LENGTH field are combined with the STREAM DEVICE TRANSFER LENGTH FIELD in the segment descriptor to determine the length of the stream read or write operation as specified in table 60.

Table 60 — Stream device transfer lengths

FIXED bit	STREAM BLOCK LENGTH field	Description
0	000000h	Use variable length reads or writes. The number of bytes for each read or write is specified by the STREAM DEVICE TRANSFER LENGTH field in the segment descriptor.
0	000001h - FFFFFFh	The command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.
1	000000h	The command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST
1	000001h - FFFFFFh	Use fixed record length reads or writes. The number of bytes for each read or write shall be the product of the STREAM BLOCK LENGTH field and the STREAM DEVICE TRANSFER LENGTH field in the segment descriptor.

The PAD bit is used in conjunction with the CAT bit (see 6.3.7.2) in the segment descriptor to determine what action should be taken when a segment of the copy does not fit exactly into an integer number of destination blocks (see 6.3.7.2).

All read commands issued to sequential-access type devices shall have the SILI bit equal to zero.

The copy manager shall not read ahead from sources of stream device type (i.e., the read operations required by a segment descriptor for which the source is a stream device shall not be started until all write operations for previous segment descriptors have completed).

6.3.6.6 Device type specific target descriptor parameters for processor device types

The format for the device type specific target descriptor parameters for the processor device type (i.e., device type code value 03h) is shown in table 61.

Table 61 — Device type specific target descriptor parameters for processor device types

Bit / Byte	7	6	5	4	3	2	1	0
28	Reserved					PAD	Reserved	
29	Reserved							
31								

The PAD bit is used in conjunction with the CAT bit (see 6.3.7.2) in the segment descriptor to determine what action should be taken when a segment of the copy does not fit exactly into an integer number of SEND or RECEIVE commands (see 6.3.7.2).

When the processor device is a source, the number of bytes to be transferred by a SEND command shall be specified by STREAM DEVICE TRANSFER LENGTH field in the segment descriptor. When the processor device is a destination, the number of bytes to be transferred by a RECEIVE command shall be specified by STREAM DEVICE TRANSFER LENGTH field in the segment descriptor.

6.3.7 Segment descriptors

6.3.7.1 Segment descriptors introduction

Segment descriptors (see table 62) begin with an eight byte header.

Table 62 — Segment descriptor header

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE CODE (00h-3Fh)							
1	Reserved						DC	CAT
2	(MSB)			DESCRIPTOR LENGTH (n-7)				
3								(LSB)
4	(MSB)			SOURCE TARGET DESCRIPTOR INDEX				
5								(LSB)
6	(MSB)			DESTINATION TARGET DESCRIPTOR INDEX				
7								(LSB)
8	Segment descriptor parameters							
n								

The DESCRIPTOR TYPE CODE field is described in 6.3.5. Support for each segment descriptor format is optional. If a copy manager receives an unsupported descriptor type code in a segment descriptor, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to UNSUPPORTED SEGMENT DESCRIPTOR TYPE CODE.

The destination count (DC) bit is only applicable to segment descriptors with descriptor type code values of 02h and 0Dh (see 6.3.7.5). The DC bit is reserved for all other segment descriptors.

The CAT bit is described in 6.3.7.2.

The DESCRIPTOR LENGTH field contains the length in bytes of the fields that follow the DESCRIPTOR LENGTH field in the segment descriptor. For each descriptor type code value, the length should be constant.

The SOURCE TARGET DESCRIPTOR INDEX field contains an index into the target descriptor list (see 6.3.1) identifying the source copy target device. The DESTINATION TARGET DESCRIPTOR INDEX field contains an index into the target descriptor list (see 6.3.1) identifying the destination copy target device. Some segment descriptor formats do not require a SOURCE TARGET DESCRIPTOR INDEX field or a DESTINATION TARGET DESCRIPTOR INDEX field, in which case the field is reserved.

If the copy target device identified by a SOURCE TARGET DESCRIPTOR INDEX field or a DESTINATION TARGET DESCRIPTOR INDEX field is not accessible to the copy manager, then the command shall be terminated with CHECK CONDITION status, with the sense key set to COPY ABORTED, and the additional sense code set to UNREACHABLE COPY TARGET.

6.3.7.2 Segment descriptor processing

In processing a segment descriptor, the copy manager may be required to:

a) Read source data by issuing data input commands to the source device;
b) Process data, an operation that generally designates data as destination data intended for transfer to the destination device; and
c) Write some or all of the destination data to the destination device.

The number of blocks to read and write, the number of bytes to process, and the nature of processing are determined by the segment descriptor type code, the parameters of the segment descriptor, and the amount of residual source or destination data retained from the previous segment, if any.

Except as otherwise specified by particular segment descriptor type codes:

a) Just enough whole-block read operations shall be performed to supply, together with residual source data from the previous segment or segments, the number of bytes to be processed;
b) Processing consists of removing bytes from the source data and designating them as destination data, without change; and
c) As many whole-block write operations as possible shall be performed with the destination data, including any residual destination data from the previous segment or segments.

Any residual source data from the previous segment or segments shall be processed before any data read from the source device during processing of the current segment descriptor. Any residual destination data from the previous segment or segments shall be written before any data processed during processing of the current segment descriptor.

Exceptions and clarifications to these general rules are described in table 63 and the subclauses it references.

Table 63 — Descriptor Type Code Dependent Copy Manager Processing (part 1 of 2)

Segment Descriptor Type Code	Reference	Description
00h (block→ stream) or 0Bh (block→ stream+application client)	6.3.7.3	The number of bytes processed is determined by the BLOCK DEVICE NUMBER OF BLOCKS field for the source blocks (see applicable type code definition subclauses for details). [a]
02h (block→ block) or 0Dh (block→ block+application client) with DC=0	6.3.7.5	
02h (block→ block) or 0Dh (block→ block+application client) with DC=1	6.3.7.5	The number of blocks or byte range specified shall be output to the destination device. If residual destination data is sufficient to perform the output, then no data shall be processed. Otherwise, just as much data as needed shall be processed (which may involve reading data from the source device) so that the destination data (which includes any residual destination data from the previous segment) is sufficient. [a]
01h (stream→ block) or 0Ch (stream→ block+application client)	6.3.7.3	
09h (stream→ block<o>)	6.3.7.12	
03h (stream→ stream) or 0Eh (stream→ stream+application client)	6.3.7.6	The number of bytes specified in the segment descriptor shall be processed. [a]
04h (inline→ stream)	6.3.7.7	The specified number of bytes of inline or embedded data shall be appended to the destination data, and no source data shall be processed.
05h (embedded→ stream)	6.3.7.8	
06h (stream→ discard)	6.3.7.9	The specified number of bytes shall be removed from the source data and discarded.

[a] For segment descriptor type codes 0Bh, 0Ch, 0Dh and 0Eh, a copy of the processed data shall also be held for retrieval by the application client.

Table 63 — Descriptor Type Code Dependent Copy Manager Processing (part 2 of 2)

Segment Descriptor Type Code	Reference	Description
07h (verify device operation)	6.3.7.10	No data shall be processed and no read or write operations shall be performed on copy target devices. Residual source or destination data, if any, shall be retained or discarded as if the CAT bit were equal to one.
10h (filemark→ tape)	6.3.7.14	
11h (space→ tape)	6.3.7.15	
12h (locate→ tape)	6.3.7.16	
14h (register persistent reservation key)	6.3.7.18	
08h (block<o>→ stream)	6.3.7.11	The required blocks shall be read from the source device, the designated byte range shall be extracted as source data, and the designated number of bytes (starting with residual source data, if any) shall be processed.
0Ah (block<o>→ block<o>)	6.3.7.13	The source byte range specified shall be read into source data, the number of bytes specified shall be moved from source data to destination data, and the specified destination byte range shall be written using destination data.
0Fh (stream→ discard+application client)	6.3.7.9	The specified number of bytes shall be removed from the source data and held for retrieval by the application client.
13h (<i>tape→ <i>tape)	6.3.7.17	The data movement shall not involve processing as described in this subclause. Residual source or destination data, if any, shall not be used and shall be retained or discarded as if the CAT bit were equal to one.

[a] For segment descriptor type codes 0Bh, 0Ch, 0Dh and 0Eh, a copy of the processed data shall also be held for retrieval by the application client.

Reads and writes shall be performed using whole-block transfer lengths determined by the block size, transfer length, or both. Therefore some source data may remain unprocessed and some destination data may not have been transferred at the end of a segment. If so, the residue shall be handled according to the CAT bit in the segment descriptor and the PAD bits of the source and destination target descriptors, as defined in table 64.

Table 64 — PAD and CAT bit definitions

PAD bit in		CAT bit	Copy manager action
Source target descriptor	Destination target descriptor		
0 or 1	0 or 1	1	Any residual source data shall be retained as source data for a subsequent segment descriptor. Any residual destination data shall be retained as destination data for a subsequent segment descriptor. It shall not be an error if either the source or destination target index in the following segment descriptor does not match the corresponding target index with which residual data was originally associated. If the CAT bit is set to one on the last segment of an EXTENDED COPY command, any residual data shall be discarded and this shall not be considered an error.
1	1	0	Any residual source data shall be discarded. Any residual destination data shall be padded with zeroes to make a whole block transfer. [a]
0	1	0	Any residual source data shall be handled as if the CAT bit is equal to one (i.e., discarded on the last segment and retained otherwise). Any residual destination data shall be padded with zeroes to make a whole block transfer. [a]
1	0	0	Any residual source or destination data shall be discarded.
0	0	0	If there is residual source or destination data, the EXTENDED COPY command shall be terminated with CHECK CONDITION status, with the sense key set to COPY ABORTED, and the additional sense code set to UNEXPECTED INEXACT SEGMENT.

[a] When the CAT bit is set to zero and the destination target descriptor has the PAD bit set to one, the EXTENDED COPY command shall be terminated with CHECK CONDITION status, with the sense key set to COPY ABORTED, and the additional sense code set to UNEXPECTED INEXACT SEGMENT if any of the following conditions are met:

a) If any residual destination data is present after writing the designated byte range for a segment descriptor of type 09h (stream→ block <o>) or 0Ah (block<o>→ block<o>); or
b) If any residual destination data is present after the designated number of blocks have been written for a segment descriptor of type 02h (block→ block) with DC set to one, 0Dh (block→ block+application client) with DC set to one, 01h (stream→ block) or 0Ch (stream→ block+application client).

A few segment descriptors have either no source or no destination and handling of the PAD bit for those descriptors shall be as follows. For segment descriptor types 04h (inline→ stream, see 6.3.7.7) and 05h (embedded→ stream, see 6.3.7.8), the handling shall be as if the PAD is set to zero for the source target descriptor. For segment descriptor types 06h and 0Fh (stream→ discard and stream→ discard+application client, see 6.3.7.9), handling shall be as if the PAD is set to zero for the destination target descriptor.

6.3.7.3 Block device to stream device operations

The segment descriptor format shown in table 65 is used by the copy operations that move data from a block device to a stream device or vice versa.

Table 65 — Block device to or from stream device segment descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE CODE (00h, 01h, 0Bh, or 0Ch)							
1	Reserved							CAT
2	(MSB)			DESCRIPTOR LENGTH (0014h)				
3								(LSB)
4	(MSB)			SOURCE TARGET DESCRIPTOR INDEX				
5								(LSB)
6	(MSB)			DESTINATION TARGET DESCRIPTOR INDEX				
7								(LSB)
8	Reserved							
9	(MSB)			STREAM DEVICE TRANSFER LENGTH				
10								
11								(LSB)
12	Reserved							
13	Reserved							
14	(MSB)			BLOCK DEVICE NUMBER OF BLOCKS				
15								(LSB)
16	(MSB)			BLOCK DEVICE LOGICAL BLOCK ADDRESS				
23								(LSB)

The DESCRIPTOR TYPE CODE field is described in 6.3.5 and 6.3.7.1. Two DESCRIPTOR TYPE CODE values use the segment descriptor format shown in table 65 and described in this subclause.

For descriptor type code 00h (block→ stream) or descriptor type code 0Bh (block→ stream+application client), the copy manager shall copy the data from the source block device identified by the SOURCE TARGET DESCRIPTOR INDEX field to the destination stream device identified by the DESTINATION TARGET DESCRIPTOR INDEX field using the logical blocks starting at the location identified by the BLOCK DEVICE LOGICAL BLOCK ADDRESS field. As many blocks shall be read as necessary to process (see 6.3.7.2) a number of bytes equal to the contents of the DISK BLOCK LENGTH field in the target descriptor for the source device times the contents of the BLOCK DEVICE NUMBER OF BLOCKS field. The data shall be written to the stream device starting at the current position of the media.

For descriptor type code 0Bh (block→ stream+application client), the copy manager also shall hold a copy of the processed data for delivery to the application client upon completion of the EXTENDED COPY command in response to a RECEIVE COPY RESULTS command with RECEIVE DATA service action as described in 6.17.3. The minimum amount of held data supported by the copy manager is returned in the response data for the RECEIVE COPY RESULTS command with OPERATING PARAMETERS service action (see 6.17.4). If the copy manager supports the 0Bh descriptor type code, it also shall support the RECEIVE COPY RESULTS command with RECEIVE DATA service action.

The CAT bit is described in 6.3.7.2.

The DESCRIPTOR LENGTH field shall contain 20 (0014h). The SOURCE TARGET DESCRIPTOR INDEX and DESTINATION TARGET DESCRIPTOR INDEX fields are described in 6.3.7.1.

The STREAM DEVICE TRANSFER LENGTH field specifies the amount of data to be written on each write operation to the stream device. See 6.3.6.5 for a description of how data in the STREAM DEVICE TRANSFER LENGTH field in the segment descriptor interacts with data in the STREAM BLOCK LENGTH field in the device type specific target descriptor parameters for the sequential-access device type.

The BLOCK DEVICE NUMBER OF BLOCKS field specifies the length, in source logical blocks, of data to be processed (see 6.3.7.2) in the segment. A value of zero shall not be considered as an error. No data shall be processed, but any residual destination data retained from a previous segment shall be written if possible to the destination in whole-block transfers. A value of zero shall not modify the handling of residual data.

The BLOCK DEVICE LOGICAL BLOCK ADDRESS field specifies the starting logical block address on the block device for this segment.

6.3.7.4 Stream device to block device operations

The segment descriptor format shown in table 65 (see 6.3.7.3) also is used by the copy operations that move data from a stream device to a block device. Two DESCRIPTOR TYPE CODE values use the segment descriptor format shown in table 65 and described in this subclause.

For descriptor type code 01h (stream→ block) or descriptor type code 0Ch (stream→ block+application client), the copy manager shall copy the data from the source stream device identified by the SOURCE TARGET DESCRIPTOR INDEX field to the destination block device identified by the DESTINATION TARGET DESCRIPTOR INDEX field using the stream data starting at the current position of the stream device. The data shall be written to logical blocks starting at the location identified by the BLOCK DEVICE LOGICAL BLOCK ADDRESS field and continuing for the number of blocks specified in the BLOCK DEVICE NUMBER OF BLOCKS field.

For descriptor type code 0Ch (stream→ block+application client), the copy manager also shall hold a copy of the processed data for delivery to the application client upon completion of the EXTENDED COPY command in response to a RECEIVE COPY RESULTS command with RECEIVE DATA service action as described in 6.17.3. The minimum amount of held data supported by the copy manager is returned in the response data for the RECEIVE COPY RESULTS command with OPERATING PARAMETERS service action (see 6.17.4). If the copy manager supports the 0Ch descriptor type code, it also shall support the RECEIVE COPY RESULTS command with RECEIVE DATA service action.

The CAT bit is described in 6.3.7.2.

The DESCRIPTOR LENGTH field shall contain 20 (0014h). The SOURCE TARGET DESCRIPTOR INDEX and DESTINATION TARGET DESCRIPTOR INDEX fields are described in 6.3.7.1.

The STREAM DEVICE TRANSFER LENGTH field specifies the amount of data to be read from the source stream device on each read operation. See 6.3.6.5 for a description of how data in the STREAM DEVICE TRANSFER LENGTH field in the segment descriptor interacts with data in the STREAM BLOCK LENGTH field in the device type specific target descriptor parameters for the sequential-access device type.

The BLOCK DEVICE NUMBER OF BLOCKS field specifies the number blocks to be written by the segment. A value of zero specifies that no blocks shall be written in this segment. This shall not be considered as an error.

The BLOCK DEVICE LOGICAL BLOCK ADDRESS field specifies the starting logical block address on the block device for this segment.

6.3.7.5 Block device to block device operations

The segment descriptor format shown in table 66 is used by the copy operations that move data from a block device to a block device.

Table 66 — Block device to block device segment descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE CODE (02h or 0Dh)							
1	Reserved						DC	CAT
2	(MSB)			DESCRIPTOR LENGTH (0018h)				
3								(LSB)
4	(MSB)			SOURCE TARGET DESCRIPTOR INDEX				
5								(LSB)
6	(MSB)			DESTINATION TARGET DESCRIPTOR INDEX				
7								(LSB)
8	Reserved							
9	Reserved							
10	(MSB)			BLOCK DEVICE NUMBER OF BLOCKS				
11								(LSB)
12	(MSB)			SOURCE BLOCK DEVICE LOGICAL BLOCK ADDRESS				
19								(LSB)
20	(MSB)			DESTINATION BLOCK DEVICE LOGICAL BLOCK ADDRESS				
27								(LSB)

The DESCRIPTOR TYPE CODE field is described in 6.3.5 and 6.3.7.1. Two DESCRIPTOR TYPE CODE values use the segment descriptor format shown in table 66 and described in this subclause.

For descriptor type code 02h (block→ block) or descriptor type code 0Dh (block→ block+application client), the copy manager shall copy the data from the source block device identified by the SOURCE TARGET DESCRIPTOR INDEX field to the destination block device identified by the DESTINATION TARGET DESCRIPTOR INDEX field using the logical blocks starting at the location identified by the SOURCE BLOCK DEVICE LOGICAL BLOCK ADDRESS field. The data shall be written to logical blocks starting at the location identified by the DESTINATION BLOCK DEVICE LOGICAL BLOCK ADDRESS field.

If the DC bit equals zero, then as many blocks shall be read as necessary to process (see 6.3.7.2) a number of bytes equal to the contents of the DISK BLOCK LENGTH field in the target descriptor for the source device times the contents of the BLOCK DEVICE NUMBER OF BLOCKS field, and as many writes as possible shall be performed using any residual destination data from the previous segment and the data processed in this segment. If the DC bit equals one, then the number of blocks specified by the BLOCK DEVICE NUMBER OF BLOCKS field shall be written to the destination block device, as many bytes shall be processed as necessary for these writes to be performed, and as many blocks shall be read as necessary to supply the data to be processed.

For descriptor type code 0Dh (block→ block+application client), the copy manager also shall hold a copy of the processed data for delivery to the application client upon completion of the EXTENDED COPY command in response to a RECEIVE COPY RESULTS command with RECEIVE DATA service action as described in 6.17.3. The minimum amount of held data supported by the copy manager is returned in the response data for the RECEIVE COPY RESULTS command with OPERATING PARAMETERS service action (see 6.17.4). If the copy manager supports the 0Dh descriptor type code, it also shall support the RECEIVE COPY RESULTS command with RECEIVE DATA service action.

The CAT bit is described in 6.3.7.2.

The destination count (DC) bit specifies whether the BLOCK DEVICE NUMBER OF BLOCKS field refers to the source or destination device. A DC bit set to zero specifies that the BLOCK DEVICE NUMBER OF BLOCKS field refers to the source device. A DC bit set to one specifies that the BLOCK DEVICE NUMBER OF BLOCKS field refers to the destination device.

The DESCRIPTOR LENGTH field shall contain 24 (0018h). The SOURCE TARGET DESCRIPTOR INDEX and DESTINATION TARGET DESCRIPTOR INDEX fields are described in 6.3.7.1.

If DC is set to zero, the BLOCK DEVICE NUMBER OF BLOCKS field specifies the number of blocks to be processed. If DC is set to one, the BLOCK DEVICE NUMBER OF BLOCKS field specifies the number of blocks to be written to the destination device. A value of zero shall not be considered as an error. If the DC bit equals one, a value of zero specifies that no destination blocks shall be written and the only processing to be performed is that any residual source or destination data from the previous segment shall be handled as residual data as described in 6.3.7.2. If the DC bit equals zero, a value of zero specifies that no source blocks shall be read and no source data shall be processed, but any residual destination data from a previous segment shall be written if possible to the destination in whole-block transfers, and any residual data shall be handled as described in 6.3.7.2.

The SOURCE BLOCK DEVICE LOGICAL BLOCK ADDRESS field specifies the logical block address from which the reading of data shall start.

The DESTINATION BLOCK DEVICE LOGICAL BLOCK ADDRESS field specifies the logical block address to which the writing of data shall begin.

6.3.7.6 Stream device to stream device operations

The segment descriptor format shown in table 67 is used by the copy operations that move data from a stream device to a stream device.

Table 67 — Stream device to stream device segment descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE CODE (03h or 0Eh)							
1	Reserved							CAT
2	(MSB)			DESCRIPTOR LENGTH (0010h)				
3								(LSB)
4	(MSB)			SOURCE TARGET DESCRIPTOR INDEX				
5								(LSB)
6	(MSB)			DESTINATION TARGET DESCRIPTOR INDEX				
7								(LSB)
8	Reserved							
9	(MSB)			SOURCE STREAM DEVICE TRANSFER LENGTH				
10								
11								(LSB)
12	Reserved							
13	(MSB)			DESTINATION STREAM DEVICE TRANSFER LENGTH				
14								
15								(LSB)
16	(MSB)			BYTE COUNT				
19								(LSB)

The DESCRIPTOR TYPE CODE field is described in 6.3.5 and 6.3.7.1. Two DESCRIPTOR TYPE CODE values use the segment descriptor format shown in table 67 and described in this subclause.

For descriptor type code 03h (stream→ stream) or descriptor type code 0Eh (stream→ stream+application client), the copy manager shall copy the data from the source stream device identified by the SOURCE TARGET DESCRIPTOR INDEX field to the destination stream device identified by the DESTINATION TARGET DESCRIPTOR INDEX field. Data shall be read from the source stream device starting at the current position of the source stream device. Data shall be written to the destination stream device starting at the current position of the destination stream device. The BYTE COUNT field defines the number of bytes to be processed (see 6.3.7.2) by the copy manager. The copy manager shall perform read operations as necessary to supply the source data, and as many write operations as possible using the destination data.

For descriptor type code 0Eh (stream→ stream+application client), the copy manager also shall hold a copy of the processed data for delivery to the application client upon completion of the EXTENDED COPY command in response to a RECEIVE COPY RESULTS command with RECEIVE DATA service action as described in 6.17.3. The minimum amount of held data supported by the copy manager is returned in the response data for the RECEIVE COPY RESULTS command with OPERATING PARAMETERS service action (see 6.17.4). If the copy manager supports the 0Eh descriptor type code, it also shall support the RECEIVE COPY RESULTS command with RECEIVE DATA service action.

The CAT bit is described in 6.3.7.2.

The DESCRIPTOR LENGTH field shall contain 16 (0010h). The SOURCE TARGET DESCRIPTOR INDEX and DESTINATION TARGET DESCRIPTOR INDEX fields are described in 6.3.7.1.

The SOURCE STREAM DEVICE TRANSFER LENGTH field specifies the amount of data to be read from the source stream device on each read operation. See 6.3.6.5 for a description of how data in the SOURCE STREAM DEVICE TRANSFER LENGTH field in the segment descriptor interacts with data in the STREAM BLOCK LENGTH field in the device type specific target descriptor parameters for the source sequential-access device type.

The DESTINATION STREAM DEVICE TRANSFER LENGTH field specifies the amount of data to be written to the destination stream device on each write operation. See 6.3.6.5 for a description of how data in the DESTINATION STREAM DEVICE TRANSFER LENGTH field in the segment descriptor interacts with data in the STREAM BLOCK LENGTH field in the device type specific target descriptor parameters for the destination sequential-access device type.

The BYTE COUNT field specifies the number of bytes that shall be processed for this segment descriptor. A value of zero shall not be considered as an error, and specifies that no source blocks shall be read and no source data shall be processed. However, a value of zero specifies that any residual destination data from a previous segment shall be written if possible to the destination in whole-block transfers, and any residual data shall be handled as described in 6.3.7.2.

6.3.7.7 Inline data to stream device operation

The segment descriptor format shown in table 68 instructs the copy manager to write inline data from the EXTENDED COPY parameter list to a stream device.

Table 68 — Inline data to stream device segment descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE CODE (04h)							
1	Reserved							CAT
2	(MSB)			DESCRIPTOR LENGTH (0010h)				
3								(LSB)
4	Reserved							
5	Reserved							
6	(MSB)			DESTINATION TARGET DESCRIPTOR INDEX				
7								(LSB)
8	Reserved							
9	(MSB)			STREAM DEVICE TRANSFER LENGTH				
10								
11								(LSB)
12	(MSB)			INLINE DATA OFFSET				
15								(LSB)
16	(MSB)			INLINE DATA NUMBER OF BYTES				
19								(LSB)

The DESCRIPTOR TYPE CODE field is described in 6.3.5 and 6.3.7.1. Descriptor type code 04h (inline→ stream) instructs the copy manager to write inline data from the EXTENDED COPY parameter list to a stream device. The inline data shall be read from the optional inline data at the end of the EXTENDED COPY parameter list. The data shall be written to the destination stream device identified by the DESTINATION TARGET DESCRIPTOR INDEX field starting at the current position of the stream device. Any residual destination data from a previous segment

descriptor shall be written before the data of the current segment descriptor. Any residual source data from a previous segment descriptor shall not be processed (see 6.3.7.2), and shall be handled as residual source data.

The CAT bit is described in 6.3.7.2.

The DESCRIPTOR LENGTH field shall contain 16 (0010h). The DESTINATION TARGET DESCRIPTOR INDEX field is described in 6.3.7.1.

The STREAM DEVICE TRANSFER LENGTH field specifies the amount of data to be written to the stream device on each write operation. See 6.3.6.5 for a description of how data in the STREAM DEVICE TRANSFER LENGTH field in the segment descriptor interacts with data in the STREAM BLOCK LENGTH field in the device type specific target descriptor parameters for the destination sequential-access device type.

The value in the INLINE DATA OFFSET field is added to the location of the first byte of inline data in the EXTENDED COPY parameter list (see table 51) to locate the first byte of inline data to be written to the stream device. The INLINE DATA OFFSET value shall be a multiple of 4.

The INLINE DATA NUMBER OF BYTES field specifies the number of bytes of inline data that are to be transferred to the stream device. A value of zero shall not be considered an error.

If the sum of the INLINE DATA OFFSET and the INLINE DATA NUMBER OF BYTES values exceeds the value in the INLINE DATA LENGTH field (see table 51), the copy manager shall terminate the command with CHECK CONDITION status, with the sense key set to COPY ABORTED, and the additional sense code set to INLINE DATA LENGTH EXCEEDED.

6.3.7.8 Embedded data to stream device operation

The segment descriptor format shown in table 69 instructs the copy manager to write embedded data from the segment descriptor to a stream device.

Table 69 — Embedded data to stream device segment descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE CODE (05h)							
1	Reserved							CAT
2	(MSB)			DESCRIPTOR LENGTH (n-3)				
3								(LSB)
4	Reserved							
5	Reserved							
6	(MSB)			DESTINATION TARGET DESCRIPTOR INDEX				
7								(LSB)
8	Reserved							
9	(MSB)			STREAM DEVICE TRANSFER LENGTH				
10								
11								(LSB)
12	(MSB)			EMBEDDED DATA NUMBER OF BYTES				
13								(LSB)
14				Reserved				
15								
16				EMBEDDED DATA				
n								

The DESCRIPTOR TYPE CODE field is described in 6.3.5 and 6.3.7.1. Descriptor type code 05h (embedded→ stream) instructs the copy manager to write embedded data from the segment descriptor to a stream device. The embedded data shall be read from the segment descriptor. The data shall be written to the destination stream device identified by the DESTINATION TARGET DESCRIPTOR INDEX field starting at the current position of the stream device. Any residual destination data from a previous segment descriptor shall be written before the data of the current segment descriptor. Any residual source data from a previous segment descriptor shall not be processed (see 6.3.7.2), and shall be handled as residual source data.

The CAT bit is described in 6.3.7.2.

The DESCRIPTOR LENGTH field shall contain the length in bytes of the fields that follow the DESCRIPTOR LENGTH field, including the embedded data. The value in the DESCRIPTOR LENGTH field shall be a multiple of 4.

The DESTINATION TARGET DESCRIPTOR INDEX field is described in 6.3.7.1.

The STREAM DEVICE TRANSFER LENGTH field specifies the amount of data to be written to the stream device on each write operation. See 6.3.6.5 for a description of how data in the STREAM DEVICE TRANSFER LENGTH field in the segment descriptor interacts with data in the STREAM BLOCK LENGTH field in the device type specific target descriptor parameters for the destination sequential-access device type.

The EMBEDDED DATA NUMBER OF BYTES field specifies the number of bytes of embedded data that are to be transferred to the stream device. A value of zero shall not be considered an error. The EMBEDDED DATA NUMBER OF BYTES value shall be less than or equal to the DESCRIPTOR LENGTH value minus 12.

6.3.7.9 Stream device to discard operation

The segment descriptor format shown in table 70 instructs the copy manager to read data from a stream device and not copy it to any destination device.

Table 70 — Stream device to discard segment descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE CODE (06h or 0Fh)							
1	Reserved							CAT
2	(MSB)			DESCRIPTOR LENGTH (000Ch)				
3								(LSB)
4	(MSB)			SOURCE TARGET DESCRIPTOR INDEX				
5								(LSB)
6	Reserved							
7	Reserved							
8	Reserved							
9	(MSB)			STREAM DEVICE TRANSFER LENGTH				
10								
11								(LSB)
12	(MSB)			NUMBER OF BYTES				
15								(LSB)

The DESCRIPTOR TYPE CODE field is described in 6.3.5 and 6.3.7.1. Two DESCRIPTOR TYPE CODE values use the segment descriptor format shown in table 70 and described in this subclause.

For descriptor type code 06h (stream→ discard) or descriptor type code 0Fh (stream→ discard+application client), the copy manager shall read data as necessary from the source stream device identified by the SOURCE TARGET DESCRIPTOR INDEX field starting at the current position of the source stream device. The number of bytes specified by the NUMBER OF BYTES field shall be removed from the source data, starting with any residual source data from the previous segment.

For descriptor type code 06h (stream→ discard) the removed data shall be discarded and not written to any destination device. For descriptor type code 0Fh (stream→ discard+application client) the removed data shall be held for delivery to the application client upon completion of the EXTENDED COPY command in response to a RECEIVE COPY RESULTS command with RECEIVE DATA service action as described in 6.17.3. The minimum amount of held data supported by the copy manager is returned in the response data for the RECEIVE COPY RESULTS command with OPERATING PARAMETERS service action (see 6.17.4). If the copy manager supports the 0Fh (stream→ discard+application client) descriptor type code, it also shall support the RECEIVE COPY RESULTS command with RECEIVE DATA service action.

The CAT bit is described in 6.3.7.2.

The DESCRIPTOR LENGTH field shall contain 12 (000Ch). The DESTINATION TARGET DESCRIPTOR INDEX field is described in 6.3.7.1.

The SOURCE STREAM DEVICE TRANSFER LENGTH field specifies the amount of data to be read from the source stream device on each read operation. See 6.3.6.5 for a description of how data in the SOURCE STREAM DEVICE TRANSFER LENGTH field in the segment descriptor interacts with data in the STREAM BLOCK LENGTH field in the device type specific target descriptor parameters for the source sequential-access device type.

The NUMBER OF BYTES field specifies the number of bytes to be removed from the source data.

6.3.7.10 Verify device operation

The segment descriptor format shown in table 71 instructs the copy manager to verify the accessibility of a SCSI device.

Table 71 — Verify device operation segment descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE CODE (07h)							
1	Reserved							
2	(MSB)			DESCRIPTOR LENGTH (0008h)				
3								(LSB)
4	(MSB)			SOURCE TARGET DESCRIPTOR INDEX				
5								(LSB)
6	Reserved							
7								
8	Reserved							TUR
9	Reserved							
11								

The DESCRIPTOR TYPE CODE field is described in 6.3.5 and 6.3.7.1. Descriptor type code 07h instructs the copy manager to verify the accessibility of the device identified by the SOURCE TARGET DESCRIPTOR INDEX field.

The DESCRIPTOR LENGTH field shall contain 8 (0008h). The SOURCE TARGET DESCRIPTOR INDEX field is described in 6.3.7.1.

Support for a value of one in the TUR (Test Unit Ready) bit is optional. If setting the TUR bit to one is supported and the TUR bit is set to one, then a TEST UNIT READY command (see 6.33) shall be used to determine the readiness of the device. If setting the TUR to one is not supported and the TUR bit is set to one, then the EXTENDED COPY command shall be terminated with CHECK CONDITION status, with the sense key set to COPY ABORTED, and the additional sense code set to INVALID FIELD IN PARAMETER LIST. The SENSE-KEY SPECIFIC field shall be set as described in 6.3.3. If the TUR bit is set to zero, then the accessibility should be verified without disturbing established unit attention conditions or ACA conditions (e.g., using the INQUIRY command (see 6.4)).

6.3.7.11 Block device with offset to stream device operation

The segment descriptor format shown in table 72 is used to instruct the copy manager to move data from a block device with a byte offset to a stream device or vice versa.

Table 72 — Block device with offset to or from stream device segment descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE CODE (08h or 09h)							
1	Reserved							CAT
2	(MSB)			DESCRIPTOR LENGTH (0018h)				
3								(LSB)
4	(MSB)			SOURCE TARGET DESCRIPTOR INDEX				
5								(LSB)
6	(MSB)			DESTINATION TARGET DESCRIPTOR INDEX				
7								(LSB)
8	Reserved							
9	(MSB)			STREAM DEVICE TRANSFER LENGTH				
10								
11								(LSB)
12	(MSB)			NUMBER OF BYTES				
15								(LSB)
16	(MSB)			BLOCK DEVICE LOGICAL BLOCK ADDRESS				
23								(LSB)
24	Reserved							
25	Reserved							
26	(MSB)			BLOCK DEVICE BYTE OFFSET				
27								(LSB)

The DESCRIPTOR TYPE CODE field is described in 6.3.5 and 6.3.7.1. Descriptor type code 08h (block<o>→ stream) instructs the copy manager to copy the data from the source block device identified by the SOURCE TARGET DESCRIPTOR INDEX field to the destination stream device identified by the DESTINATION TARGET DESCRIPTOR INDEX field using data starting at the location identified by the BLOCK DEVICE BYTE OFFSET field in the logical block identified by the BLOCK DEVICE LOGICAL BLOCK ADDRESS field and continuing for the number of bytes specified in the NUMBER OF BYTES field. The data shall be written to the stream device starting at the current position of the media.

The CAT bit is described in 6.3.7.2.

The DESCRIPTOR LENGTH field shall contain 24 (0018h). The SOURCE TARGET DESCRIPTOR INDEX and DESTINATION TARGET DESCRIPTOR INDEX fields are described in 6.3.7.1.

The STREAM DEVICE TRANSFER LENGTH field specifies the amount of data to be written on each write operation to the stream device. See 6.3.6.5 for a description of how data in the STREAM DEVICE TRANSFER LENGTH field in the segment descriptor interacts with data in the STREAM BLOCK LENGTH field in the device type specific target descriptor parameters for the sequential-access device type.

The NUMBER OF BYTES field specifies the number bytes to be read. A value of zero specifies that no bytes shall be transferred in this segment. This shall not be considered as an error.

The BLOCK DEVICE LOGICAL BLOCK ADDRESS field specifies the starting logical block address on the source block device for this segment.

The BLOCK DEVICE BYTE OFFSET field specifies the offset into the first source block at which to begin reading bytes.

6.3.7.12 Stream device to block device with offset operation

The segment descriptor format shown in table 72 (see 6.3.7.11) also is used to instruct the copy manager to move data from a stream device to a block device with a byte offset.

The DESCRIPTOR TYPE CODE field is described in 6.3.5 and 6.3.7.1. Descriptor type code 09h (stream→block<o>) instructs the copy manager to copy the data from the source stream device identified by the SOURCE TARGET DESCRIPTOR INDEX field to the destination block device identified by the DESTINATION TARGET DESCRIPTOR INDEX field using the stream data starting at the current position of the stream device. The data shall be written starting at the location identified by the BLOCK DEVICE BYTE OFFSET field in the logical block identified by the BLOCK DEVICE LOGICAL BLOCK ADDRESS field and continuing for the number of bytes specified in the NUMBER OF BYTES field.

The content of the starting logical block on the destination device before the starting offset shall be preserved. The content on the ending logical block beyond the end of the transfer shall be preserved. The copy manager may implement this operation by reading the starting and ending logical blocks, modifying a portion of the blocks as required, and writing the full blocks to the destination device.

The CAT bit is described in 6.3.7.2.

The DESCRIPTOR LENGTH field shall contain 24 (0018h). The SOURCE TARGET DESCRIPTOR INDEX and DESTINATION TARGET DESCRIPTOR INDEX fields are described in 6.3.7.1.

The STREAM DEVICE TRANSFER LENGTH field specifies the amount of data to be written on each write operation to the stream device. See 6.3.6.5 for a description of how data in the STREAM DEVICE TRANSFER LENGTH field in the segment descriptor interacts with data in the STREAM BLOCK LENGTH field in the device type specific target descriptor parameters for the sequential-access device type.

The NUMBER OF BYTES field specifies the number bytes to be read. A value of zero specifies that no bytes shall be transferred in this segment. This shall not be considered as an error.

The BLOCK DEVICE LOGICAL BLOCK ADDRESS field specifies the starting logical block address on the destination block device for this segment.

The BLOCK DEVICE BYTE OFFSET field specifies the offset into the first destination block at which to begin writing data to the destination block device.

6.3.7.13 Block device with offset to block device with offset operation

The segment descriptor format shown in table 73 instructs the copy manager to move data from a block device with a byte offset to a block device with a byte offset.

Table 73 — Block device with offset to block device with offset segment descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE CODE (0Ah)							
1	Reserved							CAT
2	(MSB)			DESCRIPTOR LENGTH (001Ch)				
3								(LSB)
4	(MSB)			SOURCE TARGET DESCRIPTOR INDEX				
5								(LSB)
6	(MSB)			DESTINATION TARGET DESCRIPTOR INDEX				
7								(LSB)
8	(MSB)			NUMBER OF BYTES				
11								(LSB)
12	(MSB)			SOURCE BLOCK DEVICE LOGICAL BLOCK ADDRESS				
19								(LSB)
20	(MSB)			DESTINATION BLOCK DEVICE LOGICAL BLOCK ADDRESS				
27								(LSB)
28	(MSB)			SOURCE BLOCK DEVICE BYTE OFFSET				
29								(LSB)
30	(MSB)			DESTINATION BLOCK DEVICE BYTE OFFSET				
31								(LSB)

The DESCRIPTOR TYPE CODE field is described in 6.3.5 and 6.3.7.1. Descriptor type code 0Ah (block<o>→ block<o>) instructs the copy manager to copy the data from the source block device identified by the SOURCE TARGET DESCRIPTOR INDEX field to the destination block device identified by the DESTINATION TARGET DESCRIPTOR INDEX field using data starting at the location identified by the source BLOCK DEVICE BYTE OFFSET field in the logical block identified by the SOURCE BLOCK DEVICE LOGICAL BLOCK ADDRESS field and continuing for the number of bytes specified in the NUMBER OF BYTES field. The data shall be written starting at the location identified by the DESTINATION BLOCK DEVICE BYTE OFFSET field in the logical block identified by the DESTINATION BLOCK DEVICE LOGICAL BLOCK ADDRESS field.

The content of the starting logical block on the destination device before the starting offset shall be preserved. The content on the ending logical block beyond the end of the transfer shall be preserved. The copy manager may implement this operation by reading the starting and ending logical blocks, modifying a portion of the blocks as required, and writing the full blocks to the destination device.

The CAT bit is described in 6.3.7.2.

The DESCRIPTOR LENGTH field shall contain 28 (001Ch). The SOURCE TARGET DESCRIPTOR INDEX and DESTINATION TARGET DESCRIPTOR INDEX fields are described in 6.3.7.1.

The NUMBER OF BYTES field specifies the number bytes to be read. A value of zero specifies that no bytes shall be transferred in this segment. This shall not be considered as an error.

The SOURCE BLOCK DEVICE LOGICAL BLOCK ADDRESS field specifies the starting address on the source block device for this segment.

The DESTINATION BLOCK DEVICE LOGICAL BLOCK ADDRESS field specifies the starting logical block address on the destination block device for this segment.

The SOURCE BLOCK DEVICE BYTE OFFSET field specifies the offset into the first source block at which to begin reading bytes.

The DESTINATION BLOCK DEVICE BYTE OFFSET field specifies the offset into the first destination block at which to begin writing data to the destination block device.

6.3.7.14 Write filemarks operation

The segment descriptor format shown in table 74 instructs the copy manager to write filemarks or setmarks on the destination tape device.

Table 74 — Write filemarks operation segment descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE CODE (10h)							
1	Reserved							
2	(MSB)							
3	DESCRIPTOR LENGTH (0008h)							(LSB)
4	Reserved							
5	Reserved							
6	(MSB)							
7	DESTINATION TARGET DESCRIPTOR INDEX							(LSB)
8	Reserved						WSMK	IMMED
9	(MSB)							
10	TRANSFER LENGTH							
11								(LSB)

The DESCRIPTOR TYPE CODE field is described in 6.3.5 and 6.3.7.1. Descriptor type code 10h (filemark→ tape) instructs the copy manager to write filemarks or setmarks to the destination tape device identified by the DESTINATION TARGET DESCRIPTOR INDEX field starting at the current position of the tape device. If the PERIPHERAL DEVICE TYPE field in the target descriptor identified by the DESTINATION TARGET DESCRIPTOR INDEX field does not contain 01h, the copy manager shall terminate the command with CHECK CONDITION status, with the sense key set to COPY ABORTED, and the additional sense code set to INVALID OPERATION FOR COPY SOURCE OR DESTINATION.

The DESCRIPTOR LENGTH field shall contain 8 (0008h). The DESTINATION TARGET DESCRIPTOR INDEX field is described in 6.3.7.1.

If the write setmark (WSMK) bit is set to one, the TRANSFER LENGTH field specifies the number of setmarks to be written. If the WSMK bit is set to zero, the TRANSFER LENGTH field specifies the number of filemarks to be written.

If the immediate (IMMED) bit in the segment descriptor is set to one, then the copy manager shall issue a WRITE FILEMARKS command to the destination tape device with the immediate bit is set to one. If the IMMED bit is set to zero, then the copy manager shall issue a WRITE FILEMARKS command to the destination tape device with the immediate bit is set to zero.

6.3.7.15 Space operation

The segment descriptor format shown in table 75 instructs the copy manager to send a SPACE command (see SSC-2) to the destination tape device.

Table 75 — Space operation segment descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE CODE (11h)							
1	Reserved							
2	(MSB)			DESCRIPTOR LENGTH (0008h)				
3								(LSB)
4	Reserved							
5	Reserved							
6	(MSB)			DESTINATION TARGET DESCRIPTOR INDEX				
7								(LSB)
8	Reserved					CODE		
9	(MSB)			COUNT				
10								
11								(LSB)

The DESCRIPTOR TYPE CODE field is described in 6.3.5 and 6.3.7.1. Descriptor type code 11h (space→ tape) instructs the copy manager to send a SPACE command to the destination tape device identified by the DESTINATION TARGET DESCRIPTOR INDEX field. If the PERIPHERAL DEVICE TYPE field in the target descriptor identified by the DESTINATION TARGET DESCRIPTOR INDEX field does not contain 01h, the copy manager shall terminate the command with CHECK CONDITION status, with the sense key set to COPY ABORTED, and the additional sense code set to INVALID OPERATION FOR COPY SOURCE OR DESTINATION.

The DESCRIPTOR LENGTH field shall contain 8 (0008h). The DESTINATION TARGET DESCRIPTOR INDEX field is described in 6.3.7.1.

The CODE and COUNT fields contents in the SPACE command sent to the destination tape device shall be copied from the CODE and COUNT fields in the segment descriptor. All other fields in the SPACE command sent to the destination tape device that affect the positioning of the tape shall be set to zero.

6.3.7.16 Locate operation

The segment descriptor format shown in table 76 instructs the copy manager to send a LOCATE command (see SSC-2) to the destination tape device.

Table 76 — Locate operation segment descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE CODE (12h)							
1	Reserved							
2	(MSB)			DESCRIPTOR LENGTH (0008h)				
3								(LSB)
4	Reserved							
5	Reserved							
6	(MSB)			DESTINATION TARGET DESCRIPTOR INDEX				
7								(LSB)
8	(MSB)			BLOCK ADDRESS				
11								(LSB)

The DESCRIPTOR TYPE CODE field is described in 6.3.5 and 6.3.7.1. Descriptor type code 12h (locate→ tape) instructs the copy manager to send a LOCATE command to the destination tape device identified by the DESTINATION TARGET DESCRIPTOR INDEX field. If the PERIPHERAL DEVICE TYPE field in the target descriptor identified by the DESTINATION TARGET DESCRIPTOR INDEX field does not contain 01h, the copy manager shall terminate the command with CHECK CONDITION status, with the sense key set to COPY ABORTED, and the additional sense code set to INVALID OPERATION FOR COPY SOURCE OR DESTINATION.

The DESCRIPTOR LENGTH field shall contain 8 (0008h). The DESTINATION TARGET DESCRIPTOR INDEX field is described in 6.3.7.1.

The BLOCK ADDRESS field contents in the LOCATE command sent to the destination tape device shall be copied from the BLOCK ADDRESS field in the segment descriptor. All other fields in the LOCATE command sent to the destination tape device that affect the positioning of the tape shall be set to zero.

NOTE 19 - The restrictions described above for the LOCATE command limit the operation to locating SCSI logical block addresses in the current tape partition.

6.3.7.17 Tape device image copy operation

The segment descriptor format shown in table 77 instructs the copy manager to perform an image copy from the source tape device to the destination tape device.

Table 77 — Tape device image copy segment descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE CODE (13h)							
1	Reserved							
2	(MSB)			DESCRIPTOR LENGTH (0008h)				
3								(LSB)
4	(MSB)			SOURCE TARGET DESCRIPTOR INDEX				
5								(LSB)
6	(MSB)			DESTINATION TARGET DESCRIPTOR INDEX				
7								(LSB)
8	(MSB)			COUNT				
11								(LSB)

The DESCRIPTOR TYPE CODE field is described in 6.3.5 and 6.3.7.1. Descriptor type code 13h (<i>tape→ <i>tape) instructs the copy manager to create a compatible image of the source device medium identified by the SOURCE TARGET DESCRIPTOR INDEX field on the destination device medium identified by the DESTINATION TARGET DESCRIPTOR INDEX field beginning at their current positions. If the PERIPHERAL DEVICE TYPE field in the target descriptor identified by the SOURCE TARGET DESCRIPTOR INDEX field or the DESTINATION TARGET DESCRIPTOR INDEX field does not contain 01h, the copy manager shall terminate the command with CHECK CONDITION status, with the sense key set to COPY ABORTED, and the additional sense code set to INVALID OPERATION FOR COPY SOURCE OR DESTINATION.

The DESCRIPTOR LENGTH field shall contain 8 (0008h). The SOURCE TARGET DESCRIPTOR INDEX and DESTINATION TARGET DESCRIPTOR INDEX fields are described in 6.3.7.1.

The tape image copy operation terminates when:

a) The source device encounters an end-of-partition as defined by the source device;
b) The source device encounters an end-of-data as defined by the source device (i.e., BLANK CHECK sense key);
c) The copy manager has copied the number of consecutive filemarks specified in the COUNT field from the source device to the destination device; or
d) The copy manager has copied the number of consecutive filemarks and/or setmarks specified in the COUNT field from the source device to the destination device, if the RSMK bit in the Device Configuration mode page (see SSC-2) of the source device is set to one.

A COUNT field of zero specifies that the EXTENDED COPY command shall not terminate due to any number of consecutive filemarks or setmarks. Other error or exception conditions (e.g., early-warning, end-of-partition on destination device) may cause the EXTENDED COPY command to terminate prior to completion. If this occurs, the residue shall not be calculated and the INFORMATION field in the sense data shall be set to zero.

6.3.7.18 Register persistent reservation key operation

The segment descriptor format shown in table 78 instructs the copy manager to register an I_T nexus using the reservation key (see 5.6.6) specified by the RESERVATION KEY field with the SCSI target device specified by the DESTINATION TARGET DESCRIPTOR INDEX field.

Table 78 — Register persistent reservation key segment descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE CODE (14h)							
1	Reserved							
2	(MSB)			DESCRIPTOR LENGTH (0018h)				
3								(LSB)
4	Reserved							
5	Reserved							
6	(MSB)			DESTINATION TARGET DESCRIPTOR INDEX				
7								(LSB)
8	(MSB)			RESERVATION KEY				
15								(LSB)
16	(MSB)			SERVICE ACTION RESERVATION KEY				
23								(LSB)
24	Reserved							
27								

The DESCRIPTOR TYPE CODE field is described in 6.3.5 and 6.3.7.1. Descriptor type code 14h instructs the copy manager to register an I_T nexus using the reservation key specified by the RESERVATION KEY field with the SCSI target device identified by the DESTINATION TARGET DESCRIPTOR INDEX field using a PERSISTENT RESERVE OUT command with a REGISTER service action (see 6.12.2).

The DESCRIPTOR LENGTH field shall contain 24 (0018h). The DESTINATION TARGET DESCRIPTOR INDEX field is described in 6.3.7.1.

The RESERVATION KEY and SERVICE ACTION RESERVATION KEY field contents in the PERSISTENT RESERVE OUT command sent to the destination device shall be copied from the RESERVATION KEY and SERVICE ACTION RESERVATION KEY fields in the segment descriptor.

The application client sending the EXTENDED COPY command may need to remove the reservation key held by the copy manager as described in 5.6.10 prior to sending the EXTENDED COPY command.

6.3.7.19 Third party persistent reservations source I_T nexus

The segment descriptor format shown in table 79 instructs the copy manager to send a PERSISTENT RESERVATION OUT command with REGISTER AND MOVE service action (see 5.6.7) with the specified I_T nexus after all other segment descriptors have been processed. If an error is detected any time after receiving a third party persistent source reservation I_T nexus segment descriptor, the PERSISTENT RESERVATION OUT command REGISTER AND MOVE service action shall be processed before status is returned for the EXTENDED COPY command.

This segment descriptor should be placed at or near the beginning of the list of segment descriptors to assure the copy manager processes the PERSISTENT RESERVATION OUT command with REGISTER AND MOVE service action in the event of an error that terminates the processing of segment descriptors. If an error is detected in a

segment descriptor and third party persistent reservations source I_T nexus segment descriptor has not been processed, the copy manager shall not send a PERSISTENT RESERVATION OUT command with REGISTER AND MOVE service action.

Placing more than one source third party persistent reservations source I_T nexus segment descriptor in the list of descriptors is not an error. All source third party persistent reservations source I_T nexus segment descriptors known to the copy manager shall be processed after all other segment descriptors have been processed.

Table 79 — Third party persistent reservations source I_T nexus segment descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE CODE (15h)							
1	Reserved							
2	(MSB)			DESCRIPTOR LENGTH (n-3)				
3								(LSB)
4	Reserved							
5	Reserved							
6	(MSB)			DESTINATION TARGET DESCRIPTOR INDEX				
7								(LSB)
8	(MSB)			RESERVATION KEY				
15								(LSB)
16	(MSB)			SERVICE ACTION RESERVATION KEY				
23								(LSB)
24	Reserved							
25	Reserved						UNREG	APTPL
26	(MSB)			RELATIVE TARGET PORT IDENTIFIER				
27								(LSB)
28	(MSB)			TRANSPORTID PARAMETER DATA LENGTH (n - 31)				
31								(LSB)
31	TransportID							
n								

The DESCRIPTOR TYPE CODE field is described in 6.3.5 and 6.3.7.1. Descriptor type code 15h instructs the copy manager to send PERSISTENT RESERVATION OUT command with REGISTER AND MOVE service action (see 6.12) to the target port identified by the DESTINATION TARGET DESCRIPTOR INDEX field.

The DESCRIPTOR LENGTH field shall contain the length in bytes of the fields that follow the DESCRIPTOR LENGTH field. The value in the DESCRIPTOR LENGTH field shall be a multiple of 4.

If the PERIPHERAL DEVICE TYPE field in the target descriptor identified by the DESTINATION TARGET DESCRIPTOR INDEX field does not contain 01h, the copy manager shall terminate the command with CHECK CONDITION status, with the sense key set to COPY ABORTED, and the additional sense code set to INVALID OPERATION FOR COPY SOURCE OR DESTINATION.

Bytes 8 through n of the segment descriptor shall be sent as the parameter list (see 6.12.4) for the PERSISTENT RESERVE OUT command with REGISTER AND MOVE service action.

For a description of the RESERVATION KEY field, SERVICE ACTION RESERVATION KEY field, UNREG bit, APTPL bit, RELATIVE TARGET PORT IDENTIFIER field, TRANSPORTID DESCRIPTOR LENGTH field, and TransportID, see 6.12.4.

6.4 INQUIRY command

6.4.1 INQUIRY command introduction

The INQUIRY command (see table 80) requests that information regarding the logical unit and SCSI target device be sent to the application client.

Table 80 — INQUIRY command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (12h)							
1	Reserved						Obsolete	EVPD
2	PAGE CODE							
3	(MSB)	ALLOCATION LENGTH						
4								(LSB)
5	CONTROL							

An enable vital product data (EVPD) bit set to one specifies that the device server shall return the vital product data specified by the PAGE CODE field (see 6.4.4).

If the EVPD bit is set to zero, the device server shall return the standard INQUIRY data (see 6.4.2). If the PAGE CODE field is not set to zero when the EVPD bit is set to zero, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

When the EVPD bit is set to one, the PAGE CODE field specifies which page of vital product data information the device server shall return (see 7.6).

The ALLOCATION LENGTH field is defined in 4.3.4.6. If EVPD is set to zero, the allocation length should be at least five, so that the ADDITIONAL LENGTH field in the parameter data (see 6.4.2) is returned. If EVPD is set to one, the allocation length should be should be at least four, so that the PAGE LENGTH field in the parameter data (see 7.6) is returned.

In response to an INQUIRY command received by an incorrect logical unit, the SCSI target device shall return the INQUIRY data with the peripheral qualifier set to the value defined in 6.4.2. The INQUIRY command shall return CHECK CONDITION status only when the device server is unable to return the requested INQUIRY data.

If an INQUIRY command is received from an initiator port with a pending unit attention condition (i.e., before the device server reports CHECK CONDITION status), the device server shall perform the INQUIRY command and shall not clear the unit attention condition (see SAM-3).

The INQUIRY data should be returned even though the device server is not ready for other commands. The standard INQUIRY data should be available without incurring any media access delays. If the device server does store some of the standard INQUIRY data or VPD data on the media, it may return ASCII spaces (20h) in ASCII fields and zeros in other fields until the data is available from the media.

The INQUIRY data may change as the SCSI target device and its logical units perform their initialization sequence. (E.g., logical units may provide a minimum command set from nonvolatile memory until they load the final firmware from the media. After the firmware has been loaded, more options may be supported and therefore different INQUIRY data may be returned.)

If the INQUIRY data changes for any reason, the device server shall establish a unit attention condition for the initiator port associated with every I_T nexus (see SAM-3), with the additional sense code set to INQUIRY DATA HAS CHANGED.

NOTE 20 - The INQUIRY command may be used by an application client after a hard reset or power on condition to determine the device types for system configuration.

6.4.2 Standard INQUIRY data

The standard INQUIRY data (see table 81) shall contain at least 36 bytes.

Table 81 — Standard INQUIRY data format

Bit / Byte	7	6	5	4	3	2	1	0
0	PERIPHERAL QUALIFIER			PERIPHERAL DEVICE TYPE				
1	RMB	Reserved						
2	VERSION							
3	Obsolete	Obsolete	NORMACA	HISUP	RESPONSE DATA FORMAT			
4	ADDITIONAL LENGTH (n-4)							
5	SCCS	ACC	TPGS		3PC	Reserved		PROTECT
6	BQUE	ENCSERV	VS	MULTIP	MCHNGR	Obsolete	Obsolete	ADDR16[a]
7	Obsolete	Obsolete	WBUS16[a]	SYNC[a]	LINKED	Obsolete	CMDQUE	VS
8	(MSB)	T10 VENDOR IDENTIFICATION						
15								(LSB)
16	(MSB)	PRODUCT IDENTIFICATION						
31								(LSB)
32	(MSB)	PRODUCT REVISION LEVEL						
35								(LSB)
36	Vendor specific							
55								
56	Reserved				CLOCKING[a]		QAS[a]	IUS[a]
57	Reserved							
58	(MSB)	VERSION DESCRIPTOR 1						
59								(LSB)
	. . .							
72	(MSB)	VERSION DESCRIPTOR 8						
73								(LSB)
74	Reserved							
95								
	Vendor specific parameters							
96	Vendor specific							
n								

[a] The meanings of these fields are specific to SPI-5 (see 6.4.3). For SCSI transport protocols other than the SCSI Parallel Interface, these fields are reserved.

The PERIPHERAL QUALIFIER field and PERIPHERAL DEVICE TYPE field identify the peripheral device connected to the logical unit. If the SCSI target device is not capable of supporting a peripheral device connected to this logical unit, the device server shall set these fields to 7Fh (i.e., PERIPHERAL QUALIFIER field set to 011b and PERIPHERAL DEVICE TYPE field set to 1Fh).

The peripheral qualifier is defined in table 82 and the peripheral device type is defined in table 83.

Table 82 — Peripheral qualifier

Qualifier	Description
000b	A peripheral device having the specified peripheral device type is connected to this logical unit. If the device server is unable to determine whether or not a peripheral device is connected, it also shall use this peripheral qualifier. This peripheral qualifier does not mean that the peripheral device connected to the logical unit is ready for access.
001b	A peripheral device having the specified peripheral device type is not connected to this logical unit. However, the device server is capable of supporting the specified peripheral device type on this logical unit.
010b	Reserved
011b	The device server is not capable of supporting a peripheral device on this logical unit. For this peripheral qualifier the peripheral device type shall be set to 1Fh. All other peripheral device type values are reserved for this peripheral qualifier.
100b - 111b	Vendor specific

Table 83 — Peripheral device type (part 1 of 2)

Code	Doc. [a]	Description
00h	SBC-2	Direct access block device (e.g., magnetic disk)
01h	SSC-2	Sequential-access device (e.g., magnetic tape)
02h	SSC	Printer device
03h	SPC-2	Processor device
04h	SBC	Write-once device (e.g., some optical disks)
05h	MMC-4	CD/DVD device
06h		Scanner device (obsolete)
07h	SBC	Optical memory device (e.g., some optical disks)
08h	SMC-2	Medium changer device (e.g., jukeboxes)
09h		Communications device (obsolete)
0Ah - 0Bh		Obsolete
0Ch	SCC-2	Storage array controller device (e.g., RAID)
0Dh	SES	Enclosure services device
0Eh	RBC	Simplified direct-access device (e.g., magnetic disk)
0Fh	OCRW	Optical card reader/writer device
10h	BCC	Bridge Controller Commands
11h	OSD	Object-based Storage Device
12h	ADC	Automation/Drive Interface
13h - 1Dh		Reserved

[a] All standards are subject to revision, and parties to agreements based on this standard are encouraged to investigate the possibility of applying the most recent editions of the listed standards.

[b] All well known logical units use the same peripheral device type code.

Table 83 — Peripheral device type (part 2 of 2)

Code	Doc. [a]	Description
1Eh		Well known logical unit [b]
1Fh		Unknown or no device type

[a] All standards are subject to revision, and parties to agreements based on this standard are encouraged to investigate the possibility of applying the most recent editions of the listed standards.

[b] All well known logical units use the same peripheral device type code.

A removable medium (RMB) bit set to zero indicates that the medium is not removable. A RMB bit set to one indicates that the medium is removable.

The VERSION field indicates the implemented version of this standard and is defined in table 84.

Table 84 — Version

Code	Description		
00h	The device does not claim conformance to any standard.		
02h	Obsolete		
03h	The device complies to ANSI INCITS 301-1997 (SPC).		
04h	The device complies to ANSI INCITS 351-2001 (SPC-2).		
05h	The device complies to this standard.		
Code	**Description**	**Code**	**Description**
01h	Obsolete (SCSI=001b)	06h - 07h	Reserved
08h - 0Ch	Obsolete (ECMA=001b)	0Dh - 3Fh	Reserved
40h - 44h	Obsolete (ISO=01b)	45h - 47h	Reserved
48h - 4Ch	Obsolete (ISO=01b & ECMA=001b)	4Dh - 7Fh	Reserved
80h - 84h	Obsolete (ISO=10b)	85h - 87h	Reserved
88h - 8Ch	Obsolete (ECMA=001b)	8Dh - FFh	Reserved

The Normal ACA Supported (NORMACA) bit set to one indicates that the device server supports a NACA bit set to one in the CDB CONTROL byte and supports the ACA task attribute (see SAM-3). A NORMACA bit set to zero indicates that the device server does not support a NACA bit set to one and does not support the ACA task attribute.

A hierarchical support (HISUP) bit set to zero indicates the SCSI target device does not use the hierarchical addressing model to assign LUNs to logical units. A HISUP bit set to one indicates the SCSI target device uses the hierarchical addressing model to assign LUNs to logical units.

A RESPONSE DATA FORMAT field value of two indicates that the data shall be in the format defined in this standard. Response data format values less than two are obsolete. Response data format values greater than two are reserved.

The ADDITIONAL LENGTH field indicates the length in bytes of the remaining standard INQUIRY data. The relationship between the ADDITIONAL LENGTH field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6.

An SCC Supported (SCCS) bit set to one indicates that the SCSI target device contains an embedded storage array controller component. See SCC-2 for details about storage array controller devices. An SCCS bit set to zero indicates that the SCSI target device does not contain an embedded storage array controller component.

An Access Controls Coordinator (ACC) bit set to one indicates that the SCSI target device contains an access controls coordinator (see 3.1.4) that may be addressed through this logical unit. An ACC bit set to zero indicates that no access controls coordinator may be addressed through this logical unit. If the SCSI target device contains an

access controls coordinator that may be addressed through any logical unit other than the ACCESS CONTROLS well known logical unit (see 8.3), then the ACC bit shall be set to one for LUN 0.

The contents of the target port group support (TPGS) field (see table 85) indicate the support for asymmetric logical unit access (see 5.8).

Table 85 — TPGS field

Code	Description
00b	The SCSI target device does not support asymmetric logical unit access or supports a form of asymmetric access that is vendor specific. Neither the REPORT TARGET GROUPS nor the SET TARGET GROUPS commands is supported.
01b	Only implicit asymmetric logical unit access (see 5.8.2.7) is supported. The SCSI target device is capable of changing target port asymmetric access states without a SET TARGET PORT GROUPS command. The REPORT TARGET PORT GROUPS command is supported and the SET TARGET PORT GROUPS command is not supported.
10b	Only explicit asymmetric logical unit access (see 5.8.2.8) is supported. The SCSI target device only changes target port asymmetric access states as requested with the SET TARGET PORT GROUPS command. Both the REPORT TARGET PORT GROUPS command and the SET TARGET PORT GROUPS command are supported.
11b	Both explicit and implicit asymmetric logical unit access are supported. Both the REPORT TARGET PORT GROUPS command and the SET TARGET PORT GROUPS commands are supported.

A Third-Party Copy (3PC) bit set to one indicates that the SCSI target device supports third-party copy commands such as the EXTENDED COPY command (see 6.3). A 3PC bit set to zero indicates that the SCSI target device does not support such commands.

A PROTECT bit set to zero indicates that the logical unit does not support protection information (see 7.6.4 and SBC-2). A PROTECT bit set to one indicates that the logical unit supports protection information.

The BQUE bit combines with the CMDQUE bit to indicate whether the logical unit supports the full task management model or the basic task management model as described in table 86.

An Enclosure Services (ENCSERV) bit set to one indicates that the SCSI target device contains an embedded enclosure services component. See SES for details about enclosure services, including a device model for an embedded enclosure services device. An ENCSERV bit set to zero indicates that the SCSI target device does not contain an embedded enclosure services component.

A Multi Port (MULTIP) bit set to one indicates that this is a multi-port (two or more ports) SCSI target device and conforms to the SCSI multi-port device requirements found in the applicable standards (e.g., SAM-3, a SCSI transport protocol standard and possibly provisions of a command standard). A MULTIP bit set to zero indicates that this SCSI target device has a single port and does not implement the multi-port requirements.

A medium changer (MCHNGR) bit set to one indicates that the SCSI target device supports commands to control an attached media changer. See 5.10 and SMC-2 for details about medium changers, including a device model for an attached medium changer device. The MCHNGR bit is valid only when the RMB bit is equal to one. A MCHNGR bit set to zero indicates that the SCSI target device does not support commands to control an attached media changer.

A linked command (LINKED) bit set to one indicates that the device server supports linked commands (see SAM-3). A LINKED bit set to zero indicates the device server does not support linked commands.

The CMDQUE bit and BQUE bit indicate whether the logical unit supports the full task management model (see SAM-3) or the basic task management model (see SAM-3) as described in table 86.

Table 86 — BQUE and CMDQUE bits definition

BQUE	CMDQUE	Description
0	0	Obsolete
0	1	Full task management model supported
1	0	Basic task management model supported
1	1	Illegal combination of BQUE and CMDQUE bits

The T10 VENDOR IDENTIFICATION field contains eight bytes of left-aligned ASCII data (see 4.4.1) identifying the vendor of the product. The T10 vendor identification shall be one assigned by INCITS. A list of assigned T10 vendor identifications is in Annex E and on the T10 web site (http://www.t10.org).

NOTE 21 - The T10 web site (http://www.t10.org) provides a convenient means to request an identification code.

The PRODUCT IDENTIFICATION field contains sixteen bytes of left-aligned ASCII data (see 4.4.1) defined by the vendor.

The PRODUCT REVISION LEVEL field contains four bytes of left-aligned ASCII data defined by the vendor.

The VERSION DESCRIPTOR fields provide for identifying up to eight standards to which the SCSI target device claims conformance. The value in each VERSION DESCRIPTOR field shall be selected from those listed at http://www.t10.org/lists/2stds.html. All version descriptor values not listed are reserved. In the event that the T10 world wide web site is no longer active, access may be possible via the INCITS world wide web site (http://www.incits.org), the ANSI world wide web site (http://www.ansi.org), the IEC site (http://www.iec.ch/), the ISO site (http://www.iso.ch/), or the ISO/IEC JTC 1 web site (http://www.jtc1.org/). It is recommended that the first version descriptor be used for the SCSI architecture standard, followed by the physical transport standard if any, followed by the SCSI transport protocol standard, followed by the appropriate SPC version, followed by the device type command set, followed by a secondary command set if any.

6.4.3 SCSI Parallel Interface specific INQUIRY data

Portions of bytes 6 and 7 and all of byte 56 of the standard INQUIRY data shall be used only by SCSI target devices that implement the SCSI Parallel Interface. These fields are noted in table 81. For details on how the SPI-specific fields relate to the SCSI Parallel Interface see SPI-n (where n is 2 or greater). Table 87 shows just the SPI-specific standard INQUIRY fields. The definitions of the SCSI Parallel Interface specific fields shall be as follows.

Table 87 — SPI-specific standard INQUIRY bits

Bit / Byte	7	6	5	4	3	2	1	0
6	see table 81							ADDR16
7	see table 81		WBUS16	SYNC	see table 81	Obsolete	see table 81	
	. . .							
56	Reserved				CLOCKING		QAS	IUS

A wide SCSI address 16 (ADDR16) bit of one indicates that the SCSI target device supports 16-bit wide SCSI addresses. A value of zero indicates that the SCSI target device does not support 16-bit wide SCSI addresses.

A wide bus 16 (WBUS16) bit of one indicates that the SCSI target device supports 16-bit wide data transfers. A value of zero indicates that the SCSI target device does not support 16-bit wide data transfers.

A synchronous transfer (SYNC) bit of one indicates that the SCSI target device supports synchronous data transfer. A value of zero indicates the SCSI target device does not support synchronous data transfer.

The obsolete bit 2 in byte 7 indicates whether the SCSI target device supports an obsolete data transfers management mechanism defined in SPI-2.

Table 88 defines the relationships between the ADDR16 and WBUS16 bits.

Table 88 — Maximum logical device configuration table

ADDR16	WBUS16	Description
0	0	8 bit wide data path on a single cable with 8 SCSI IDs supported
0	1	16 bit wide data path on a single cable with 8 SCSI IDs supported
1	1	16 bit wide data path on a single cable with 16 SCSI IDs supported

The CLOCKING field shall not apply to asynchronous transfers and is defined in table 89.

Table 89 — CLOCKING field

Code	Description
00b	Indicates the target port supports only ST
01b	Indicates the target port supports only DT
10b	Reserved
11b	Indicates the target port supports ST and DT

A quick arbitration and selection supported (QAS) bit of one indicates that the target port supports quick arbitration and selection. A value of zero indicates that the target port does not support quick arbitration and selection.

An information units supported (IUS) bit of one indicates that the SCSI target device supports information unit transfers. A value of zero indicates that the SCSI target device does not support information unit transfers.

NOTE 22 - The acronyms ST and DT and the terms 'quick arbitration and selection' and 'information units' are defined in SPI-5.

6.4.4 Vital product data

The application client requests the vital product data information by setting the EVPD bit to one and specifying the page code of a vital product data. See 7.6 for details about vital product data. The information returned consists of configuration data (e.g., vendor identification, product identification, model, serial number), manufacturing data (e.g., plant and date of manufacture), field replaceable unit data and other vendor specific or device specific data. If the device server does not implement the requested page, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

The device server should have the ability to process the INQUIRY command even when an error occurs that prohibits normal command completion. In such a case, CHECK CONDITION status should be returned for commands other than INQUIRY or REQUEST SENSE. The sense data returned may contain the field replaceable unit code. The vital product data may be obtained for the failing device using the INQUIRY command.

This standard defines a format that allows device-independent application client software to display the vital product data returned by the INQUIRY command. The contents of the data may be vendor specific, and may be unusable without detailed information about the device.

This standard does not define the location or method of storing the vital product data. The retrieval of the data may require completion of initialization operations within the device, that may induce delays before the data is available to the application client. Time-critical requirements are an implementation consideration and are not addressed in this standard.

6.5 LOG SELECT command

The LOG SELECT command (see table 90) provides a means for an application client to manage statistical information maintained by the SCSI target device about the SCSI target device or its logical units. Device servers that implement the LOG SELECT command shall also implement the LOG SENSE command. Structures in the form of log parameters within log pages are defined as a way to manage the log data. The LOG SELECT command provides for sending zero or more log pages via the Data-Out Buffer. This standard defines the format of the log pages, but does not define the conditions and events that are logged.

Table 90 — LOG SELECT command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (4Ch)							
1	Reserved						PCR	SP
2	PC		Reserved					
3	Reserved							
6								
7	(MSB)							
8	PARAMETER LIST LENGTH							(LSB)
9	CONTROL							

A parameter code reset (PCR) bit set to one and a parameter list length of zero shall cause all implemented parameters to be set to the vendor specific default values (e.g., zero). If the PCR bit is set to one and the parameter list length is greater than zero, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB. A PCR bit set to zero specifies that the log parameters shall not be reset.

A save parameters (SP) bit set to one specifies that after performing the specified LOG SELECT operation the device server shall save to nonvolatile memory all parameters identified as saveable by the DS bit in the log page (see 7.2). A SP bit set to zero specifies that parameters shall not be saved.

Saving of log parameters is optional and indicated for each log parameter by the DS bit in the log page. Log parameters also may be saved at vendor specific times subject to the TSD bit (see 7.2) in the log parameter and the GLTSD bit in the Control mode page (see 7.4.6). If the logical unit does not implement saved parameters for any log parameter and the SP bit is set to one, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

It is not an error to set the SP bit to one and to set the DS bit of a log parameter to one. In this case, the parameter value for that log parameter is not saved.

The page control (PC) field defines the type of parameter values to be selected. The PC field is defined in table 91.

Table 91 — Page control (PC) field

PC	LOG SELECT parameter values	LOG SENSE parameter values
00b	Current threshold values	Threshold values
01b	Current cumulative values	Cumulative values
10b	Default threshold values	Default threshold values
11b	Default cumulative values	Default cumulative values

The current cumulative values may be updated by the device server or by the application client using the LOG SELECT command to reflect the cumulative number of events experienced by the logical unit. Fields in the

parameter control byte (see 7.2) of each log parameter control the updating and saving of the current cumulative parameters.

The device server shall set the current threshold parameters to the default threshold values in response to a LOG SELECT command with the PC field set to 10b and the parameter list length field set to zero.

The device server shall set all cumulative parameters to their default values in response to a LOG SELECT command with the PC field set to 11b and the parameter list length field set to zero.

The current threshold value may only be modified by the application client via the LOG SELECT command. If the application client attempts to change current threshold values that are not available or not implemented for that log parameter, then the device server shall terminate the LOG SELECT command with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST. The saving of current threshold parameters and the criteria for the current threshold being met are controlled by bits in the parameter control byte (see 7.2).

NOTE 23 - Log pages or log parameters that are not available may become available at some later time (e.g., after the logical unit has become ready).

The PARAMETER LIST LENGTH field specifies the length in bytes of the parameter list that shall be located in the Data-Out Buffer. A parameter list length of zero specifies that no log pages shall be transferred. This condition shall not be considered an error. If an application client sends page codes or parameter codes within the parameter list that are reserved or not implemented by the logical unit, then the device server shall terminate the LOG SELECT command with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

If a parameter list length results in the truncation of any log parameter, the device server shall terminate the command with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST. The additional sense code should be set to PARAMETER LIST LENGTH ERROR or may be set to INVALID FIELD IN CDB.

The application client should send log pages in ascending order by page code value if the Data-Out Buffer contains multiple log pages. If the Data-Out Buffer contains multiple log parameters within a log page, then they should be sent in ascending order by parameter code value. If the application client sends log pages out of order or parameter codes out of order, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

NOTE 24 - Application clients should issue LOG SENSE commands prior to issuing LOG SELECT commands to determine supported log pages and page lengths.

The SCSI target device may provide independent sets of log parameters for each logical unit or for each combination of logical units and I_T nexuses. If the SCSI target device does not support independent sets of log parameters and any log parameters are changed that affect other I_T nexuses, then the device server shall establish a unit attention condition (see SAM-3) for the initiator port associated with every I_T nexus except the I_T nexus on which the LOG SELECT command was received, with the additional sense code set to LOG PARAMETERS CHANGED.

If an application client sends a log parameter that is not supported by the logical unit, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

Additional information about the LOG SELECT command is in Annex C.

6.6 LOG SENSE command

The LOG SENSE command (see table 92) provides a means for the application client to retrieve statistical or other operational information maintained by the SCSI target device about the SCSI target device or its logical units. It is a complementary command to the LOG SELECT command.

Table 92 — LOG SENSE command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (4Dh)							
1	Reserved						PPC	SP
2	PC		PAGE CODE					
3	Reserved							
4								
5	(MSB)	PARAMETER POINTER						
6								(LSB)
7	(MSB)	ALLOCATION LENGTH						
8								(LSB)
9	CONTROL							

The parameter pointer control (PPC) bit controls the type of parameters requested from the device server:

a) A PPC bit set to one specifies that the device server shall return a log page with parameter code values that have changed since the last LOG SELECT or LOG SENSE command. The device server shall return only those parameter codes that are greater than or equal to the contents of the PARAMETER POINTER field in ascending order of parameter codes from the specified log page;
b) A PPC bit set to zero specifies that the device server shall return those parameter codes that are greater than or equal to the contents of the PARAMETER POINTER field in ascending order of parameter codes from the specified log page; and
c) A PPC bit set to zero and a PARAMETER POINTER field set to zero specifies that the device server shall return all available log parameters from the specified log page.

Saving parameters is an optional function of the LOG SENSE command. If the logical unit does not implement saving log parameters and if the save parameters (SP) bit is set to one, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

An SP bit set to zero specifies the device server shall perform the specified LOG SENSE command and shall not save any log parameters. If saving log parameters is implemented, an SP bit set to one specifies that the device server shall perform the specified LOG SENSE command and shall save all log parameters identified as saveable by the DS bit (see 7.2) to a nonvolatile, vendor specific location.

The page control (PC) field specifies the type of parameter values to be selected (see 6.5 for the definition of the page control field). The parameter values returned by a LOG SENSE command are determined as follows:

a) The specified parameter values at the last update (i.e., in response to a LOG SELECT or LOG SENSE command or done automatically by the device server for cumulative values);
b) The saved values, if saved parameters are implemented and an update has not occurred since the last logical unit reset; or
c) The default values, if saved values are not available or not implemented and an update has not occurred since the last logical unit reset.

The PAGE CODE field specifies which log page of data is being requested (see 7.2). If the log page code is reserved or not implemented, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

The PARAMETER POINTER field allows the application client to request parameter data beginning from a specific parameter code to the maximum allocation length or the maximum parameter code supported by the logical unit, whichever is less. If the value of the PARAMETER POINTER field is larger than the largest available parameter code known to the device server for the specified log page, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

The ALLOCATION LENGTH field is defined in 4.3.4.6.

Log parameters within the specified log page shall be transferred in ascending order according to parameter code.

Additional information about the LOG SENSE command is in Annex C.

6.7 MODE SELECT(6) command

The MODE SELECT(6) command (see table 93) provides a means for the application client to specify medium, logical unit, or peripheral device parameters to the device server. Device servers that implement the MODE SELECT(6) command shall also implement the MODE SENSE(6) command. Application clients should issue MODE SENSE(6) prior to each MODE SELECT(6) to determine supported mode pages, page lengths, and other parameters.

Table 93 — MODE SELECT(6) command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (15h)							
1	Reserved			PF	Reserved			SP
2	Reserved							
3								
4	PARAMETER LIST LENGTH							
5	CONTROL							

Logical units shall share mode parameter header and block descriptor values across all I_T nexuses. I_T nexus loss shall not affect mode parameter header, block descriptor, and mode page values.

Logical units shall maintain current and saved values of each mode page based on any of the policies listed in table 94. The mode page policy used for each mode page may be reported in the Mode Page Policy VPD page (see 7.6.6).

Table 94 — Mode page policies

Mode page policy	Number of mode page copies
Shared	One copy of the mode page that is shared by all I_T nexuses.
Per target port	A separate copy of the mode page for each target port with each copy shared by all initiator ports.
Per I_T nexus	A separate copy of the mode page for each I_T nexus

After a logical unit reset, each mode parameter header, block descriptor, and mode page shall revert to saved values if supported or default values if saved values are not supported.

If an application client sends a MODE SELECT command that changes any parameters applying to other I_T nexuses, the device server shall establish a unit attention (see SAM-3) condition for the initiator port associated with every I_T nexus except the I_T nexus on which the MODE SELECT command was received, with the additional sense code set to MODE PARAMETERS CHANGED.

A page format (PF) bit set to zero specifies that all parameters after the block descriptors are vendor specific. A PF bit set to one specifies that the MODE SELECT parameters following the header and block descriptor(s) are structured as pages of related parameters and are as defined in this standard.

A save pages (SP) bit set to zero specifies that the device server shall perform the specified MODE SELECT operation, and shall not save any mode pages. If the logical unit implements no distinction between current and saved mode pages and the SP bit is set to zero, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB. An SP bit set to one specifies that the device server shall perform the specified MODE SELECT operation, and shall save to a nonvolatile vendor specific location all the saveable mode pages including any sent in the Data-Out Buffer. Mode pages that are saved are specified by the parameter saveable (PS) bit that is returned in the first byte of each

mode page by the MODE SENSE command (see 7.4). If the PS bit is set to one in the MODE SENSE data, then the mode page shall be saveable by issuing a MODE SELECT command with the SP bit set to one. If the logical unit does not implement saved mode pages and the SP bit is set to one, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

The PARAMETER LIST LENGTH field specifies the length in bytes of the mode parameter list that shall be contained in the Data-Out Buffer. A parameter list length of zero specifies that the Data-Out Buffer shall be empty. This condition shall not be considered as an error.

If the parameter list length results in the truncation of any mode parameter header, mode parameter block descriptor(s), or mode page, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to PARAMETER LIST LENGTH ERROR.

The mode parameter list for the MODE SELECT and MODE SENSE commands is defined in 7.4. Parts of each mode parameter list are defined in a device-type dependent manner. Definitions for the parts of each mode parameter list that are unique for each device-type may be found in the applicable command standards (see 3.1.18).

The device server shall terminate the MODE SELECT command with CHECK CONDITION status, set the sense key to ILLEGAL REQUEST, set the additional sense code to INVALID FIELD IN PARAMETER LIST, and shall not change any mode parameters in response to any of the following conditions:

a) If the application client sets any field that is reported as not changeable by the device server to a value other than its current value;
b) If the application client sets any field in the mode parameter header or block descriptor(s) to an unsupported value;
c) If an application client sends a mode page with a page length not equal to the page length returned by the MODE SENSE command for that mode page;
d) If the application client sends an unsupported value for a mode parameter and rounding is not implemented for that mode parameter; or
e) If the application client sets any reserved field in the mode parameter list to a non-zero value and the device server checks reserved fields.

If the application client sends a value for a mode parameter that is outside the range supported by the device server and rounding is implemented for that mode parameter, the device server handles the condition by either:

a) Rounding the parameter to an acceptable value and terminating the command as described in 5.4; or
b) Terminating the command with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

A device server may alter any mode parameter in any mode page, even those reported as non-changeable, as a result of changes to other mode parameters.

The device server validates the non-changeable mode parameters against the current values that existed for those mode parameters prior to the MODE SELECT command.

NOTE 25 - The current values calculated by the device server may affect the application client's operation. The application client may issue a MODE SENSE command after each MODE SELECT command, to determine the current values.

6.8 MODE SELECT(10) command

The MODE SELECT(10) command (see table 95) provides a means for the application client to specify medium, logical unit, or peripheral device parameters to the device server. See the MODE SELECT(6) command (6.7) for a description of the fields and operation of this command. Application clients should issue MODE SENSE(10) prior to each MODE SELECT(10) to determine supported mode pages, page lengths, and other parameters. Device servers that implement the MODE SELECT(10) command shall also implement the MODE SENSE(10) command.

Table 95 — MODE SELECT(10) command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (55h)							
1	Reserved			PF	Reserved			SP
2	Reserved							
6								
7	(MSB)			PARAMETER LIST LENGTH				
8								(LSB)
9	CONTROL							

6.9 MODE SENSE(6) command

6.9.1 MODE SENSE(6) command introduction

The MODE SENSE(6) command (see table 96) provides a means for a device server to report parameters to an application client. It is a complementary command to the MODE SELECT(6) command. Device servers that implement the MODE SENSE(6) command shall also implement the MODE SELECT(6) command.

Table 96 — MODE SENSE(6) command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (1Ah)							
1	Reserved				DBD	Reserved		
2	PC		PAGE CODE					
3	SUBPAGE CODE							
4	ALLOCATION LENGTH							
5	CONTROL							

A disable block descriptors (DBD) bit set to zero specifies that the device server may return zero or more block descriptors in the returned MODE SENSE data (see 7.4). A DBD bit set to one specifies that the device server shall not return any block descriptors in the returned MODE SENSE data.

The page control (PC) field specifies the type of mode parameter values to be returned in the mode pages. The PC field is defined in table 97.

Table 97 — Page control (PC) field

Code	Type of parameter	Reference
00b	Current values	6.9.2
01b	Changeable values	6.9.3
10b	Default values	6.9.4
11b	Saved values	6.9.5

The PC field only affects the mode parameters within the mode pages, however the PS bit, SPF bit, PAGE CODE field, SUBPAGE CODE field, and PAGE LENGTH field should return current values (i.e., as if PC is set to 00b). The mode parameter header and mode parameter block descriptor should return current values.

Some SCSI target devices may not distinguish between current and saved mode parameters and report identical values in response to a PC field of either 00b or 11b. See also the description of the save pages (SP) bit in the MODE SELECT command.

The PAGE CODE and SUBPAGE CODE fields specify which mode pages and subpages to return (see table 98).

Table 98 — Mode page code usage for all devices

Page Code	Subpage Code	Description
00h	vendor specific	Vendor specific (does not require page format)
01h - 1Fh	00h	See specific device types (page_0 format)
	01h - DFh	See specific device types (sub_page format)
	E0h - FEh	Vendor specific (sub_page format)
	FFh	Return all subpages for the specified device specific mode page in the page_0 format for subpage 00h and in the sub_page format for subpages 01h - FEh
20h - 3Eh	00h	Vendor specific (page_0 format required)
	01h - FEh	Vendor specific (sub_page format required)
	FFh	Return all subpages for the specified vendor specific mode page in the page_0 format for subpage 00h and in the sub_page format for subpages 01h - FEh
3Fh	00h	Return all subpage 00h mode pages in page_0 format
	01h - FEh	Reserved
	FFh	Return all subpages for all mode pages in the page_0 format for subpage 00h and in the sub_page format for subpages 01h - FEh

The ALLOCATION LENGTH field is defined in 4.3.4.6.

An application client may request any one or all of the supported mode pages from the device server. If an application client issues a MODE SENSE command with a page code or subpage code value not implemented by the logical unit, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

If an application client requests all supported mode pages, the device server shall return the supported pages in ascending page code order beginning with mode page 01h. If mode page 00h is implemented, the device server shall return mode page 00h after all other mode pages have been returned.

If the PC field and the PAGE CODE field are both set to zero, the device server should return a mode parameter header and block descriptor, if applicable.

The mode parameter list for all device types for MODE SELECT and MODE SENSE is defined in 7.4. Parts of the mode parameter list are specifically defined for each device type. Definitions for the parts of each mode parameter list that are unique for each device-type may be found in the applicable command standards (see 3.1.18).

6.9.2 Current values

A PC field value of 00b requests that the device server return the current values of the mode parameters. The current values returned are:

a) The current values of the mode parameters established by the last successful MODE SELECT command;
b) The saved values of the mode parameters if a MODE SELECT command has not successfully completed since the mode parameters were restored to their saved values (see 6.7); or
c) The default values of the mode parameters if a MODE SELECT command has not successfully completed since the mode parameters were restored to their default values (see 6.7).

6.9.3 Changeable values

A PC field value of 01b requests that the device server return a mask denoting those mode parameters that are changeable. In the mask, the bits in the fields of the mode parameters that are changeable all shall be set to one and the bits in the fields of the mode parameters that are non-changeable (i.e., defined by the logical unit) all shall be set to zero.

If the logical unit does not implement changeable parameters mode pages and the device server receives a MODE SENSE command with 01b in the PC field, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

An attempt to change a non-changeable mode parameter using the MODE SELECT command shall result in an error condition (see 6.7).

The application client should issue a MODE SENSE command with the PC field set to 01b and the PAGE CODE field set to 3Fh to determine which mode pages are supported, which mode parameters within the mode pages are changeable, and the supported length of each mode page prior to issuing any MODE SELECT commands.

6.9.4 Default values

A PC field value of 10b requests that the device server return the default values of the mode parameters. Unsupported parameters shall be set to zero. Default values should be accessible even if the logical unit is not ready.

6.9.5 Saved values

A PC field value of 11b requests that the device server return the saved values of the mode parameters. Mode parameters not supported by the logical unit shall be set to zero. If saved values are not implemented, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to SAVING PARAMETERS NOT SUPPORTED.

The method of saving parameters is vendor specific. The parameters are preserved in such a manner that they are retained when the device is powered down. All saveable mode pages should be considered saved when a MODE SELECT command issued with the SP bit set to one has returned a GOOD status or after the successful completion of a FORMAT UNIT command.

6.9.6 Initial responses

After a logical unit reset, the device server shall respond in the following manner:

a) If default values are requested, report the default values;

b) If saved values are requested, report valid restored mode parameters, or restore the mode parameters and report them. If the saved values of the mode parameters are not able to be accessed from the nonvolatile vendor specific location, the command shall be terminated with CHECK CONDITION status, with the sense key set to NOT READY. If saved parameters are not implemented, respond as defined in 6.9.5; or
c) If current values are requested and the current values have been sent by the application client via a MODE SELECT command, the current values shall be returned. If the current values have not been sent, the device server shall return:
   A) The saved values, if saving is implemented and saved values are available; or
   B) The default values.

6.10 MODE SENSE(10) command

The MODE SENSE(10) command (see table 99) provides a means for a device server to report parameters to an application client. It is a complementary command to the MODE SELECT(10) command. Device servers that implement the MODE SENSE(10) command shall also implement the MODE SELECT(10) command.

Table 99 — MODE SENSE(10) command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (5Ah)							
1	Reserved			LLBAA	DBD	Reserved		
2	PC		PAGE CODE					
3	SUBPAGE CODE							
4	Reserved							
6								
7	(MSB)		ALLOCATION LENGTH					
8								(LSB)
9	CONTROL							

If the Long LBA Accepted (LLBAA) bit is set to one, the device server is allowed to return parameter data with the LONGLBA bit equal to one (see 7.4.3). If LLBAA bit is set to zero, the LONGLBA bit shall be zero in the parameter data returned by the device server.

See the MODE SENSE(6) command (6.9) for a description of the other fields and operation of this command.

6.11 PERSISTENT RESERVE IN command

6.11.1 PERSISTENT RESERVE IN command introduction

The PERSISTENT RESERVE IN command (see table 100) is used to obtain information about persistent reservations and reservation keys (i.e., registrations) that are active within a device server. This command is used in conjunction with the PERSISTENT RESERVE OUT command (see 6.12).

Table 100 — PERSISTENT RESERVE IN command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (5Eh)							
1	Reserved			SERVICE ACTION				
2	Reserved							
6								
7	(MSB)			ALLOCATION LENGTH				
8								(LSB)
9	CONTROL							

The ALLOCATION LENGTH field is defined in 4.3.4.6. The PERSISTENT RESERVE IN parameter data includes a length field that indicates the number of parameter data bytes available to be returned. The allocation length should be set to a value large enough to return the length field for the specified service action.

The service action codes for the PERSISTENT RESERVE IN command are defined in table 101.

Table 101 — PERSISTENT RESERVE IN service action codes

Code	Name	Description	Reference
00h	READ KEYS	Reads all registered reservation keys (i.e., registrations) as described in 5.6.5.2	6.11.2
01h	READ RESERVATION	Reads the current persistent reservations as described in 5.6.5.3	6.11.3
02h	REPORT CAPABILITIES	Returns capability information	6.11.4
03h	READ FULL STATUS	Reads complete information about all registrations and the persistent reservations, if any	6.11.5
04h - 1Fh	Reserved	Reserved	

6.11.2 READ KEYS service action

The READ KEYS service action requests that the device server return a parameter list containing a header and a list of each currently registered I_T nexus' reservation key. If multiple I_T nexuses have registered with the same key, then that key value shall be listed multiple times, once for each such registration.

For more information on READ KEYS see 5.6.5.2.

The format for the parameter data provided in response to a PERSISTENT RESERVE IN command with the READ KEYS service action is shown in table 102.

Table 102 — PERSISTENT RESERVE IN parameter data for READ KEYS

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)			PRGENERATION				
3								(LSB)
4	(MSB)			ADDITIONAL LENGTH (n-7)				
7								(LSB)
	Reservation key list							
8	(MSB)			First reservation key				
15								(LSB)
				⋮				
n-7	(MSB)			Last reservation key				
n								(LSB)

The Persistent Reservations Generation (PRGENERATION) field shall contain a 32-bit counter maintained by the device server that shall be incremented every time a PERSISTENT RESERVE OUT command requests a REGISTER service action, a REGISTER AND IGNORE EXISTING KEY service action, a REGISTER AND MOVE service action, a CLEAR service action, a PREEMPT service action, or a PREEMPT AND ABORT service action. The counter shall not be incremented by a PERSISTENT RESERVE IN command, by a PERSISTENT RESERVE OUT command that performs a RESERVE or RELEASE service action, or by a PERSISTENT RESERVE OUT command that is terminated due to an error or reservation conflict. Regardless of the APTPL bit value the PRgeneration value shall be set to zero by a power on.

The ADDITIONAL LENGTH field contains a count of the number of bytes in the Reservation key list. The relationship between the ADDITIONAL LENGTH field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6.

The reservation key list contains the 8-byte reservation keys for all I_T nexuses that have been registered (see 5.6.6).

6.11.3 READ RESERVATION service action

6.11.3.1 READ RESERVATION service action introduction

The READ RESERVATION service action requests that the device server return a parameter list containing a header and the persistent reservation, if any, that is present in the device server.

For more information on READ RESERVATION see 5.6.5.3.

6.11.3.2 Format of PERSISTENT RESERVE IN parameter data for READ RESERVATION

When no persistent reservation is held, the format for the parameter data provided in response to a PERSISTENT RESERVE IN command with the READ RESERVATION service action is shown in table 103.

Table 103 — PERSISTENT RESERVE IN parameter data for READ RESERVATION with no reservation held

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
3				PRGENERATION				(LSB)
4	(MSB)							
7				ADDITIONAL LENGTH (0)				(LSB)

The PRGENERATION field shall be as defined for the PERSISTENT RESERVE IN command with READ KEYS service action parameter data (see 6.11.2).

The ADDITIONAL LENGTH field shall be set to zero, indicating that no persistent reservation is held.

When a persistent reservation is held, the format for the parameter data provided in response to a PERSISTENT RESERVE IN command with the READ RESERVATION service action is shown in table 104.

Table 104 — PERSISTENT RESERVE IN parameter data for READ RESERVATION with reservation

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
3				PRGENERATION				(LSB)
4	(MSB)							
7				ADDITIONAL LENGTH (10h)				(LSB)
8	(MSB)							
15				RESERVATION KEY				(LSB)
16								
19				Obsolete				
20				Reserved				
21			SCOPE			TYPE		
22								
23				Obsolete				

The PRGENERATION field shall be as defined for the PERSISTENT RESERVE IN command with READ KEYS service action parameter data.

The ADDITIONAL LENGTH field contains a count of the number of bytes to follow and shall be set to 16. The relationship between the ADDITIONAL LENGTH field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6.

The RESERVATION KEY field shall contain the reservation key under which the persistent reservation is held (see 5.6.9).

The SCOPE field shall be set to LU_SCOPE (see 6.11.3.3).

The TYPE field shall contain the persistent reservation type (see 6.11.3.4) specified in the PERSISTENT RESERVE OUT command that created the persistent reservation.

The obsolete fields in bytes 16 through 19, byte 22, and byte 23 were defined in a previous standard.

6.11.3.3 Persistent reservations scope

The SCOPE field (see table 105) shall be set to LU_SCOPE, specifying that the persistent reservation applies to the entire logical unit.

Table 105 — Persistent reservation scope codes

Code	Name	Description
0h	LU_SCOPE	Persistent reservation applies to the full logical unit
1h - 2h		Obsolete
3h - Fh		Reserved

The LU_SCOPE scope shall be implemented by all device servers that implement PERSISTENT RESERVE OUT.

6.11.3.4 Persistent reservations type

The TYPE field (see table 106) specifies the characteristics of the persistent reservation being established for all logical blocks within the logical unit. Table 31 (see 5.6.1) defines the persistent reservation types under which each command defined in this standard is allowed to be processed. Each other command standard (see 3.1.18) defines the persistent reservation types under which each command defined in that command standard is allowed to be processed.

Table 106 — Persistent reservation type codes (part 1 of 2)

Code	Name	Description
0h		Obsolete
1h	Write Exclusive	**Access Restrictions**: Some commands (e.g., media-access write commands) are only allowed for the persistent reservation holder (see 5.6.9). **Persistent Reservation Holder:** There is only one persistent reservation holder.
2h		Obsolete
3h	Exclusive Access	**Access Restrictions**: Some commands (e.g., media-access commands) are only allowed for the persistent reservation holder (see 5.6.9). **Persistent Reservation Holder:** There is only one persistent reservation holder.
4h		Obsolete
5h	Write Exclusive – Registrants Only	**Access Restrictions**: Some commands (e.g., media-access write commands) are only allowed for registered I_T nexuses. **Persistent Reservation Holder:** There is only one persistent reservation holder (see 5.6.9).
6h	Exclusive Access – Registrants Only	**Access Restrictions**: Some commands (e.g., media-access commands) are only allowed for registered I_T nexuses. **Persistent Reservation Holder:** There is only one persistent reservation holder (see 5.6.9).

Table 106 — Persistent reservation type codes (part 2 of 2)

Code	Name	Description
7h	Write Exclusive – All Registrants	**Access Restrictions**: Some commands (e.g., media-access write commands) are only allowed for registered I_T nexuses. **Persistent Reservation Holder:** Each registered I_T nexus is a persistent reservation holder (see 5.6.9).
8h	Exclusive Access – All Registrants	**Access Restrictions**: Some commands (e.g., media-access commands) are only allowed for registered I_T nexuses. **Persistent Reservation Holder:** Each registered I_T nexus is a persistent reservation holder (see 5.6.9).
9h - Fh	Reserved	

6.11.4 REPORT CAPABILITIES service action

The REPORT CAPABILITIES service action requests that the device server return information on persistent reservation features.

The format for the parameter data provided in response to a PERSISTENT RESERVE IN command with the REPORT CAPABILITIES service action is shown in table 107.

Table 107 — PERSISTENT RESERVE IN parameter data for REPORT CAPABILITIES

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1				LENGTH (0008h)				(LSB)
2		Reserved		CRH	SIP_C	ATP_C	Reserved	PTPL_C
3	TMV			Reserved				PTPL_A
4								
5				PERSISTENT RESERVATION TYPE MASK				
6								
7				Reserved				

The LENGTH field indicates the length in bytes of the parameter data. The relationship between the LENGTH field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6.

A Compatible Reservation Handling (CRH) bit set to one indicates that the device server supports the exceptions to the SPC-2 RESERVE and RELEASE commands described in 5.6.3. A CRH bit set to zero indicates that RESERVE(6) command, RESERVE(10) command, RELEASE(6) command, and RELEASE(10) command are processed as defined in SPC-2.

A Specify Initiator Ports Capable (SIP_C) bit set to one indicates that the device server supports the SPEC_I_PT bit in the PERSISTENT RESERVE OUT command parameter data (see 6.12.3). An SIP_C bit set to zero indicates that the device server does not support the SPEC_I_PT bit in the PERSISTENT RESERVE OUT command parameter data.

An All Target Ports Capable (ATP_C) bit set to one indicates that the device server supports the ALL_TG_PT bit in the PERSISTENT RESERVE OUT command parameter data. An ATP_C bit set to zero indicates that the device server does not support the ALL_TG_PT bit in the PERSISTENT RESERVE OUT command parameter data.

A Persist Through Power Loss Capable (PTPL_C) bit set to one indicates that the device server supports the persist through power loss capability (see 5.6.4) for persistent reservations and the APTPL bit in the PERSISTENT RESERVE OUT command parameter data. An PTPL_C bit set to zero indicates that the device server does not support the persist through power loss capability.

A Type Mask Valid (TMV) bit set to one indicates that the PERSISTENT RESERVATION TYPE MASK field contains a bit map indicating which persistent reservation types are supported by the device server. A TMV bit set to zero indicates that the PERSISTENT RESERVATION TYPE MASK field shall be ignored.

A Persist Through Power Loss Activated (PTPL_A) bit set to one indicates that the persist through power loss capability is activated (see 5.6.4). A PTPL_A bit set to zero indicates that the persist through power loss capability is not activated.

The PERSISTENT RESERVATION TYPE MASK field (see table 108) contains a bit map that indicates the persistent reservation types that are supported by the device server.

Table 108 — Persistent Reservation Type Mask format

Bit / Byte	7	6	5	4	3	2	1	0
4	WR_EX_AR	EX_AC_RO	WR_EX_RO	Reserved	EX_AC	Reserved	WR_EX	Reserved
5	Reserved							EX_AC_AR

A Write Exclusive – All Registrants (WR_EX_AR) bit set to one indicates that the device server supports the Write Exclusive – All Registrants persistent reservation type. An WR_EX_AR bit set to zero indicates that the device server does not support the Write Exclusive – All Registrants persistent reservation type.

An Exclusive Access – Registrants Only (EX_AC_RO) bit set to one indicates that the device server supports the Exclusive Access – Registrants Only persistent reservation type. An EX_AC_RO bit set to zero indicates that the device server does not support the Exclusive Access – Registrants Only persistent reservation type.

A Write Exclusive – Registrants Only (WR_EX_RO) bit set to one indicates that the device server supports the Write Exclusive – Registrants Only persistent reservation type. An WR_EX_RO bit set to zero indicates that the device server does not support the Write Exclusive – Registrants Only persistent reservation type.

An Exclusive Access (EX_AC) bit set to one indicates that the device server supports the Exclusive Access persistent reservation type. An EX_AC bit set to zero indicates that the device server does not support the Exclusive Access persistent reservation type.

A Write Exclusive (WR_EX) bit set to one indicates that the device server supports the Write Exclusive persistent reservation type. An WR_EX bit set to zero indicates that the device server does not support the Write Exclusive persistent reservation type.

An Exclusive Access – All Registrants (EX_AC_AR) bit set to one indicates that the device server supports the Exclusive Access – All Registrants persistent reservation type. An EX_AC_AR bit set to zero indicates that the device server does not support the Exclusive Access – All Registrants persistent reservation type.

6.11.5 READ FULL STATUS service action

The READ FULL STATUS service action requests that the device server return a parameter list describing the registration and persistent reservation status of each currently registered I_T nexus for the logical unit.

For more information on READ FULL STATUS see 5.6.5.4.

The format for the parameter data provided in response to a PERSISTENT RESERVE IN command with the READ FULL STATUS service action is shown in table 109.

Table 109 — PERSISTENT RESERVE IN parameter data for READ FULL STATUS

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
3				PRGENERATION				(LSB)
4	(MSB)							
7				ADDITIONAL LENGTH (n-7)				(LSB)
	Full status descriptors							
8								
	First full status descriptor (see table 110)							
	. . .							
n	Last full status descriptor (see table 110)							

The PRGENERATION field shall be as defined for the PERSISTENT RESERVE IN command with READ KEYS service action parameter data (see 6.11.2).

The ADDITIONAL LENGTH field contains a count of the number of bytes to follow in the full status descriptors. The relationship between the ADDITIONAL LENGTH field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6.

The format of the full status descriptors is shown in table 110. Each full status descriptor describes one or more registered I_T nexuses. The device server shall return persistent reservations status information for every registered I_T nexus.

Table 110 — PERSISTENT RESERVE IN full status descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
7				RESERVATION KEY				(LSB)
8								
11	Reserved							
12	Reserved						ALL_TG_PT	R_HOLDER
13	SCOPE				TYPE			
14								
17	Reserved							
18	(MSB)							
19				RELATIVE TARGET PORT IDENTIFIER				(LSB)
20	(MSB)							
23				ADDITIONAL DESCRIPTOR LENGTH (n-23)				(LSB)
24								
n	TRANSPORTID							

The RESERVATION KEY field contains the reservation key.

A Reservation Holder (R_HOLDER) bit set to one indicates that all I_T nexuses described by this full status descriptor are registered and are persistent reservation holders (see 5.6.9). A R_HOLDER bit set to zero indicates that all I_T nexuses described by this full status descriptor are registered but are not persistent reservation holders.

An All Target Ports (ALL_TG_PT) bit set to zero indicates that this full status descriptor represents a single I_T nexus. An ALL_TG_PT bit set to one indicates that:

a) This full status descriptor represents all the I_T nexuses that are associated with both:
   A) The initiator port specified by the TRANSPORTID field; and
   B) Every target port in the SCSI target device;
b) All the I_T nexuses are registered with the same reservation key; and
c) All the I_T nexuses are either reservation holders or not reservation holders as indicated by the R_HOLDER bit.

The device server is not required to return an ALL_TG_PT bit set to one. Instead, it may return separate full status descriptors for each I_T nexus.

If the R_HOLDER bit is set to one (i.e., if the I_T nexus described by this full status descriptor is a reservation holder), the SCOPE field and the TYPE field are as defined in the READ RESERVATION service action parameter data (see 6.11.3). If the R_HOLDER bit is set to zero, the contents of the SCOPE field and the TYPE field are not defined by this standard.

If the ALL_TG_PT bit set to zero, the RELATIVE TARGET PORT IDENTIFIER field contains the relative port identifier (see 3.1.88) of the target port that is part of the I_T nexus described by this full status descriptor. If the ALL_TG_PT bit is set to one, the contents of the RELATIVE TARGET PORT IDENTIFIER field are not defined by this standard.

The ADDITIONAL DESCRIPTOR LENGTH field contains a count of the number of bytes that follow in the descriptor (i.e., the size of the TransportID).

The TRANSPORTID field contains a TransportID (see 7.5.4) identifying the initiator port that is part of the I_T nexus or I_T nexuses described by this full status descriptor.

6.12 PERSISTENT RESERVE OUT command

6.12.1 PERSISTENT RESERVE OUT command introduction

The PERSISTENT RESERVE OUT command (see table 111) is used to request service actions that reserve a logical unit for the exclusive or shared use of a particular I_T nexus. The command uses other service actions to manage and remove such persistent reservations.

I_T nexuses performing PERSISTENT RESERVE OUT service actions are identified by a registered reservation key provided by the application client. An application client may use the PERSISTENT RESERVE IN command to

obtain the reservation key, if any, for the I_T nexus holding a persistent reservation and may use the PERSISTENT RESERVE OUT command to preempt that persistent reservation.

Table 111 — PERSISTENT RESERVE OUT command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (5Fh)							
1	Reserved				SERVICE ACTION			
2	SCOPE				TYPE			
3	Reserved							
4								
5	(MSB)			PARAMETER LIST LENGTH				
8								(LSB)
9	CONTROL							

If a PERSISTENT RESERVE OUT command is attempted, but there are insufficient device server resources to complete the operation, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INSUFFICIENT REGISTRATION RESOURCES.

The PERSISTENT RESERVE OUT command contains fields that specify a persistent reservation service action, the intended scope of the persistent reservation, and the restrictions caused by the persistent reservation. The SCOPE field and TYPE field are defined in 6.11.3.3 and 6.11.3.4. If a SCOPE field specifies a scope that is not implemented, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

Fields contained in the PERSISTENT RESERVE OUT parameter list specify the information required to perform a particular persistent reservation service action.

The PARAMETER LIST LENGTH field specifies the number of bytes of parameter data for the PERSISTENT RESERVE OUT command.

The parameter list shall be 24 bytes in length and the PARAMETER LIST LENGTH field shall contain 24 (18h), if the following conditions are true:

a) The SPEC_I_PT bit (see 6.12.3) is set to zero; and
b) The service action is not REGISTER AND MOVE.

If the SPEC_I_PT bit is set to zero, the service action is not REGISTER AND MOVE, and the parameter list length is not 24, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to PARAMETER LIST LENGTH ERROR.

If the parameter list length is larger than the device server is able to process, the command should be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to PARAMETER LIST LENGTH ERROR.

6.12.2 PERSISTENT RESERVE OUT service actions

When processing the PERSISTENT RESERVE OUT service actions, the device server shall increment the PRgeneration value as specified in 6.11.2.

The PERSISTENT RESERVE OUT command service actions are defined in table 112.

Table 112 — PERSISTENT RESERVE OUT service action codes

Code	Name	Description	PRGENERATION field incremented (see 6.11.2)	Parameter list format
00h	REGISTER	Register a reservation key with the device server (see 5.6.6) or unregister a reservation key (see 5.6.10.3).	Yes	Basic (see 6.12.3)
01h	RESERVE	Creates a persistent reservation having a specified SCOPE and TYPE (see 5.6.8). The SCOPE and TYPE of a persistent reservation are defined in 6.11.3.3 and 6.11.3.4.	No	Basic (see 6.12.3)
02h	RELEASE	Releases the selected persistent reservation (see 5.6.10.2).	No	Basic (see 6.12.3)
03h	CLEAR	Clears all reservation keys (i.e., registrations) and all persistent reservations (see 5.6.10.6).	Yes	Basic (see 6.12.3)
04h	PREEMPT	Preempts persistent reservations and/or removes registrations (see 5.6.10.4).	Yes	Basic (see 6.12.3)
05h	PREEMPT AND ABORT	Preempts persistent reservations and/or removes registrations and aborts all tasks for all preempted I_T nexuses (see 5.6.10.4 and 5.6.10.5).	Yes	Basic (see 6.12.3)
06h	REGISTER AND IGNORE EXISTING KEY	Register a reservation key with the device server (see 5.6.6) or unregister a reservation key (see 5.6.10.3).	Yes	Basic (see 6.12.3)
07h	REGISTER AND MOVE	Register a reservation key for another I_T nexus with the device server and move a persistent reservation to that I_T nexus (see 5.6.7)	Yes	Register and move (see 6.12.4)
08h - 1Fh	Reserved			

6.12.3 Basic PERSISTENT RESERVE OUT parameter list

The parameter list format shown in table 113 shall be used by the PERSISTENT RESERVE OUT command with any service action except the REGISTER AND MOVE service action. All fields shall be sent, even if the field is not required for the specified service action and scope values.

Table 113 — PERSISTENT RESERVE OUT parameter list

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)			RESERVATION KEY				
7								(LSB)
8	(MSB)			SERVICE ACTION RESERVATION KEY				
15								(LSB)
16				Obsolete				
19								
20	Reserved				SPEC_I_PT	ALL_TG_PT	Reserved	APTPL
21	Reserved							
22				Obsolete				
23								
24				Additional parameter data				
n								

The obsolete fields in bytes 16 through 19, byte 22 and byte 23 were defined in a previous standard.

The RESERVATION KEY field contains an 8-byte value provided by the application client to the device server to identify the I_T nexus that is the source of the PERSISTENT RESERVE OUT command. The device server shall verify that the contents of the RESERVATION KEY field in a PERSISTENT RESERVE OUT command parameter data matches the registered reservation key for the I_T nexus from which the command was received, except for:

a) The REGISTER AND IGNORE EXISTING KEY service action where the RESERVATION KEY field shall be ignored; and
b) The REGISTER service action for an unregistered I_T nexus where the RESERVATION KEY field shall contain zero.

Except as noted above, when a PERSISTENT RESERVE OUT command specifies a RESERVATION KEY field other than the reservation key registered for the I_T nexus the device server shall return a RESERVATION CONFLICT status. Except as noted above, the reservation key of the I_T nexus shall be verified to be correct regardless of the SERVICE ACTION and SCOPE field values.

The SERVICE ACTION RESERVATION KEY field contains information needed for the following service actions: REGISTER, REGISTER AND IGNORE EXISTING KEY, PREEMPT, and PREEMPT AND ABORT. The SERVICE ACTION RESERVATION KEY field is ignored for the following service actions: RESERVE, RELEASE, and CLEAR.

For the REGISTER service action and REGISTER AND IGNORE EXISTING KEY service action, the SERVICE ACTION RESERVATION KEY field contains:

a) The new reservation key to be registered in place of the registered reservation key specified in the RESERVATION KEY field; or
b) Zero to unregister the registered reservation key specified in the RESERVATION KEY field.

For the PREEMPT service action and PREEMPT AND ABORT service action, the SERVICE ACTION RESERVATION KEY field contains the reservation key of:

a) The registrations to be removed; and
b) If the SERVICE ACTION RESERVATION KEY field identifies a persistent reservation holder (see 5.6.9), persistent reservations that are to be preempted.

If the Specify Initiator Ports (SPEC_I_PT) bit is set to zero, the device server shall apply the registration only to the I_T nexus that sent the PERSISTENT RESERVE OUT command. If the SPEC_I_PT bit is set to one for the REGISTER service action or the REGISTER AND IGNORE EXISTING KEY service action, then the additional parameter data shall include a list of transport IDs (see table 114) and the device server shall also apply the registration to the I_T nexus for each initiator port specified by a TransportID. If a registration fails for any initiator port (e.g., if the logical unit does not have enough resources available to hold the registration information), none of the other registrations shall be made.

Table 114 — PERSISTENT RESERVE OUT specify initiator ports additional parameter data

Bit / Byte	7	6	5	4	3	2	1	0
24	TRANSPORTID PARAMETER DATA LENGTH (n - 27)							
27								
	TransportIDs list							
28	First TransportID							
	. . .							
	Last TransportID							
n								

The TRANSPORTID PARAMETER DATA LENGTH field specifies the number of bytes of TransportIDs that follow.

The command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST:

a) If the value in the parameter list length field in the CDB does not include all of the additional parameter list bytes specified by the TRANSPORTID PARAMETER DATA LENGTH field; or
b) If the value in the TRANSPORTID PARAMETER DATA LENGTH field results in the truncation of a TransportID.

The format of a TransportID is specified in 7.5.4.

The All Target Ports (ALL_TG_PT) bit is valid only for the REGISTER service action and the REGISTER AND IGNORE EXISTING KEY service action, and shall be ignored for all other service actions. Support for the ALL_TG_PT bit is optional. If the device server receives a REGISTER service action or a REGISTER AND IGNORE EXISTING KEY service action with the ALL_TG_PT bit set to one, it shall create the specified registration on all target ports in the SCSI target device known to the device server (i.e., as if the same registration request had been received individually through each target port). If the device server receives a REGISTER service action or a REGISTER AND IGNORE EXISTING KEY service action with the ALL_TG_PT bit set to zero, it shall apply the registration only to the target port through which the PERSISTENT RESERVE OUT command was received.

The Activate Persist Through Power Loss (APTPL) bit is valid only for the REGISTER service action and the REGISTER AND IGNORE EXISTING KEY service action, and shall be ignored for all other service actions. Support for an APTPL bit equal to one is optional. If a device server that does not support an APTPL bit set to one receives that value in a REGISTER service action or a REGISTER AND IGNORE EXISTING KEY service action,

the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

If the last valid APTPL bit value received by the device server is zero, the loss of power in the SCSI target device shall release the persistent reservation for the logical unit and remove all registered reservation keys (see 5.6.6). If the last valid APTPL bit value received by the device server is one, the logical unit shall retain any persistent reservation(s) that may be present and all reservation keys (i.e., registrations) for all I_T nexuses even if power is lost and later returned (see 5.6.4).

Table 115 summarizes which fields are set by the application client and interpreted by the device server for each service action and scope value.

Table 115 — PERSISTENT RESERVE OUT service actions and valid parameters (part 1 of 2)

Service action	Allowed SCOPE	Parameters (part 1 of 2)			
		TYPE	RESERVATION KEY	SERVICE ACTION RESERVATION KEY	APTPL
REGISTER	ignored	ignored	valid	valid	valid
REGISTER AND IGNORE EXISTING KEY	ignored	ignored	ignored	valid	valid
RESERVE	LU_SCOPE	valid	valid	ignored	ignored
RELEASE	LU_SCOPE	valid	valid	ignored	ignored
CLEAR	ignored	ignored	valid	ignored	ignored
PREEMPT	LU_SCOPE	valid	valid	valid	ignored
PREEMPT AND ABORT	LU_SCOPE	valid	valid	valid	ignored
REGISTER AND MOVE	LU_SCOPE	valid	valid	valid	not applicable [a]

[a] The parameter list format for the REGISTER AND MOVE service action is described in 6.12.4.

Table 115 — PERSISTENT RESERVE OUT service actions and valid parameters (part 2 of 2)

Service action	Allowed SCOPE	Parameters (part 2 of 2)	
		ALL_TG_PT	SPEC_I_PT
REGISTER	ignored	valid	valid
REGISTER AND IGNORE EXISTING KEY	ignored	valid	valid
RESERVE	LU_SCOPE	ignored	ignored
RELEASE	LU_SCOPE	ignored	ignored
CLEAR	ignored	ignored	ignored
PREEMPT	LU_SCOPE	ignored	ignored
PREEMPT AND ABORT	LU_SCOPE	ignored	ignored
REGISTER AND MOVE	LU_SCOPE	not applicable [a]	not applicable [a]

[a] The parameter list format for the REGISTER AND MOVE service action is described in 6.12.4.

6.12.4 PERSISTENT RESERVE OUT command with REGISTER AND MOVE service action parameters

The parameter list format shown in table 116 shall be used by the PERSISTENT RESERVE OUT command with REGISTER AND MOVE service action.

Table 116 — PERSISTENT RESERVE OUT command with REGISTER AND MOVE service action parameter list

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)			RESERVATION KEY				
7								(LSB)
8	(MSB)			SERVICE ACTION RESERVATION KEY				
15								(LSB)
16				Reserved				
17				Reserved			UNREG	APTPL
18	(MSB)			RELATIVE TARGET PORT IDENTIFIER				
19								(LSB)
20	(MSB)			TRANSPORTID PARAMETER DATA LENGTH (n - 23)				
23								(LSB)
24				TransportID				
n								

The RESERVATION KEY field contains an 8-byte value provided by the application client to the device server to identify the I_T nexus that is the source of the PERSISTENT RESERVE OUT command. The device server shall verify that the contents of the RESERVATION KEY field in a PERSISTENT RESERVE OUT command parameter data

matches the registered reservation key for the I_T nexus from which the command was received. If a PERSISTENT RESERVE OUT command specifies a RESERVATION KEY field other than the reservation key registered for the I_T nexus, the device server shall return a RESERVATION CONFLICT status.

The SERVICE ACTION RESERVATION KEY field contains the reservation key to be registered to the specified I_T nexus.

The Activate Persist Through Power Loss (APTPL) bit set to one is optional. If a device server that does not support an APTPL bit set to one receives that value, it shall return CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

If the last valid APTPL bit value received by the device server is zero, the loss of power in the SCSI target device shall release the persistent reservation for the logical unit and remove all registered reservation keys (see 5.6.5). If the last valid APTPL bit value received by the device server is one, the logical unit shall retain any persistent reservation(s) that may be present and all reservation keys (i.e., registrations) for all I_T nexuses even if power is lost and later returned (see 5.6.4).

The unregister (UNREG) bit set to zero specifies that the device server shall not unregister the I_T nexus on which the PERSISTENT RESERVE OUT command REGISTER AND MOVE service action was received. An UNREG bit set to one specifies that the device server shall unregister the I_T nexus on which the PERSISTENT RESERVE OUT command REGISTER AND MOVE service action was received.

The RELATIVE TARGET PORT IDENTIFIER field specifies the relative port identifier (see 3.1.88) of the target port in the I_T nexus to which the persistent reservation is to be moved.

The TRANSPORTID DESCRIPTOR LENGTH field specifies the number of bytes of the TransportID that follows, shall be a minimum of 24 bytes, and shall be a multiple of 4.

The TransportID specifies the initiator port in the I_T nexus to which the persistent reservation is to be moved. The format of the TransportID is defined in 7.5.4.

The command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST:

a) If the value in the parameter list length field in the CDB does not include all of the parameter list bytes specified by the TRANSPORTID PARAMETER DATA LENGTH field; or
b) If the value in the TRANSPORTID PARAMETER DATA LENGTH field results in the truncation of a TransportID.

6.13 PREVENT ALLOW MEDIUM REMOVAL command

The PREVENT ALLOW MEDIUM REMOVAL command (see table 117) requests that the logical unit enable or disable the removal of the medium. The logical unit shall not allow medium removal if any initiator port currently has medium removal prevented.

Table 117 — PREVENT ALLOW MEDIUM REMOVAL command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (1Eh)							
1	Reserved							
3								
4	Reserved							PREVENT
5	CONTROL							

Table 118 defines the PREVENT field values and their meanings.

Table 118 — PREVENT field

PREVENT	Description
00b	Medium removal shall be allowed from both the data transport element and the attached medium changer, if any.
01b	Medium removal shall be prohibited from the data transport element but allowed from the attached medium changer, if any.
10b [a]	Medium removal shall be allowed for the data transport element but prohibited for the attached medium changer.
11b [a]	Medium removal shall be prohibited for both the data transport element and the attached medium changer.

[a] PREVENT values 10b and 11b are valid only when the RMB bit is set to one and the MCHNGR bit is set to one are both equal to one in the standard INQUIRY data (see 6.4.2).

The prevention of medium removal shall begin when any application client issues a PREVENT ALLOW MEDIUM REMOVAL command with a PREVENT field of 01b or 11b (i.e., medium removal prevented). The prevention of medium removal for the logical unit shall terminate after:

a) One of the following occurs for each I_T nexus that previously had medium removal prevented:
   A) Receipt of a PREVENT ALLOW MEDIUM REMOVAL command with a PREVENT field of 00b or 10b;
   B) An I_T nexus loss; or
b) A power on;
c) A hard reset; or
d) A a logical unit reset.

If possible, the device server shall perform an synchronize cache operation before terminating the prevention of medium removal.

If a persistent reservation or registration is being preempted by a PERSISTENT RESERVE OUT command with PREEMPT AND ABORT service action (see 5.6.10.5), the equivalent of a PREVENT ALLOW MEDIUM REMOVAL command with the PREVENT field set to zero shall be processed for each the I_T nexuses associated with the persistent reservation or registrations being preempted. This allows an application client to override the prevention of medium removal function for an initiator port that is no longer operating correctly.

While a prevention of medium removal condition is in effect, the logical unit shall inhibit mechanisms that normally allow removal of the medium by an operator.

6.14 READ ATTRIBUTE command

6.14.1 READ ATTRIBUTE command introduction

The READ ATTRIBUTE command (see table 119) allows an application client to read attribute values from medium auxiliary memory.

Table 119 — READ ATTRIBUTE command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (8Ch)							
1	Reserved			SERVICE ACTION				
2	Restricted (see SMC-2)							
4								
5	VOLUME NUMBER							
6	Reserved							
7	PARTITION NUMBER							
8	(MSB)			FIRST ATTRIBUTE IDENTIFIER				
9								(LSB)
10	(MSB)			ALLOCATION LENGTH				
13								(LSB)
14	Reserved							
15	CONTROL							

If the medium auxiliary memory is not accessible because there is no medium present, the READ ATTRIBUTE command shall be terminated with CHECK CONDITION status, with the sense key set to NOT READY, and the additional sense code set to MEDIUM NOT PRESENT.

If the medium is present but the medium auxiliary memory is not accessible, the READ ATTRIBUTE command shall be terminated with CHECK CONDITION status, with the sense key set to MEDIUM ERROR, and the additional sense code set to LOGICAL UNIT NOT READY, AUXILIARY MEMORY NOT ACCESSIBLE.

If the medium auxiliary memory is not operational, the READ ATTRIBUTE command shall be terminated with CHECK CONDITION status, with the sense key set to MEDIUM ERROR, and the additional sense code set to AUXILIARY MEMORY READ ERROR.

The service actions defined for the READ ATTRIBUTE command are shown in table 120.

Table 120 — READ ATTRIBUTE service action codes

Code	Name	Description	Subclause
00h	ATTRIBUTE VALUES	Return attribute values.	6.14.2
01h	ATTRIBUTE LIST	Return a list of available attribute identifiers, identifiers that are not in the nonexistent state or unsupported state (see 5.11).	6.14.3
02h	VOLUME LIST	Return a list of known volume numbers.	6.14.4
03h	PARTITION LIST	Return a list of known partition numbers.	6.14.5
04h	Restricted		
05h-1Fh	Reserved		

The VOLUME NUMBER field specifies a volume (see SSC-2) within the medium auxiliary memory. The number of volumes of the medium auxiliary memory shall equal that of the attached medium. If the medium only has a single volume, then its volume number shall be zero.

The PARTITION NUMBER field specifies a partition (see SSC-2) within a volume. The number of partitions of the medium auxiliary memory shall equal that of the attached medium. If the medium only has a single partition, then its partition number shall be zero.

If the combination of volume number and partition number is not valid, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

The FIRST ATTRIBUTE IDENTIFIER field specifies the attribute identifier of the first attribute to be returned. If the specified attribute is in the unsupported state or nonexistent state (see 5.11), the READ ATTRIBUTE command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

The ALLOCATION LENGTH field is defined in 4.3.4.6.

The format of parameter data returned by the READ ATTRIBUTE command depends on the service action specified.

6.14.2 ATTRIBUTE VALUES service action

The READ ATTRIBUTE command with ATTRIBUTE VALUES service action returns parameter data containing the attributes specified by the PARTITION NUMBER, VOLUME NUMBER, and FIRST ATTRIBUTE IDENTIFIER fields in the CDB.

The returned parameter data shall contain the requested attributes in ascending numerical order by attribute identifier value and in the format shown in table 121.

Table 121 — READ ATTRIBUTE with ATTRIBUTE VALUES service action parameter list format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
3				AVAILABLE DATA (n-3)				(LSB)
	Attribute(s)							
4				Attribute 0 (see 7.3.1)				
				. . .				
n				Attribute x (see 7.3.1)				

The AVAILABLE DATA field shall contain the number of bytes of attribute information in the parameter list. The relationship between the AVAILABLE DATA field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6.

The format of the attributes is described in 7.3.1.

6.14.3 ATTRIBUTE LIST service action

The READ ATTRIBUTE command with ATTRIBUTE LIST service action returns parameter data containing the attribute identifiers for the attributes that are not in the unsupported state and not in the nonexistent state (see 5.11) in the specified partition and volume number. The contents of FIRST ATTRIBUTE IDENTIFIER field in the CDB shall be ignored. The returned parameter data shall contain the requested attribute identifiers in ascending numerical order by attribute identifier value and in the format shown in table 122.

Table 122 — READ ATTRIBUTE with ATTRIBUTE LIST service action parameter list format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
3				AVAILABLE DATA (n-3)				(LSB)
	Attribute identifiers							
4				ATTRIBUTE IDENTIFIER 0				
5								
				. . .				
n-1				ATTRIBUTE IDENTIFIER x				
n								

The AVAILABLE DATA field shall contain the number of bytes of attribute identifiers in the parameter list. The relationship between the AVAILABLE DATA field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6.

An ATTRIBUTE IDENTIFIER field is returned for each attribute that is not in the unsupported state and not in the nonexistent state (see 5.11) in the specified partition and volume number. See 7.3.2 for a description of the attribute identifier values.

6.14.4 VOLUME LIST service action

The READ ATTRIBUTE command with VOLUME LIST service action returns parameter data (see table 123) identifying the supported number of volumes. The contents of VOLUME NUMBER, PARTITION NUMBER, and FIRST ATTRIBUTE IDENTIFIER fields in the CDB shall be ignored.

Table 123 — READ ATTRIBUTE with VOLUME LIST service action parameter list format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1				AVAILABLE DATA (0002h)				(LSB)
2				FIRST VOLUME NUMBER				
3				NUMBER OF VOLUMES AVAILABLE				

The AVAILABLE DATA field shall contain two. The relationship between the AVAILABLE DATA field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6.

The FIRST VOLUME NUMBER field indicates the first volume available. Volume numbering should start at zero.

The NUMBER OF VOLUMES AVAILABLE field indicates the number of volumes available.

6.14.5 PARTITION LIST service action

The READ ATTRIBUTE command with PARTITION LIST service action returns parameter data (see table 124) identifying the number of partitions supported in the specified volume number. The contents of PARTITION NUMBER and FIRST ATTRIBUTE IDENTIFIER fields in the CDB shall be ignored.

Table 124 — READ ATTRIBUTE with PARTITION LIST service action parameter list format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1				AVAILABLE DATA (0002h)				(LSB)
2				FIRST PARTITION NUMBER				
3				NUMBER OF PARTITIONS AVAILABLE				

The AVAILABLE DATA field shall contain two. The relationship between the AVAILABLE DATA field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6.

The FIRST PARTITION NUMBER field indicates the first partition available on the specified volume number. Partition numbering should start at zero.

The NUMBER OF PARTITIONS AVAILABLE field indicates the number of partitions available on the specified volume number.

6.15 READ BUFFER command

6.15.1 READ BUFFER command introduction

The READ BUFFER command (see table 125) is used in conjunction with the WRITE BUFFER command as a diagnostic function for testing memory in the SCSI device and the integrity of the service delivery subsystem. This command shall not alter the medium.

Table 125 — READ BUFFER command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (3Ch)							
1	Reserved			MODE				
2	BUFFER ID							
3	(MSB)			BUFFER OFFSET				
5								(LSB)
6	(MSB)			ALLOCATION LENGTH				
8								(LSB)
9	CONTROL							

The function of this command and the meaning of fields within the CDB depend on the contents of the MODE field. The MODE field is defined in table 126.

Table 126 — READ BUFFER MODE field

MODE	Description
00h	Combined header and data [a]
01h	Vendor specific [a]
02h	Data
03h	Descriptor
0Ah	Echo buffer
0Bh	Echo buffer descriptor
1Ah	Enable expander communications protocol and Echo buffer
04h - 09h	Reserved
0Ch - 19h	Reserved
1Bh - 1Fh	Reserved

[a] Modes 00h and 01h are not recommended.

If the mode is not set to one, the ALLOCATION LENGTH field is defined in 4.3.4.6.

6.15.2 Combined header and data mode (00h)

In this mode, a four-byte header followed by data bytes is returned to the application client in the Data-In Buffer. The allocation length should be set to four or greater. The BUFFER ID and the BUFFER OFFSET fields are reserved.

The four-byte READ BUFFER header (see table 127) is followed by data bytes from the buffer.

Table 127 — READ BUFFER header

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved							
1	(MSB)			BUFFER CAPACITY				
3								(LSB)
4				Data				
n								

The BUFFER CAPACITY field specifies the total number of data bytes available in the buffer. The buffer capacity is not reduced to reflect the actual number of bytes written using the WRITE BUFFER command. The relationship between the BUFFER CAPACITY field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6. Following the READ BUFFER header, the device server shall transfer data from the buffer.

6.15.3 Vendor specific mode (01h)

In this mode, the meanings of the BUFFER ID, BUFFER OFFSET, and ALLOCATION LENGTH fields are not specified by this standard.

6.15.4 Data mode (02h)

In this mode, the Data-In Buffer is filled only with logical unit buffer data. The BUFFER ID field specifies a buffer within the logical unit from which data shall be transferred. The vendor assigns buffer ID codes to buffers within the logical unit. Buffer ID zero shall be supported. If more than one buffer is supported, then additional buffer ID codes shall be assigned contiguously, beginning with one. Buffer ID code assignments for the READ BUFFER command shall be the same as for the WRITE BUFFER command. If an unsupported buffer ID code is selected, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

The BUFFER OFFSET field contains the byte offset within the specified buffer from which data shall be transferred. The application client should conform to the offset boundary requirements returned in the READ BUFFER descriptor (see 6.15.5). If the device server is unable to accept the specified buffer offset, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

6.15.5 Descriptor mode (03h)

In this mode, a maximum of four bytes of READ BUFFER descriptor information is returned. The device server shall return the descriptor information for the buffer specified by the BUFFER ID field (see the description of the buffer ID in 6.15.4). If there is no buffer associated with the specified buffer ID, the device server shall return all zeros in

the READ BUFFER descriptor. The BUFFER OFFSET field is reserved in this mode. The allocation length should be set to four or greater. The READ BUFFER descriptor is defined as shown in table 128.

Table 128 — READ BUFFER descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	OFFSET BOUNDARY							
1	BUFFER CAPACITY							
2								
3								

The OFFSET BOUNDARY field returns the boundary alignment within the selected buffer for subsequent WRITE BUFFER and READ BUFFER commands. The value contained in the OFFSET BOUNDARY field shall be interpreted as a power of two.

The value contained in the BUFFER OFFSET field of subsequent WRITE BUFFER and READ BUFFER commands should be a multiple of $2^{\text{offset boundary}}$ as shown in table 129.

Table 129 — Buffer offset boundary

Offset boundary	$2^{\text{Offset boundary}}$	Buffer offsets
0h	$2^0 = 1$	Byte boundaries
1h	$2^1 = 2$	Even-byte boundaries
2h	$2^2 = 4$	Four-byte boundaries
3h	$2^3 = 8$	Eight-byte boundaries
4h	$2^4 = 16$	16-byte boundaries
.	.	.
FFh	Not applicable	0 is the only supported buffer offset

The BUFFER CAPACITY field shall return the size of the selected buffer in bytes.

NOTE 26 - In a system employing multiple application clients, a buffer may be altered between the WRITE BUFFER and READ BUFFER commands by another application client. Buffer testing applications should ensure that only a single application client is active. Use of reservations to all logical units on the device or linked commands may be helpful in avoiding buffer alteration between these two commands.

6.15.6 Echo buffer mode (0Ah)

In this mode the device server transfers data to the application client from the echo buffer that was written by the most recent WRITE BUFFER command with the mode field set to echo buffer (see 6.35.9) received on the same I_T nexus. The READ BUFFER command shall return the same number of bytes of data as received in the prior WRITE BUFFER command with the mode field set to echo buffer, limited by the allocation length as described in 4.3.4.6.

The BUFFER ID and BUFFER OFFSET fields are ignored in this mode.

If no WRITE BUFFER command with the mode set to echo buffer received on this I_T nexus has completed without an error, then the READ BUFFER command shall terminate with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to COMMAND SEQUENCE ERROR. If the data in the echo buffer has been overwritten by another I_T nexus, the READ BUFFER command shall be terminated with CHECK CONDITION status, with the sense key set to ABORTED COMMAND, and the additional sense code set to ECHO BUFFER OVERWRITTEN.

After a WRITE BUFFER command with the mode set to echo buffer has completed without an error, the application client may send multiple READ BUFFER commands with the mode set to echo buffer in order to read the echo buffer data multiple times.

6.15.7 Echo buffer descriptor mode (0Bh)

In this mode, a maximum of four bytes of READ BUFFER descriptor information is returned. The device server shall return the descriptor information for the echo buffer. If there is no echo buffer implemented, the device server shall return all zeros in the READ BUFFER descriptor. The BUFFER ID field and BUFFER OFFSET field are reserved in this mode. The allocation length should be set to four or greater. The READ BUFFER descriptor is defined as shown in table 130.

Table 130 — Echo buffer descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved							EBOS
1	Reserved							
2	Reserved			(MSB)				
3	BUFFER CAPACITY							(LSB)

The BUFFER CAPACITY field shall return the size of the echo buffer in bytes aligned to a four-byte boundary. The maximum echo buffer size is 4 096 bytes.

If the echo buffer is implemented, the echo buffer descriptor shall be implemented.

An echo buffer overwritten supported (EBOS) bit set to one indicates either:

a) The device server returns the ECHO BUFFER OVERWRITTEN additional sense code if the data being read from the echo buffer is not the data previously written by the same I_T nexus, or
b) The device server ensures echo buffer data returned to each I_T nexus is the same as that previously written by that I_T nexus.

An EBOS bit set to zero specifies that the echo buffer may be overwritten by any intervening command received on any I_T nexus.

A READ BUFFER command with the mode set to echo buffer descriptor may be used to determine the echo buffer capacity and supported features before a WRITE BUFFER command with the mode set to echo buffer (see 6.35.9) is sent.

6.15.8 Enable expander communications protocol and Echo buffer (1Ah)

Receipt of a READ BUFFER command with this mode (1Ah) causes a communicative expander (see SPI-5) to enter the expanded communications protocol mode. Device servers in SCSI target devices that receive a READ BUFFER command with this mode shall process it as if it were a READ BUFFER command with mode 0Ah (see 6.15.6).

6.16 READ MEDIA SERIAL NUMBER command

The READ MEDIA SERIAL NUMBER command (see table 131) reports the media serial number reported by the device and the currently mounted media.

Table 131 — READ MEDIA SERIAL NUMBER command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (ABh)							
1	Reserved			SERVICE ACTION (01h)				
2	Reserved							
5								
6	(MSB)			ALLOCATION LENGTH				
9								(LSB)
10	Reserved							
11	CONTROL							

The ALLOCATION LENGTH field is defined in 4.3.4.6.

The READ MEDIA SERIAL NUMBER parameter data format is shown in table 132.

Table 132 — READ MEDIA SERIAL NUMBER parameter data format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)			MEDIA SERIAL NUMBER LENGTH (4n-4)				
3								(LSB)
4	MEDIA SERIAL NUMBER							
4n-1								

The MEDIA SERIAL NUMBER LENGTH field shall contain the number of bytes in the MEDIA SERIAL NUMBER field. The media serial number length shall be a multiple of four. The relationship between the MEDIA SERIAL NUMBER LENGTH field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6.

The MEDIA SERIAL NUMBER field shall contain the vendor specific serial number of the media currently installed. If the number of bytes in the vendor specific serial number is not a multiple of four, then up to three bytes containing zero shall be appended to the highest numbered bytes of the MEDIA SERIAL NUMBER field.

If the media serial number is not available (e.g., the currently installed media has no valid media serial number), zero shall be returned in the MEDIA SERIAL NUMBER LENGTH field.

If the media serial number is not accessible because there is no media present, the command shall be terminated with CHECK CONDITION status, with the sense key set to NOT READY, and the additional sense code set to MEDIUM NOT PRESENT.

6.17 RECEIVE COPY RESULTS command

6.17.1 RECEIVE COPY RESULTS command introduction

The RECEIVE COPY RESULTS command (see table 133) provides a means for the application client to receive information about the copy manager or the results of a previous or current EXTENDED COPY command (see 6.3).

Table 133 — RECEIVE COPY RESULTS command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (84h)							
1	Reserved			SERVICE ACTION				
2	LIST IDENTIFIER							
3	Reserved							
9								
10	(MSB)			ALLOCATION LENGTH				
13								(LSB)
14	Reserved							
15	CONTROL							

The service actions defined for the RECEIVE COPY RESULTS command are shown in table 134.

Table 134 — RECEIVE COPY RESULTS service action codes

Code	Name	Description	Returns Data While EXTENDED COPY Is In Progress	Reference
00h	COPY STATUS	Return the current copy status of the EXTENDED COPY command identified by the LIST IDENTIFIER field.	Yes	6.17.2
01h	RECEIVE DATA	Return the held data read by EXTENDED COPY command identified by the LIST IDENTIFIER field.	No	6.17.3
03h	OPERATING PARAMETERS	Return copy manager operating parameters.	Yes	6.17.4
04h	FAILED SEGMENT DETAILS	Return copy target device sense data and other information about the progress of processing a segment descriptor whose processing was not completed during processing of the EXTENDED COPY command identified by the LIST IDENTIFIER field.	No	6.17.5
05h-1Eh	Reserved			
1Fh	Vendor Specific			

The LIST IDENTIFIER field specifies the EXTENDED COPY command (see 6.3) about which information is to be transferred. The RECEIVE COPY RESULTS command shall return information from the EXTENDED COPY command received on the same I_T nexus with a list identifier that matches the list identifier specified in the RECEIVE COPY RESULTS CDB.

If the LIST IDENTIFIER field specifies an EXTENDED COPY command that had the NRCR bit set to one in the parameter data (see 6.3), the copy manager may respond to a RECEIVE COPY RESULTS command as if the EXTENDED COPY command had never been received.

The actual length of the RECEIVE COPY RESULTS parameter data is available in the AVAILABLE DATA parameter data field. The ALLOCATION LENGTH field is defined in 4.3.4.6. See the RECEIVE COPY RESULTS service action definitions for additional requirements.

6.17.2 COPY STATUS service action

In response to the COPY STATUS service action, the copy manager shall return the current status of the EXTENDED COPY command (see 6.3) specified by the LIST IDENTIFIER field in the CDB. If no EXTENDED COPY command known to the copy manager has a matching list identifier, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

Table 135 shows the format of the information returned by the copy manager in response to the COPY STATUS service action. If a device server supports the EXTENDED COPY command, it shall also support the RECEIVE COPY RESULTS command with COPY STATUS service action.

Table 135 — Parameter data for the COPY STATUS service action

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
3				AVAILABLE DATA (00000008h)				(LSB)
4	HDD			COPY MANAGER STATUS				
5	(MSB)							
6				SEGMENTS PROCESSED				(LSB)
7				TRANSFER COUNT UNITS				
8	(MSB)							
11				TRANSFER COUNT				(LSB)

After completion of an EXTENDED COPY command, the copy manager shall preserve all data returned by a COPY STATUS service action for a vendor specific period of time. The copy manager shall discard the COPY STATUS data when:

a) A RECEIVE COPY RESULTS command with COPY STATUS service action is received on the same I_T nexus with a matching list identifier;
b) When another EXTENDED COPY command is received on the same I_T nexus and the list identifier matches the list identifier associated with the data preserved for the COPY STATUS service action;
c) When the copy manager detects a logical unit reset or I_T nexus loss; or
d) When the copy manager requires the resources used to preserve the data.

The AVAILABLE DATA field shall contain the number of bytes present in the parameter data that follows. The relationship between the AVAILABLE DATA field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6.

The held data discarded (HDD) bit indicates whether held data has been discarded. If HDD bit is set to one, held data has been discarded as described in 6.17.4. If HDD bit is set to zero, held data has not been discarded.

The COPY MANAGER STATUS field contains the current status of the EXTENDED COPY command specified by the LIST IDENTIFIER field in the CDB as defined in table 136.

Table 136 — COPY MANAGER STATUS field

Code	Meaning
00h	Operation in progress
01h	Operation completed without errors
02h	Operation completed with errors
03h - 7Fh	Reserved

The SEGMENTS PROCESSED field contains the number of segments the copy manager has processed for the EXTENDED COPY command specified by the LIST IDENTIFIER field in the CDB including the segment currently being processed. This field shall be zero if the copy manager has not yet begun processing segment descriptors.

The TRANSFER COUNT UNITS field specifies the units for the TRANSFER COUNT field as defined in table 137.

Table 137 — COPY STATUS TRANSFER COUNT UNITS field

Code	Meaning [a]	Multiplier to convert TRANSFER COUNT field to bytes
00h	Bytes	1
01h	Kibibytes	2^{10} or 1024
02h	Mebibytes	2^{20}
03h	Gebibytes	2^{30}
04h	Tebibytes	2^{40}
05h	Pebibytes	2^{50}
06h	Exbibytes	2^{60}
07h - FFh	Reserved	

[a] See 3.6.4.

The TRANSFER COUNT field specifies the amount of data written to a destination device for the EXTENDED COPY command specified by the LIST IDENTIFIER field in the CDB prior to receiving the RECEIVE COPY RESULTS command with COPY STATUS service action.

6.17.3 RECEIVE DATA service action

If the copy manager supports those segment descriptors that require data to be held for transfer to the application client, then the RECEIVE DATA service action causes the copy manager to return the held data using the format shown in table 138. If a copy manager supports any of the segment descriptor type codes that require data to be held for the application client (see 6.3.5), then it shall also support the RECEIVE COPY RESULTS command with RECEIVE DATA service action.

The EXTENDED COPY command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB if any of the following conditions exist:

a) No EXTENDED COPY command known to the copy manager has a list identifier that matches the LIST IDENTIFIER field in the CDB; or
b) If the LIST IDENTIFIER field in the CDB identifies an EXTENDED COPY command that still is being processed by the copy manager.

Table 138 — Parameter data for the RECEIVE DATA service action

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
3	AVAILABLE DATA (n-3)							(LSB)
4	HELD DATA							
n								

Following completion of an EXTENDED COPY command, the copy manager shall preserve all data returned by a RECEIVE DATA service action for a vendor specific period of time. The application client should issue a RECEIVE COPY RESULTS command with RECEIVE DATA service action as soon as practical following completion of the EXTENDED COPY command to insure that the data is not discarded by the copy manager. The copy manager shall discard the buffered inline data:

a) After all data held for a specific EXTENDED COPY command has been successfully transferred to the application client;
b) When a RECEIVE COPY RESULTS command with RECEIVE DATA service action has been received on the same I_T nexus with a matching list identifier, with the ALLOCATION LENGTH field set to zero;
c) When another EXTENDED COPY command is received on the same I_T nexus and the list identifier matches the list identifier associated with the data preserved for RECEIVE DATA service action;
d) When the copy manager detects a logical unit reset or I_T nexus loss; or
e) When the copy manager requires the resources used to preserve the data.

The AVAILABLE DATA field shall contain the number of bytes of held data available for delivery to the application client. The relationship between the AVAILABLE DATA field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6.

The HELD DATA field contains the data held by the copy manager for delivery to the application client as prescribed by several segment descriptor type codes. Unless the copy manager's held data limit (see 6.17.4) is exceeded, the first byte held in response to the first segment descriptor in the EXTENDED COPY parameter list prescribing the holding of data (i.e., the oldest byte held) is returned in byte 4. The last byte held in response to the last segment descriptor in the EXTENDED COPY parameter list prescribing the holding of data (i.e., the newest byte held) is returned in byte n.

6.17.4 OPERATING PARAMETERS service action

In response to the OPERATING PARAMETERS service action, the copy manager shall return its operating parameter information in the format shown in table 139. If a device server supports the EXTENDED COPY command (see 6.3), then it shall also support the RECEIVE COPY RESULTS command with OPERATING PARAMETERS service action.

Table 139 — Parameter data for the OPERATING PARAMETERS service action

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
3				AVAILABLE DATA (n-3)				(LSB)
4								
7				Reserved				
8	(MSB)							
9				MAXIMUM TARGET DESCRIPTOR COUNT				(LSB)
10	(MSB)							
11				MAXIMUM SEGMENT DESCRIPTOR COUNT				(LSB)
12	(MSB)							
15				MAXIMUM DESCRIPTOR LIST LENGTH				(LSB)
16	(MSB)							
19				MAXIMUM SEGMENT LENGTH				(LSB)
20	(MSB)							
23				MAXIMUM INLINE DATA LENGTH				(LSB)
24	(MSB)							
27				HELD DATA LIMIT				(LSB)
28	(MSB)							
31				MAXIMUM STREAM DEVICE TRANSFER SIZE				(LSB)
32								
35				Reserved				
36				MAXIMUM CONCURRENT COPIES				
37				DATA SEGMENT GRANULARITY (log 2)				
38				INLINE DATA GRANULARITY (log 2)				
39				HELD DATA GRANULARITY (log 2)				
40								
42				Reserved				
43				IMPLEMENTED DESCRIPTOR LIST LENGTH (n-43)				
44								
n				List of implemented descriptor type codes (ordered)				

The AVAILABLE DATA field shall contain the number of bytes following the AVAILABLE DATA field in the parameter data (i.e., the total number of parameter data bytes minus 4). The relationship between the AVAILABLE DATA field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6.

The MAXIMUM TARGET COUNT field contains the maximum number of target descriptors that the copy manager allows in a single EXTENDED COPY target descriptor list.

The MAXIMUM SEGMENT COUNT field contains the maximum number of segment descriptors that the copy manager allows in a single EXTENDED COPY segment descriptor list.

The MAXIMUM DESCRIPTOR LIST LENGTH field contains the maximum length, in bytes, of the target descriptor list and segment descriptor list. This length includes the embedded data but excludes inline data that follows the descriptors.

The MAXIMUM SEGMENT LENGTH field indicates the length, in bytes, of the largest amount of data that the copy manager supports writing via a single segment. Bytes introduced as a result of the PAD bit being set to one (see 6.3.7) are not counted towards this limit. A value of zero indicates that the copy manager places no limits on the amount of data written by a single segment.

The MAXIMUM INLINE DATA LENGTH field indicates the length, in bytes, of the largest amount of inline data that the copy manager supports in the EXTENDED COPY parameter list. This does not include data included as embedded data within the segment descriptors. The MAXIMUM INLINE DATA LENGTH field applies only to segment descriptors containing the 04h descriptor type code (see 6.3.7.7). The field shall be set to zero when the 04h descriptor type code is not supported by the copy manager.

The HELD DATA LIMIT field indicates the length, in bytes, of the minimum amount of data the copy manager guarantees to hold for return to the application client via the RECEIVE COPY RESULTS command with RECEIVE DATA service action (see 6.17.3). If the processing of segment descriptors requires more data to be held, the copy manager may discard some of the held data in a vendor specific manner that retains the held bytes from the most recently processed segment descriptors. The discarding of held data bytes shall not be considered an error. If held data is discarded, the HDD bit shall be set as described in 6.17.2.

The MAXIMUM CONCURRENT COPIES field contains the maximum number of EXTENDED COPY commands supported for concurrent processing by the copy manager.

The DATA SEGMENT GRANULARITY field indicates the length of the smallest data block that copy manager permits in a non-inline segment descriptor (i.e., segment descriptors with type codes other than 04h). The amount of data transferred by a single segment descriptor shall be a multiple of the granularity. The DATA SEGMENT GRANULARITY value is expressed as a power of two. Bytes introduced as a result of the PAD bit being set to one (see 6.3.7) are not counted towards the data length granularity.

The INLINE DATA GRANULARITY field indicates the length of the of the smallest block of inline data that the copy manager permits being written by a segment descriptor containing the 04h descriptor type code (see 6.3.7.7). The amount of inline data written by a single segment descriptor shall be a multiple of the granularity. The INLINE DATA GRANULARITY value is expressed as a power of two. Bytes introduced as a result of the PAD bit being set to one (see 6.3.7) are not counted towards the length granularity.

If the copy manager encounters a data or inline segment descriptor that violates either the data segment granularity or the inline data granularity, the EXTENDED COPY command shall be terminated with CHECK CONDITION status, with the sense key set to COPY ABORTED, and the additional sense code set to COPY SEGMENT GRANULARITY VIOLATION.

The HELD DATA GRANULARITY field indicates the length of the smallest block of held data that the copy manager shall transfer to the application client in response to a RECEIVE COPY RESULTS command with RECEIVE DATA service action (see 6.17.3). The amount of data held by the copy manager in response to any one segment descriptor shall be a multiple of this granularity. The HELD DATA GRANULARITY value is expressed as a power of two.

The MAXIMUM STREAM DEVICE TRANSFER SIZE field indicates the maximum transfer size, in bytes, supported for stream devices.

The IMPLEMENTED DESCRIPTOR LIST LENGTH field contains the length, in bytes, of the list of implemented descriptor type codes.

The list of implemented descriptor type codes contains one byte for each segment or target DESCRIPTOR TYPE CODE value (see 6.3.5) supported by the copy manager, with a unique supported DESCRIPTOR TYPE CODE value in each byte. The DESCRIPTOR TYPE CODE values shall appear in the list in ascending numerical order.

6.17.5 FAILED SEGMENT DETAILS service action

In response to the FAILED SEGMENT DETAILS service action, the copy manager shall return details of the segment processing failure that caused termination of the EXTENDED COPY command (see 6.3) specified by the LIST IDENTIFIER field in the CDB. Table 140 shows the format of the information returned by the copy manager in response to a FAILED SEGMENT DETAILS service action. If a device server supports the EXTENDED COPY command (see 7.4), then it shall also support the RECEIVE COPY RESULTS command with FAILED SEGMENT DETAILS service action.

When processing of an EXTENDED COPY command is aborted and processing of a segment descriptor is incomplete, the copy manager shall preserve details about the progress in processing of that descriptor. These details enable the application client to obtain information it needs to determine the state in which copy target devices, in particular stream devices, have been left by incomplete processing.

The EXTENDED COPY command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB if any of the following conditions exist:

a) No EXTENDED COPY command known to the copy manager has a list identifier that matches the LIST IDENTIFIER field in the CDB; or
b) If the LIST IDENTIFIER field in the CDB identifies an EXTENDED COPY command that still is being processed by the copy manager.

Table 140 — Parameter data for the FAILED SEGMENT DETAILS service action

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
3				AVAILABLE DATA (n-3)				(LSB)
4								
55				Reserved				
56				EXTENDED COPY COMMAND STATUS				
57				Reserved				
58	(MSB)							
59				SENSE DATA LENGTH (n-59)				(LSB)
60								
n				SENSE DATA				

The application client should issue a RECEIVE COPY RESULTS command with FAILED SEGMENT DETAILS service action immediately following failure of the EXTENDED COPY command to insure that the information is not discarded by the copy manager. The copy manager shall discard the failed segment details:

a) After all failed segment details held for a specific EXTENDED COPY command have been successfully transferred to the application client;
b) When a RECEIVE COPY RESULTS command with FAILED SEGMENT DETAILS service action has been received on the same I_T nexus with a matching list identifier, with the ALLOCATION LENGTH field set to zero;
c) When another EXTENDED COPY command is received on the same I_T nexus using the same list identifier;
d) When the copy manager detects a logical unit reset or I_T nexus loss; or
e) When the copy manager requires the resources used to preserve the data.

The AVAILABLE DATA field shall contain the number of bytes of failed segment details available for delivery to the application client. If no failed segment details data is available for the specified list identifier then the AVAILABLE

DATA field shall be set to zero and no data beyond the AVAILABLE DATA field shall be returned. The relationship between the AVAILABLE DATA field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6.

The COPY COMMAND STATUS field contains the SCSI status value that was returned for the EXTENDED COPY command identified by the LIST IDENTIFIER field in the CDB.

The SENSE DATA LENGTH field indicates how many bytes of sense data are present in the SENSE DATA field.

The SENSE DATA field contains a copy of the sense data that the copy manager prepared as part of terminating the EXTENDED COPY command identified by the list identifier with a CHECK CONDITION status.

6.18 RECEIVE DIAGNOSTIC RESULTS command

The RECEIVE DIAGNOSTIC RESULTS command (see table 141) requests that data be sent to the application client Data-In Buffer. The data is either data based on the most recent SEND DIAGNOSTIC command (see 6.28) or is a diagnostic page specified by the PAGE CODE field.

Table 141 — RECEIVE DIAGNOSTIC RESULTS command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (1Ch)							
1	Reserved							PCV
2	PAGE CODE							
3	(MSB)			ALLOCATION LENGTH				
4								(LSB)
5	CONTROL							

A page code valid (PCV) bit set to zero specifies that the device server return parameter data based on the most recent SEND DIAGNOSTIC command (e.g., the diagnostic page with the same page code as that specified in the most recent SEND DIAGNOSTIC command). The response to a RECEIVE DIAGNOSTIC RESULTS command with the PCV bit set to zero is vendor-specific if:

a) The most recent SEND DIAGNOSTIC command was not a SEND DIAGNOSTIC command defining parameter data to return;
b) A RECEIVE DIAGNOSTIC RESULTS command with a PCV bit set to one has been processed since the last SEND DIAGNOSTIC command was processed; or
c) No SEND DIAGNOSTIC command defining parameter data to return has been processed since power on, hard reset, or logical unit reset.

A page code valid (PCV) bit set to one specifies that the device server return the diagnostic page specified in the PAGE CODE field. Page code values are defined in 7.1 or in another command standard (see 3.1.18).

NOTE 27 - Logical units compliant with previous versions of this standard (e.g., SPC-2) may transfer more than one diagnostic page in the parameter data if the PCV bit is set to zero and the previous SEND DIAGNOSTIC command sent more than one diagnostic page in the parameter list.

NOTE 28 - To ensure that the diagnostic command information is not destroyed by a command sent from another I_T nexus, the logical unit should be reserved.

NOTE 29 - Although diagnostic software is generally device-specific, this command and the SEND DIAGNOSTIC command provide a means to isolate the operating system software from the device-specific diagnostic software. The operating system may remain device-independent.

The ALLOCATION LENGTH field is defined in 4.3.4.6.

See 7.1 for RECEIVE DIAGNOSTIC RESULTS diagnostic page format definitions.

6.19 REPORT ALIASES command

The REPORT ALIASES command (see table 142) requests that the device server return the alias list. The alias list is managed using the CHANGE ALIASES command (see 6.2). If the CHANGE ALIASES command is supported, the REPORT ALIASES command shall also be supported.

The REPORT ALIASES command is a service action of the MAINTENANCE IN command. Additional MAINTENANCE IN service actions are defined in SCC-2 and in this standard. The MAINTENANCE IN service actions defined in SCC-2 apply only to logical units that return a device type of 0Ch or the SCCS bit set to one in their standard INQUIRY data (see 6.4.2).

Table 142 — REPORT ALIASES command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (A3h)							
1	Reserved			SERVICE ACTION (0Bh)				
2	Reserved							
5								
6	(MSB)			ALLOCATION LENGTH				
9								(LSB)
10	Reserved							
11	CONTROL							

The ALLOCATION LENGTH field is defined in 4.3.4.6.

The parameter data returned by a REPORT ALIASES command (see table 143) contains zero or more alias entries.

Table 143 — REPORT ALIASES parameter data

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
3	ADDITIONAL LENGTH (n-3)							(LSB)
4	Reserved							
5	Reserved							
6	NUMBER OF ALIASES (X)							
7								
	Alias entry (or entries)							
8	Alias entry 0 (see 6.2.2)							
	. . .							
	Alias entry x (see 6.2.2)							
n								

The ADDITIONAL LENGTH field indicates the number of bytes in the remaining parameter data. The relationship between the ADDITIONAL LENGTH field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6.

The NUMBER OF ALIASES field indicates the number of alias entries in the alias list and shall not be changed if the CDB contains an insufficient allocation length.

The parameter data shall include one alias entry for each alias in the alias list. The format of an alias entry is described in 6.2.2.

6.20 REPORT DEVICE IDENTIFIER command

The REPORT DEVICE IDENTIFIER command (see table 144) requests that the device server send device identification information to the application client. As defined in the SCC-2 standard, the REPORT DEVICE IDENTIFIER command is the REPORT PERIPHERAL DEVICE/COMPONENT DEVICE IDENTIFIER service action of the MAINTENANCE IN command. Additional MAINTENANCE IN and MAINTENANCE OUT service actions are defined in SCC-2 and in this standard.

The MAINTENANCE IN service actions defined only in SCC-2 shall apply only to SCSI devices that return a device type of 0Ch or the SCCS bit set to one in their standard INQUIRY data. When a SCSI device returns a device type of 0Ch or the SCCS bit set to one in its standard INQUIRY data, the implementation requirements for the SCC-2

MAINTENANCE IN service actions shall be as specified in SCC-2. Otherwise the MAINTENANCE IN service action definitions and implementation requirements stated in this standard shall apply.

Table 144 — REPORT DEVICE IDENTIFIER command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (A3h)							
1	Reserved			SERVICE ACTION (05h)				
2	Reserved							
3								
4	Restricted							
5								
6	(MSB)	ALLOCATION LENGTH						
9								(LSB)
10	Reserved						Restricted	Reserved
11	CONTROL							

SCC-2 defines specific usages for bytes 4 and 5, and bit 1 in byte 10, however these fields are reserved for the REPORT DEVICE IDENTIFIER command defined by this standard.

The ALLOCATION LENGTH field is defined in 4.3.4.6.

The REPORT DEVICE IDENTIFIER parameter data (see table 145) contains a four-byte field that contains the length in bytes of the parameter data and the logical unit's identifier.

Table 145 — REPORT DEVICE IDENTIFIER parameter data

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)	IDENTIFIER LENGTH (n-3)						
3								(LSB)
4	IDENTIFIER							
n								

The IDENTIFIER LENGTH field indicates the length in bytes of the IDENTIFIER field. The relationship between the IDENTIFIER LENGTH field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6. The identifier length shall initially equal zero, and shall be changed only by a successful SET DEVICE IDENTIFIER command.

The IDENTIFIER field shall contain a vendor specific value. The value reported shall be the last value written by a successful SET DEVICE IDENTIFIER command. The value of the identifier shall be changed only by a SET DEVICE IDENTIFIER command. The identifier value shall persist through logical unit resets, I_T nexus losses, media format operations, and media replacement.

The logical unit shall return the same identifier to all application clients.

Processing a REPORT DEVICE IDENTIFIER may require the enabling of a nonvolatile memory within the logical unit. If the nonvolatile memory is not ready, the command shall be terminated with CHECK CONDITION status, rather than wait for the nonvolatile memory to become ready. The sense key shall be set to NOT READY and the additional sense code shall be set as described in table 184 (see 6.33). This information should allow the application client to determine the action required to cause the device server to become ready.

6.21 REPORT LUNS command

The REPORT LUNS command (see table 146) requests that the peripheral device logical unit inventory accessible to the I_T nexus be sent to the application client. The logical unit inventory is a list that shall include the logical unit numbers of all logical units having a PERIPHERAL QUALIFIER value of 000b (see 6.4.2). Logical unit numbers for logical units with PERIPHERAL QUALIFIER values other than 000b and 011b may be included in the logical unit inventory. Logical unit numbers for logical units with a PERIPHERAL QUALIFIER value of 011b shall not be included in the logical unit inventory.

Table 146 — REPORT LUNS command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (A0h)							
1	Reserved							
2	SELECT REPORT							
3	Reserved							
5								
6	(MSB)			ALLOCATION LENGTH				
9								(LSB)
10	Reserved							
11	CONTROL							

The SELECT REPORT field (see table 147) specifies the types of logical unit addresses that shall be reported.

Table 147 — SELECT REPORT field

Code	Description
00h	The list shall contain the logical units accessible to the I_T nexus with the following addressing methods (see SAM-3): a) Logical unit addressing method, b) Peripheral device addressing method; and c) Flat space addressing method. If there are no logical units, the LUN LIST LENGTH field shall be zero.
01h	The list shall contain only well known logical units, if any. If there are no well known logical units, the LUN LIST LENGTH field shall be zero.
02h	The list shall contain all logical units accessible to the I_T nexus.
03h - FFh	Reserved

The ALLOCATION LENGTH field is defined in 4.3.4.6. The allocation length should be at least 16.

NOTE 30 - Device servers compliant with SPC return CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB when the allocation length is less than 16 bytes.

The REPORT LUNS command shall return CHECK CONDITION status only when the device server is unable to return the requested report of the logical unit inventory.

If a REPORT LUNS command is received from an I_T nexus with a pending unit attention condition (i.e., before the device server reports CHECK CONDITION status), the device server shall perform the REPORT LUNS command. If the unit attention condition was established because of a change in the logical unit inventory, that unit attention condition shall be cleared for the initiator port associated with that I_T nexus by the REPORT LUNS command.

Unit attention conditions established for other reasons shall not be cleared by the REPORT LUNS command (see SAM-3).

The REPORT LUNS parameter data should be returned even though the device server is not ready for other commands. The report of the logical unit inventory should be available without incurring any media access delays. If the device server is not ready with the logical unit inventory or if the inventory list is null for the requesting I_T nexus and the SELECT REPORT field set to 02h, then the device server shall provide a default logical unit inventory that contains at least LUN 0 or the REPORT LUNS well known logical unit (see 8.2). A non-empty peripheral device logical unit inventory that does not contain either LUN 0 or the REPORT LUNS well known logical unit is valid.

If a REPORT LUNS command is received for a logical unit that the SCSI target device does not support and the device server is not capable of returning the logical unit inventory, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to LOGICAL UNIT NOT SUPPORTED.

If the logical unit inventory changes for any reason (e.g., completion of initialization, removal of a logical unit, or creation of a logical unit), then the device server shall establish a unit attention condition (see SAM-3) for the initiator port associated with every I_T nexus, with the additional sense code set to REPORTED LUNS DATA HAS CHANGED.

The processing of a REPORT LUNS command that returns the logical unit inventory by any logical unit shall clear the REPORTED LUNS DATA HAS CHANGED unit attention condition for all logical units accessible to the I_T nexus on which the command was received.

The device server shall report those devices in the logical unit inventory using the format shown in table 148.

Table 148 — REPORT LUNS parameter data format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
3	LUN LIST LENGTH (n-7)							(LSB)
4	Reserved							
7								
	LUN list							
8	First LUN							
15								
	. . .							
n-7	Last LUN							
n								

The LUN LIST LENGTH field shall contain the length in bytes of the LUN list that is available to be transferred. The LUN list length is the number of logical unit numbers in the logical unit inventory multiplied by eight. The relationship between the LUN LIST LENGTH field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6.

6.22 REPORT PRIORITY command

The REPORT PRIORITY command (see table 149) requests the priority that has been assigned to one or more I_T nexuses associated with the logical unit (i.e., I_T_L nexuses).

The REPORT PRIORTY command is a service action of the MAINTENANCE IN command. Additional MAINTENANCE IN service actions are defined in SCC-2 and in this standard. The MAINTENANCE IN service actions defined in SCC-2 apply only to logical units that return a device type of 0Ch or the SCCS bit set to one in their standard INQUIRY data (see 6.4.2).

Table 149 — REPORT PRIORITY command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (A3h)							
1	Reserved			SERVICE ACTION (0Eh)				
2	PRIORITY REPORTED			Reserved				
3	Reserved							
5								
6	(MSB)			ALLOCATION LENGTH				
9								(LSB)
10	Reserved							
11	CONTROL							

The PRIORITY REPORTED field (see table 150) specifies the information to be returned in the parameter data.

Table 150 — PRIORITY REPORTED field

Code	Description
00b	Only the priority for the I_T nexus on which the command was received shall be reported in the REPORT PRIORITY parameter data.
01b	The priority for each I_T nexus that is not set to the initial priority shall be reported in the REPORT PRIORITY parameter data.
10b - 11b	Reserved

The ALLOCATION LENGTH field is defined in 4.3.4.6. The allocation length should be at least four.

The format of the parameter data returned by the REPORT PRIORITY command is shown in table 151.

Table 151 — REPORT PRIORITY parameter data format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
3	PRIORITY PARAMETER DATA LENGTH (n-3)							(LSB)
	Priority descriptors							
4	First priority descriptor (see table 152)							
	. . .							
n	Last priority descriptor (see table 152)							

The PRIORITY PARAMETER DATA LENGTH field indicates the number of bytes of parameter data that follow.

Each priority descriptor (see table 152) contains priority information for a single I_T_L nexus.

Table 152 — Priority descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved				CURRENT PRIORITY			
1	Reserved							
2	(MSB)							
3	RELATIVE TARGET PORT IDENTIFIER							(LSB)
4	Reserved							
5	Reserved							
6	(MSB)							
7	ADDITIONAL DESCRIPTOR LENGTH (n-7)							(LSB)
8	TRANSPORTID							
n								

The CURRENT PRIORITY field contains the priority assigned to the I_T_L nexus represented by this descriptor. If the PRIORITY REPORTED field in this command is set to 00b and the priority for the I_T_L nexus associated with this command is set to the initial priority, then the CURRENT PRIORITY field shall be set to zero. The priority assigned to an I_T_L nexus may be used as a task priority for tasks received via that I_T_L nexus (see SAM-3).

The RELATIVE TARGET PORT IDENTIFIER field contains the relative port identifier (see 3.1.88) of the target port that is part of the I_T_L nexus to which the current priority applies.

The ADDITIONAL DESCRIPTOR LENGTH field indicates the number of bytes that follow in the descriptor (i.e., the size of the TransportID).

The TRANSPORTID field contains a TransportID (see 7.5.4) identifying the initiator port that is part of the I_T_L nexus to which the current priority applies.

6.23 REPORT SUPPORTED OPERATION CODES command

6.23.1 REPORT SUPPORTED OPERATION CODES command introduction

The REPORT SUPPORTED OPERATION CODES command (see table 153) requests information on commands the addressed logical unit supports. An application client may request a list of all operation codes and service actions supported by the logical unit or the command support data for a specific command.

The REPORT SUPPORTED OPERATION CODES command is a service action of the MAINTENANCE IN command. Additional MAINTENANCE IN service actions are defined in SCC-2 and in this standard. The MAINTENANCE IN service actions defined in SCC-2 apply only to logical units that return a device type of 0Ch or the SCCS bit set to one in their standard INQUIRY data (see 6.4.2).

Table 153 — REPORT SUPPORTED OPERATION CODES command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (A3h)							
1	Reserved			SERVICE ACTION (0Ch)				
2	Reserved					REPORTING OPTIONS		
3	REQUESTED OPERATION CODE							
4	(MSB)			REQUESTED SERVICE ACTION				
5								(LSB)
6	(MSB)			ALLOCATION LENGTH				
9								(LSB)
10	Reserved							
11	CONTROL							

The REPORTING OPTIONS field (see table 154) specifies the information to be returned in the parameter data.

Table 154 — REPORT SUPPORTED OPERATION CODES reporting options

Reporting Option	Description	Parameter Data Reference
000b	A list of all operation codes and service actions supported by the logical unit shall be returned in the all_commands parameter data format. The REQUESTED OPERATION CODE CDB field and REQUESTED SERVICE ACTION CDB field shall be ignored.	6.23.2
001b	The command support data for the operation code specified in the REQUESTED OPERATION CODE field shall be returned in the one_command parameter data format. The REQUESTED SERVICE ACTION CDB field shall be ignored. If the REQUESTED OPERATION CODE field specifies an operation code that has service actions, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.	6.23.3
010b	The command support data for the operation code and service action specified in the REQUESTED OPERATION CODE CDB field and REQUESTED SERVICE ACTION CDB field shall be returned in the one_command parameter data format. If the REQUESTED OPERATION CODE CDB field specifies an operation code that does not have service actions, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.	6.23.3
011b-111b	Reserved	

The REQUESTED OPERATION CODE field specifies the operation code of the command to be returned in the one_command parameter data format (see 6.23.3).

The REQUESTED SERVICE ACTION field specifies the service action of the command to be returned in the one_command parameter data format.

The ALLOCATION LENGTH field is defined in 4.3.4.6.

6.23.2 All_commands parameter data format

The REPORT SUPPORTED OPERATION CODES all_commands parameter data format (see table 155) begins with a four-byte header that contains the length in bytes of the parameter data followed by a list of supported commands. Each command descriptor contains information about a single supported command CDB (i.e., one operation code and service action combination, or one non-service-action operation code). The list of command descriptors shall contain all commands supported by the logical unit.

Table 155 — All_commands parameter data

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
3				COMMAND DATA LENGTH (n-3)				(LSB)
	Command descriptors							
4				Command descriptor 0 (see table 156)				
				. . .				
n				Command descriptor x (see table 156)				

The COMMAND DATA LENGTH field indicates the length in bytes of the command descriptor list.

Each command descriptor (see table 156) contains information about a single supported command CDB.

Table 156 — Command descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE							
1	Reserved							
2	(MSB)							
3				SERVICE ACTION				(LSB)
4	Reserved							
5	Reserved							SERVACTV
6	(MSB)							
7				CDB LENGTH				(LSB)

The OPERATION CODE field contains the operation code of a command supported by the logical unit.

The SERVICE ACTION field contains a supported service action of the supported operation code indicated by the OPERATION CODE field. If the operation code indicated in the OPERATION CODE field does not have a service actions, the SERVICE ACTION field shall be set to 00h.

A service action valid (SERVACTV) bit set to zero indicates the operation code indicated by the OPERATION CODE field does not have service actions and the SERVICE ACTION field contents are reserved. A SERVACTV bit set to one indicates the operation code indicated by the OPERATION CODE field has service actions and the contents of the SERVICE ACTION field are valid.

The CDB LENGTH field contains the length of the command CDB in bytes for the operation code indicated in the OPERATION CODE field, and if the SERVACTV bit is set to the service action indicated by the SERVICE ACTION field.

6.23.3 One_command parameter data format

The REPORT SUPPORTED OPERATION CODES one_command parameter data format (see table 157) contains information about the CDB and a usage map for bits in the CDB for the command specified by the REPORTING OPTIONS, REQUESTED OPERATION CODE, and REQUESTED SERVICE ACTION fields in the REPORT SUPPORTED OPERATION CODES CDB.

Table 157 — One_command parameter data

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved							
1	Reserved						SUPPORT	
2	(MSB)			CDB SIZE (n-3)				
3								(LSB)
4	CDB USAGE DATA							
n								

The SUPPORT field is defined in table 158.

Table 158 — SUPPORT values

Support	Description
000b	Data about the requested SCSI command is not currently available. All data after byte 1 is not valid. A subsequent request for command support data may be successful.
001b	The device server does not support the requested command. All data after byte 1 is undefined.
010b	Reserved
011b	The device server supports the requested command in conformance with a SCSI standard. The parameter data format conforms to the definition in table 157.
100b	Reserved
101b	The device server supports the requested command in a vendor specific manner. The parameter data format conforms to the definition in table 157.
110b - 111b	Reserved

The CDB SIZE field contains the size of the CDB USAGE DATA field in the parameter data, and the number of bytes in the CDB for command being queried (i.e., the command specified by the REPORTING OPTIONS, REQUESTED OPERATION CODE, and REQUESTED SERVICE ACTION fields in the REPORT SUPPORTED OPERATION CODES CDB).

The CDB USAGE DATA field contains information about the CDB for the command being queried. The first byte of the CDB USAGE DATA field shall contain the operation code for the command being queried. If the command being queried contains a service action, then that service action code shall be placed in the CDB USAGE DATA field in the same location as the SERVICE ACTION field of the command CDB. All other bytes of the CDB USAGE DATA field shall contain a usage map for bits in the CDB for the command being queried.

The bits in the usage map shall have a one-for-one correspondence to the CDB for the command being queried. If the device server evaluates a bit in the CDB for the command being queried, the usage map shall contain a one in the corresponding bit position. If any bit representing part of a field is returned as one, all bits for the field shall be

returned as one. If the device server ignores or treats as reserved a bit in the CDB for the command being queried, the usage map shall contain a zero in the corresponding bit position. The usage map bits for a given CDB field all shall have the same value.

For example, the CDB usage bit map for the REPORT SUPPORTED OPERATION CODES command is: A3h, 0Ch, 03h, FFh, FFh, FFh, FFh, FFh, FFh, FFh, 00h, 07h. This example assumes that the logical unit only supports the low-order three bits of the CDB CONTROL byte. The first byte contains the operation code, and the second byte contains three reserved bits and the service action. The remaining bytes contain the usage map.

6.24 REPORT SUPPORTED TASK MANAGEMENT FUNCTIONS command

The REPORT SUPPORTED TASK MANAGEMENT FUNCTIONS command (see table 159) requests information on task management functions (see SAM-3) the addressed logical unit supports.

The REPORT SUPPORTED TASK MANAGEMENT FUNCTIONS command is a service action of the MAINTENANCE IN command. Additional MAINTENANCE IN service actions are defined in SCC-2 and in this standard. The MAINTENANCE IN service actions defined in SCC-2 apply only to logical units that return a device type of 0Ch or the SCCS bit set to one in their standard INQUIRY data (see 6.4.2).

Table 159 — REPORT SUPPORTED TASK MANAGEMENT FUNCTIONS command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (A3h)							
1	Reserved			SERVICE ACTION (0Dh)				
2	Reserved							
5								
6	(MSB)			ALLOCATION LENGTH (4h or larger)				
9								(LSB)
10	Reserved							
11	CONTROL							

The ALLOCATION LENGTH field specifies the number of bytes that have been allocated for the returned parameter data. The allocation length should be at least four. If the allocation length is less than four, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

The format of the parameter data returned by the REPORT TASK MANAGEMENT FUNCTIONS command is shown in table 160.

Table 160 — REPORT SUPPORTED TASK MANAGEMENT FUNCTIONS parameter data

Bit / Byte	7	6	5	4	3	2	1	0
0	ATS	ATSS	CACAS	CTSS	LURS	QTS	TRS	WAKES
1	Reserved							
3								

An ABORT TASK supported (ATS) bit set to one indicates the ABORT TASK task management function (see SAM-3) is supported by the logical unit. An ATS bit set to zero indicates the ABORT TASK task management function is not supported.

An ABORT TASK SET supported (ATSS) bit set to one indicates the ABORT TASK SET task management function (see SAM-3) is supported by the logical unit. An ATSS bit set to zero indicates the ABORT TASK SET task management function is not supported.

A CLEAR ACA supported (CACAS) bit set to one indicates the CLEAR ACA task management function (see SAM-3) is supported by the logical unit. An CACAS bit set to zero indicates the CLEAR ACA task management function is not supported.

A CLEAR TASK SET supported (CTSS) bit set to one indicates the CLEAR TASK SET task management function (see SAM-3) is supported by the logical unit. An CTSS bit set to zero indicates the CLEAR TASK SET task management function is not supported.

A LOGICAL UNIT RESET supported (LURS) bit set to one indicates the LOGICAL UNIT RESET task management function (see SAM-3) is supported by the logical unit. An LURS bit set to zero indicates the LOGICAL UNIT RESET task management function is not supported.

A QUERY TASK supported (QTS) bit set to one indicates the QUERY TASK task management function (see SAM-3) is supported by the logical unit. An QTS bit set to zero indicates the QUERY TASK task management function is not supported.

A TARGET RESET supported (TRS) bit set to one indicates the TARGET RESET task management function (see SAM-2) is supported by the logical unit. An TRS bit set to zero indicates the TARGET RESET task management function is not supported.

A WAKEUP supported (WAKES) bit set to one indicates the WAKEUP task management function (see SAM-2) is supported by the logical unit. An WAKES bit set to zero indicates the WAKEUP task management function is not supported.

6.25 REPORT TARGET PORT GROUPS command

The REPORT TARGET PORT GROUPS command (see table 161) requests that the device server send target port group information to the application client. This command shall be supported by logical units that report in the standard INQUIRY data (see 6.4.2) that they support asymmetric logical unit access (i.e., return a non-zero value in the TPGS field).

The REPORT TARGET PORT GROUPS command is a service action of the MAINTENANCE IN command. Additional MAINTENANCE IN service actions are defined in SCC-2 and in this standard. The MAINTENANCE IN service actions defined only in SCC-2 apply only to logical units that return a device type of 0Ch or the SCCS bit set to one in their standard INQUIRY data.

Table 161 — REPORT TARGET PORT GROUPS command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (A3h)							
1	Reserved			SERVICE ACTION (0Ah)				
2	Reserved							
5								
6	(MSB)			ALLOCATION LENGTH				
9								(LSB)
10	Reserved							
11	CONTROL							

The ALLOCATION LENGTH field is defined in 4.3.4.6.

Returning REPORT TARGET PORT GROUPS parameter data may require the enabling of a nonvolatile memory. If the nonvolatile memory is not ready, the command shall be terminated with CHECK CONDITION status, rather than wait for the nonvolatile memory to become ready. The sense key shall be set to NOT READY and the additional sense code shall be set as described in table 184 (see 6.33).

The format for the parameter data returned by the REPORT TARGET PORT GROUPS command is shown in table 162.

Table 162 — REPORT TARGET PORT GROUPS parameter data format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)			RETURN DATA LENGTH (n-3)				
3								(LSB)
	Target port group descriptor(s)							
4	First target port group descriptor (see table 163)							
	⋮							
n	Last port group descriptor (see table 163)							

The RETURN DATA LENGTH field indicates the length in bytes of the list of target port groups. The relationship between the RETURN DATA LENGTH field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6.

There shall be one target port group descriptor (see table 163) for each target port group.

Table 163 — Target port group descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	PREF	Reserved			ASYMMETRIC ACCESS STATE			
1	T_SUP	Reserved			U_SUP	S_SUP	AN_SUP	AO_SUP
2	(MSB)	TARGET PORT GROUP						
3								(LSB)
4	Reserved							
5	STATUS CODE							
6	Vendor specific							
7	TARGET PORT COUNT							
	Target port descriptor(s)							
8	First target port descriptor (see table 166)							
11								
	. . .							
n-3	Last port group descriptor (see table 166)							
n								

A preferred target port (PREF) bit set to one indicates that the target port group is a preferred target port group for accessing the addressed logical unit (see 5.8.2.6). A PREF bit set to zero indicates the target port group is not a preferred target port group.

The ASYMMETRIC ACCESS STATE field (see table 164) contains the target port group's current asymmetric access state (see 5.8.2.4).

Table 164 — ASYMMETRIC ACCESS STATE field

Code	State
0h	Active/optimized
1h	Active/non-optimized
2h	Standby
3h	Unavailable
4h-Eh	Reserved
Fh	Transitioning between states

If any of the T_SUP bit, U_SUP bit, S_SUP bit, AN_SUP bit, or AO_SUP bit are set to one, then the T_SUP bit, U_SUP bit, S_SUP bit, AN_SUP bit, and AO_SUP bit are as defined in this standard. If the T_SUP bit, U_SUP bit, S_SUP bit, AN_SUP bit, and AO_SUP bit are all set to zero, then which asymmetric access states are supported is vendor specific.

A transitioning supported (T_SUP) bit set to one indicates that the device server supports returning the ASYMMETRIC ACCESS STATE field set to Fh (i.e., transitioning between states). A T_SUP bit set to zero indicates that the device server does not return an ASYMMETRIC ACCESS STATE field set to Fh.

An unavailable supported (U_SUP) bit set to one indicates that the unavailable asymmetric access state is supported. A U_SUP bit set to zero indicates that the unavailable asymmetric access state is not supported.

A standby supported (S_SUP) bit set to one indicates that the standby asymmetric access state is supported. An S_SUP bit set to zero indicates that the standby asymmetric access state is not supported.

An active/non-optimized supported (AN_SUP) bit set to one indicates that the active/non-optimized asymmetric access state is supported. An AN_SUP bit set to zero indicates that the active/non-optimized asymmetric access state is not supported.

An active/optimized supported (AO_SUP) bit set to one indicates that the active/optimized asymmetric access state is supported. An AO_SUP bit set to zero indicates that the active/optimized asymmetric access state is not supported.

The TARGET PORT GROUP field contains an identification of the target port group described by this target port group descriptor. Target port group information is also returned in the Device Identification VPD page (see 7.6.3).

The STATUS CODE field (see table 165) indicates why a target port group may be in a specific target port group asymmetric access state. It provides a mechanism to indicate error conditions.

Table 165 — STATUS CODE field

Code	Description
00h	No status available.
01h	Target port group asymmetric access state altered by SET TARGET PORT GROUPS command.
02h	Target port group asymmetric access state altered by implicit asymmetrical logical unit access behavior.
03h-FFh	Reserved

The TARGET PORT COUNT field indicates the number of target ports that are in that target port group and the number of target port descriptors in the target port group descriptor. Every target port group shall contain at least one target port. The target port group descriptor shall include one target port descriptor for each target port in the target port group.

Table 166 — Target port descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	Obsolete							
1								
2	(MSB)	RELATIVE TARGET PORT IDENTIFIER						
3								(LSB)

The RELATIVE TARGET PORT IDENTIFIER field contains a relative port identifier (see 3.1.88) of a target port in the target port group.

6.26 REPORT TIMESTAMP command

The REPORT TIMESTAMP command (see table 167) requests that the device server return the value of the logical unit's timestamp.

The REPORT TIMESTAMP command is a service action of the MAINTENANCE IN command. Additional MAINTENANCE IN service actions are defined in SCC-2 and in this standard. The MAINTENANCE IN service actions defined only in SCC-2 apply only to logical units that return a device type of 0Ch or the SCCS bit set to one in their standard INQUIRY data (see 6.4.2).

Table 167 — REPORT TIMESTAMP command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (A3h)							
1	Reserved			SERVICE ACTION (0Fh)				
2	Reserved							
5								
6	(MSB)			ALLOCATION LENGTH				
9								(LSB)
10	Reserved							
11	CONTROL							

The ALLOCATION LENGTH field is defined in 4.3.4.6.

The format for the parameter data returned by the REPORT TIMESTAMP command is shown in table 168.

Table 168 — REPORT TIMESTAMP parameter data format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)			TIMESTAMP PARAMETER DATA LENGTH (0Ah)				
1								(LSB)
2	Reserved					TIMESTAMP ORIGIN		
3	Reserved							
4	TIMESTAMP							
9								
10	Reserved							
11	Reserved							

The TIMESTAMP PARAMETER DATA LENGTH field indicates the number of bytes of parameter data that follow. The relationship between the TIMESTAMP PARAMETER DATA LENGTH field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6.

The TIMESTAMP ORIGIN field indicates the origin of the timestamp (see 5.13).

The TIMESTAMP field contains the current value of the timestamp (see 5.13).

6.27 REQUEST SENSE command

The REQUEST SENSE command (see table 169) requests that the device server transfer sense data to the application client.

Table 169 — REQUEST SENSE command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (03h)							
1	Reserved							DESC
2	Reserved							
3								
4	ALLOCATION LENGTH							
5	CONTROL							

The descriptor format (DESC) bit specifies which sense data format shall be returned. If DESC is set to zero, fixed format sense data (see 4.5.3) shall be returned. If DESC is set to one and descriptor format sense data (see 4.5.2) is supported, descriptor format sense data shall be returned.

The ALLOCATION LENGTH field is defined in 4.3.4.6. Application clients should request 252 bytes of sense data to ensure they retrieve all the sense data. If fewer than 252 bytes are requested, sense data may be lost since the REQUEST SENSE command with any allocation length clears the sense data.

Sense data shall be available and cleared under the conditions defined in SAM-3. If the device server has no other sense data available to return, it shall return the sense key set to NO SENSE and the additional sense code set to NO ADDITIONAL SENSE INFORMATION.

If the logical unit is in a power condition other than the active power condition when a REQUEST SENSE command is received and there is no ACA condition, it shall return the sense key set to NO SENSE and the additional sense code set to one of the following:

a) LOW POWER CONDITION ON if the reason for entry into the power condition is unknown;
b) IDLE CONDITION ACTIVATED BY TIMER if the logical unit entered the idle power condition due to the idle condition timer (see 7.4.12);
c) STANDBY CONDITION ACTIVATED BY TIMER if the logical unit entered the standby power condition due to the standby condition timer (see 7.4.12);
d) IDLE CONDITION ACTIVATED BY COMMAND if the logical unit entered the idle power condition due to receipt of a command requiring the idle power condition while it was in the standby power condition; or
e) Another additional sense code based on requirements specified in a command standard (see 3.1.18).

On completion of the command the logical unit shall return to the same power condition that was active before the REQUEST SENSE command was received. A REQUEST SENSE command shall not reset any power condition timers.

The device server shall return CHECK CONDITION status for a REQUEST SENSE command only to report exception conditions specific to the REQUEST SENSE command itself.

Examples of conditions that cause a REQUEST SENSE command to return a CHECK CONDITION status are:

a) An invalid field value is detected in the CDB;
b) The device server does not support the REQUEST SENSE command (see 4.3.1);
c) An unrecovered error is detected by the service delivery subsystem; or
d) A malfunction prevents return of the sense data.

If a REQUEST SENSE command is received on an I_T nexus with a pending unit attention condition (i.e., before the device server reports CHECK CONDITION status) and there is an exception condition specific to the REQUEST SENSE command itself, then the device server shall not clear the pending unit attention condition (see SAM-3).

If a recovered error occurs during the processing of the REQUEST SENSE command, the device server shall return the sense data with GOOD status. If a device server returns CHECK CONDITION status for a REQUEST SENSE command, all sense data may be invalid.

In response to a REQUEST SENSE command issued to a logical unit that reports a peripheral qualifier of 011b in its standard INQUIRY data (see 6.4.2) the device server shall return GOOD status and parameter data that contains sense data. The sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to LOGICAL UNIT NOT SUPPORTED.

In response to a REQUEST SENSE command issued to a logical unit that reports a peripheral qualifier of 001b in its standard INQUIRY data, the device server shall return GOOD status and parameter data that contains sense data. The sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to LOGICAL UNIT NOT SUPPORTED.

In response to a REQUEST SENSE command issued to a logical unit that reports a peripheral qualifier of 000b in its standard INQUIRY data because it has a peripheral device connected but is not ready for access, the device server shall return GOOD status and parameter data that contains sense data appropriate to the condition that is making the logical unit not operational.

In response to a REQUEST SENSE command issued to a logical unit that reports a peripheral qualifier of 000b in its standard INQUIRY data because the device server is unable to determine whether or not a peripheral device is connected, the device server shall return GOOD status and parameter data that contains sense data with the sense key set to NO SENSE.

Device servers shall return at least 18 bytes of parameter data in response to a REQUEST SENSE command if the allocation length is 18 or greater and the DESC bit is set to zero. Application clients may determine how much sense data has been returned by examining the ALLOCATION LENGTH field in the CDB and the ADDITIONAL SENSE LENGTH field in the sense data. Device servers shall not adjust the additional sense length to reflect truncation if the allocation length is less than the sense data available.

6.28 SEND DIAGNOSTIC command

The SEND DIAGNOSTIC command (see table 170) requests the device server to perform diagnostic operations on the SCSI target device, on the logical unit, or on both. Logical units that support this command shall implement, at a minimum, the default self-test feature (i.e., the SELFTEST bit equal to one and a parameter list length of zero).

Table 170 — SEND DIAGNOSTIC command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (1Dh)							
1	SELF-TEST CODE			PF	Reserved	SELFTEST	DEVOFFL	UNITOFFL
2	Reserved							
3	(MSB)			PARAMETER LIST LENGTH				
4								(LSB)
5	CONTROL							

If the SELFTEST bit is set to one, the SELF-TEST CODE field shall contain 000b. If the SELFTEST bit is set to zero, the contents of SELF-TEST CODE field are specified in table 171.

Table 171 — SELF-TEST CODE field

Code	Name	Description
000b		This value shall be used when the SELFTEST bit is set to one, or when the SELFTEST bit is set to zero and the PF bit is set to one.
001b	Background short self-test	The device server shall start its short self-test (see 5.5.2) in the background mode (see 5.5.3.3). The PARAMETER LIST LENGTH field shall contain zero.
010b	Background extended self-test	The device server shall start its extended self-test (see 5.5.2) in the background mode (see 5.5.3.3). The PARAMETER LIST LENGTH field shall contain zero.
011b	Reserved	
100b	Abort background self-test	The device server shall abort the current self-test running in background mode. The PARAMETER LIST LENGTH field shall contain zero. This value is only valid if a previous SEND DIAGNOSTIC command specified a background self-test function and that self-test has not completed. If either of these conditions is not met, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.
101b	Foreground short self-test	The device server shall start its short self-test (see 5.5.2) in the foreground mode (see 5.5.3.2). The PARAMETER LIST LENGTH field shall contain zero.
110b	Foreground extended self-test	The device server shall start its extended self-test (see 5.5.2) in the foreground mode (see 5.5.3.2). The PARAMETER LIST LENGTH field shall contain zero.
111b	Reserved	

A page format (PF) bit set to one specifies that the SEND DIAGNOSTIC parameters and any parameters returned by a following RECEIVE DIAGNOSTIC RESULTS command with the PCV bit set to zero shall contain a single diagnostic page as defined in 7.1.

NOTE 31 - Logical units compliant with previous versions of this standard (e.g., SPC-2) may transfer more than one diagnostic page in the SEND DIAGNOSTIC command's parameter list and by doing so may request that more than one diagnostic page be transmitted in the RECEIVE DIAGNOSTIC RESULTS command's parameter data.

A PF bit set to zero specifies that all SEND DIAGNOSTIC parameters are vendor specific. If the PARAMETER LIST LENGTH field is set to zero and the SEND DIAGNOSTIC command is not going to be followed by a corresponding RECEIVE DIAGNOSTIC RESULTS command with the PCV bit set to zero, then the application client shall set the PF bit to zero. The implementation of the PF bit is optional.

A self-test (SELFTEST) bit set to one specifies that the device server shall perform the logical unit default self-test. If the self-test successfully passes, the command shall be terminated with GOOD status. If the self-test fails, the command shall be terminated with CHECK CONDITION status, with the sense key set to HARDWARE ERROR.

A SELFTEST bit set to zero specifies that the device server shall perform the diagnostic operation specified by the SELF-TEST CODE field or in the parameter list. The diagnostic operation may require the device server to return parameter data that contains diagnostic results. If the return of parameter data is not required, the return of GOOD status indicates successful completion of the diagnostic operation. If the return of parameter data is required, the device server shall either:

a) Perform the requested diagnostic operation, prepare the parameter data to be returned and indicate completion by returning GOOD status. The application client issues a RECEIVE DIAGNOSTIC RESULTS command to recover the parameter data; or
b) Accept the parameter list, and if no errors are detected in the parameter list, return GOOD status. The requested diagnostic operation and the preparation of the parameter data to be returned are performed upon receipt of a RECEIVE DIAGNOSTIC RESULTS command.

A unit offline (UNITOFFL) bit set to one specifies that the device server may perform diagnostic operations that may affect the user accessible medium on the logical unit (e.g., write operations to the user accessible medium, or repositioning of the medium on sequential access devices). The device server may ignore the UNITOFFL bit. A UNITOFFL bit set to zero prohibits any diagnostic operations that may be detected by subsequent tasks. When the SELFTEST bit is set to zero, the UNITOFFL bit shall be ignored.

A SCSI target device offline (DEVOFFL) bit set to one grants permission to the device server to perform diagnostic operations that may affect all the logical units in the SCSI target device (e.g., alteration of reservations, log parameters, or sense data). The device server may ignore the DEVOFFL bit. A DEVOFFL bit set to zero prohibits diagnostic operations that may be detected by subsequent tasks. When the SELFTEST bit is set to zero, the DEVOFFL bit shall be ignored.

The PARAMETER LIST LENGTH field specifies the length in bytes of the parameter list that shall be transferred from the application client Data-Out Buffer to the device server. A parameter list length of zero specifies that no data shall be transferred. This condition shall not be considered an error. If PF bit is set to one and the specified parameter list length results in the truncation of the diagnostic page (e.g., the parameter list length does not match the page length specified in the diagnostic page), then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

NOTE 32 - To ensure that the diagnostic command information is not destroyed by a command sent from another I_T nexus, the logical unit should be reserved.

6.29 SET DEVICE IDENTIFIER command

The SET DEVICE IDENTIFIER command (see table 172) requests that the device identifier information in the logical unit be set to the value received in the SET DEVICE IDENTIFIER parameter list. As defined in the SCC-2 standard, the SET DEVICE IDENTIFIER command is the SET PERIPHERAL DEVICE/COMPONENT DEVICE IDENTIFIER service action of the MAINTENANCE OUT command. Additional MAINTENANCE IN and MAINTENANCE OUT service actions are defined in SCC-2 and in this standard.

The MAINTENANCE OUT service actions defined only in SCC-2 shall apply only to SCSI devices that return a device type of 0Ch or the SCCS bit set to one in their standard INQUIRY data. When a SCSI devices returns a device type of 0Ch or the SCCS bit set to one in its standard INQUIRY data, the implementation requirements for the SCC-2 MAINTENANCE OUT service actions shall be as specified in SCC-2. Otherwise the MAINTENANCE OUT service action definitions and implementation requirements stated in this standard shall apply.

On successful completion of a SET DEVICE IDENTIFIER command that changes the device identifier saved by the logical unit, the device server shall establish a unit attention condition (see SAM-3) for the initiator port associated with every I_T nexus except the I_T nexus on which the SET IDENTIFIER command was received, with the additional sense code set to DEVICE IDENTIFIER CHANGED.

Table 172 — SET DEVICE IDENTIFIER command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (A4h)							
1	Reserved			SERVICE ACTION (06h)				
2	Reserved							
3								
4	Restricted							
5								
6	(MSB)			PARAMETER LIST LENGTH				
9								(LSB)
10	Reserved						Restricted	Reserved
11	CONTROL							

SCC-2 defines specific usages for bytes 4 and 5, and bit 1 in byte 10, however these fields are reserved for the SET DEVICE IDENTIFIER command defined by this standard.

The PARAMETER LIST LENGTH field specifies the length in bytes of the identifier that shall be transferred from the application client to the device server. The maximum value for this field shall be 512 bytes. A parameter list length of zero specifies that no data shall be transferred, and that subsequent REPORT DEVICE IDENTIFIER commands shall return an Identifier length of zero. Logical units that implement this command shall be capable of accepting a parameter list length of 64 bytes or less. If the parameter list length exceeds 64 bytes and the logical unit is not capable of storing the requested number of bytes, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

The SET DEVICE IDENTIFIER parameter list (see table 173) contains the identifier to be set by the addressed logical unit.

Table 173 — SET DEVICE IDENTIFIER parameter list

Bit / Byte	7	6	5	4	3	2	1	0
0	IDENTIFIER							
n								

The IDENTIFIER field is a value selected by the application client using mechanisms outside the scope of this standard to be returned in subsequent REPORT DEVICE IDENTIFIER commands.

6.30 SET PRIORITY command

The SET PRIORITY command (see table 174) requests that a priority be set to the specified value. The priority set by this command shall remain in effect until one of the following occurs:

a) Another SET PRIORITY command is received;
b) Hard reset;
c) Logical unit reset; or
d) Power off.

The priority set by this command shall not be affected by an I_T nexus loss.

The priority set by a SET PRIORITY command may be used as a task priority (see SAM-3) for tasks received by the logical unit via an I_T nexus (i.e., an I_T_L nexus).

The SET PRIORITY command is a service action of the MAINTENANCE OUT command. Additional MAINTENANCE OUT service actions are defined in SCC-2 and in this standard. The MAINTENANCE OUT service actions defined only in SCC-2 apply only to logical units that return a device type of 0Ch or the SCCS bit set to one in their standard INQUIRY data.

Table 174 — SET PRIORITY command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (A4h)							
1	Reserved			SERVICE ACTION (0Eh)				
2	I_T_L NEXUS TO SET			Reserved				
3	Reserved							
5								
6	(MSB)			PARAMETER LIST LENGTH				
9								(LSB)
10	Reserved							
11	CONTROL							

The I_T_L NEXUS TO SET field (see table 175) specifies the I_T_L nexus and the location of the priority value to be assigned to that I_T_L nexus.

Table 175 — I_T_L NEXUS TO SET field

Code	Description
00b	The priority for the I_T_L nexus associated with this command shall be set to the value contained in the PRIORITY TO SET field in the SET PRIORITY parameter list (see table 176). All fields in the SET PRIORITY parameter list except the PRIORITY TO SET field shall be ignored. If the parameter list length is zero, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to PARAMETER LIST LENGTH ERROR.
01b	The priority for the I_T_L nexus specified by the logical unit that is processing this command, the RELATIVE TARGET PORT IDENTIFIER field, and the TRANSPORTID field in the SET PRIORITY parameter list (see table 176) shall be set to the value specified by the PRIORITY TO SET field in the SET PRIORITY parameter list. If the parameter list length results in the truncation of the RELATIVE TARGET PORT IDENTIFIER field, the ADDITIONAL DESCRIPTOR LENGTH field, or the TRANSPORTID field, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to PARAMETER LIST LENGTH ERROR. On successful completion of a SET PRIORITY command a unit attention condition shall be established for the initiator port associated with the I_T nexus specified by the TRANSPORTID field and the RELATIVE TARGET PORT IDENTIFIER field, with the additional sense code set to PRIORITY CHANGED.
10b	The priority value specified in the INITIAL PRIORITY field of the Control Extension mode page (see 7.4.7) shall be used for all I_T_L nexuses associated with the logical unit that is processing this command regardless of any prior priority. The contents of the SET PRIORITY parameter list shall be ignored. On successful completion of a SET PRIORITY command a unit attention condition shall be established for the initiator port associated with every other I_T_L nexus, with the additional sense code set to PRIORITY CHANGED.
11b	Reserved

The PARAMETER LIST LENGTH field specifies the length in bytes of the SET PRIORITY parameter list (see table 176) that shall be contained in the Data-Out Buffer. A parameter list length of zero specifies that the Data-Out Buffer shall be empty. This condition shall not be considered as an error.

Table 176 — SET PRIORITY parameter list format

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved				PRIORITY TO SET			
1	Reserved							
2	(MSB)			RELATIVE TARGET PORT IDENTIFIER				
3								(LSB)
4	Reserved							
5	Reserved							
6	(MSB)			ADDITIONAL LENGTH (n-7)				
7								(LSB)
8	TRANSPORTID							
n								

The PRIORITY TO SET field specifies the priority to be assigned to the I_T_L nexus specified by the I_T_L NEXUS TO SET field in the CDB. The value in the PRIORITY TO SET field shall be returned in subsequent REPORT PRIORITY commands (see 6.22) until one of the conditions described in this subclause occurs. A priority to set value of zero specifies the I_T_L nexus specified by the I_T_L NEXUS TO SET field shall be set to the value specified in the INITIAL PRIORITY field of the Control Extension mode page (see 7.4.7). The contents of the I_T_L NEXUS TO SET field may specify that the PRIORITY TO SET field shall be ignored.

The RELATIVE TARGET PORT IDENTIFIER field contains the relative port identifier (see 3.1.88) of the target port that is part of the I_T_L nexus for which the priority is to be set. The contents of the I_T_L NEXUS TO SET field may specify that the RELATIVE TARGET PORT IDENTIFIER field shall be ignored.

The ADDITIONAL LENGTH field specifies the number of bytes that follow in the SET PRIORITY parameter list (i.e., the size of the TransportID).

The TRANSPORTID field contains a TransportID (see 7.5.4) identifying the initiator port that is part of the I_T_L nexus for which the priority is to be set. The contents of the I_T_L NEXUS TO SET field may specify that the TRANSPORTID field shall be ignored.

6.31 SET TARGET PORT GROUPS command

The SET TARGET PORT GROUPS command (see table 177) requests the device server to set the asymmetric access state of all of the target ports in the specified target port groups. See 5.8 for details regarding the transition between target port group asymmetric access states. This command is mandatory for all logical units that report in the standard INQUIRY data (see 6.4.2) that they support explicit asymmetric logical units access (i.e., the TPGS field contains either 10b or 11b).

The SET TARGET PORT GROUPS command is a service action of the MAINTENANCE OUT command. Additional MAINTENANCE OUT service actions are defined in SCC-2 and in this standard. The MAINTENANCE OUT service actions defined only in SCC-2 apply only to logical units that return a device type of 0Ch or the SCCS bit set to one in their standard INQUIRY data.

Table 177 — SET TARGET PORT GROUPS command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (A4h)							
1	Reserved			SERVICE ACTION (0Ah)				
2	Reserved							
5								
6	(MSB)			PARAMETER LIST LENGTH				
9								(LSB)
10	Reserved							
11	CONTROL							

The PARAMETER LIST LENGTH field specifies the length in bytes of the target port group management parameters that shall be transferred from the application client to the device server. A parameter list length of zero specifies that no data shall be transferred, and that no change shall be made in the asymmetric access state of any target port groups. If the parameter list length violates the vendor specific length requirements, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

The allowable values to which target port asymmetric access states may be set is vendor specific and should be reported in the REPORT TARGET PORT GROUP parameter data (see 6.25).

Target port groups that are not specified in a parameter list may change asymmetric access states as a result of the SET TARGET PORT GROUPS command. This shall not be considered an implicit target port group asymmetric access state change.

If the SET TARGET PORT GROUPS attempts to establish an invalid combination of target port asymmetric access states or attempts to establish an unsupported asymmetric access state, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

If the SET TARGET PORT GROUPS command has been performed, the completion of the command depends upon which of the following conditions apply:

a) If the transition is treated as a single indivisible event (see 5.8.2.5), then the SET TARGET PORT GROUPS command shall not complete until the transition to the requested state has completed; or
b) If the transition is not treated as a single indivisible event (i.e., the device server supports other commands (see 5.8.2.5) when those commands are routed though a target port that is transitioning between asymmetric access states), then the SET TARGET PORT GROUPS command may complete before the transition into the requested state has completed.

If the SET TARGET PORT GROUPS command is not performed successfully, the completion of the command depends upon which of the following conditions apply:

a) If the processing of a SET TARGET PORT GROUPS command requires the enabling of a nonvolatile memory and the nonvolatile memory is not ready, then the command shall be terminated with CHECK CONDITION status, rather than wait for the logical unit to become ready. The sense key shall be set to NOT READY and the additional sense code shall be set as described in table 184 (see 6.33); or
b) If a failure occurred before the transition was completed, the command shall be terminated with CHECK CONDITION status, with the sense key set to HARDWARE ERROR, and the additional sense code set to SET TARGET PORT GROUPS COMMAND FAILED.

If two SET TARGET PORT GROUPS commands are performed concurrently, the target port group asymmetric access state change behavior is vendor specific. A target should not process multiple SET TARGET PORT GROUPS concurrently.

The SET TARGET PORT GROUPS parameter data format is shown in table 178.

Table 178 — SET TARGET PORT GROUPS parameter list format

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved							
3								
	Set target port group descriptor(s)							
4	Set target port group descriptor 0 (see table 179)							
7								
	⋮							
n-3	Set target port group descriptor x (see table 179)							
n								

The format of the set target port group descriptor is defined in table 179.

Table 179 — Set target port group descriptor parameter list

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved				ASYMMETRIC ACCESS STATE			
1	Reserved							
2	(MSB)	TARGET PORT GROUP						
3								(LSB)

The ASYMMETRIC ACCESS STATE field (see table 180) specifies the asymmetric access state (see 5.8.2.4) to which all of the target ports in the specified target port group shall transition (see 5.8.2.5).

Table 180 — ASYMMETRIC ACCESS STATE field

Value	State
0h	Active/optimized
1h	Active/non-optimized
2h	Standby
3h	Unavailable
4h-Eh	Reserved
Fh	Illegal Request [a]

[a] If the ASYMMETRIC ACCESS STATE field is any set target port group descriptor contains Fh, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

The TARGET PORT GROUP field specifies a target port group for which the asymmetric access state shall be changed.

6.32 SET TIMESTAMP command

The SET TIMESTAMP command (see table 181) requests the device server to initialize the timestamp (see 5.13), if the SCSIP bit is set to one or the TCMOS bit is set to one in the Control Extension mode page (see 7.4.7). If the SCSIP bit is set to zero, the SET TIMESTAMP command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

The SET TIMESTAMP command is a service action of the MAINTENANCE OUT command. Additional MAINTENANCE OUT service actions are defined in SCC-2 and in this standard. The MAINTENANCE OUT service actions defined only in SCC-2 apply only to logical units that return a device type of 0Ch or the SCCS bit set to one in their standard INQUIRY data (see 6.4.2).

Table 181 — SET TIMESTAMP command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (A4h)							
1	Reserved			SERVICE ACTION (0Fh)				
2	Reserved							
5								
6	(MSB)			PARAMETER LIST LENGTH				
9								(LSB)
10	Reserved							
11	CONTROL							

The PARAMETER LIST LENGTH field specifies the length in bytes of the SET TIMESTAMP parameters that shall be transferred from the application client to the device server. A parameter list length of zero indicates that no data shall be transferred, and that no change shall be made to the timestamp.

The format for the parameter data returned by the SET TIMESTAMP command is shown in table 182.

Table 182 — SET TIMESTAMP parameter data format

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved							
3								
4	TIMESTAMP							
9								
10	Reserved							
11	Reserved							

The TIMESTAMP field shall contain the initial value of the timestamp in the format defined in 5.13. The timestamp should be the number of milliseconds that have elapsed since midnight, 1 January 1970 UT. If the high order byte in the TIMESTAMP field is greater than F0h, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

On successful completion of a SET TIMESTAMP command the device server shall generate a unit attention condition for the initiator port associated with every I_T nexus except the I_T nexus on which the SET TIMESTAMP command was received (see SAM-3), with the additional sense code set to TIMESTAMP CHANGED.

6.33 TEST UNIT READY command

The TEST UNIT READY command (see table 183) provides a means to check if the logical unit is ready. This is not a request for a self-test. If the logical unit is able to accept an appropriate medium-access command without returning CHECK CONDITION status, this command shall return a GOOD status. If the logical unit is unable to become operational or is in a state such that an application client action (e.g., START UNIT command) is required to make the logical unit ready, the command shall be terminated with CHECK CONDITION status, with the sense key set to NOT READY.

Table 183 — TEST UNIT READY command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (00h)							
1	Reserved							
4								
5	CONTROL							

Table 184 defines the suggested GOOD and CHECK CONDITION status responses to the TEST UNIT READY command. Other conditions, including deferred errors, may result in other responses (e.g., BUSY or RESERVATION CONFLICT status).

Table 184 — Preferred TEST UNIT READY responses

Status	Sense Key	Additional Sense Code
GOOD	not applicable	not applicable
CHECK CONDITION	ILLEGAL REQUEST	LOGICAL UNIT NOT SUPPORTED
CHECK CONDITION	NOT READY	LOGICAL UNIT DOES NOT RESPOND TO SELECTION
CHECK CONDITION	NOT READY	MEDIUM NOT PRESENT
CHECK CONDITION	NOT READY	LOGICAL UNIT NOT READY, CAUSE NOT REPORTABLE
CHECK CONDITION	NOT READY	LOGICAL UNIT IS IN PROCESS OF BECOMING READY
CHECK CONDITION	NOT READY	LOGICAL UNIT NOT READY, INITIALIZING COMMAND REQUIRED
CHECK CONDITION	NOT READY	LOGICAL UNIT NOT READY, MANUAL INTERVENTION REQUIRED
CHECK CONDITION	NOT READY	LOGICAL UNIT NOT READY, FORMAT IN PROGRESS

6.34 WRITE ATTRIBUTE command

The WRITE ATTRIBUTE command (see table 185) allows an application client to write attributes to medium auxiliary memory. Device servers that implement the WRITE ATTRIBUTE command shall also implement the READ ATTRIBUTE command (see 6.14). Application clients should issue READ ATTRIBUTE commands prior to using this command to discover device server support for medium auxiliary memory.

Table 185 — WRITE ATTRIBUTE command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (8Dh)							
1	Reserved							
2	Restricted (see SMC-2)							
4								
5	VOLUME NUMBER							
6	Reserved							
7	PARTITION NUMBER							
8	Reserved							
9								
10	(MSB)							
13	PARAMETER LIST LENGTH							(LSB)
14	Reserved							
15	CONTROL							

The VOLUME NUMBER field specifies a volume (see SSC-2) within the medium auxiliary memory. The number of volumes of the medium auxiliary memory shall equal that of the attached medium. If the medium only has a single volume, then its volume number shall be zero.

The PARTITION NUMBER field specifies a partition (see SSC-2) within a volume. The number of partitions of the medium auxiliary memory shall equal that of the attached medium. If the medium only has a single partition, then its partition number shall be zero.

If the combination of volume number and partition number is not valid, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

The PARAMETER LIST LENGTH field specifies the length in bytes of the parameter list contained in the Data-Out Buffer. A parameter list length of zero specifies that no parameter data is present; this shall not be considered an error. If the parameter list length results in the truncation of an attribute, the WRITE ATTRIBUTE command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to PARAMETER LIST LENGTH ERROR.

The parameter list shall have the format shown in table 186. Attributes should be sent in ascending numerical order. If the attributes are not in order, then no attributes shall be changed and the WRITE ATTRIBUTE command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

Table 186 — WRITE ATTRIBUTE parameter list format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
3	PARAMETER DATA LENGTH (n-3)							(LSB)
	Attribute(s)							
4	Attribute 0 (see 7.3.1)							
	. . .							
n	Attribute x (see 7.3.1)							

The PARAMETER DATA LENGTH field should contain the number of bytes of attribute data and shall be ignored by the device server.

The format of the attributes is described in 7.3.1.

If there is not enough space to write the attributes to the medium auxiliary memory, then no attributes shall be changed and the WRITE ATTRIBUTE command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to AUXILIARY MEMORY OUT OF SPACE.

If the medium auxiliary memory is not accessible because there is no medium present, then no attributes shall be changed and the WRITE ATTRIBUTE command shall be terminated with CHECK CONDITION status, with the sense key set to NOT READY, and the additional sense code set to MEDIUM NOT PRESENT.

If the medium is present but the medium auxiliary memory is not accessible, then no attributes shall be changed and the WRITE ATTRIBUTE command shall be terminated with CHECK CONDITION status, with the sense key set to MEDIUM ERROR, and the additional sense code set to LOGICAL UNIT NOT READY, AUXILIARY MEMORY NOT ACCESSIBLE.

If the medium auxiliary memory is not operational (e.g., bad checksum), the WRITE ATTRIBUTE command shall be terminated with CHECK CONDITION status, with the sense key set to MEDIUM ERROR, and the additional sense code set to AUXILIARY MEMORY WRITE ERROR.

If the WRITE ATTRIBUTE command parameter data contains an attribute with an ATTRIBUTE LENGTH field (see 7.3.1) set to zero, then one of the following actions shall occur:

a) If the attribute state is unsupported or read only (see 5.11), then no attributes shall be changed and the WRITE ATTRIBUTE command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST;
b) If the attribute state is read/write, the attribute shall be changed to the nonexistent state. This attribute shall not be returned in response to a READ ATTRIBUTE command and not be reported by the READ ATTRIBUTE command with ATTRIBUTE LIST service action; or
c) If the attribute state is nonexistent, the attribute in the WRITE ATTRIBUTE command parameter list shall be ignored; this shall not be considered an error.

No attributes shall be changed, the WRITE ATTRIBUTE command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST if the parameter data contains any of the following:

a) An attempt to change an attribute in the read only state (see 5.11);
b) An attribute with incorrect ATTRIBUTE LENGTH field (see 7.3.1) contents; or
c) An attribute with unsupported ATTRIBUTE VALUE field (see 7.3.1) contents.

6.35 WRITE BUFFER command

6.35.1 WRITE BUFFER command introduction

The WRITE BUFFER command (see table 187) is used in conjunction with the READ BUFFER command as a diagnostic function for testing logical unit memory in the SCSI target device and the integrity of the service delivery subsystem. Additional modes are provided for:

a) Downloading microcode;
b) Downloading and saving microcode; and
c) Downloading application logs (see 5.12).

Table 187 — WRITE BUFFER command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (3Bh)							
1	Reserved			MODE				
2	BUFFER ID							
3	(MSB)							
5	BUFFER OFFSET							(LSB)
6	(MSB)							
8	PARAMETER LIST LENGTH							(LSB)
9	CONTROL							

This command shall not alter any medium of the logical unit when the data mode or the combined header and data mode is specified.

The function of this command and the meaning of fields within the CDB depend on the contents of the MODE field. The MODE field is defined in table 188.

Table 188 — WRITE BUFFER MODE field (part 1 of 2)

MODE	Description
00h	Write combined header and data [a]
01h	Vendor specific [a]
02h	Write data
04h	Download microcode
05h	Download microcode and save

[a] Modes 00h and 01h are not recommended.

[b] When downloading microcode with buffer offsets, the WRITE BUFFER command mode should be 06h or 07h.

Table 188 — WRITE BUFFER MODE field (part 2 of 2)

MODE	Description
06h	Download microcode with offsets [b]
07h	Download microcode with offsets and save [b]
0Ah	Echo buffer
1Ah	Enable expander communications protocol and Echo buffer
1Bh	Disable expander communications protocol
1Ch	Download application log
03h	Reserved
08h - 09h	Reserved
0Bh - 19h	Reserved
1Dh - 1Fh	Reserved

[a] Modes 00h and 01h are not recommended.

[b] When downloading microcode with buffer offsets, the WRITE BUFFER command mode should be 06h or 07h.

6.35.2 Combined header and data mode (00h)

In this mode, data to be transferred is preceded by a four-byte header. The four-byte header consists of all reserved bytes. The BUFFER ID and the BUFFER OFFSET fields shall be zero. The PARAMETER LIST LENGTH field specifies the maximum number of bytes that shall be transferred from the Data-Out Buffer. This number includes four bytes of header, so the data length to be stored in the device server's buffer is parameter list length minus four. The application client should attempt to ensure that the parameter list length is not greater than four plus the BUFFER CAPACITY field value (see 6.15.2) that is returned in the header of the READ BUFFER command (mode 0h). If the parameter list length exceeds the buffer capacity, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

6.35.3 Vendor specific mode (01h)

In this mode, the meaning of the BUFFER ID, BUFFER OFFSET, and PARAMETER LIST LENGTH fields are not specified by this standard.

6.35.4 Data mode (02h)

In this mode, the Data-Out Buffer contains buffer data destined for the logical unit. The BUFFER ID field identifies a specific buffer within the logical unit. The vendor assigns buffer ID codes to buffers within the logical unit. Buffer ID zero shall be supported. If more than one buffer is supported, then additional buffer ID codes shall be assigned contiguously, beginning with one. If an unsupported buffer ID code is selected, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

Data are written to the logical unit buffer starting at the location specified by the BUFFER OFFSET field. The application client should conform to the offset boundary requirements returned in the READ BUFFER descriptor. If the device server is unable to accept the specified buffer offset, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

The PARAMETER LIST LENGTH field specifies the maximum number of bytes that shall be transferred from the Data-Out Buffer to be stored in the specified buffer beginning at the buffer offset. The application client should attempt to ensure that the parameter list length plus the buffer offset does not exceed the capacity of the specified buffer. The capacity of the buffer is indicated by the BUFFER CAPACITY field in the READ BUFFER descriptor (see 6.15.5). If the BUFFER OFFSET and PARAMETER LIST LENGTH fields specify a transfer in excess of the buffer capacity,

the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

6.35.5 Download microcode mode (04h)

In this mode, vendor specific microcode or control information shall be transferred to the control memory space of the logical unit. After a hard reset, the device operation shall revert to a vendor specific condition. The meanings of the BUFFER ID, BUFFER OFFSET, and PARAMETER LIST LENGTH fields are not specified by this standard and are not required to be zero-filled. When the microcode download has completed successfully the device server shall establish a unit attention condition (see SAM-3) for the initiator port associated with every I_T nexus except the I_T nexus on which the WRITE BUFFER command was received, with the additional sense code set to MICROCODE HAS BEEN CHANGED.

If the logical unit is unable to accept this command because of some device condition, each WRITE BUFFER command with this mode (04h) shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to COMMAND SEQUENCE ERROR.

6.35.6 Download microcode and save mode (05h)

In this mode, vendor specific microcode or control information shall be transferred to the logical unit and, if the WRITE BUFFER command is completed successfully, also shall be saved in a nonvolatile memory space (e.g., semiconductor, disk, or other). The downloaded code shall then be effective after each hard reset until it is supplanted in another download microcode and save operation or download microcode with offsets and save operation. The meanings of the BUFFER ID, BUFFER OFFSET, and PARAMETER LIST LENGTH fields are not specified by this standard and are not required to be zero-filled. When the download microcode and save command has completed successfully the device server shall establish a unit attention condition (see SAM-3) for the initiator port associated with every I_T nexus except the I_T nexus on which the WRITE BUFFER command was received with the additional sense code set to MICROCODE HAS BEEN CHANGED.

If the logical unit is unable to accept this command because of some device condition, each WRITE BUFFER command with this mode (05h) shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to COMMAND SEQUENCE ERROR.

6.35.7 Download microcode with offsets mode (06h)

In this mode, the application client may split the transfer of the vendor specific microcode or control information over two or more WRITE BUFFER commands. If the last WRITE BUFFER command of a set of one or more commands completes successfully, then the microcode or control information shall be transferred to the control memory space of the logical unit. After a hard reset, the device shall revert to a vendor specific condition. In this mode, the Data-Out Buffer contains vendor specific, self-describing microcode or control information.

Since the downloaded microcode or control information may be sent using several commands, when the logical unit detects the last download microcode with offsets WRITE BUFFER command has been received, the device server shall perform any logical unit required verification of the complete set of downloaded microcode or control information prior to returning GOOD status for the last command. After the last command completes successfully the device server shall establish a unit attention condition (see SAM-3) for the initiator port associated with every I_T nexus except the I_T nexus on which the set of WRITE BUFFER commands was received, with the additional sense code set to MICROCODE HAS BEEN CHANGED.

If the complete set of WRITE BUFFER commands required to effect a microcode or control information change (i.e., one or more commands) are not received before a logical unit reset or I_T nexus loss occurs, the change shall not be effective and the new microcode or control information shall be discarded.

The BUFFER ID field specifies a buffer within the logical unit. The vendor assigns buffer ID codes to buffers within the logical unit. A buffer ID value of zero shall be supported. If more than one buffer is supported, then additional buffer ID codes shall be assigned contiguously, beginning with one. If an unsupported buffer ID code is specified, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

The microcode or control information are written to the logical unit buffer starting at the location specified by the BUFFER OFFSET field. The application client shall send commands that conform to the offset boundary requirements (see 6.15.5). If the device server is unable to accept the specified buffer offset, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

The PARAMETER LIST LENGTH field specifies the maximum number of bytes that shall be present in the Data-Out Buffer to be stored in the specified buffer beginning at the buffer offset. The application client should ensure that the parameter list length plus the buffer offset does not exceed the capacity of the specified buffer. The capacity of the buffer is indicated by the BUFFER CAPACITY field in the READ BUFFER descriptor (see 6.15.5). If the BUFFER OFFSET and PARAMETER LIST LENGTH fields specify a transfer in excess of the buffer capacity, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

If the logical unit is unable to accept this command because of some device condition, each WRITE BUFFER command with this mode (06h) shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to COMMAND SEQUENCE ERROR.

6.35.8 Download microcode with offsets and save mode (07h)

In this mode, the application client may split the transfer of the vendor specific microcode or control information over two or more WRITE BUFFER commands. If the last WRITE BUFFER command of a set of one or more commands completes successfully, then the microcode or control information shall be saved in a nonvolatile memory space (e.g., semiconductor, disk, or other). The saved downloaded microcode or control information shall then be effective after each hard reset until it is supplanted by another download microcode with save operation or download microcode with offsets and save operation. In this mode, the Data-Out Buffer contains vendor specific, self-describing microcode or control information.

Since the downloaded microcode or control information may be sent using several commands, when the logical unit detects the last download microcode with offsets and save mode WRITE BUFFER command has been received, the device server shall perform any logical unit required verification of the complete set of downloaded microcode or control information prior to returning GOOD status for the last command. After the last command completes successfully the device server shall establish a unit attention condition (see SAM-3) for the initiator port associated with every I_T nexus except the I_T nexus on which the set of WRITE BUFFER commands was received, with the additional sense code set to MICROCODE HAS BEEN CHANGED.

If the complete set of WRITE BUFFER commands required to effect a microcode or control information change (i.e., one or more commands) are not received before a logical unit reset or I_T nexus loss occurs, the change shall not be effective and the new microcode or control information shall be discarded.

The BUFFER ID field specifies a buffer within the logical unit. The vendor assigns buffer ID codes to buffers within the logical unit. A buffer ID value of zero shall be supported. If more than one buffer is supported, then additional buffer ID codes shall be assigned contiguously, beginning with one. If an unsupported buffer ID code is specified, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

The microcode or control information are written to the logical unit buffer starting at the location specified by the BUFFER OFFSET field. The application client shall conform to the offset boundary requirements. If the device server is unable to accept the specified buffer offset, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

The PARAMETER LIST LENGTH field specifies the maximum number of bytes that shall be present in the Data-Out Buffer to be stored in the specified buffer beginning at the buffer offset. The application client should ensure that the parameter list length plus the buffer offset does not exceed the capacity of the specified buffer. The capacity of the buffer is indicated by the BUFFER CAPACITY field in the READ BUFFER descriptor (see 6.15.5). If the BUFFER OFFSET and PARAMETER LIST LENGTH fields specify a transfer in excess of the buffer capacity, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

If the logical unit is unable to accept this command because of some device condition, each WRITE BUFFER command with this mode (07h) shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to COMMAND SEQUENCE ERROR.

6.35.9 Write data to echo buffer mode (0Ah)

In this mode the device server transfers data from the application client and stores it in an echo buffer. An echo buffer is assigned in the same manner by the device server as it would for a write operation. Data shall be sent aligned on four-byte boundaries. The BUFFER ID and BUFFER OFFSET fields are ignored in this mode.

NOTE 33 - It is recommended that the logical unit assign echo buffers on a per I_T nexus basis to limit the number of exception conditions that may occur when I_T nexuses are present.

Upon successful completion of a WRITE BUFFER command the data shall be preserved in the echo buffer unless there is an intervening command to any logical unit in which case the data may be changed.

The PARAMETER LIST LENGTH field specifies the maximum number of bytes that shall be transferred from the Data-Out Buffer to be stored in the echo buffer. The application client should ensure that the parameter list length does not exceed the capacity of the echo buffer. The capacity of the echo buffer is indicated by the BUFFER CAPACITY field in the READ BUFFER echo buffer descriptor (see 6.15.7). If the PARAMETER LIST LENGTH field specifies a transfer in excess of the buffer capacity, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

6.35.10 Enable expander communications protocol and Echo buffer mode (1Ah)

Receipt of a WRITE BUFFER command with this mode (1Ah) causes a communicative expander (see SPI-5) to enter the expanded communications protocol mode. Device servers in SCSI target devices that receive a WRITE BUFFER command with this mode shall process it as if it were a WRITE BUFFER command with mode 0Ah (see 6.35.9).

6.35.11 Disable expander communications protocol mode (1Bh)

Receipt of a WRITE BUFFER command with this mode (1Bh) causes a communicative expander (see SPI-5) to exit the expanded communications protocol mode and return to simple expander operation. Device servers in SCSI target devices that receive a WRITE BUFFER command with this mode shall terminate the command with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

6.35.12 Download application log mode (1Ch)

In this mode the device server transfers data from the application client and stores it in an application log (see 5.12). The format of the application log data is as specified in table 189. The BUFFER ID field and BUFFER OFFSET field are ignored in this mode.

Upon successful completion of a WRITE BUFFER command the data shall be appended to the application log.

The PARAMETER LIST LENGTH field specifies the maximum number of bytes that shall be transferred from the Data-Out Buffer to be stored in the application log. If the PARAMETER LIST LENGTH field specifies a transfer that exceeds the application log's capacity, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

Table 189 — Application log data WRITE BUFFER format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)			T10 VENDOR IDENTIFICATION				
7								(LSB)
8	(MSB)			ERROR TYPE				
9								(LSB)
10				Reserved				
11								
12	(MSB)			TIME STAMP				
17								(LSB)
18				Reserved				
19								
20	Reserved					CODE SET		
21	ERROR LOCATION FORMAT							
22	(MSB)			ERROR LOCATION LENGTH (m-25)				
23								(LSB)
24	(MSB)			VENDOR SPECIFIC LENGTH (n-m)				
25								(LSB)
26	(MSB)			ERROR LOCATION				
m								(LSB)
m+1				VENDOR SPECIFIC				
n								

The T10 VENDOR IDENTIFICATION field contains eight bytes of left-aligned ASCII data (see 4.4.1) identifying the vendor of the product. The T10 vendor identification shall be one assigned by INCITS. A list of assigned T10 vendor identifications is in Annex E and on the T10 web site (http://www.T10.org).

The ERROR TYPE field (see table 190) specifies the error detected by the application client.

Table 190 — ERROR TYPE field

Code	Description
0000h	No error specified by the application client
0001h	An unknown error was detected by the application client
0002h	The application client detected corrupted data
0003h	The application client detected a permanent error
0004h	The application client detected a service response of SERVICE DELIVERY OR TARGET FAILURE (SAM-3).
0005h - 7FFFh	Reserved
8000h - FFFFh	Vendor specific

The TIME STAMP field shall contain:

a) The number of milliseconds that have elapsed since midnight, 1 January 1970 UT (see 3.1.122); or
b) Zero, if the application client is not able to determine the UT of the log entry.

The CODE SET field specifies the code set used for the application log information (see table 191) and shall only apply to information contained in the VENDOR SPECIFIC field.

NOTE 34 - The CODE SET field is intended to be an aid to software that displays the application log information.

Table 191 — CODE SET field

Code	Description
0h	Reserved
1h	The application log information is binary
2h	The application log information is ASCII printable characters (i.e., code values 20h through 7Eh)
3h	The application log information is ISO/IEC 10646-1 (UTF-8) codes
4h - Fh	Reserved

The ERROR LOCATION FORMAT field specifies the format (see table 192) of the ERROR LOCATION field.

Table 192 — ERROR LOCATION FORMAT field

Code	Description
00h	No error specified by the application client
01h	The ERROR LOCATION field specifies the logical block (e.g., LBA) associated with the error information contained within the application log.
02h - 7Fh	Reserved
80h - FFh	Vendor specific

The ERROR LOCATION LENGTH field specifies the length of the ERROR LOCATION field. The ERROR LOCATION LENGTH field value shall be a multiple of four. An error location length value of zero specifies there is no error location information.

The VENDOR SPECIFIC LENGTH field specifies the length of the VENDOR SPECIFIC field. The VENDOR SPECIFIC LENGTH field value shall be a multiple of four. A vendor specific length value of zero specifies there is no vendor specific information.

The ERROR LOCATION field specifies the location at which the application client detected the error.

The VENDOR SPECIFIC field provides vendor specific information on the error.

7 Parameters for all device types

7.1 Diagnostic parameters

7.1.1 Diagnostic page format and page codes for all device types

This subclause describes the diagnostic page structure and the diagnostic pages that are applicable to all SCSI devices. Diagnostic pages specific to each device type are described in the command standard (see 3.1.18) that applies to that device type.

A SEND DIAGNOSTIC command with a PF bit set to one specifies that the SEND DIAGNOSTIC parameter list consists of a single diagnostic page and that the data returned by the subsequent RECEIVE DIAGNOSTIC RESULTS command that has the PCV bit set to zero shall use the diagnostic page format defined in table 193. A RECEIVE DIAGNOSTIC RESULTS command with a PCV bit set to one specifies that the device server return a diagnostic page using the format defined in table 193.

Table 193 — Diagnostic page format

Bit / Byte	7	6	5	4	3	2	1	0
0	PAGE CODE							
1	Page code specific							
2	(MSB)			PAGE LENGTH (n-3)				
3								(LSB)
4	Diagnostic parameters							
n								

Each diagnostic page defines a function or operation that the device server shall perform as a result of a SEND DIAGNOSTIC command or the information being returned as a result of a RECEIVE DIAGNOSTIC RESULTS command with the PCV bit equal to one. The diagnostic parameters contain data that is formatted according to the page code specified.

The PAGE CODE field identifies the diagnostic page (see table 194).

Table 194 — Diagnostic page codes

Page Code	Diagnostic Page Name	Reference
00h	Supported Diagnostic Pages	7.1.2
01h - 2Fh	Defined by SES-2 for: a) Enclosure services devices (i.e., SCSI devices with the PERIPHERAL DEVICE TYPE field set to 0Dh in standard INQUIRY data); and b) SCSI devices with the ENCSERV bit set to one in standard INQUIRY data (see 6.4.2).	SES-2
30h - 3Eh	Reserved	
3Fh	See specific SCSI transport protocol for definition	
40h - 7Fh	See specific device type for definition	
80h - FFh	Vendor specific	

The PAGE LENGTH field contains the length in bytes of the diagnostic parameters that follow this field. If the application client sends a SEND DIAGNOSTIC command with a parameter list containing a PAGE LENGTH field that

results in the truncation of any parameter, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

The diagnostic parameters are defined for each diagnostic page code. The diagnostic parameters within a diagnostic page may be defined differently in a SEND DIAGNOSTIC command than in a RECEIVE DIAGNOSTIC RESULTS command.

7.1.2 Supported diagnostic pages

The Supported Diagnostic Pages diagnostic page (see table 195) returns the list of diagnostic pages implemented by the device server. This diagnostic page shall be implemented if the device server implements the diagnostic page format option of the SEND DIAGNOSTIC and RECEIVE DIAGNOSTIC RESULTS commands.

Table 195 — Supported diagnostic pages

Bit / Byte	7	6	5	4	3	2	1	0
0	PAGE CODE (00h)							
1	Reserved							
2	(MSB)			PAGE LENGTH (n-3)				
3								(LSB)
4	SUPPORTED PAGE LIST							
n								

The definition of this diagnostic page for the SEND DIAGNOSTIC command includes only the first four bytes. If the PAGE LENGTH field is not zero, the device server shall terminate the SEND DIAGNOSTIC command with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST. This diagnostic page instructs the device server to make available the list of all supported diagnostic pages to be returned by a subsequent RECEIVE DIAGNOSTIC RESULTS command.

The definition of this diagnostic page for the RECEIVE DIAGNOSTIC RESULTS command includes the list of diagnostic pages supported by the device server.

The PAGE LENGTH field specifies the length in bytes of the following supported page list.

The SUPPORTED PAGE LIST field shall contain a list of all diagnostic page codes, one per byte, implemented by the device server in ascending order beginning with page code 00h.

7.2 Log parameters

7.2.1 Log page structure and page codes for all device types

This subclause describes the log page structure and the log pages that are applicable to all SCSI devices. Log pages specific to each device type are described in the command standard (see 3.1.18) that applies to that device type. The LOG SELECT command (see 6.5) supports the ability to send zero or more log pages. The LOG SENSE command (see 6.6) returns a single log page specified in the PAGE CODE field of the CDB.

Each log page begins with a four-byte page header followed by zero or more variable-length log parameters defined for that log page. The log page format is defined in table 196.

Table 196 — Log page format

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved			PAGE CODE				
1	Reserved							
2	(MSB)			PAGE LENGTH (n-3)				
3								(LSB)
	Log parameter(s)							
4				Log parameter (First)				
x+3				(Length x)				
				.				
				.				
n-y+1				Log parameter (Last)				
n				(Length y)				

The value in the PAGE CODE field is the number of the log page is being transferred.

The value in the PAGE LENGTH field is the length in bytes of the following log parameters. If the application client sends a log page length that results in the truncation of any parameter, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

Most log pages contain one or more special data structures called log parameters (see table 197). Log parameters may be data counters of a particular event(s), the conditions under which certain operations were performed, or list parameters that contain a character string description of a particular event.

Table 197 — Log parameter

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)			PARAMETER CODE				
1								(LSB)
2	DU	DS	TSD	ETC	TMC		LBIN	LP
3	PARAMETER LENGTH (n-3)							
4				PARAMETER VALUE				
n								

Each log parameter begins with a four-byte parameter header followed by one or more bytes of PARAMETER VALUE data.

The PARAMETER CODE field identifies the log parameter being transferred for that log page.

The DU bit, DS bit, TSD bit, ETC bit, TMC field, LBIN bit, and LP bit are collectively referred to as the parameter control byte. These fields are described in this subclause.

For cumulative log parameter values, indicated by the PC field of the LOG SELECT and LOG SENSE commands, the disable update (DU) bit is defined as follows:

a) DU set to zero indicates that the device server shall update the log parameter value to reflect all events that should be noted by that parameter; or
b) DU set to one indicates that the device server shall not update the log parameter value except in response to a LOG SELECT command that specifies a new value for the parameter.

NOTE 35 - When updating cumulative log parameter values, a device server may use volatile memory to hold these values until a LOG SELECT or LOG SENSE command is received with an SP bit set to one or a vendor specific event occurs. As a result the updated cumulative log parameter values may be lost if a power cycle occurs.

The DU bit is not defined for threshold values, indicated by the PC field of the LOG SENSE command, or for list parameters as indicated by the LP bit. The device server shall ignore the value of the DU bit in any such log parameters received with a LOG SELECT command.

A disable save (DS) bit set to zero indicates that the logical unit supports saving for that log parameter. The device server shall save the current cumulative or the current threshold parameter value, depending on the value in the PC field of the CDB, in response to a LOG SELECT or LOG SENSE command with an SP bit set to one. A DS bit set to one indicates that the logical unit does not support saving that log parameter in response to a LOG SELECT or LOG SENSE command with an SP bit set to one.

A target save disable (TSD) bit set to zero indicates that the logical unit implicitly saves the log parameter at vendor specific intervals. This implicit saving operation shall be done frequently enough to insure that the cumulative parameter values retain statistical significance (i.e., across power cycles). A TSD bit set to one indicates that either the logical unit does not implicitly save the log parameter or implicit saving of the log parameter has been disabled individually by an application client setting the TSD bit to one. An application client may disable the implicit saving for all log parameters without changing any TSD bits using the GLTSD bit in the Control mode page (see 7.4.6).

An enable threshold comparison (ETC) bit set to one indicates that a comparison to the threshold value is performed whenever the cumulative value is updated. An ETC bit set to zero indicates that a comparison is not performed. The value of the ETC bit is the same for cumulative and threshold parameters.

The threshold met criteria (TMC) field (see table 198) defines the basis for comparison of the cumulative and threshold values. The TMC field is valid only if the ETC bit is set to one. The value of the TMC field is the same for cumulative and threshold parameters.

Table 198 — Threshold met criteria

Code	Basis for comparison		
00b	Every update of the cumulative value		
01b	Cumulative value	equal to	threshold value
10b	Cumulative value	not equal to	threshold value
11b	Cumulative value	greater than	threshold value

If the ETC bit is set to one and the result of the comparison is true, a unit attention condition shall be established for the initiator port associated with every I_T nexus, with the additional sense code set to THRESHOLD CONDITION MET.

The list binary (LBIN) bit is only valid if the LP bit is set to one. If the LP bit is set to one and the LBIN bit is set to zero, then the list parameter is ASCII data (see 4.4.1). If the LP bit is set to one and the LBIN bit is set to one, then the list parameter is binary data.

The list parameter (LP) bit indicates the format of the log parameter. If an application client attempts to set the value of the LP bit to a value other than the one returned for the same parameter in the LOG SENSE command, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

An LP bit set to zero indicates that the parameter is a data counter. Data counters are associated with one or more events; the data counter is updated whenever one of these events occurs by incrementing the counter value. If each data counter has associated with it a vendor specific maximum value, then upon reaching this maximum value, the data counter shall not be incremented (i.e., it does not wrap). When a data counter reaches its maximum value, the device server shall set the associated DU bit to one. If the data counter is at or reaches its maximum value during the processing of a command, the device server shall complete the command. If the command completes correctly, except for the data counter being at its maximum value, and if the RLEC bit of the Control mode page (see 7.4.6) is set to one, then the device server shall terminate the command with CHECK CONDITION status, with the sense key set to RECOVERED ERROR, and the additional sense code set to LOG COUNTER AT MAXIMUM.

An LP bit set to one indicates that the parameter is a list parameter. List parameters are not counters and thus the ETC and TMC fields shall be set to zero.

If more than one list parameter is defined in a single log page, the following rules apply to assigning parameter codes:

a) The parameter updated last shall have a higher parameter code than the previous parameter, except as defined in rule b); and
b) When the maximum parameter code value supported by the logical unit is reached, the device server shall assign the lowest parameter code value to the next log parameter (i.e., wrap-around parameter codes). If the associated command completes correctly, except for the parameter code being at its maximum value, and if the RLEC bit of the Control mode page (see 7.4.6) is set to one, then the command shall be terminated with CHECK CONDITION status, with the sense key set to RECOVERED ERROR, and the additional sense code set to LOG LIST CODES EXHAUSTED.

NOTE 36 - List parameters may be used to store the locations of defective blocks in the following manner. When a defective block is identified, a list parameter is updated to reflect the location and cause of the defect. When the next defect is encountered, the list parameter with the next higher parameter code is updated to record this defect. The size of the log page may be made vendor specific to accommodate memory limitations. It is recommended that one or more data counter parameters be defined for the log page to keep track of the number of valid list parameters and the parameter code of the parameter with the oldest recorded defect. This technique may be adapted to record other types of information.

The PARAMETER LENGTH field specifies the length in bytes of the following parameter value. If the application client sends a parameter length value that results in the truncation of the parameter value, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

If the application client sends a log parameter value that is outside the range supported by the logical unit, and rounding is implemented for that parameter, the device server may either:

a) Round to an acceptable value and terminate the command as described in 5.4; or
b) Terminate the command with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

When any counter in a log page reaches its maximum value, incrementing of all counters in that log page shall cease until reinitialized by the application client via a LOG SELECT command. If the RLEC bit of the Control mode page is set to one, then the device server shall report the exception condition.

The page code assignments for the log pages are listed in table 199.

Table 199 — Log page codes

Page Code	Log Page Name	Reference
0Fh	Application Client	7.2.2
01h	Buffer Over-Run/Under-Run	7.2.3
2Fh	Informational Exceptions	7.2.5
0Bh	Last *n* Deferred Errors or Asynchronous Events	7.2.6
07h	Last *n* Error Events	7.2.7
06h	Non-Medium Error	7.2.8
18h	Protocol Specific Port	7.2.9
03h	Read Error Counter	7.2.4
04h	Read Reverse Error Counter	7.2.4
10h	Self-Test Results	7.2.10
0Eh	Start-Stop Cycle Counter	7.2.11
00h	Supported Log Pages	7.2.12
0Dh	Temperature	7.2.13
05h	Verify Error Counter	7.2.4
02h	Write Error Counter	7.2.4
08h - 0Ah	Reserved (may be used by specific device types)	
0Ch	Reserved (may be used by specific device types)	
11h - 17h	Reserved (may be used by specific device types)	
19h - 2Eh	Reserved (may be used by specific device types)	
3Fh	Reserved	
30h - 3Eh	Vendor specific	
Annex D contains a listing of log pages codes in numeric order.		

Additional information about the LOG SELECT command is in Annex C.

7.2.2 Application Client log page

The Application Client log page (see table 200) provides a place for application clients to store information. The page code for the application client page is 0Fh.

Table 200 — Application client log page

Bit / Byte	7	6	5	4	3	2	1	0
0	PAGE CODE (0Fh)							
1	Reserved							
2	(MSB)	PAGE LENGTH (n-3)						
3								(LSB)
	Application client log parameters							
4	First application client log parameter							
	. . .							
n	Last application client log parameter							

The PAGE CODE and PAGE LENGTH fields are described in 7.2.1.

Parameter codes 0000h through 0FFFh are for general usage application client data. The intended use for this information is to aid in describing the system configuration and system problems, but the specific definition of the data is application client specific. The general usage application client data parameters all have the format shown in table 201.

Table 201 — General usage application client parameter data

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)	PARAMETER CODE						
1								(LSB)
2	DU	DS	TSD	ETC	TMC		LBIN	LP
3	PARAMETER LENGTH (FCh)							
4	GENERAL USAGE PARAMETER BYTES							
255								

For general usage application client data, the value in the PARAMETER CODE field shall be between 0000h and 0FFFh. The first supported general usage application client parameter code shall be 0000h and additional supported parameters shall be sequentially numbered. If any general usage parameter codes are implemented, the device shall support at least 64 general usage parameter descriptors and they shall be parameter codes 0000h through 003Fh.

For the general usage application client parameter, the PARAMETER LENGTH value for each parameter shall be FCh.

The state of the log parameter control bits for parameters 0000h through 0FFFh is specified in table 202.

Table 202 — Parameter control bits for general usage parameters (0000h through 0FFFh)

Bit	Value	Description
DU	1	Value provided by application client
DS	0	Device server supports saving of parameter
TSD	0	Device server manages saving of parameter
ETC	0	No threshold comparison is made on this value
TMC	xx	Ignored when the ETC bit is set to zero
LBIN	1	The parameter is in binary format
LP	1	The parameter is a list parameter

The values stored in the GENERAL USAGE PARAMETER BYTES represent data sent to the device server in a previous LOG SELECT command. If a previous LOG SELECT command has not occurred, the data is vendor specific.

In the application client log page, parameter codes 1000h through FFFFh are reserved.

7.2.3 Buffer Over-Run/Under-Run log page

The Buffer Over-Run/Under-Run log page (page code 01h) defines 24 data counters that may be used to record the number of buffer over-runs or under-runs for the logical unit. A logical unit that implements this log page may implement one or more of the defined data counters.

A buffer over-run or under-run may occur when a SCSI initiator device does not transmit data to or from the logical unit's buffer fast enough to keep up with reading or writing the media. A buffer over-run condition may occur during a read operation when a buffer full condition prevents continued transfer of data from the media to the buffer. A buffer under-run condition may occur during a write operation when a buffer empty condition prevents continued transfer of data to the media from the buffer. Most devices incur a delay at this point while the media is repositioned.

Table 203 defines the PARAMETER CODE field for the buffer over-run/under-run counters.

Table 203 — Parameter code field for buffer over-run/under-run counters

Bit Byte	7	6	5	4	3	2	1	0
0	Reserved							
1	COUNT BASIS			CAUSE				TYPE

The PARAMETER CODE field for buffer over-run/under-run counters contains a 16-bit value comprised of eight reserved bits, a COUNT BASIS field (see table 204), a CAUSE field (see table 205), and a TYPE bit. These are concatenated to determine the value of the parameter code for that log parameter. (E.g., a counter for parameter code value of 0023h specifies a count basis of 001b, a cause of 0001b, and a type of 1b. This counter is incremented once per command that experiences an over-run due to the service delivery subsystem being busy.)

The COUNT BASIS field defines the criteria for incrementing the counter. The criteria are defined in table 204.

Table 204 — Count basis definition

Count basis	Description
000b	Undefined
001b	Per command
010b	Per I_T nexus loss
011b	Per unit of time
100b - 111b	Reserved

NOTE 37 - The per unit of time count basis is device type specific. Direct access block devices typically use a latency period (i.e., one revolution of the medium) as the unit of time.

The CAUSE field indicates the reason that the over-run or under-run occurred. The following causes are defined in table 205.

Table 205 — CAUSE field definition

Cause	Description
0h	Undefined
1h	Service delivery subsystem busy
2h	Transfer rate too slow
3h - Fh	Reserved

The TYPE bit indicates whether the counter records under-runs or over-runs. A TYPE bit set to zero specifies a buffer under-run condition and a TYPE bit set to one specifies a buffer over-run condition.

The counters contain the total number of times buffer over-run or under-run conditions have occurred since the last time the counter was cleared. The counter shall be incremented for each occurrence of an under-run or over-run condition and may be incremented more than once for multiple occurrences during the processing of a single command.

7.2.4 Error counter log pages

This subclause defines the error counter log pages (see table 206).

Table 206 — Error counter log page codes

Page Code	Loge Page Name
03h	Read Error Counter
04h	Read Reverse Error Counter
05h	Verify Error Counter
02h	Write Error Counter

The log page format is defined in 7.2.1. A log page may return one or more log parameters that record events defined by the parameter codes. Table 207 defines the parameter codes for the error counter log pages.

Table 207 — Parameter codes for error counter log pages

Parameter code	Description
0000h	Errors corrected without substantial delay
0001h	Errors corrected with possible delays
0002h	Total (e.g., rewrites or rereads)
0003h	Total errors corrected
0004h	Total times correction algorithm processed
0005h	Total bytes processed
0006h	Total uncorrected errors
0007h - 7FFFh	Reserved
8000h - FFFFh	Vendor specific

NOTE 38 - The exact definition of the error counters is not part of this standard. These counters should not be used to compare products because the products may define errors differently.

7.2.5 Informational Exceptions log page

The Informational Exceptions log page (see table 208) provides a place for reporting detail about informational exceptions. The page code for the Informational Exceptions log page is 2Fh.

Table 208 — Informational Exceptions log page

Bit / Byte	7	6	5	4	3	2	1	0
0	PAGE CODE (2Fh)							
1	Reserved							
2	(MSB)	PAGE LENGTH (n-3)						
3								(LSB)
	Informational exceptions log parameters							
4	First informational exceptions log parameter							
	:							
n	Last informational exceptions log parameter							

The PAGE CODE and PAGE LENGTH fields are described in 7.2.1.

Table 209 defines the parameter codes.

Table 209 — Informational exceptions parameter codes

Parameter code	Description
0000h	Informational exceptions general parameter data
0001h - FFFFh	Vendor specific

The informational exceptions general parameter data page has the format shown in table 210.

Table 210 — Informational exceptions general parameter data

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1				PARAMETER CODE (0000h)				(LSB)
2	DU	DS	TSD	ETC	TMC		LBIN	LP
3	PARAMETER LENGTH (n-3)							
4	INFORMATIONAL EXCEPTION ADDITIONAL SENSE CODE							
5	INFORMATIONAL EXCEPTION ADDITIONAL SENSE CODE QUALIFIER							
6	MOST RECENT TEMPERATURE READING							
7	Vendor specific							
n								

The values of the log parameter control bits for self test results log parameters are specified in table 211.

Table 211 — Parameter control bits for Informational exceptions log parameter (0000h)

Bit	Value	Description
DU	0	Value provided by device server
DS	0	Device server supports saving of parameter
TSD	0	Device server manages saving of parameter
ETC	0	No threshold comparison is made on this value
TMC	xx	Ignored when the ETC bit is set to zero
LBIN	1	The parameter is in binary format
LP	1	The parameter is a list parameter

The PARAMETER LENGTH field is described in 7.2.1. The parameter length shall be at least 04h.

If the INFORMATIONAL EXCEPTION ADDITIONAL SENSE CODE field contains zero, no informational exception condition is pending and contents of the INFORMATIONAL EXCEPTION ADDITIONAL SENSE CODE QUALIFIER field are unspecified. If the INFORMATIONAL EXCEPTION ADDITIONAL SENSE CODE field contains any value other than zero, an informational exception condition exists that has an additional sense code indicated by INFORMATIONAL EXCEPTION ADDITIONAL SENSE CODE field and an ADDITIONAL SENSE CODE QUALIFIER indicated by the INFORMATIONAL EXCEPTION ADDITIONAL SENSE CODE QUALIFIER field.

The MOST RECENT TEMPERATURE READING field indicates the temperature in degrees Celsius of the SCSI target device at the time the LOG SENSE command is performed. Temperatures equal to or less than zero degrees Celsius shall be indicated by a value of zero. If the device server is unable to detect a valid temperature because of a sensor failure or other condition, the value returned shall be FFh. The temperature should be reported with an accuracy of plus or minus three Celsius degrees while the device is operating at a steady state within the environmental limits specified for the device.

7.2.6 Last *n* Deferred Errors or Asynchronous Events log page

The Last *n* Deferred Errors or Asynchronous Events log page (page code 0Bh) provides for a number of deferred errors or asynchronous events sense data records using the list parameter format of the log page. The number of these deferred errors or asynchronous events records supported, *n*, is vendor specific. Each deferred error or asynchronous event record contains SCSI sense data for a deferred error or asynchronous event that has occurred. The parameter code associated with the record indicates the relative time at which the deferred error or asynchronous event occurred. A higher parameter code indicates that the deferred error or asynchronous event occurred later in time.

The content of the PARAMETER VALUE field of each log parameter is the SCSI sense data describing the deferred error.

The fields DU bit, TSD bit, ETC bit, and TMC field shall be set to zero. The LBIN bit shall be set to one to indicate binary information. The LP bit shall be set to one to indicate a list parameter.

7.2.7 Last *n* Error Events log page

The Last *n* Error Events log page (page code 07h) provides for a number of error-event records using the list parameter format of the log page. The number of these error-event records supported, *n*, is vendor specific. Each error-event record contains vendor specific diagnostic information for a single error encountered by the device. The parameter code associated with error-event record indicates the relative time at which the error occurred. A higher parameter code indicates that the error event occurred later in time.

The content of the PARAMETER VALUE field of each log parameter is ASCII data (see 4.4.1) that may describe the error event. The contents of the character string is not defined by this standard.

When the last supported parameter code is used by an error-event record, the recording on this log page of all subsequent error information shall cease until one or more of the list parameters with the highest parameter codes have been reinitialized. If the RLEC bit of the Control mode page (see 7.4.6) is set to one, the command shall be terminated with CHECK CONDITION status, with the sense key set to RECOVERED ERROR, and the additional sense code set to LOG LIST CODES EXHAUSTED.

7.2.8 Non-Medium Error log page

The Non-Medium Error log page (page code 06h) provides for summing the occurrences of recoverable error events other than write, read, or verify failures. No discrimination among the various types of events is provided by parameter code (see table 212). Vendor specific discrimination may be provided through the vendor specific parameter codes.

Table 212 — Non-medium error event parameter codes

Parameter code	Description
0000h	Non-medium error count
0001h - 7FFFh	Reserved
8000h - FFFFh	Vendor specific error counts

7.2.9 Protocol Specific Port log page

The Protocol Specific Port log page (see table 213) provides SCSI transport protocol specific parameters that are associated with the SCSI targets ports in the SCSI target device. This log page may be implemented in any logical

unit, including the TARGET LOG PAGES well-known logical unit (see 8.4). See the SCSI transport protocol standard (see 3.1.82) for definitions of the protocol specific log parameters.

Table 213 — Protocol Specific Port log page

Bit / Byte	7	6	5	4	3	2	1	0
0	PAGE CODE (18h)							
1	Reserved							
2	(MSB)			PAGE LENGTH (n-3)				
3								(LSB)
	Protocol specific port log parameters							
4				First protocol specific port log parameter				
				⋮				
n				Last protocol specific port log parameter				

Table 214 shows the format of a protocol specific port log parameter.

Table 214 — Protocol specific port log parameter format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)			PARAMETER CODE				
1								(LSB)
2	DU	DS	TSD	ETC	TMC		LBIN	LP
3	PARAMETER LENGTH (x-3)							
4	Reserved				PROTOCOL IDENTIFIER			
5				SCSI transport protocol specific				
x								

The PARAMETER CODE field contains the relative target port identifier (see 3.1.88) of the target port for which the parameter data applies.

The contents of the DU, DS, TSD, ETC, LBIN, and LP bits and the TMC field are defined in 7.2.1.

The PARAMETER LENGTH field indicates the number of bytes remaining in the log parameter.

The PROTOCOL IDENTIFIER field contains one of the values shown in table 261 (see 7.5.1) to identify the SCSI transport protocol standard that defines the SCSI transport protocol specific data in this log parameter. The SCSI transport protocol specific data is defined by the corresponding SCSI transport protocol standard.

7.2.10 Self-Test Results log page

The Self-Test Results log page (see table 215) provides the results from the 20 most recent self-tests (see 5.5). Results from the most recent self-test or the self-test currently in progress shall be reported in the first self-test log parameter; results from the second most recent self-test shall be reported in the second self-test log parameter; etc. If fewer than 20 self-tests have occurred, the unused self-test log parameter entries shall be zero filled.

Table 215 — Self-Test Results log page

Bit / Byte	7	6	5	4	3	2	1	0
0	PAGE CODE (10h)							
1	Reserved							
2	(MSB)							
3	PAGE LENGTH (190h)							(LSB)
	Self-test results log parameters							
4	First self-test results log parameter							
23								
	. . .							
384	Twentieth self-test results log parameter							
403								

The PAGE CODE and PAGE LENGTH fields are described in 7.2.1.

Table 216 shows the format of one self-test log parameter.

Table 216 — Self-test results log parameter format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1	PARAMETER CODE (0001h to 0014h)							(LSB)
2	DU	DS	TSD	ETC	TMC		LBIN	LP
3	PARAMETER LENGTH (10h)							
4	SELF-TEST CODE			Reserved	SELF-TEST RESULTS			
5	SELF-TEST NUMBER							
6	(MSB)							
7	TIMESTAMP							(LSB)
8	(MSB)							
15	ADDRESS OF FIRST FAILURE							(LSB)
16	Reserved				SENSE KEY			
17	ADDITIONAL SENSE CODE							
18	ADDITIONAL SENSE CODE QUALIFIER							
19	Vendor specific							

The PARAMETER CODE field identifies the log parameter being transferred. The PARAMETER CODE field for the results of the most recent self-test shall contain 0001h; the PARAMETER CODE field for the results of the second most recent test shall contain 0002h; etc.

The values of the log parameter control bits for self test results log parameters is specified in table 217.

Table 217 — Parameter control bits for self-test results log parameters

Bit	Value	Description
DU	0	Value provided by device server
DS	0	Device server supports saving of parameter
TSD	0	Device server manages saving of parameter
ETC	0	No threshold comparison is made on this value
TMC	xx	Ignored when the ETC bit is set to zero
LBIN	1	The parameter is in binary format
LP	1	The parameter is a list parameter

The PARAMETER LENGTH field shall contain 10h.

The SELF-TEST CODE field contains the value in the SELF-TEST CODE field of the SEND DIAGNOSTIC command that initiated this self-test (see 6.28).

Table 218 defines the content of the SELF-TEST RESULTS field.

Table 218 — SELF-TEST RESULTS field

Code	Description
0h	The self-test completed without error.
1h	The background self-test was aborted by the application client using a SEND DIAGNOSTIC command (see 6.28) with the SELF-TEST CODE field set to 100b (i.e., abort background self-test).
2h	The self-test routine was aborted by an application client using a method other than a SEND DIAGNOSTIC command with the SELF-TEST CODE field set to 100b (e.g., by a task management function, or by issuing an exception command as defined in 5.5.3).
3h	An unknown error occurred while the device server was processing the self-test and the device server was unable to complete the self-test.
4h	The self-test completed with a failure in a test segment, and the test segment that failed is not known.
5h	The first segment of the self-test failed.
6h	The second segment of the self-test failed.
7h	Another segment of the self-test failed and which test is indicated by the contents of the SELF-TEST NUMBER field.
8h-Eh	Reserved
Fh	The self-test is in progress.

The SELF-TEST NUMBER field identifies the self-test that failed and consists of either:

a) The number of the segment that failed during the self-test; or
b) The number of the test that failed and the number of the segment in which the test was run, using a vendor specific method for placing the two values in the one field.

When the segment in which the failure occurred is not able to be identified or need not be identified, the SELF-TEST NUMBER field shall contain 00h.

The TIMESTAMP field contains the total accumulated power-on hours for the device server at the time the self-test was completed. If the test is still in progress, the content of the TIMESTAMP field shall be zero. If the power-on hours for the device server at the time the self-test was completed is greater than FFFFh then the content of the TIMESTAMP field shall be FFFFh.

The ADDRESS OF FIRST FAILURE field contains information that locates the failure on the media. If the logical unit implements logical blocks, the content of the ADDRESS OF FIRST FAILURE field is the first logical block address where a self-test error occurred. This implies nothing about the quality of any other logical block on the logical unit, since the testing during which the error occurred may not have been performed in a sequential manner. This value shall not change (e.g., as the result of block reassignment). The content of the ADDRESS OF FIRST FAILURE field shall be FFFF FFFF FFFF FFFFh if no errors occurred during the self-test or if the error that occurred is not related to an identifiable media address.

The SENSE KEY field, ADDITIONAL SENSE CODE field, and ADDITIONAL SENSE CODE QUALIFIER field may contain a hierarchy of additional information relating to error or exception conditions that occurred during the self-test represented in the same format used by the sense data (see 4.5).

7.2.11 Start-Stop Cycle Counter log page

This subclause defines the Start-Stop Cycle Counter log page (page code 0Eh). A device server that implements the Start-Stop Cycle Counter log page shall implement one or more of the defined parameters. Table 219 shows the Start-Stop Cycle Counter log page with all parameters present.

Table 219 — Start-Stop Cycle Counter log page (part 1 of 2)

Bit / Byte	7	6	5	4	3	2	1	0
0	PAGE CODE (0Eh)							
1	Reserved							
2	(MSB)			PAGE LENGTH (24h)				
3								(LSB)
4	(MSB)			PARAMETER CODE 0001h Date of Manufacture				
5								(LSB)
6	DU	DS	TSD	ETC	TMC		LBIN	LP
7	PARAMETER LENGTH (06h)							
8	(MSB)			YEAR OF MANUFACTURE (4 ASCII characters)				
11								(LSB)
12	(MSB)			WEEK OF MANUFACTURE (2 ASCII characters)				
13								(LSB)
14	(MSB)			PARAMETER CODE 0002h Accounting Date				
15								(LSB)
16	DU	DS	TSD	ETC	TMC		LBIN	LP
17	PARAMETER LENGTH (06h)							
18	(MSB)			ACCOUNTING DATE YEAR (4 ASCII characters)				
21								(LSB)
22	(MSB)			ACCOUNTING DATE WEEK (2 ASCII characters)				
23								(LSB)

Table 219 — Start-Stop Cycle Counter log page (part 2 of 2)

Bit / Byte	7	6	5	4	3	2	1	0
24	(MSB)			PARAMETER CODE 0003h				
25				Specified cycle count over device lifetime				(LSB)
26	DU	DS	TSD	ETC	TMC		LBIN	LP
27	PARAMETER LENGTH (04h)							
28	(MSB)			SPECIFIED CYCLE COUNT OVER DEVICE LIFETIME				
31				(4-byte binary number)				(LSB)
32	(MSB)			PARAMETER CODE 0004h				
33				Accumulated start-stop cycles				(LSB)
34	DU	DS	TSD	ETC	TMC		LBIN	LP
35	PARAMETER LENGTH (04h)							
36	(MSB)			ACCUMULATED START-STOP CYCLES				
39				(4-byte binary number)				(LSB)

The year and week in the year that the SCSI target device was manufactured shall be contained in the parameter value of the log parameter in which the parameter code is 0001h. The date of manufacture shall not be saveable by the application client using the LOG SELECT command (i.e., the log parameter DS bit shall be set to one). The date is expressed in numeric ASCII characters (30h – 39h) in the form YYYYWW, as shown in table 219. For the log parameter in which the parameter code value is 0001h, the values of the parameter control bits are defined in table 220.

Table 220 — Parameter control bits for date of manufacture parameter (0001h)

Bit	Value	Description
DU	0	Value provided by device server
DS	1	Device server does not support saving of parameter
TSD	0	Device server manages saving of parameter
ETC	0	No threshold comparison is made on this value
TMC	xx	Ignored when the ETC bit is set to zero
LBIN	0	The parameter is in ASCII format
LP	1	The parameter is a list parameter

The accounting date specified by parameter code 0002h may be saved using a LOG SELECT command to indicate when the device was placed in service. If the parameter is not yet set or is not settable, the default value placed in the parameter field shall be 6 ASCII space characters (20h). The field shall not be checked for validity by

the device server. For the log parameter in which the parameter code value is 0002h, the values of the parameter control bits are defined in table 221.

Table 221 — Parameter control bits for accounting date parameter (0002h)

Bit	Value	Description
DU	0	Value provided by device server
DS	0 or 1	Device server optionally supports saving of parameter
TSD	0	Device server manages saving of parameter
ETC	0	No threshold comparison is made on this value
TMC	xx	Ignored when the ETC bit is set to zero
LBIN	0	The parameter is in ASCII format
LP	1	The parameter is a list parameter

The parameter value in the specified cycle count over device lifetime log parameter (parameter code 0003h) shall contain a four-byte binary value that indicates how many stop-start cycles may typically be performed over the lifetime of the SCSI target device without degrading the SCSI target device's operation or reliability outside the limits specified by the manufacturer of the SCSI target device. The specified cycle count over device lifetime parameter shall not be saveable by the application client using the LOG SELECT command (i.e., the log parameter DS bit shall be set to one). For the log parameter in which the parameter code value is 0003h, the values of the parameter control bits are defined in table 222.

Table 222 — Parameter control bits for start-stop cycle counter parameters (0003h and 0004h)

Bit	Value	Description
DU	0	Value provided by device server
DS	1	Device server does not support saving of parameter
TSD	0	Device server manages saving of parameter
ETC	0	No threshold comparison is made on this value
TMC	xx	Ignored when the ETC bit is set to zero
LBIN	1	The parameter is in binary format
LP	1	The parameter is a list parameter

The parameter value in the accumulated start-stop cycles log parameter (parameter code 0004h) shall contain a four-byte binary value that indicates how many stop-start cycles the SCSI target device has detected since its date of manufacture. The accumulated start-stop cycles parameter shall not be saveable by the application client using the LOG SELECT command (i.e., the log parameter DS bit shall be set to one). The time at which the count is incremented during a start-stop cycle is vendor specific. For rotating magnetic storage devices, a single start-stop cycle is defined as an operational cycle that begins with the disk spindle at rest, continues while the disk accelerates to its normal operational rotational rate, continues during the entire period the disk is rotating, continues as the disk decelerates toward a resting state, and ends when the disk is no longer rotating. For devices without a spindle or with multiple spindles, the definition of a single start-stop cycle is vendor specific. The count is incremented by one for each complete start-stop cycle. No comparison with the value of parameter 0003h shall be performed by the device server. For the log parameter in which the parameter code value is 0004h, the values of the parameter control bits are defined in table 222.

7.2.12 Supported Log Pages log page

The Supported Log Pages log page (see table 223) returns the list of log pages implemented by the logical unit. Logical units that implement the LOG SENSE command shall implement this log page.

This log page is not defined for the LOG SELECT command. This log page returns the list of supported log pages for the specified logical unit.

Table 223 — Supported log pages

Bit / Byte	7	6	5	4	3	2	1	0
0	PAGE CODE (00h)							
1	Reserved							
2	(MSB)	PAGE LENGTH (n-3)						
3								(LSB)
4	SUPPORTED PAGE LIST							
n								

The PAGE LENGTH field indicates the length in bytes of the following supported log page list.

The SUPPORTED PAGE LIST field shall contain a list of all log page codes implemented by the logical unit in ascending order beginning with page code 00h.

7.2.13 Temperature log page

This subclause defines the Temperature log page (page code 0Dh). A device server that implements the Temperature log page shall implement parameter 0000h and may implement parameter 0001h. Table 224 shows the Temperature log page with all parameters present.

Table 224 — Temperature log page

Bit / Byte	7	6	5	4	3	2	1	0
0	PAGE CODE (0Dh)							
1	Reserved							
2	(MSB)	PAGE LENGTH (0Ch)						
3								(LSB)
4	(MSB)	PARAMETER CODE 0000h Temperature						
5								(LSB)
6	DU	DS	TSD	ETC	TMC		LBIN	LP
7	PARAMETER LENGTH (02h)							
8	Reserved							
9	TEMPERATURE (degrees Celsius)							
10	(MSB)	PARAMETER CODE 0001h Reference temperature						
11								(LSB)
12	DU	DS	TSD	ETC	TMC		LBIN	LP
13	PARAMETER LENGTH (02h)							
14	Reserved							
15	REFERENCE TEMPERATURE (degrees Celsius)							

The parameter value in the temperature log parameter (parameter code 0000h) shall contain a one-byte binary value that indicates the temperature of the SCSI target device in degrees Celsius at the time the LOG SENSE command is performed. Temperatures equal to or less than zero degrees Celsius shall be indicated by a value of zero. If the device server is unable to detect a valid temperature because of a sensor failure or other condition, then the value returned shall be FFh. The temperature should be reported with an accuracy of plus or minus three Celsius degrees while the SCSI target device is operating at a steady state within its environmental limits. No comparison is performed between the temperature value specified in parameter 0000h and the reference temperature specified in parameter 0001h. The state of the parameter control bits for parameter 0000h is specified in table 225.

Table 225 — Parameter control bits for temperature parameters (0000h and 0001h)

Bit	Value	Description
DU	0	Value provided by device server
DS	1	Device server does not support saving of parameter
TSD	0	Device server manages saving of parameter
ETC	0	No threshold comparison is made on this value
TMC	xx	Ignored when the ETC bit is set to zero
LBIN	1	The parameter is in binary format
LP	1	The parameter is a list parameter

A reference temperature for the device may be returned by the device server as follows:

a) If a reference temperature is returned, the parameter value in the reference temperature log parameter (parameter code 0001h) shall contain a one-byte binary value that indicates the maximum reported sensor temperature in degrees Celsius at which the SCSI target device is capable of operating continuously without degrading the SCSI target device's operation or reliability beyond manufacturer accepted limits; or
b) If no reference temperature is returned, then:
   A) The log parameter with parameter code 0001h may not be included in the log page; or
   B) The parameter value in the reference temperature log parameter (parameter code 0001h) may be set to FFh.

The reference temperature may change for vendor specific reasons. The state of the parameter control bits for parameter 0001h is specified in table 225.

7.3 Medium auxiliary memory attributes

7.3.1 Attribute format

Each medium auxiliary memory attribute shall be communicated between the application client and device server in the format shown in table 226. This format shall be used in the parameter data for the WRITE ATTRIBUTE command (see 6.34) and the READ ATTRIBUTE command (see 6.14). The attribute format in this standard implies nothing about the physical representation of an attribute in the medium auxiliary memory.

Table 226 — MAM ATTRIBUTE format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1				ATTRIBUTE IDENTIFIER				(LSB)
2	READ ONLY			Reserved			FORMAT	
3	(MSB)							
4				ATTRIBUTE LENGTH (n-4)				(LSB)
5								
n				ATTRIBUTE VALUE				

The ATTRIBUTE IDENTIFIER field contains a code value identifying the attribute (see 7.3.2).

The READ ONLY bit indicates whether the attribute is in the read only state (see 5.11). If the READ ONLY bit is set to one, the attribute is in the read only state. If the READ ONLY bit is set to zero, the attribute is in the read/write state.

The FORMAT field (see table 227) specifies the format of the data in the ATTRIBUTE VALUE field.

Table 227 — MAM attribute formats

Format	Name	Description
00b	BINARY	The ATTRIBUTE VALUE field contains binary data.
01b	ASCII	The ATTRIBUTE VALUE field contains left-aligned ASCII data (see 4.4.1).
10b	TEXT	The attribute contains textual data. The character set is as described in the TEXT LOCALIZATION IDENTIFIER attribute (see 7.3.2.4.6).
11b		Reserved

The ATTRIBUTE LENGTH field specifies the length in bytes of the ATTRIBUTE VALUE field.

The ATTRIBUTE VALUE field contains the current value, for the READ ATTRIBUTE command (see 6.14), or intended value, for the WRITE ATTRIBUTE command (see 6.34), of the attribute.

7.3.2 Attribute identifier values

7.3.2.1 Attribute identifier values overview

The values in the ATTRIBUTE IDENTIFIER field (see 7.3.1) are assigned according to the attribute type (see 5.11) and whether the attribute is standard or vendor specific (see table 228).

Table 228 — MAM attribute identifier range assignments

Attribute Identifiers	Attribute Type	Standardized	Subclause
0000h - 03FFh	Device	Yes	7.3.2.2
0400h - 07FFh	Medium	Yes	7.3.2.3
0800h - 0BFFh	Host	Yes	7.3.2.4
0C00h - 0FFFh	Device	Vendor specific	
1000h - 13FFh	Medium	Vendor specific	
1400h - 17FFh	Host	Vendor specific	
1800h - FFFFh	Reserved		

Device servers may accept and process a WRITE ATTRIBUTE command containing standardized host type attribute identifier values (i.e., 0800h-0BFFh) or vendor specific host type attribute identifier values (i.e., 1400h-17FFh). Standardized host type attribute identifier values may be checked as described in 7.3.2.4.

7.3.2.2 Device type attributes

Device type attributes (see table 229) shall be maintained and updated by the device server when the medium and associated medium auxiliary memory are present. All supported medium type attributes shall have a status of read only (see 5.11).

Table 229 — Device type attributes

Attribute Identifier	Name	Attribute Length (in bytes)	Format	Subclause
0000h	REMAINING CAPACITY IN PARTITION	8	BINARY	7.3.2.2.1
0001h	MAXIMUM CAPACITY IN PARTITION	8	BINARY	7.3.2.2.1
0002h	Restricted			
0003h	LOAD COUNT	8	BINARY	7.3.2.2.2
0004h	MAM SPACE REMAINING	8	BINARY	7.3.2.2.3
0005h - 0006h	Restricted			
0007h	INITIALIZATION COUNT	2	BINARY	7.3.2.2.4
0008h - 020Ah	Reserved			
020Ah	DEVICE MAKE/SERIAL NUMBER AT LAST LOAD	40	ASCII	7.3.2.2.5
020Bh	DEVICE MAKE/SERIAL NUMBER AT LOAD-1	40	ASCII	7.3.2.2.5
020Ch	DEVICE MAKE/SERIAL NUMBER AT LOAD-2	40	ASCII	7.3.2.2.5
020Dh	DEVICE MAKE/SERIAL NUMBER AT LOAD-3	40	ASCII	7.3.2.2.5
020Eh - 021Fh	Reserved			
0220h	TOTAL MBYTES WRITTEN IN MEDIUM LIFE	8	BINARY	7.3.2.2.6
0221h	TOTAL MBYTES READ IN MEDIUM LIFE	8	BINARY	7.3.2.2.6
0222h	TOTAL MBYTES WRITTEN IN CURRENT/LAST LOAD	8	BINARY	7.3.2.2.7
0223h	TOTAL MBYTES READ IN CURRENT/LAST LOAD	8	BINARY	7.3.2.2.7
0224h - 033Fh	Reserved			
0340h	MEDIUM USAGE HISTORY	90	BINARY	7.3.2.2.8
0341h	PARTITION USAGE HISTORY	60	BINARY	7.3.2.2.8
0342h - 03FFh	Reserved			

7.3.2.2.1 REMAINING CAPACITY IN PARTITION and MAXIMUM CAPACITY IN PARTITION: Are native capacities (i.e., assuming no data compression for the specified medium partition). These values are expressed in increments of 1 048 576 bytes (e.g., a value of one means 1 048 576 bytes and a value of two means 2 097 152 bytes).

7.3.2.2.2 LOAD COUNT: Indicates how many times this medium has been fully loaded. This attribute should not be reset to zero by any action of the device server.

7.3.2.2.3 MAM SPACE REMAINING: Indicates the space currently available in the medium auxiliary memory. The total medium auxiliary memory capacity is reported in the MAM CAPACITY attribute (see 7.3.2.3.4).

NOTE 39 - It may not always be possible to utilize all of the available space in a given medium auxiliary memory implementation. Depending on the internal organization of the memory and the software that controls it, fragmentation issues may mean that certain attribute sizes may not be fully accommodated as the medium auxiliary memory nears its maximum capacity.

7.3.2.2.4 INITIALIZATION COUNT: Indicates the number of times that a device server has logically formatted the medium. This value is cumulative over the life of the medium and shall not be reset to zero.

7.3.2.2.5 DEVICE VENDOR/SERIAL NUMBER AT LAST LOAD, DEVICE VENDOR/SERIAL NUMBER AT LOAD –1, DEVICE VENDOR/SERIAL NUMBER AT LOAD –2 and DEVICE VENDOR/SERIAL NUMBER AT LOAD –3:

Give a history of the last four device servers in which the medium has been loaded. The format of the attributes is shown in table 230.

Table 230 — DEVICE VENDOR/SERIAL NUMBER attribute format

Bit Byte	7	6	5	4	3	2	1	0
0	(MSB)							
7				T10 VENDOR IDENTIFICATION				(LSB)
8	(MSB)							
39				PRODUCT SERIAL NUMBER				(LSB)

The T10 VENDOR IDENTIFICATION field shall be the same value returned in the Standard INQUIRY data (see 6.4.2).

The PRODUCT SERIAL NUMBER field contains ASCII data (see 4.4.1) that is a vendor specific serial number. If the product serial number is not available, the PRODUCT SERIAL NUMBER field shall contain ASCII spaces (20h).

7.3.2.2.6 TOTAL MBYTES WRITTEN IN MEDIUM LIFE and TOTAL MBYTES READ IN MEDIUM LIFE: Indicate the total number of data bytes that are transferred to or from the medium, after any data compression has been applied, over the entire medium life. These values are cumulative and shall not be reset to zero. These values are expressed in increments of 1 048 576 bytes (e.g., a value of one means 1 048 576 bytes and a value of two means 2 097 152 bytes).

7.3.2.2.7 TOTAL MBYTES WRITTEN IN CURRENT/LAST LOAD and TOTAL MBYTES READ IN CURRENT/LAST LOAD: Indicate the total number of data bytes that are transferred to or from the medium, after any data compression has been applied, during the current load if the medium is currently loaded, or the last load if the medium is currently unloaded. The device server should reset these attributes to zero when the medium is loaded. These values are expressed in increments of 1 048 576 bytes (e.g., a value of one means 1 048 576 bytes and a value of two means 2 097 152 bytes).

7.3.2.2.8 MEDIUM USAGE HISTORY: Provides counters (see table 231) for the entire medium. The value in each field is the sum for all partitions. If a field is not used, it should be set to zero.

Table 231 — MEDIUM USAGE HISTORY attribute format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
5				CURRENT AMOUNT OF DATA WRITTEN				(LSB)
6	(MSB)							
11				CURRENT WRITE RETRIES COUNT				(LSB)
12	(MSB)							
17				CURRENT AMOUNT OF DATA READ				(LSB)
18	(MSB)							
23				CURRENT READ RETRIES COUNT				(LSB)
24	(MSB)							
29				PREVIOUS AMOUNT OF DATA WRITTEN				(LSB)
30	(MSB)							
35				PREVIOUS WRITE RETRIES COUNT				(LSB)
36	(MSB)							
41				PREVIOUS AMOUNT OF DATA READ				(LSB)
42	(MSB)							
47				PREVIOUS READ RETRIES COUNT				(LSB)
48	(MSB)							
53				TOTAL AMOUNT OF DATA WRITTEN				(LSB)
54	(MSB)							
59				TOTAL WRITE RETRIES COUNT				(LSB)
60	(MSB)							
65				TOTAL AMOUNT OF DATA READ				(LSB)
66	(MSB)							
71				TOTAL READ RETRIES COUNT				(LSB)
72	(MSB)							
77				LOAD COUNT				(LSB)
78	(MSB)							
83				TOTAL CHANGE PARTITION COUNT				(LSB)
84	(MSB)							
89				TOTAL PARTITION INITIALIZE COUNT				(LSB)

The CURRENT AMOUNT OF DATA WRITTEN field indicates the amount of data written to the medium during this load of the medium. This value is expressed in mebibytes (see 3.6.4).

The CURRENT WRITE RETRIES COUNT field indicates the total number of times a write retry occurred during this load of the medium.[1]

The CURRENT AMOUNT OF DATA READ field indicates the amount of data read from the medium during this load of the medium. This value is expressed in mebibytes (see 3.6.4).

The CURRENT READ RETRIES COUNT field indicates the number of times a read retry occurred during this load of the medium.[1.]

The PREVIOUS AMOUNT OF DATA WRITTEN field indicates the amount of data written to the medium during the previous load of the medium. This value is expressed in mebibytes (see 3.6.4).

The PREVIOUS WRITE RETRIES COUNT field indicates the total number of times a write retry occurred during the previous load of the medium.[1.]

The PREVIOUS AMOUNT OF DATA READ field indicates the amount of data read from the medium during the previous load of the medium. This value is expressed in mebibytes (see 3.6.4).

The PREVIOUS READ RETRIES COUNT field indicates the number of times a read retry occurred during the previous load of the medium.[1.]

The TOTAL AMOUNT OF DATA WRITTEN field indicates the amount of data written to the medium since the last medium format. This value is expressed in mebibytes (see 3.6.4).

The TOTAL WRITE RETRIES COUNT field indicates the total number of times a write retry occurred since the last medium format.[1.]

The TOTAL AMOUNT OF DATA READ field indicates the amount of data read from the medium since the last medium format. This value is expressed in mebibytes (see 3.6.4).

The TOTAL READ RETRIES COUNT field indicates the number of times a read retry occurred since the last medium format.[1.]

The LOAD COUNT field indicates the number of loads since the last medium format. This count accumulates over the life of the medium but it is reset to zero after a medium format.

The TOTAL CHANGE PARTITION COUNT field indicates the number of times that switches between partitions have been performed on the medium. This count accumulates over the life of the medium but it is reset to zero after a medium format.

The TOTAL PARTITION INITIALIZE COUNT field indicates number of times that any of the partitions on the medium have been erased. This count accumulates over the life of the medium but it is reset to zero after a medium format.

1. The definition of one retry as counted by this attribute field is not part of this standard. This counter should not be used to compare products because the products may define errors differently.

7.3.2.2.9 PARTITION USAGE HISTORY: Provides counters (see table 232) for the partition specified by the PARTITION NUMBER field in the CDB. If a field is not used, it should be set to zero.

Table 232 — PARTITION USAGE HISTORY attribute format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)			CURRENT AMOUNT OF DATA WRITTEN				
3								(LSB)
4	(MSB)			CURRENT WRITE RETRIES COUNT				
7								(LSB)
8	(MSB)			CURRENT AMOUNT OF DATA READ				
11								(LSB)
12	(MSB)			CURRENT READ RETRIES COUNT				
15								(LSB)
16	(MSB)			PREVIOUS AMOUNT OF DATA WRITTEN				
19								(LSB)
20	(MSB)			PREVIOUS WRITE RETRIES COUNT				
23								(LSB)
24	(MSB)			PREVIOUS AMOUNT OF DATA READ				
27								(LSB)
28	(MSB)			PREVIOUS READ RETRIES COUNT				
31								(LSB)
32	(MSB)			TOTAL AMOUNT OF DATA WRITTEN				
35								(LSB)
36	(MSB)			TOTAL WRITE RETRIES COUNT				
39								(LSB)
40	(MSB)			TOTAL AMOUNT OF DATA READ				
43								(LSB)
44	(MSB)			TOTAL READ RETRIES COUNT				
47								(LSB)
48	(MSB)			LOAD COUNT				
51								(LSB)
52	(MSB)			CHANGE PARTITION COUNT				
55								(LSB)
56	(MSB)			PARTITION INITIALIZE COUNT				
59								(LSB)

The CURRENT AMOUNT OF DATA WRITTEN field indicates the amount of data written to the medium in the partition specified by the PARTITION NUMBER field in the CDB during this load of the medium. This value is expressed in mebibytes (see 3.6.4).

The CURRENT WRITE RETRIES COUNT field indicates the total number of times a write retry occurred in the partition specified by the PARTITION NUMBER field in the CDB during this load of the medium.[2]

The CURRENT AMOUNT OF DATA READ field indicates the amount of data read from the medium in the partition specified by the PARTITION NUMBER field in the CDB during this load of the medium. This value is expressed mebibytes (see 3.6.4).

The CURRENT READ RETRIES COUNT field indicates the number of times a read retry occurred in the partition specified by the PARTITION NUMBER field in the CDB during this load of the medium.[2.]

The PREVIOUS AMOUNT OF DATA WRITTEN field indicates the amount of data written to the medium in the partition specified by the PARTITION NUMBER field in the CDB during the previous load of the medium. This value is expressed in mebibytes (see 3.6.4).

The PREVIOUS WRITE RETRIES COUNT field indicates the total number of times a write retry occurred in the partition specified by the PARTITION NUMBER field in the CDB during the previous load of the medium.[2.]

The PREVIOUS AMOUNT OF DATA READ field indicates the amount of data read from the medium in the partition specified by the PARTITION NUMBER field in the CDB during the previous load of the medium. This value is expressed in mebibytes (see 3.6.4).

The PREVIOUS READ RETRIES COUNT field indicates the number of times a read retry occurred in the partition specified by the PARTITION NUMBER field in the CDB during the previous load of the medium.[2.]

The TOTAL AMOUNT OF DATA WRITTEN field indicates the amount of data written to the medium in the partition specified by the PARTITION NUMBER field in the CDB since the last medium format. This value is expressed in mebibytes (see 3.6.4).

The TOTAL WRITE RETRIES COUNT field indicates the total number of times a write retry occurred in the partition specified by the PARTITION NUMBER field in the CDB since the last medium format.[2.]

The TOTAL AMOUNT OF DATA READ field indicates the amount of data read from the medium in the partition specified by the PARTITION NUMBER field in the CDB since the last medium format. This value is expressed in mebibytes (see 3.6.4).

The TOTAL READ RETRIES COUNT field indicates the number of times a read retry occurred in the partition specified by the PARTITION NUMBER field in the CDB since the last medium format.[2.]

The LOAD COUNT field indicates the number of loads in the partition specified by the PARTITION NUMBER field in the CDB since the last medium format. This count accumulates over the life of the medium but it is reset to zero after a medium format.

The TOTAL CHANGE PARTITION COUNT field indicates the number of times that switches to the partition specified by the PARTITION NUMBER field in the CDB have been performed on the medium. This count accumulates over the life of the medium but it is reset to zero after a medium format.

The TOTAL PARTITION INITIALIZE COUNT field indicates number of times that the partition specified by the PARTITION NUMBER field in the CDB has been initialized. This count accumulates over the life of the medium but it is reset to zero after a medium format.

2. The definition of one retry as counted by this attribute field is not part of this standard. This counter should not be used to compare products because the products may define errors differently.

7.3.2.3 Medium type attributes

Medium type attributes (see table 233) are stored in the medium auxiliary memory by the manufacturer. The device server shall not alter medium type attributes. All supported medium type attributes shall have a status of read only (see 5.11).

Table 233 — Medium type attributes

Attribute Identifier	Name	Attribute Length (in bytes)	Format	Subclause
0400h	MEDIUM MANUFACTURER	8	ASCII	7.3.2.3.1
0401h	MEDIUM SERIAL NUMBER	32	ASCII	7.3.2.3.2
0402h - 0405h	Restricted			
0406h	MEDIUM MANUFACTURE DATE	8	ASCII	7.3.2.3.3
0407h	MAM CAPACITY	8	BINARY	7.3.2.3.4
0408h	MEDIUM TYPE	1	BINARY	7.3.2.3.5
0409h	MEDIUM TYPE INFORMATION	2	BINARY	7.3.2.3.5
040Ah	NUMERIC MEDIUM SERIAL NUMBER	unspecified	unspecified	7.3.2.3.6
040Bh - 07FFh	Reserved			

7.3.2.3.1 MEDIUM MANUFACTURER: Contains eight bytes of left-aligned ASCII data (see 4.4.1) identifying the vendor of the media. The medium manufacturer shall be a T10 vendor identification assigned by INCITS. A list of assigned T10 vendor identifications is in Annex E and on the T10 web site (http://www.T10.org).

NOTE 40 - The T10 web site (http://www.t10.org) provides a convenient means to request an identification code.

7.3.2.3.2 MEDIUM SERIAL NUMBER: Contains the manufacturer's serial number for the medium.

7.3.2.3.3 MEDIUM MANUFACTURE DATE: Contains the date of manufacture of the medium. The format is YYYYMMDD (i.e., four numeric ASCII characters for the year followed by two numeric ASCII characters for the month followed by two numeric ASCII characters for the day with no intervening spaces).

7.3.2.3.4 MAM CAPACITY: Is the total capacity of the medium auxiliary memory, in bytes, at manufacture time. It does not indicate the available space of an unused medium auxiliary memory because some of the medium auxiliary memory space may be reserved for device-specific use making it inaccessible to the application client.

7.3.2.3.5 MEDIUM TYPE and MEDIUM TYPE INFORMATION: Give information about non-data media and other types of media. The MEDIUM TYPE INFORMATION attribute is interpreted according to the type of medium indicated by the MEDIUM TYPE (see table 234).

Table 234 — MEDIUM TYPE and MEDIUM TYPE INFORMATION attributes

MEDIUM TYPE	Description	MEDIUM TYPE INFORMATION
00h	Data medium	Reserved
01h	Cleaning medium	Maximum number of cleaning cycles permitted
02h-7Fh	Reserved	Reserved
80h	Write-once medium	Reserved
81h-FFh	Reserved	Reserved

7.3.2.3.6 NUMERIC MEDIUM SERIAL NUMBER: Contains the manufacturer's serial number for the medium in a vendor specific format.

7.3.2.4 Host type attributes

Application clients may use the WRITE ATTRIBUTE and READ ATTRIBUTE commands to maintain the attributes shown in table 235. All existent host type attributes shall have a status of read/write (see 5.11).

Table 235 — Host type attributes

Attribute Identifier	Name	Attribute Length (in bytes)	Format	Subclause
0800h	APPLICATION VENDOR	8	ASCII	7.3.2.4.1
0801h	APPLICATION NAME	32	ASCII	7.3.2.4.2
0802h	APPLICATION VERSION	8	ASCII	7.3.2.4.3
0803h	USER MEDIUM TEXT LABEL	160	TEXT	7.3.2.4.4
0804h	DATE AND TIME LAST WRITTEN	12	ASCII	7.3.2.4.5
0805h	TEXT LOCALIZATION IDENTIFIER	1	BINARY	7.3.2.4.6
0806h	BARCODE	32	ASCII	7.3.2.4.7
0807h	OWNING HOST TEXTUAL NAME	80	TEXT	7.3.2.4.8
0808h	MEDIA POOL	160	TEXT	7.3.2.4.9
0809h	PARTITION USER TEXT LABEL	16	ASCII	7.3.2.4.10
080Ah	LOAD/UNLOAD AT PARTITION	1	BINARY	7.3.2.4.11
080Bh - BFFh	Reserved			

7.3.2.4.1 APPLICATION VENDOR: Contains eight bytes of left-aligned ASCII data (see 4.4.1) identifying the manufacturer of the application client (e.g., class driver or backup program) that last sent a WRITE ATTRIBUTE command to the device server while this medium auxiliary memory was accessible. The application vendor shall be a T10 vendor identification assigned by INCITS. A list of assigned T10 vendor identifications is in Annex E and on the T10 web site (http://www.T10.org).

NOTE 41 - The T10 web site (http://www.t10.org) provides a convenient means to request an identification code.

7.3.2.4.2 APPLICATION NAME: Contains the name of the application client.

7.3.2.4.3 APPLICATION VERSION: Contains the version of the application client.

7.3.2.4.4 USER MEDIUM TEXT LABEL: Is the user level identifier for the medium.

7.3.2.4.5 DATE & TIME LAST WRITTEN: Contains when the application client last wrote to the medium auxiliary memory. The format is YYYYMMDDHHMM (i.e., four numeric ASCII characters for the year followed by two numeric ASCII characters for the month followed by two numeric ASCII characters for the day followed by two numeric ASCII characters between 00 and 24 for the hour followed by two numeric ASCII characters for the minute with no intervening spaces).

7.3.2.4.6 TEXT LOCALIZATION IDENTIFIER: Defines the character set (see table 236) used for attributes with a TEXT format (see 7.3.1).

Table 236 — TEXT LOCALIZATION IDENTIFIER attribute values

Value	Meaning
00h	No code specified (ASCII)
01h	ISO/IEC 8859-1 (Europe, Latin America)
02h	ISO/IEC 8859-2 (Eastern Europe)
03h	ISO/IEC 8859-3 (SE Europe/miscellaneous)
04h	ISO/IEC 8859-4 (Scandinavia/Baltic)

Table 236 — TEXT LOCALIZATION IDENTIFIER attribute values

Value	Meaning
05h	ISO/IEC 8859-5 (Cyrillic)
06h	ISO/IEC 8859-6 (Arabic)
07h	ISO/IEC 8859-7 (Greek)
08h	ISO/IEC 8859-8 (Hebrew)
09h	ISO/IEC 8859-9 (Latin 5)
0Ah	ISO/IEC 8859-10 (Latin 6)
0Bh - 7Fh	Reserved
80h	ISO/IEC 10646-1 (UCS-2BE)
81h	ISO/IEC 10646-1 (UTF-8)
82h - FFh	Reserved

7.3.2.4.7 BARCODE: Is contents of a barcode associated with the medium in the medium auxiliary memory.

7.3.2.4.8 OWNING HOST TEXTUAL NAME: Indicates the host from which that USER MEDIUM TEXT LABEL (see 7.3.2.4.4) originates.

7.3.2.4.9 MEDIA POOL: Indicates the media pool to which this medium belongs.

7.3.2.4.10 PARTITION USER TEXT LABEL: Is a user level identifier for the partition specified by the PARTITION NUMBER field in the CDB.

7.3.2.4.11 LOAD/UNLOAD AT PARTITION: Indicates whether the media is capable of being loaded or unloaded at the partition specified by the PARTITION NUMBER field in the CDB. If loads and unloads are enabled for the specified partition, the value of this attribute shall be one. If loads and unloads are not enabled for the specified partition, the value of this attribute shall be zero. All attribute values other than zero and one are reserved. If LOAD/UNLOAD AT PARTITION is disabled, then loads and unloads are performed at the beginning of the media instead of at the specified partition. If this attribute is in the nonexistent state (see 5.11), then the default action shall be to load and unload at the beginning of media.

7.4 Mode parameters

7.4.1 Mode parameters overview

This subclause describes the mode parameter headers, block descriptors, and mode pages used with MODE SELECT command (see 6.7 and 6.8) and MODE SENSE command (see 6.9 and 6.10) that are applicable to all SCSI devices. Subpages are identical to mode pages except that they include a SUBPAGE CODE field that further differentiates the mode page contents. Mode pages specific to each device type are described in the command standard (see 3.1.18) that applies to that device type.

7.4.2 Mode parameter list format

The mode parameter list shown in table 237 contains a header, followed by zero or more block descriptors, followed by zero or more variable-length mode pages. Parameter lists are defined for each device type.

Table 237 — Mode parameter list

Bit Byte	7	6	5	4	3	2	1	0
	Mode parameter header							
	Block descriptor(s)							
	Mode page(s) or vendor specific (e.g., page code set to zero)							

7.4.3 Mode parameter header formats

The mode parameter header that is used by the MODE SELECT(6) command (see 6.7) and the MODE SENSE(6) command (see 6.9) is defined in table 238.

Table 238 — Mode parameter header(6)

Bit Byte	7	6	5	4	3	2	1	0
0	MODE DATA LENGTH							
1	MEDIUM TYPE							
2	DEVICE-SPECIFIC PARAMETER							
3	BLOCK DESCRIPTOR LENGTH							

The mode parameter header that is used by the MODE SELECT(10) command (see 6.8) and the MODE SENSE(10) command (see 6.10) is defined in table 239.

Table 239 — Mode parameter header(10)

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

When using the MODE SENSE command, the MODE DATA LENGTH field indicates the length in bytes of the following data that is available to be transferred. The mode data length does not include the number of bytes in the MODE DATA LENGTH field. When using the MODE SELECT command, this field is reserved.

NOTE 42 - Logical units that support more than 256 bytes of block descriptors and mode pages may need to implement ten-byte mode commands. The mode data length field in the six-byte CDB header limits the returned data to 256 bytes.

The contents of the MEDIUM TYPE field are unique for each device type. Refer to the mode parameters subclause of the specific device type command standard (see 3.1.18) for definition of these values. Some device types reserve this field.

The DEVICE-SPECIFIC PARAMETER field is unique for each device type. Refer to the mode parameters subclause of the specific device type command standard (see 3.1.18) for definition of this field. Some device types reserve all or part of this field.

If the Long LBA (LONGLBA) bit is set to zero, the mode parameter block descriptor(s), if any, are each eight bytes long and have the format described in 7.4.4.1. If the LONGLBA bit is set to one, the mode parameter block descriptor(s), if any, are each sixteen bytes long and have a format described in a command standard (see 3.1.18).

The BLOCK DESCRIPTOR LENGTH field contains the length in bytes of all the block descriptors. It is equal to the number of block descriptors times eight if the LONGLBA bit is set to zero or times sixteen if the LONGLBA bit is set to one, and does not include mode pages or vendor specific parameters (e.g., page code set to zero), if any, that may follow the last block descriptor. A block descriptor length of zero indicates that no block descriptors are included in the mode parameter list. This condition shall not be considered an error.

7.4.4 Mode parameter block descriptor formats

7.4.4.1 General block descriptor format

When the LONGLBA bit is set to zero (see 7.4.3), the mode parameter block descriptor format for all device types except direct access block devices (see SBC-2) is shown in table 240.

Table 240 — General mode parameter block descriptor

Bit Byte	7	6	5	4	3	2	1	0
0	DENSITY CODE							
1	(MSB)							
2				NUMBER OF BLOCKS				
3								(LSB)
4	Reserved							
5	(MSB)							
6				BLOCK LENGTH				
7								(LSB)

Block descriptors specify some of the medium characteristics for all or part of a logical unit. Support for block descriptors is optional. Each block descriptor contains a DENSITY CODE field, a NUMBER OF BLOCKS field, and a BLOCK LENGTH field. Block descriptor values are always current (i.e., saving is not supported). A unit attention condition (see 6.7 and SAM-3) shall be established when any block descriptor values are changed. Command standards (see 3.1.18) may place additional requirements on the general mode parameter block descriptor. Requirements in the command standards that conflict with requirements defined in this subclause shall take precedence over the requirements defined in this subclause.

The DENSITY CODE field is unique for each device type. Refer to the mode parameters subclause of the specific device type command standard (see 3.1.18) for definition of this field. Some device types reserve all or part of this field.

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

If the number of logical blocks on the medium exceeds the maximum value that may be specified in the NUMBER OF BLOCKS field, a value of FFFFFFh indicates that all of the remaining logical blocks of the logical unit shall have the medium characteristics specified.

NOTE 43 - There may be implicit association between parameters defined in the mode pages and block descriptors. In this case, the device server may change parameters not explicitly sent with the MODE SELECT command. A subsequent MODE SENSE command may be used to detect these changes.

NOTE 44 - The number of remaining logical blocks may be unknown for some device types.

The BLOCK LENGTH field specifies the length in bytes of each logical block described by the block descriptor. For sequential-access devices, a block length of zero indicates that the logical block size written to the medium is specified by the TRANSFER LENGTH field in the CDB (see SSC-2).

7.4.5 Mode page and subpage formats and page codes

The page_0 mode page format is defined in table 241.

Table 241 — Page_0 mode page format

Bit / Byte	7	6	5	4	3	2	1	0
0	PS	SPF (0b)	PAGE CODE					
1	PAGE LENGTH (n-1)							
2	Mode parameters							
n								

The sub_page mode page format is defined in table 242.

Table 242 — Sub_page mode page format

Bit / Byte	7	6	5	4	3	2	1	0
0	PS	SPF (1b)	PAGE CODE					
1	SUBPAGE CODE							
2	(MSB)	PAGE LENGTH (n-3)						
3								(LSB)
4	Mode parameters							
n								

Each mode page contains a PS bit, an SPF bit, a PAGE CODE field, a PAGE LENGTH field, and a set of mode parameters. The page codes are defined in this subclause and in the mode parameter subclauses in the command standard (see 3.1.18) for the specific device type. Each mode page with a SPF bit set to one contains a SUBPAGE CODE field.

A SubPage Format (SPF) bit set to zero indicates that the page_0 mode page format is being used. A SPF bit set to one indicates that the sub_page mode page format is being used.

When using the MODE SENSE command, a parameters saveable (PS) bit set to one indicates that the mode page may be saved by the logical unit in a nonvolatile, vendor specific location. A PS bit set to zero indicates that the device server is not able to save the supported parameters. When using the MODE SELECT command, the PS bit is reserved.

The PAGE CODE and SUBPAGE CODE fields identify the format and parameters defined for that mode page. Some page codes are defined as applying to all device types and other page codes are defined for the specific device type. The page codes that apply to a specific device type are defined in the command standard (see 3.1.18) for that device type. The applicability of each subpage code matches that of the page code with which it is associated.

When using the MODE SENSE command, if page code 00h (vendor specific mode page) is implemented, the device server shall return that mode page last in response to a request to return all mode pages (page code 3Fh). When using the MODE SELECT command, this mode page should be sent last.

The PAGE LENGTH field specifies the length in bytes of the mode parameters that follow. If the application client does not set this value to the value that is returned for the mode page by the MODE SENSE command, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST. The logical unit may implement a mode page

that is less than the full mode page length defined, provided no field is truncated and the PAGE LENGTH field correctly specifies the actual length implemented.

The mode parameters for each mode page are defined in the following subclauses, or in the mode parameters subclause in the command standard (see 3.1.18) for the specific device type. Mode parameters not implemented by the logical unit shall be set to zero.

Table 243 defines the mode pages that are applicable to all device types that implement the MODE SELECT and MODE SENSE commands.

Table 243 — Mode page codes and subpage codes

Page code	Subpage code	Mode Page Name	Reference
0Ah	00h	Control	7.4.6
0Ah	01h	Control Extension	7.4.7
02h	00h	Disconnect-Reconnect	7.4.8
15h	00h	Extended	7.4.9
16h	00h	Extended Device-Type Specific	7.4.10
1Ch	00h	Informational Exceptions Control	7.4.11
09h	00h	obsolete	3.3.7
1Ah	00h	Power Condition	7.4.12
18h	00h	Protocol Specific LUN	7.4.13
18h	01h - FEh	(See specific SCSI transport protocol)	
19h	00h	Protocol Specific Port	7.4.14
19h	01h - FEh	(See specific SCSI transport protocol)	
01h	00h - FEh	(See specific device type)	
03h - 08h	00h - FEh	(See specific device type)	
0Bh - 14h	00h - FEh	(See specific device type)	
1Bh	00h - FEh	(See specific device type)	
1Dh - 1Fh	00h - FEh	(See specific device type)	
20h - 3Eh	00h - FEh	(See specific device type)	
00h	not applicable	Vendor specific (does not require page format)	
3Fh	00h	Return all pages [a]	
3Fh	FFh	Return all pages and subpages [a]	
00h - 3Eh	FFh	Return all subpages [a]	
All page code and subpage code combinations not shown in this table are reserved. Annex D contains a listing of mode page and subpage codes in numeric order.			
[a] Valid only for the MODE SENSE command			

7.4.6 Control mode page

The Control mode page (see table 244) provides controls over SCSI features that are applicable to all device types (e.g., task set management and error logging). If a field in this mode page is changed while there is a task already in the task set, it is vendor specific whether the old or new value of the field applies to that task. The mode page policy (see 6.7) for this mode page shall be shared, or per I_T nexus.

Table 244 — Control mode page

Bit / Byte	7	6	5	4	3	2	1	0
0	PS	SPF (0b)	PAGE CODE (0Ah)					
1	PAGE LENGTH (0Ah)							
2	TST			TMF_ONLY	Reserved	D_SENSE	GLTSD	RLEC
3	QUEUE ALGORITHM MODIFIER				Reserved	QERR		Obsolete
4	VS	RAC	UA_INTLCK_CTRL		SWP	Obsolete		
5	ATO	TAS	Reserved			AUTOLOAD MODE		
6	Obsolete							
7								
8	(MSB)	BUSY TIMEOUT PERIOD						
9								(LSB)
10	(MSB)	EXTENDED SELF-TEST COMPLETION TIME						
11								(LSB)

The PS bit, SPF bit, PAGE CODE field, and PAGE LENGTH field are described in 7.4.5.

A task set type (TST) field specifies the type of task set in the logical unit (see table 245).

Table 245 — Task set type (TST) field

Code	Description
000b	The logical unit maintains one task set for all I_T nexuses
001b	The logical unit maintains separate task sets for each I_T nexus
010b - 111b	Reserved

Regardless of the mode page policy (see 6.7) for the Control mode page, the shared mode page policy shall be applied to the TST field. If the most recent MODE SELECT changes the setting of this field, then the device server shall establish a unit attention condition (see SAM-3) for the initiator port associated with every I_T nexus except the I_T nexus on which the MODE SELECT command was received, with the additional sense code set to MODE PARAMETERS CHANGED.

The allow task management functions only (TMF_ONLY) bit set to zero specifies that the device server shall process tasks with the ACA task attribute received on the faulted I_T nexus (see 3.1.38) when an ACA condition has been established (see SAM-3). A TMF_ONLY bit set to one specifies that the device server shall terminate all tasks received on the faulted I_T nexus with an ACA ACTIVE status when an ACA condition has been established.

A descriptor format sense data (D_SENSE) bit set to zero specifies that the device server shall return the fixed format sense data (see 4.5.3) when returning sense data in the same I_T_L_Q nexus transaction (see 3.1.46) as a CHECK CONDITION status. A D_SENSE bit set to one specifies that the device server shall return descriptor format sense data (see 4.5.2) when returning sense data in the same I_T_L_Q nexus transaction as a CHECK CONDITION status, except as defined in 4.5.1.

A global logging target save disable (GLTSD) bit set to zero specifies that the logical unit implicitly saves, at vendor specific intervals, each log parameter in which the TSD bit (see 7.2) is set to zero. A GLTSD bit set to one specifies that the logical unit shall not implicitly save any log parameters.

A report log exception condition (RLEC) bit set to one specifies that the device server shall report log exception conditions as described in 7.2.1. A RLEC bit set to zero specifies that the device server shall not report log exception conditions.

The QUEUE ALGORITHM MODIFIER field (see table 246) specifies restrictions on the algorithm used for reordering tasks having the SIMPLE task attribute (see SAM-3).

Table 246 — QUEUE ALGORITHM MODIFIER field

Code	Description
0h	Restricted reordering
1h	Unrestricted reordering allowed
2h - 7h	Reserved
8h - Fh	Vendor specific

A value of zero in the QUEUE ALGORITHM MODIFIER field specifies that the device server shall order the processing sequence of tasks having the SIMPLE task attribute such that data integrity is maintained for that I_T nexus (i.e., if the transmission of new SCSI transport protocol requests is halted at any time, the final value of all data observable on the medium shall have exactly the same value as it would have if all the tasks had been given the ORDERED task attribute).

A value of one in the QUEUE ALGORITHM MODIFIER field specifies that the device server may reorder the processing sequence of tasks having the SIMPLE task attribute in any manner. Any data integrity exposures related to task sequence order shall be explicitly handled by the application client through the selection of appropriate commands and task attributes.

The queue error management (QERR) field (see table 247) specifies how the device server shall handle other tasks when one task is terminated with CHECK CONDITION status (see SAM-3). The task set type (see the TST field definition in this subclause) defines which other tasks are affected. If the TST field equals 000b, then all tasks from all I_T nexuses are affected. If the TST field equals 001b, then only tasks from the same I_T nexus as the task that is terminated with CHECK CONDITION status are affected.

Table 247 — Queue error management (QERR) field

Code	Definition
00b	If an ACA condition is established, the affected tasks in the task set shall resume after the ACA condition is cleared (see SAM-3). Otherwise, all tasks other than the task that received the CHECK CONDITION status shall be processed as if no error occurred.
01b	All the affected tasks in the task set shall be aborted when the CHECK CONDITION status is sent. If the TAS bit is set to zero, a unit attention condition (see SAM-3) shall be established for the initiator port associated with every I_T nexus that had tasks aborted except for the I_T nexus on which the CHECK CONDITION status was returned, with the additional sense code set to COMMANDS CLEARED BY ANOTHER INITIATOR. If the TAS bit is set to one, all affected tasks in the task set for I_T nexuses other than the I_T nexus for which the CHECK CONDITION status was sent shall be completed with a TASK ABORTED status and no unit attention shall be established. For the I_T nexus to which the CHECK CONDITION status is sent, no status shall be sent for the tasks that are aborted.
10b	Reserved
11b	Affected tasks in the task set belonging to the I_T nexus on which a CHECK CONDITION status is returned shall be aborted when the status is sent.

A task aborted status (TAS) bit set to zero specifies that aborted tasks shall be terminated by the device server without any response to the application client. A TAS bit set to one specifies that tasks aborted by the actions of an I_T nexus other than the I_T nexus on which the command was received shall be terminated with a TASK ABORTED status (see SAM-3).

The report a check (RAC) bit provides control of reporting long busy conditions or CHECK CONDITION status. A RAC bit set to one specifies that the device server should return CHECK CONDITION status rather than returning BUSY status if the reason for returning the BUSY status may persist for a longer time than that specified by the BUSY TIMEOUT PERIOD field. A RAC bit set to zero specifies that the device server may return BUSY status regardless of the length of time the reason for returning BUSY status may persist.

The unit attention interlocks control (UA_INTLCK_CTRL) field (see table 248) controls the clearing of unit attention conditions reported in the same I_T_L_Q nexus transaction (see 3.1.46) as a CHECK CONDITION status and whether returning a status of BUSY, TASK SET FULL or RESERVATION CONFLICT results in the establishment of a unit attention condition (see SAM-3).

Table 248 — Unit attention interlocks control (UA_INTLCK_CTRL) field

Code	Definition
00b	The logical unit shall clear any unit attention condition reported in the same I_T_L_Q nexus transaction as a CHECK CONDITION status and shall not establish a unit attention condition when a task is terminated with BUSY, TASK SET FULL, or RESERVATION CONFLICT status.
01b	Reserved [a]
10b [a]	The logical unit shall not clear any unit attention condition reported in the same I_T_L_Q nexus transaction as a CHECK CONDITION status and shall not establish a unit attention condition when a task is terminated with BUSY, TASK SET FULL, or RESERVATION CONFLICT status.
11b [a]	The logical unit shall not clear any unit attention condition reported in the same I_T_L_Q nexus transaction as a CHECK CONDITION status and shall establish a unit attention condition for the initiator port associated with the I_T nexus on which the BUSY, TASK SET FULL, or RESERVATION CONFLICT status is being returned. Depending on the status, the additional sense code shall be set to PREVIOUS BUSY STATUS, PREVIOUS TASK SET FULL STATUS, or PREVIOUS RESERVATION CONFLICT STATUS. Until it is cleared by a REQUEST SENSE command, a unit attention condition shall be established only once for a BUSY, TASK SET FULL, or RESERVATION CONFLICT status regardless to the number of commands terminated with one of those status values.

[a] A REQUEST SENSE command still clears any unit attention condition that it reports.

A software write protect (SWP) bit set to one specifies that the logical unit shall inhibit writing to the medium after writing all cached or buffered write data, if any. When SWP is one, all commands requiring writes to the medium shall be terminated with CHECK CONDITION status, with the sense key set to DATA PROTECT, and the additional sense code set to WRITE PROTECTED. When SWP is one and the device type's command standard (see 3.1.18) defines a write protect (WP) bit in the DEVICE-SPECIFIC PARAMETER field in the mode parameter header, the WP bit shall be set to one for subsequent MODE SENSE commands. A SWP bit set to zero specifies that the logical unit may allow writing to the medium, depending on other write inhibit mechanisms implemented by the logical unit. When the SWP bit is set to zero, the value of the WP bit, if defined, is device type specific. For a list of commands affected by the SWP bit and details of the WP bit see the command standard for the specific device type.

An application tag owner (ATO) bit set to one specifies that the contents of the LOGICAL BLOCK APPLICATION TAG field in the protection information (see SBC-2), if any, shall not be modified by the device server. An ATO bit set to zero specifies that the contents of the LOGICAL BLOCK APPLICATION TAG field in the protection information, if any, may be modified by the device server. If the ATO bit is set to zero, the device server shall ignore the contents of the LOGICAL BLOCK APPLICATION TAG field in the protection information when received from the application client.

The AUTOLOAD MODE field specifies the action to be taken by a removable medium device server when a medium is inserted. For devices other than removable medium devices, this field is reserved. Table 249 shows the usage of the AUTOLOAD MODE field.

Table 249 — AUTOLOAD MODE field

Code	Definition
000b	Medium shall be loaded for full access.
001b	Medium shall be loaded for medium auxiliary memory access only.
010b	Medium shall not be loaded.
011b - 111b	Reserved

The BUSY TIMEOUT PERIOD field specifies the maximum time, in 100 milliseconds increments, that the application client allows for the device server to return BUSY status for unanticipated conditions that are not a routine part of commands from the application client. This value may be rounded down as defined in 5.4. A 0000h value in this field is undefined by this standard. An FFFFh value in this field is defined as an unlimited period.

The EXTENDED SELF-TEST COMPLETION TIME field contains advisory data that is the time in seconds that the device server requires to complete an extended self-test when the device server is not interrupted by subsequent commands and no errors occur during processing of the self-test. The application client should expect this time to increase significantly if other commands are sent to the logical unit while a self-test is in progress or if errors occur during the processing of the self-test. Device servers supporting SELF-TEST CODE field values other than 000b for the SEND DIAGNOSTIC command (see 6.28) shall support the EXTENDED SELF-TEST COMPLETION TIME field. The EXTENDED SELF-TEST COMPLETION TIME field is not changeable.

Bits 0, 1, and 2 of byte 4 as well as bytes 6 and 7 provide controls for the obsolete asynchronous event reporting feature.

7.4.7 Control Extension mode page

The Control Extension mode page (see table 250) is a subpage of the Control mode page (see 7.4.6) and provides controls over SCSI features that are applicable to all device types. The mode page policy (see 6.7) for this mode page shall be shared. If a field in this mode page is changed while there is a task already in the task set, it is vendor specific whether the old or new value of the field applies to that task.

Table 250 — Control Extension mode page

Bit / Byte	7	6	5	4	3	2	1	0
0	PS	SPF (1b)	PAGE CODE (0Ah)					
1	SUBPAGE CODE (01h)							
2	(MSB)	PAGE LENGTH (1Ch)						
3								(LSB)
4	Reserved					TCMOS	SCSIP	IALUAE
5	Reserved				INITIAL PRIORITY			
6	Reserved							
31								

The PS bit, SPF bit, PAGE CODE field, SUBPAGE CODE field, and PAGE LENGTH field are described in 7.4.5.

A SCSI precedence (SCSIP) bit set to one specifies that the timestamp changed using a SET TIMESTAMP command (see 6.32) shall take precedence over methods outside the scope of this standard. A SCSIP bit set to zero specifies that methods outside this standard may change the timestamp and that the SET TIMESTAMP command is illegal.

A timestamp changeable by methods outside this standard (TCMOS) bit set to one specifies that the timestamp may be initialized by methods outside the scope of this standard. A TCMOS bit set to zero specifies that the timestamp shall not be changed by any method except those defined by this standard.

An implicit asymmetric logical unit access enabled (IALUAE) bit set to one specifies that implicit asymmetric logical unit access state changes (see 5.8.2.7) are allowed. An IALUAE bit set to zero specifies that implicit asymmetric logical unit access state changes be disallowed and indicates that implicit asymmetric logical unit access state changes are disallowed or not supported.

The INITIAL PRIORITY field specifies the priority that may be used as the task priority (see SAM-3) for tasks received by the logical unit on any I_T nexus (i.e., on any I_T_L nexus) where a priority has not been modified by a SET PRIORITY command (see 6.30). If a MODE SELECT command specifies an initial priority value that is different than the current initial priority, then the device server shall set any priorities that have not be set with a SET PRIORITY command to a value different than the new initial priority value to the new priority. The device server shall establish a unit attention condition for the initiator port associated with every I_T_L nexus that receives a new priority, with the additional sense code set to PRIORITY CHANGED.

7.4.8 Disconnect-Reconnect mode page

The Disconnect-Reconnect mode page (see table 251) provides the application client the means to tune the performance of the service delivery subsystem. The name for this mode page, disconnect-reconnect, comes from the SCSI parallel interface. The mode page policy (see 6.7) for this mode page shall be shared or per target port. If the SCSI target device contains more than one target port, the mode page policy should be per target port.

Table 251 — Disconnect-Reconnect mode page

Bit / Byte	7	6	5	4	3	2	1	0
0	PS	SPF (0b)	PAGE CODE (02h)					
1	PAGE LENGTH (0Eh)							
2	BUFFER FULL RATIO							
3	BUFFER EMPTY RATIO							
4	(MSB)	BUS INACTIVITY LIMIT						
5								(LSB)
6	(MSB)	DISCONNECT TIME LIMIT						
7								(LSB)
8	(MSB)	CONNECT TIME LIMIT						
9								(LSB)
10	(MSB)	MAXIMUM BURST SIZE						
11								(LSB)
12	EMDP	FAIR ARBITRATION			DIMM	DTDC		
13	Reserved							
14	(MSB)	FIRST BURST SIZE						
15								(LSB)

The Disconnect-Reconnect mode page controls parameters that affect one or more target ports. The parameters that may be implemented are specified in the SCSI transport protocol standard (see 3.1.82) for the target port.

The parameters for a target port affect its behavior regardless of which initiator port is forming an I_T nexus with the target port. The parameters may be accessed by MODE SENSE (see 6.9) and MODE SELECT (see 6.7) commands directed to any logical unit accessible through the target port. If a parameter value is changed, all the device servers for all logical units accessible through the target port shall establish a unit attention condition for the initiator port associated with every I_T nexus that includes the target port except the I_T nexus on which the MODE SELECT command was received, with the additional sense code set to MODE PARAMETERS CHANGED.

If a parameter that is not appropriate for the specific SCSI transport protocol implemented by the target port is non-zero, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

An interconnect tenancy is a period of time during which a given pair of SCSI ports (i.e., an initiator port and a target port) are accessing the interconnect layer to communicate with each other (e.g., on arbitrated interconnects, a tenancy typically begins when a SCSI port successfully arbitrates for the interconnect and ends when the SCSI port releases the interconnect for use by other devices). Data and other information transfers take place during interconnect tenancies.

The PS bit, SPF bit, PAGE CODE field, and PAGE LENGTH field are described in 7.4.5.

The BUFFER FULL RATIO field specifies to the target port how full the buffer should be during read operations prior to requesting an interconnect tenancy. Target ports that do not implement the requested ratio should round down to the nearest implemented ratio as defined in 5.4.

The BUFFER EMPTY RATIO field specifies to the target port how empty the buffer should be during write operations prior to requesting an interconnect tenancy. Target ports that do not implement the requested ratio should round down to the nearest implemented ratio as defined in 5.4.

The buffer full and buffer empty ratios are numerators of a fractional multiplier that has 256 as its denominator. A value of zero indicates that the target port determines when to request an interconnect tenancy consistent with the disconnect time limit parameter. These parameters are advisory to the target port.

NOTE 45 - As an example, consider a target port with ten 512-byte buffers and a specified buffer full ratio of 3Fh. The formula is: INTEGER((ratio÷256)×number of buffers). Therefore in this example INTEGER((3Fh÷256)×10) = 2. During the read operations described in this example, the target port should request an interconnect tenancy whenever two or more buffers are full.

The BUS INACTIVITY LIMIT field specifies the maximum time that the target port is permitted to maintain an interconnect tenancy without data or information transfer. If the bus inactivity limit is exceeded, then the target port shall conclude the interconnect tenancy, within the restrictions placed on it by the applicable SCSI transport protocol. The contents of the DTDC field in this mode page also shall affect the duration of an interconnect tenancy. This value may be rounded as defined in 5.4. A value of zero specifies that there is no bus inactivity limit. Different SCSI transport protocols define different units of measure for the bus inactivity limit.

The DISCONNECT TIME LIMIT field specifies the minimum time that the target port shall wait between interconnect tenancies. This value may be rounded as defined in 5.4. A value of zero specifies that there is no disconnect time limit. Different SCSI transport protocols define different units of measure for the disconnect time limit.

The CONNECT TIME LIMIT field specifies the maximum duration of a single interconnect tenancy. If the connect time limit is exceeded, then the target port shall conclude the interconnect tenancy, within the restrictions placed on it by the applicable SCSI transport protocol. The contents of the DTDC field in this mode page also shall affect the duration of an interconnect tenancy. This value may be rounded as defined in 5.4. A value of zero specifies that there is no connect time limit. Different SCSI transport protocols define different units of measure for the connect time limit.

The MAXIMUM BURST SIZE field indicates the maximum amount of data that the target port shall transfer during a single data transfer operation. This value is expressed in increments of 512 bytes (i.e., a value of one means 512 bytes, two means 1 024 bytes, etc.). The relationship, if any, between data transfer operations and interconnect tenancies is defined in the individual SCSI transport protocol standards. A value of zero specifies there is no limit on the amount of data transferred per data transfer operation.

In terms of the SCSI transport protocol services (see SAM-3), the device server shall limit the Request Byte Count argument to the **Receive Data-Out** protocol service and the **Send Data-In** protocol service to the amount specified in the MAXIMUM BURST SIZE field.

The enable modify data pointers (EMDP) bit specifies whether or not the target port may transfer data out of order. If the EMDP bit is set to zero, the target port shall not transfer data out of order. If the EMDP bit is set to one, the target port is allowed to transfer data out of order.

The FAIR ARBITRATION field specifies whether the target port should use fair or unfair arbitration when requesting an interconnect tenancy. The field may be used to specify different fairness methods as defined in the individual SCSI transport protocol standards.

A disconnect immediate (DIMM) bit set to zero specifies that the target port may transfer data for a command during the same interconnect tenancy in which it receives the command. Whether or not the target port does so may depend upon the target port's internal algorithms, the rules of the applicable SCSI transport protocol, and settings of the other parameters in this mode page. A disconnect immediate (DIMM) bit set to one specifies that the target port shall not transfer data for a command during the same interconnect tenancy in which it receives the command.

The data transfer disconnect control (DTDC) field (see table 252) defines other restrictions on when multiple interconnect tenancies are permitted. A non-zero value in the DTDC field shall take precedence over other interconnect tenancy controls represented by other fields in this mode page.

Table 252 — Data transfer disconnect control

DTDC	Description
000b	Data transfer disconnect control is not used. Interconnect tenancies are controlled by other fields in this mode page.
001b	All data for a command shall be transferred within a single interconnect tenancy.
010b	Reserved
011b	All data and the response for a command shall be transferred within a single interconnect tenancy.
100b - 111b	Reserved

The FIRST BURST SIZE field specifies the maximum amount of data that may be transferred to the target port for a command along with the command (i.e., the first burst). This value is expressed in increments of 512 bytes (i.e., a value of one means 512 bytes, two means 1 024 bytes, etc.). The meaning of a value of zero is SCSI transport protocol specific. SCSI transport protocols supporting this field shall provide an additional mechanism to enable and disable the first burst function.

In terms of the SCSI transport protocol services (see SAM-3), the **Receive Data-Out** protocol service shall retrieve the first FIRST BURST SIZE amount of data from the first burst.

7.4.9 Extended mode page

The Extended mode page (see table 253) provides a means to specify subpages that are defined for all device types. Subpage code 00h is reserved. All Extended mode pages use the sub_page format.

Table 253 — Extended mode page

Bit / Byte	7	6	5	4	3	2	1	0
0	PS	SPF (1b)	PAGE CODE (15h)					
1	SUBPAGE CODE							
2	(MSB)			PAGE LENGTH (n-3)				
3								(LSB)
4	Subpage specific mode parameters							
n								

The PS bit, SPF bit, PAGE CODE field, and PAGE LENGTH field are described in 7.4.5.

7.4.10 Extended Device-Type Specific mode page

The Extended Device-Type Specific mode page (see table 254) provides a means to specify subpages that are defined differently for each device type. Subpage code 00h is reserved. All Extended Device-Type Specific mode pages use the sub_page format.

Table 254 — Extended Device-Type Specific mode page

Bit Byte	7	6	5	4	3	2	1	0
0	PS	SPF (1b)	PAGE CODE (16h)					
1	SUBPAGE CODE							
2	(MSB)	PAGE LENGTH (n-3)						
3								(LSB)
4	Subpage specific mode parameters							
n								

The PS bit, SPF bit, PAGE CODE field, and PAGE LENGTH field are described in 7.4.5.

7.4.11 Informational Exceptions Control mode page

The Informational Exceptions Control mode page (see table 255) defines the methods used by the device server to control the reporting and the operations of specific informational exception conditions. This page shall only apply to informational exceptions that report an additional sense code of FAILURE PREDICTION THRESHOLD EXCEEDED or an additional sense code of WARNING to the application client. The mode page policy (see 6.7) for this mode page shall be shared, or per I_T nexus.

Informational exception conditions occur as the result of vendor specific events within a logical unit. An informational exception condition may occur asynchronous to any commands issued by an application client.

NOTE 46 - Storage devices that support SMART (Self-Monitoring Analysis and Reporting Technology) for predictive failure software should use informational exception conditions.

Table 255 — Informational Exceptions Control mode page

Bit Byte	7	6	5	4	3	2	1	0
0	PS	SPF (0b)	PAGE CODE (1Ch)					
1	PAGE LENGTH (0Ah)							
2	PERF	Reserved	EBF	EWASC	DEXCPT	TEST	Reserved	LOGERR
3	Reserved				MRIE			
4	(MSB)	INTERVAL TIMER						
7								(LSB)
8	(MSB)	REPORT COUNT						
11								(LSB)

The PS bit, SPF bit, PAGE CODE field, and PAGE LENGTH field are described in 7.4.5.

If the log errors (LOGERR) bit is set to zero, the logging of informational exception conditions by a device server is vendor specific. If the LOGERR bit is set to one, the device server shall log informational exception conditions.

A TEST bit set to one shall create a test device failure at the next interval time, as specified by the INTERVAL TIMER field, if the DEXCPT bit is set to zero. When the TEST bit is set to one, the MRIE and REPORT COUNT fields shall apply as if the TEST bit were zero. The test device failure shall be reported with the additional sense code set to FAILURE PREDICTION THRESHOLD EXCEEDED (FALSE). If both the TEST bit and the DEXCPT bit are one, the MODE SELECT command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST. A TEST bit set to zero shall instruct the device server not to generate any test device failure notifications.

A disable exception control (DEXCPT) bit set to zero indicates the failure prediction threshold exceeded reporting shall be enabled. The method for reporting the failure prediction threshold exceeded when the DEXCPT bit is set to zero is determined from the MRIE field. A DEXCPT bit set to one indicates the device server shall disable reporting of the failure prediction threshold exceeded. The MRIE field is ignored when DEXCPT is set to one and EWASC is set to zero.

If the enable warning (EWASC) bit is set to zero, the device server shall disable reporting of the warning. The MRIE field is ignored when DEXCPT is set to one and EWASC is set to zero. If the EWASC bit is set to one, warning reporting shall be enabled. The method for reporting the warning when the EWASC bit is set to one is determined from the MRIE field.

If background functions are supported and the Enable Background Function (EBF) bit is set to one, then the device server shall enable background functions. If the EBF bit is set to zero, the device server shall disable the functions.

For the purposes of the EBF bit, background functions are defined as idle time functions that may impact performance that are performed by a device server operating without errors but do not impact the reliability of the logical unit (e.g., read scan).

If the performance (PERF) bit is set to zero, informational exception operations that are the cause of delays are acceptable. If the PERF bit is set to one, the device server shall not cause delays while doing informational exception operations. A PERF bit set to one may cause the device server to disable some or all of the informational exceptions operations, thereby limiting the reporting of informational exception conditions.

The value in the method of reporting informational exceptions (MRIE) field defines the method that shall be used by the device server to report informational exception conditions (see table 256). The priority of reporting multiple information exceptions is vendor specific.

Table 256 — Method of reporting informational exceptions (MRIE) field (part 1 of 2)

MRIE	Description
0h	**No reporting of informational exception condition:** The device server shall not report information exception conditions.
1h	**Asynchronous event reporting:** Obsolete
2h	**Generate unit attention:** The device server shall report informational exception conditions by establishing a unit attention condition (see SAM-3) for the initiator port associated with every I_T nexus, with the additional sense code set to indicate the cause of the informational exception condition. As defined in SAM-3, the command that has the CHECK CONDITION status with the sense key set to UNIT ATTENTION is not processed before the informational exception condition is reported.

[a] In some command standards (see 3.1.18), this is controlled by the post error (PER) bit in the Read-Write Error Recovery mode page.

Table 256 — Method of reporting informational exceptions (MRIE) field (part 2 of 2)

MRIE	Description
3h	**Conditionally generate recovered error:** The device server shall report informational exception conditions, if the reporting of recovered errors is allowed,[a] by returning a CHECK CONDITION status. If the TEST bit is set to zero, the status may be returned after the informational exception condition occurs on any command for which GOOD status or INTERMEDIATE status would have been returned. If the TEST bit is set to one, the status shall be returned on the next command received on any I_T nexus that is normally capable of returning an informational exception condition when the test bit is set to zero. The sense key shall be set to RECOVERED ERROR and the additional sense code shall indicate the cause of the informational exception condition. The command that returns the CHECK CONDITION for the informational exception shall complete without error before any informational exception condition may be reported.
4h	**Unconditionally generate recovered error:** The device server shall report informational exception conditions, regardless of whether the reporting of recovered errors is allowed,[a] by returning a CHECK CONDITION status. If the TEST bit is set to zero, the status may be returned after the informational exception condition occurs on any command for which GOOD status or INTERMEDIATE status would have been returned. If the TEST bit is set to one, the status shall be returned on the next command received on any I_T nexus that is normally capable of returning an informational exception condition when the TEST bit is set to zero. The sense key shall be set to RECOVERED ERROR and the additional sense code shall indicate the cause of the informational exception condition. The command that returns the CHECK CONDITION for the informational exception shall complete without error before any informational exception condition may be reported.
5h	**Generate no sense:** The device server shall report informational exception conditions by returning a CHECK CONDITION status. If the TEST bit is set to zero, the status may be returned after the informational exception condition occurs on any command for which GOOD status or INTERMEDIATE status would have been returned. If the TEST bit is set to one, the status shall be returned on the next command received on any I_T nexus that is normally capable of returning an informational exception condition when the TEST bit is set to zero. The sense key shall be set to NO SENSE and the additional sense code shall indicate the cause of the informational exception condition. The command that returns the CHECK CONDITION for the informational exception shall complete without error before any informational exception condition may be reported.
6h	**Only report informational exception condition on request:** The device server shall preserve the informational exception(s) information. To find out about information exception conditions the application client polls the device server by issuing a REQUEST SENSE command. The sense key shall be set to NO SENSE and the additional sense code shall indicate the cause of the informational exception condition.
7h - Bh	Reserved
Ch - Fh	Vendor specific

[a] In some command standards (see 3.1.18), this is controlled by the post error (PER) bit in the Read-Write Error Recovery mode page.

The value in the INTERVAL TIMER field is the period in 100 millisecond increments for reporting that an informational exception condition has occurred. The device server shall not report informational exception conditions more frequently than the time specified by the INTERVAL TIMER field and shall report them after the time specified by INTERVAL TIMER field has elapsed. After the informational exception condition has been reported the interval timer shall be restarted. A value of zero or FFFF FFFFh in the INTERVAL TIMER field indicates that the period for reporting an informational exception condition is vendor specific.

The value in the REPORT COUNT field is the number of times to report an informational exception condition to the application client. A value of zero in the REPORT COUNT field indicates there is no limit on the number of times the device server reports an informational exception condition.

The maintaining of the interval timer and the report counter across power cycles, hard resets, logical unit resets, and I_T nexus losses is vendor specific.

7.4.12 Power Condition mode page

The Power Condition mode page provides an application client with methods to control the power condition of a logical unit (see 5.9). These methods include:

a) Specifying that the logical unit transition to a power condition without delay; and
b) Activating and setting of idle condition and standby condition timers to specify that the logical unit wait for a period of inactivity before transitioning to a specified power condition.

The mode page policy (see 6.7) for this mode page shall be shared.

When a device server receives a command while in a power condition based on a setting in the Power Condition mode page, the logical unit shall transition to the power condition that allows the command to be processed. If either the idle condition timer or the standby condition timer has been set, then they shall be reset on receipt of the command. On completion of the command, the timer(s) shall be started.

Logical units that contain cache memory shall write all cached data to the medium for the logical unit (e.g.,as a logical unit does in response to a SYNCHRONIZE CACHE command as described in SBC-2) prior to entering into any power condition that prevents accessing the media (e.g., before a hard drive stops its spindle motor during transition to the standby power condition).

The logical unit shall use the values in the Power Condition mode page to control its power condition after a power on or a hard reset until a START STOP UNIT command setting a power condition is received.

Table 257 defines the Power Condition mode page.

Table 257 — Power Condition mode page

Bit / Byte	7	6	5	4	3	2	1	0
0	PS	SPF (0b)	PAGE CODE (1Ah)					
1	PAGE LENGTH (0Ah)							
2	Reserved							
3	Reserved						IDLE	STANDBY
4	(MSB)	IDLE CONDITION TIMER						
7								(LSB)
8	(MSB)	STANDBY CONDITION TIMER						
11								(LSB)

The PS bit, SPF bit, PAGE CODE field, and PAGE LENGTH field are described in 7.4.5.

The IDLE and STANDBY bits specify which timers are active.

If the IDLE bit is set to one and the STANDBY bit is set to zero, then the idle condition timer is active and the device server shall transition to the idle power condition when the idle condition timer is zero.

If the IDLE bit is set to zero, then the device server shall ignore the idle condition timer.

If the STANDBY bit is set to one and the IDLE bit is set to zero, then the standby condition timer is active and the device server shall transition to the standby power condition when the standby condition timer is zero.

If the STANDBY bit is set to zero, then the device server shall ignore the standby condition timer.

If both the IDLE and STANDBY bits are set to one, then both timers are active and run concurrently. When the idle condition timer is zero the device server shall transition to the idle power condition. When the standby condition timer is zero the device server shall transition to the standby power condition. If the standby condition timer is zero before the idle condition timer is zero, then the logical unit shall transition to the standby power condition.

The value in the IDLE CONDITION TIMER field specifies the inactivity time in 100 millisecond increments that the logical unit shall wait before transitioning to the idle power condition when the IDLE bit is set to one. The idle condition timer is expired when:

a) The IDLE CONDITION TIMER field is set to zero; or
b) The number of milliseconds specified by the value in the IDLE CONDITION TIMER field times 100 milliseconds has elapsed since the last activity (e.g., processing a command that requires the active power condition or performing a self test).

The value in the STANDBY CONDITION TIMER field specifies the inactivity time in 100 millisecond increments that the logical unit shall wait before transitioning to the standby power condition when the STANDBY bit is set to one. The standby condition timer is expired when:

a) The STANDBY CONDITION TIMER field is set to zero; or
b) The number of milliseconds specified by the value in the STANDBY CONDITION TIMER field times 100 milliseconds has elapsed since the last activity (e.g., processing any command or performing a self test).

7.4.13 Protocol Specific Logical Unit mode page

The Protocol Specific Logical Unit mode page (see table 258) provides protocol specific controls that are associated with a logical unit.

Table 258 — Protocol Specific Logical Unit mode page

Bit / Byte	7	6	5	4	3	2	1	0
0	PS	SPF (0b)	PAGE CODE (18h)					
1	PAGE LENGTH (n-1)							
2	Reserved					PROTOCOL IDENTIFIER		
3	Protocol specific mode parameters							
n								

During an I_T_L nexus, the Protocol Specific Logical Unit mode page controls parameters that affect both:

a) One or more target ports; and
b) The logical unit.

The parameters that may be implemented are specified in the SCSI transport protocol standard (see 3.1.82) for the target port. The mode page policy (see 6.7) for this mode page shall be shared or per target port and should be per target port.

The parameters for a target port and logical unit affect their behavior regardless of which initiator port is forming an I_T_L nexus with the target port and logical unit. If a parameter value is changed, the device server shall establish a unit attention condition for the initiator port associated with every I_T nexus except the I_T nexus on which the MODE SELECT command was received, with the additional sense code set to MODE PARAMETERS CHANGED.

The PS bit, SPF bit, PAGE CODE field, and PAGE LENGTH field are described in 7.4.5.

The value in the PROTOCOL IDENTIFIER field (see 7.5.1) defines the SCSI transport protocol to which the mode page applies. For a MODE SENSE command (see 6.9), the device server shall set the PROTOCOL IDENTIFIER field to one of the values shown in table 261 (see 7.5.1) to indicate the SCSI transport protocol used by the target port through which the MODE SENSE command is being processed. For a MODE SELECT command (see 6.7), the application client shall set the PROTOCOL IDENTIFIER field to one of the values shown in table 261 indicating the SCSI transport protocol to which the protocol specific mode parameters apply. If a device server receives a mode page containing a transport protocol identifier value other than the one used by the target port on which the MODE SELECT command was received, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

7.4.14 Protocol Specific Port mode page

The Protocol Specific Port mode page provides protocol specific controls that are associated with a SCSI port. The page_0 format (see table 259) is used for subpage 00h and sub_page format (see table 260) is used for subpages 01h through FEh. See the SCSI transport protocol standard (see 3.1.82) for definition of the protocol specific mode parameters.

Table 259 — Page_0 format Protocol Specific Port mode page

Bit / Byte	7	6	5	4	3	2	1	0
0	PS	SPF (0b)	PAGE CODE (19h)					
1	PAGE LENGTH (n-1)							
2	Reserved				PROTOCOL IDENTIFIER			
3	Protocol specific mode parameters							
n								

Table 260 — Sub_page format Protocol Specific Port mode page

Bit / Byte	7	6	5	4	3	2	1	0
0	PS	SPF (1b)	PAGE CODE (19h)					
1	SUBPAGE CODE							
2	(MSB)	PAGE LENGTH (n-3)						
3								(LSB)
4	Reserved							
5	Reserved				PROTOCOL IDENTIFIER			
6	Protocol specific mode parameters							
n								

The Protocol Specific Port mode page controls parameters that affect one or more target ports. The parameters that may be implemented are specified in the SCSI transport protocol standard (see 3.1.82) for the target port. The mode page policy (see 6.7) for this mode page shall be shared or per target port. If the SCSI target device contains more than one target port, the mode page policy should be per target port.

The parameters for a target port affect its behavior regardless of which initiator port is forming an I_T nexus with the target port. The parameters may be accessed by MODE SENSE (see 6.9) and MODE SELECT (see 6.7) commands directed to any logical unit accessible through the target port. If a parameter value is changed, the device server for all logical units accessible through the target port shall establish a unit attention condition for the

initiator port associated with every I_T nexus except the I_T nexus on which the MODE SELECT command was received, with the additional sense code set to MODE PARAMETERS CHANGED.

The PS bit, SPF bit, PAGE CODE field, SUBPAGE CODE field, and PAGE LENGTH field are described in 7.4.5.

The value in the PROTOCOL IDENTIFIER field (see 7.5.1) defines the SCSI transport protocol to which the mode page applies. For a MODE SENSE command, the device server shall set the PROTOCOL IDENTIFIER field to one of the values shown in table 261 (see 7.5.1) to indicate the SCSI transport protocol used by the target port through which the MODE SENSE command is being processed. For a MODE SELECT command, the application client shall set the PROTOCOL IDENTIFIER field to one of the values shown in table 261 indicating the SCSI transport protocol to which the protocol specific mode parameters apply. If a device server receives a mode page containing a transport protocol identifier value other than the one used by the target port on which the MODE SELECT command was received, then command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

7.5 Protocol specific parameters

7.5.1 Protocol specific parameters introduction

Some commands use protocol specific information in their CDBs or parameter lists. This subclause describes those protocol specific parameters.

Protocol specific parameters may include a PROTOCOL IDENTIFIER field (see table 261) as a reference for the SCSI transport protocol to which the protocol specific parameter applies.

Table 261 — PROTOCOL IDENTIFIER values

Protocol Identifier	Description	Protocol Standard
0h	Fibre Channel	FCP-2
1h	Parallel SCSI	SPI-5
2h	SSA	SSA-S3P
3h	IEEE 1394	SBP-3
4h	SCSI Remote Direct Memory Access Protocol	SRP
5h	Internet SCSI (iSCSI)	iSCSI
6h	SAS Serial SCSI Protocol	SAS
7h	Automation/Drive Interface Transport Protocol	ADT
8h	AT Attachment Interface (ATA/ATAPI)	ATA/ATAPI-7
9h - Eh	Reserved	
Fh	No specific protocol	

7.5.2 Alias entry protocol specific designations

7.5.2.1 Introduction to alias entry protocol specific designations

The alias entries (see 6.2.2) in the parameter data for the CHANGE ALIASES command (see 6.2) and REPORT ALIASES command (see 6.19) include FORMAT CODE, DESIGNATION LENGTH and DESIGNATION fields whose contents and meaning are based on the SCSI transport protocol specified in a PROTOCOL IDENTIFIER field (see 7.5.1). This subclause defines the SCSI transport protocol specific format codes, designation lengths, and designations.

7.5.2.2 Fibre Channel specific alias entry designations

7.5.2.2.1 Introduction to Fibre Channel specific alias entry designations

If an alias entry PROTOCOL IDENTIFIER field contains the Fibre Channel protocol identifier (0h, see table 261), the FORMAT CODE, DESIGNATION LENGTH and DESIGNATION fields shall be as defined in table 262.

Table 262 — Fibre Channel alias entry format codes

Format Code	Description	Designation Length (bytes)	Designation Subclause
00h	World Wide Port Name	8	7.5.2.2.2
01h	World Wide Port Name with N_Port checking	12	7.5.2.2.3
02h - FFh	Reserved		

7.5.2.2.2 Fibre Channel world wide port name alias entry designation

If the PROTOCOL IDENTIFIER and FORMAT CODE fields specify a Fibre Channel world wide port name designation, the alias entry DESIGNATION field shall have the format shown in table 263.

Table 263 — Fibre Channel world wide port name alias entry designation

Bit / Byte	7	6	5	4	3	2	1	0
0	See table 47 in 6.2.2.							
15								
16	FIBRE CHANNEL WORLD WIDE PORT NAME							
23								

The FIBRE CHANNEL WORLD WIDE PORT NAME field shall contain the port world wide name defined by the port login (PLOGI) extended link service (see FC-FS).

A Fibre Channel world wide port name designation is valid (see 6.2.3) if the device server has access to a SCSI domain formed by a Fibre Channel fabric and the fabric contains a port with the specified port world wide name.

7.5.2.2.3 Fibre Channel world wide port name with N_Port checking alias entry designation

If the PROTOCOL IDENTIFIER and FORMAT CODE fields specify a Fibre Channel world wide port name with N_Port checking designation, the alias entry DESIGNATION field shall have the format shown in table 264.

Table 264 — Fibre Channel world wide port name with N_Port checking alias entry designation

Bit / Byte	7	6	5	4	3	2	1	0
0	See table 47 in 6.2.2.							
15								
16	FIBRE CHANNEL WORLD WIDE PORT NAME							
23								
24	Reserved							
25	(MSB) N_PORT							
27								(LSB)

The FIBRE CHANNEL WORLD WIDE PORT NAME field shall contain the port world wide name defined by the port login (PLOGI) extended link service (see FC-FS).

The N_PORT field shall contain the FC_FS port D_ID to be used to transport frames including PLOGI and FCP-2 related frames.

A Fibre Channel world wide port name with N_Port checking designation is valid (see 6.2.3) if all of the following conditions are true:

a) The device server has access to a SCSI domain formed by a Fibre Channel fabric;
b) The fabric contains a port with the specified port World Wide Name; and
c) The value in the N_PORT field is the N_Port identifier of a Fibre Channel port whose port world wide name matches that in the FIBRE CHANNEL WORLD WIDE PORT NAME field.

7.5.2.3 RDMA specific alias entry designations

7.5.2.3.1 Introduction to RDMA specific alias entry designations

If an alias entry PROTOCOL IDENTIFIER field contains the SCSI RDMA protocol identifier (4h, see table 261), the FORMAT CODE, DESIGNATION LENGTH and DESIGNATION fields shall be as defined in table 265.

Table 265 — RDMA alias entry format codes

Format Code	Description	Designation Length (bytes)	Designation Subclause
00h	Target Port Identifier	16	7.5.2.3.2
01h	InfiniBand™ Global Identifier with Target Port Identifier checking	32	7.5.2.3.3
02h - FFh	Reserved		
Note: InfiniBand is a trademark of the InfiniBand Trade Association (see http://www.infinibandta.org/)			

7.5.2.3.2 RDMA target port identifier alias entry designation

If the PROTOCOL IDENTIFIER and FORMAT CODE fields specify a SCSI RDMA target port identifier designation, the alias entry DESIGNATION field shall have the format shown in table 266.

Table 266 — RDMA target port identifier alias entry designation

Bit / Byte	7	6	5	4	3	2	1	0
0	See table 47 in 6.2.2.							
15								
16	TARGET PORT IDENTIFIER							
31								

The TARGET PORT IDENTIFIER field shall contain an SRP target port identifier.

A SCSI RDMA target port identifier designation is valid (see 6.2.3) if the device server has access to an SRP SCSI domain containing the specified SRP target port identifier.

7.5.2.3.3 InfiniBand global identifier with target port identifier checking alias entry designation

If the PROTOCOL IDENTIFIER and FORMAT CODE fields specify an InfiniBand global identifier with target port identifier checking designation, the alias entry designation field shall have the format shown in table 267.

Table 267 — InfiniBand global identifier with target port identifier checking alias entry designation

Bit / Byte	7	6	5	4	3	2	1	0
0	See table 47 in 6.2.2.							
15								
16	INFINIBAND GLOBAL IDENTIFIER							
31								
32	TARGET PORT IDENTIFIER							
47								

The INFINIBAND GLOBAL IDENTIFIER field contains an InfiniBand global identifier (GID) of an InfiniBand port connected to an SRP target port.

The TARGET PORT IDENTIFIER field shall contain an SRP target port identifier.

An InfiniBand global identifier with target port identifier checking designation is valid (see 6.2.3) if all of the following conditions are true:

a) The device server has access to an SRP SCSI domain layered on InfiniBand;
b) The device server has access to an SRP target port based on the InfiniBand global identifier specified in the INFINIBAND GLOBAL IDENTIFIER field; and
c) The value in the TARGET PORT IDENTIFIER field is the SRP target port identifier for the SRP target port that is accessible via the InfiniBand global identifier contained in the INFINIBAND GLOBAL IDENTIFIER field.

7.5.2.4 Internet SCSI specific alias entry designations

7.5.2.4.1 Introduction to Internet SCSI specific alias entry designations

If an alias entry PROTOCOL IDENTIFIER field contains the iSCSI protocol identifier (5h, see table 261), the FORMAT CODE, DESIGNATION LENGTH and DESIGNATION fields shall be as defined in table 268.

Table 268 — iSCSI alias entry format codes

Format Code	Description	Designation Length (bytes, maximum)	Designation Subclause
00h	iSCSI name	224	7.5.2.4.2
01h	iSCSI name with binary IPv4 address	236	7.5.2.4.3
02h	iSCSI name with IPName	488	7.5.2.4.4
03h	iSCSI name with binary IPv6 address	248	7.5.2.4.5
04h - FFh	Reserved		

NOTE 47 - A designation that contains no IP addressing information or contains IP addressing information that does not address the named SCSI target device may require a device server to have access to a name server or to other discovery protocols to resolve the given iSCSI name to an IP address through which the device server may establish iSCSI Login. Access to such a service is protocol specific and vendor specific.

7.5.2.4.2 iSCSI name alias entry designation

If the PROTOCOL IDENTIFIER and FORMAT CODE fields specify iSCSI name designation, the alias entry DESIGNATION field shall have the format shown in table 269.

Table 269 — iSCSI name alias entry designation

Bit / Byte	7	6	5	4	3	2	1	0
0	See table 47 in 6.2.2.							
15								
16	(MSB)	ISCSI NAME						
4m-1								(LSB)

The null-terminated, null-padded (see 4.4.2) ISCSI NAME field shall contain the iSCSI name of an iSCSI node (see RFC 3720). The number of bytes in the ISCSI NAME field shall be a multiple of four.

An iSCSI name designation is valid if the device server has access to a SCSI domain containing an Internet protocol network and that network contains an iSCSI node with the specified iSCSI name.

7.5.2.4.3 iSCSI name with binary IPv4 address alias entry designation

If the PROTOCOL IDENTIFIER and FORMAT CODE fields specify iSCSI name with binary IPv4 address designation, the alias entry DESIGNATION field shall have the format shown in table 270.

Table 270 — iSCSI name with binary IPv4 address alias entry designation

Bit / Byte	7	6	5	4	3	2	1	0
0	See table 47 in 6.2.2.							
15								
16	(MSB)	ISCSI NAME						
4m-1								(LSB)
4m	(MSB)	IPV4 ADDRESS						
4m+3								(LSB)
4m+4	Reserved							
4m+5	Reserved							
4m+6	(MSB)	PORT NUMBER						
4m+7								(LSB)
4m+8	Reserved							
4m+9	Reserved							
4m+10	(MSB)	INTERNET PROTOCOL NUMBER						
4m+11								(LSB)

The null-terminated, null-padded (see 4.4.2) ISCSI NAME field shall contain the iSCSI name of an iSCSI node (see RFC 3720). The number of bytes in the ISCSI NAME field shall be a multiple of four.

The IPV4 ADDRESS field shall contain an IPv4 address (see RFC 791).

The PORT NUMBER field shall contain a TCP port number (see 3.1.119). The TCP port number shall conform to the requirements defined by iSCSI (see RFC 3720).

The INTERNET PROTOCOL NUMBER field shall contain an Internet protocol number (see 3.1.54). The Internet protocol> number shall conform to the requirements defined by iSCSI (see RFC 3720).

An iSCSI name designation is valid if the device server has access to a SCSI domain containing an Internet protocol network and that network contains an iSCSI node with the specified iSCSI name.

The IPv4 address, port number, and Internet protocol number provided in the designation may be used by a device server for addressing to discover and establish communication with the named iSCSI node. Alternatively, the device server may use other protocol specific or vendor specific methods to discover and establish communication with the named iSCSI node.

7.5.2.4.4 iSCSI name with IPname alias entry designation

If the PROTOCOL IDENTIFIER and FORMAT CODE fields specify iSCSI name with IPname designation, the alias entry DESIGNATION field shall have the format shown in table 271.

Table 271 — iSCSI name with IPname alias entry designation

Bit / Byte	7	6	5	4	3	2	1	0
0	See table 47 in 6.2.2.							
15								
16	(MSB)	ISCSI NAME						
k								(LSB)
k+1	(MSB)	IPNAME						
n								(LSB)
n+1	PAD (if needed)							
4m-1								
4m	Reserved							
4m+1	Reserved							
4m+2	(MSB)	PORT NUMBER						
4m+3								(LSB)
4m+4	Reserved							
4m+5	Reserved							
4m+6	(MSB)	INTERNET PROTOCOL NUMBER						
4m+7								(LSB)

The null-terminated (see 4.4.2) ISCSI NAME field shall contain the iSCSI name of an iSCSI node (see RFC 3720).

The null-terminated (see 4.4.2) IPNAME field shall contain a Internet protocol domain name (see 3.1.53).

The PAD field shall contain zero to three bytes set to zero such that the total length of the ISCSI NAME, IPNAME, and PAD fields is a multiple of four. Device servers shall ignore the PAD field.

The PORT NUMBER field shall contain a TCP port number (see 3.1.119). The TCP port number shall conform to the requirements defined by iSCSI (see RFC 3720).

The INTERNET PROTOCOL NUMBER field shall contain an Internet protocol number (see 3.1.54). The Internet protocol> number shall conform to the requirements defined by iSCSI (see RFC 3720).

An iSCSI name designation is valid if the device server has access to a SCSI domain containing an Internet protocol network and that network contains an iSCSI node with the specified iSCSI name.

The Internet protocol domain name, port number, and Internet protocol number provided in the designation may be used by a device server for addressing to discover and establish communication with the named iSCSI node. Alternatively, the device server may use other protocol specific or vendor specific methods to discover and establish communication with the named iSCSI node.

7.5.2.4.5 iSCSI name with binary IPv6 address alias entry designation

If the PROTOCOL IDENTIFIER and FORMAT CODE fields specify iSCSI name with binary IPv6 address designation, the alias entry DESIGNATION field shall have the format shown in table 272.

Table 272 — iSCSI name with binary IPv6 address alias entry designation

Bit / Byte	7	6	5	4	3	2	1	0
0	See table 47 in 6.2.2.							
15								
16	(MSB)			ISCSI NAME				
n								(LSB)
4m	(MSB)			IPV6 ADDRESS				
4m+15								(LSB)
4m+16	Reserved							
4m+17	Reserved							
4m+18	(MSB)			PORT NUMBER				
4m+19								(LSB)
4m+20	Reserved							
4m+21	Reserved							
4m+22	(MSB)			INTERNET PROTOCOL NUMBER				
4m+23								(LSB)

The null-terminated, null-padded (see 4.4.2) ISCSI NAME field shall contain the iSCSI name of an iSCSI node (see RFC 3720).

The IPV6 ADDRESS field shall contain an IPv6 address (see RFC 2373).

The PORT NUMBER field shall contain a TCP port number (see 3.1.119). The TCP port number shall conform to the requirements defined by iSCSI (see RFC 3720).

The INTERNET PROTOCOL NUMBER field shall contain an Internet protocol number (see 3.1.54). The Internet protocol> number shall conform to the requirements defined by iSCSI (see RFC 3720).

An iSCSI name designation is valid if the device server has access to a SCSI domain containing an Internet protocol network and that network contains an iSCSI node with the specified iSCSI name.

The IPv6 address, port number and Internet protocol number provided in the designation may be used by a device server for addressing to discover and establish communication with the named iSCSI node. Alternatively, the

device server may use other protocol specific or vendor specific methods to discover and establish communication with the named iSCSI node.

7.5.3 EXTENDED COPY protocol specific target descriptors

7.5.3.1 Introduction to EXTENDED COPY protocol specific target descriptors

The protocol-specific target descriptors in the parameter data of the EXTENDED COPY command (see 6.3) are described in this subclause. An introduction to EXTENDED COPY target descriptors is provided in 6.3.6.1.

NOTE 48 - In the EXTENDED COPY command the target in target descriptor refers to a copy target device (i.e., the source or destination of an EXTENDED COPY operation), not a SCSI target device. Target descriptors specify logical unit numbers or proxy tokens and may also specify initiator ports, target ports, and SCSI target devices used to access those logical units.

7.5.3.2 Fibre Channel N_Port_Name EXTENDED COPY target descriptor format

The target descriptor format shown in table 273 is used by an EXTENDED COPY command to specify an FCP copy target device using its Fibre Channel N_Port_Name.

Table 273 — Fibre Channel N_Port_Name EXTENDED COPY target descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE CODE (E0h)							
1	LU ID TYPE		NUL	PERIPHERAL DEVICE TYPE				
2	(MSB)			RELATIVE INITIATOR PORT IDENTIFIER				
3								(LSB)
4	LU IDENTIFIER							
11								
12	N_PORT_NAME							
19								
20	Reserved							
27								
28	Device type specific parameters							
31								

The DESCRIPTOR TYPE CODE field, LU ID TYPE field, PERIPHERAL DEVICE TYPE field, NUL bit, RELATIVE INITIATOR PORT IDENTIFIER field, LU IDENTIFIER field, and the device type specific parameters are described in 6.3.6.1.

The N_PORT_NAME field shall contain the N_Port_Name defined by the port login (PLOGI) extended link service (see FC-FS).

NOTE 49 - The N_Port_Name EXTENDED COPY target descriptor format necessitates translating the N_Port_Name to an N_Port_ID (see FC-FS and 7.5.3.3).

7.5.3.3 Fibre Channel N_Port_ID EXTENDED COPY target descriptor format

The target descriptor format shown in table 274 is used by an EXTENDED COPY command to specify an FCP copy target device using its Fibre Channel N_Port_ID.

Table 274 — Fibre Channel N_Port_ID EXTENDED COPY target descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE CODE (E1h)							
1	LU ID TYPE		NUL	PERIPHERAL DEVICE TYPE				
2	(MSB)			RELATIVE INITIATOR PORT IDENTIFIER				
3								(LSB)
4				LU IDENTIFIER				
11								
12				Reserved				
20								
21	(MSB)			N_PORT_ID				
22								
23								(LSB)
24				Reserved				
27								
28				Device type specific parameters				
31								

The DESCRIPTOR TYPE CODE field, LU ID TYPE field, PERIPHERAL DEVICE TYPE field, NUL bit, RELATIVE INITIATOR PORT IDENTIFIER field, LU IDENTIFIER field, and the device type specific parameters are described in 6.3.6.1.

The N_PORT_ID field shall contain the port D_ID (see FC-FS) to be used to transport frames including PLOGI and FCP-2 related frames.

NOTE 50 - Use of N_Port_ID addressing restricts this target descriptor format to a single Fibre Channel fabric.

7.5.3.4 Fibre Channel N_Port_ID with N_Port_Name checking EXTENDED COPY target descriptor format

The target descriptor format shown in table 275 is used by an EXTENDED COPY command to specify an FCP copy target device using its Fibre Channel N_Port_ID and to require the copy manager to verify that the N_Port_Name of the specified N_Port matches the value in the target descriptor.

Table 275 — Fibre Channel N_Port_ID with N_Port_Name checking target descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE CODE (E2h)							
1	LU ID TYPE		NUL	PERIPHERAL DEVICE TYPE				
2	(MSB)			RELATIVE INITIATOR PORT IDENTIFIER				
3								(LSB)
4				LU IDENTIFIER				
11								
12				N_PORT_NAME				
19								
20	Reserved							
21	(MSB)			N_PORT_ID				
22								
23								(LSB)
24				Reserved				
27								
28				Device type specific parameters				
31								

The DESCRIPTOR TYPE CODE field, LU ID TYPE field, PERIPHERAL DEVICE TYPE field, NUL bit, RELATIVE INITIATOR PORT IDENTIFIER field, LU IDENTIFIER field, and the device type specific parameters are described in 6.3.6.1.

The N_PORT_NAME field shall contain the N_Port_Name defined by the port login (PLOGI) extended link service (see FC-FS).

The N_PORT field shall contain the port D_ID (see FC-FS) to be used to transport frames including PLOGI and FCP-2 related frames.

NOTE 51 - Use of N_Port addressing restricts this target descriptor format to a single fabric.

When the copy manager first processes a segment descriptor that references this target descriptor, it shall confirm that the D_ID in the N_PORT_ID field is associated with the N_Port_Name in the N_PORT_NAME field. If the confirmation fails, the command shall be terminated because the copy target device is unavailable (see 6.3.3). The SCSI device processing this target descriptor shall track configuration changes that affect the D_ID value for the duration of the task. An application client is responsible for tracking configuration changes between commands.

7.5.3.5 SCSI Parallel T_L EXTENDED COPY target descriptor format

The target descriptor format shown in table 276 is used by an EXTENDED COPY command to specify a SPI copy target device using its SCSI target identifier.

Table 276 — SCSI Parallel T_L EXTENDED COPY target descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE CODE (E3h)							
1	LU ID TYPE		NUL	PERIPHERAL DEVICE TYPE				
2	(MSB)			RELATIVE INITIATOR PORT IDENTIFIER				
3								(LSB)
4	LU IDENTIFIER							
11								
12	Vendor specific							
13	TARGET IDENTIFIER							
14	Reserved							
27								
28	Device type specific parameters							
31								

The DESCRIPTOR TYPE CODE field, LU ID TYPE field, PERIPHERAL DEVICE TYPE field, NUL bit, RELATIVE INITIATOR PORT IDENTIFIER field, LU IDENTIFIER field, and the device type specific parameters are described in 6.3.6.1.

The TARGET IDENTIFIER field specifies the SCSI target identifier (see SPI-5).

7.5.3.6 IEEE 1394 EUI-64 EXTENDED COPY target descriptor format

The target descriptor format shown in table 277 is used by an EXTENDED COPY command to specify an SBP copy target device using its IEEE 1394 Extended Unique Identifier, 64-bits (EUI-64) and configuration ROM (Read-Only Memory) directory identifier.

Table 277 — IEEE 1394 EUI-64 EXTENDED COPY target descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE CODE (E8h)							
1	LU ID TYPE		NUL	PERIPHERAL DEVICE TYPE				
2	(MSB)			RELATIVE INITIATOR PORT IDENTIFIER				
3								(LSB)
4	LU IDENTIFIER							
11								
12	EUI-64							
19								
20	DIRECTORY ID							
22								
23	Reserved							
27								
28	Device type specific parameters							
31								

The DESCRIPTOR TYPE CODE field, LU ID TYPE field, PERIPHERAL DEVICE TYPE field, NUL bit, RELATIVE INITIATOR PORT IDENTIFIER field, LU IDENTIFIER field, and the device type specific parameters are described in 6.3.6.1.

The EUI-64 field shall contain the node's unique identifier (EUI-64) obtained from the configuration ROM bus information block, as specified by ANSI IEEE 1394a:2000.

NOTE 52 - ANSI IEEE 1394a-2000 separately labels the components of the EUI-64 as NODE_VENDOR_ID, CHIP_ID_HI and CHIP_ID_LO. Collectively these form the node's EUI-64.

The DIRECTORY ID field shall contain the copy target device's directory identifier, as specified by ISO/IEC 13213:1994.

7.5.3.7 RDMA EXTENDED COPY target descriptor format

The target descriptor format shown in table 278 is used by an EXTENDED COPY command to specify an SRP copy target device using its RDMA SRP target port identifier.

Table 278 — RDMA EXTENDED COPY target descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE CODE (E7h)							
1	LU ID TYPE		NUL	PERIPHERAL DEVICE TYPE				
2	(MSB)			RELATIVE INITIATOR PORT IDENTIFIER				
3								(LSB)
4	LU IDENTIFIER							
11								
12	TARGET PORT IDENTIFIER							
27								
28	Device type specific parameters							
31								

The DESCRIPTOR TYPE CODE field, LU ID TYPE field, PERIPHERAL DEVICE TYPE field, NUL bit, RELATIVE INITIATOR PORT IDENTIFIER field, LU IDENTIFIER field, and the device type specific parameters are described in 6.3.6.1.

The TARGET PORT IDENTIFIER field contains the SRP target port identifier (see SRP).

7.5.3.8 iSCSI binary IPv4 address EXTENDED COPY target descriptor format

The target descriptor format shown in table 279 is used by an EXTENDED COPY command to specify an iSCSI copy target device using its binary IPv4 (Internet Protocol version 4) address.

Table 279 — iSCSI binary IPv4 address EXTENDED COPY target descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE CODE (E5h)							
1	LU ID TYPE		NUL	PERIPHERAL DEVICE TYPE				
2	(MSB)			RELATIVE INITIATOR PORT IDENTIFIER				
3								(LSB)
4				LU IDENTIFIER				
11								
12	(MSB)			IPV4 ADDRESS				
15								(LSB)
16				Reserved				
21								
22	(MSB)			PORT NUMBER				
23								(LSB)
24	Reserved							
25	Reserved							
26	(MSB)			INTERNET PROTOCOL NUMBER				
27								(LSB)
28				Device type specific parameters				
31								

The DESCRIPTOR TYPE CODE field, LU ID TYPE field, PERIPHERAL DEVICE TYPE field, NUL bit, RELATIVE INITIATOR PORT IDENTIFIER field, LU IDENTIFIER field, and the device type specific parameters are described in 6.3.6.1.

The IPV4 ADDRESS field shall contain an IPv4 address (see RFC 791).

The PORT NUMBER field shall contain the TCP port number (see 3.1.119). The TCP port number shall conform to the requirements defined by iSCSI (see RFC 3720).

The INTERNET PROTOCOL NUMBER field shall contain an Internet protocol number (see 3.1.54). The Internet protocol number shall conform to the requirements defined by iSCSI (see RFC 3720).

7.5.3.9 SAS serial SCSI protocol target descriptor format

The target descriptor format shown in table 280 is used by an EXTENDED COPY command to specify a SAS copy target device using its SAS address.

Table 280 — SAS serial SCSI protocol EXTENDED COPY target descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE CODE (E9h)							
1	LU ID TYPE		NUL	PERIPHERAL DEVICE TYPE				
2	(MSB)			RELATIVE INITIATOR PORT IDENTIFIER				
3								(LSB)
4	LU IDENTIFIER							
11								
12	SAS ADDRESS							
19								
20	Reserved							
27								
28	Device type specific parameters							
31								

The DESCRIPTOR TYPE CODE field, LU ID TYPE field, PERIPHERAL DEVICE TYPE field, NUL bit, RELATIVE INITIATOR PORT IDENTIFIER field, LU IDENTIFIER field, and the device type specific parameters are described in 6.3.6.1.

The SAS ADDRESS field contains the SAS address (see SAS).

7.5.4 TransportID identifiers

7.5.4.1 Overview of TransportID identifiers

An application client may use a TransportID to specify an initiator port other than the initiator port that is transporting the command and parameter data (e.g., as an Access identifiers (see 8.3.1.3.2) in ACL ACEs, as the initiator port in the I_T nexus to which PERSISTENT RESERVE OUT command with REGISTER AND MOVE service action (see 5.6.7) is moving a persistent reservation).

TransportIDs (see table 281) shall be at least 24 bytes long and shall be a multiple of four bytes in length.

Table 281 — TransportID format

Bit / Byte	7	6	5	4	3	2	1	0
0	FORMAT CODE		Reserved		PROTOCOL IDENTIFIER			
1	SCSI transport protocol specific data							
n								

The FORMAT CODE field specifies the format of the TransportID. All format codes not specified in this standard are reserved.

The PROTOCOL IDENTIFIER field (see table 261 in 7.5.1) specifies the SCSI transport protocol to which the TransportID applies.

The format of the SCSI transport protocol specific data depends on the value in the PROTOCOL IDENTIFIER field. The SCSI transport protocol specific data in a TransportID shall only include initiator port identifiers, initiator port names, or initiator device names (see SAM-3) that persist across hard resets and I_T nexus losses. TransportID formats specific to SCSI transport protocols are listed in table 282.

Table 282 — TransportID formats for specific SCSI transport protocols

SCSI transport Protocol	Protocol Standard	Reference
Fibre Channel	FCP-2	7.5.4.2
Parallel SCSI	SPI-5	7.5.4.3
IEEE 1394	SBP-3	7.5.4.4
Remote Direct Memory Access (RDMA)	SRP	7.5.4.5
Internet SCSI (iSCSI)	iSCSI	7.5.4.6
SAS Serial SCSI Protocol	SAS-1.1	7.5.4.7

7.5.4.2 TransportID for initiator ports using SCSI over Fibre Channel

A Fibre Channel TransportID (see table 283) specifies an FCP-2 initiator port based on the N_Port_Name belonging to that initiator port.

Table 283 — Fibre Channel TransportID format

Bit / Byte	7	6	5	4	3	2	1	0
0	FORMAT CODE (00b)		Reserved		PROTOCOL IDENTIFIER (0h)			
1	Reserved							
7								
8	N_PORT_NAME							
15								
16	Reserved							
23								

The N_PORT_NAME field shall contain the N_Port_Name defined by the Physical Log In (PLOGI) extended link service (see FC-FS).

7.5.4.3 TransportID for initiator ports using a parallel SCSI bus

A parallel SCSI bus TransportID (see table 284) specifies a SPI-5 initiator port based on the SCSI address of an initiator port and the relative port identifier of the target port through which the application client accesses the SCSI target device.

Table 284 — Parallel SCSI bus TransportID format

Bit / Byte	7	6	5	4	3	2	1	0
0	FORMAT CODE (00b)		Reserved		PROTOCOL IDENTIFIER (1h)			
1	Reserved							
2	(MSB)		SCSI ADDRESS					
3								(LSB)
4			Obsolete					
5								
6	(MSB)		RELATIVE TARGET PORT IDENTIFIER					
7								(LSB)
8	Reserved							
23								

The SCSI ADDRESS field specifies the SCSI address (see SPI-5) of the initiator port.

The RELATIVE TARGET PORT IDENTIFIER field specifies the relative port identifier (see 3.1.88) of the target port for which the initiator port SCSI address applies. If the RELATIVE TARGET PORT IDENTIFIER does not reference a target port in the SCSI target device, the TransportID is invalid.

7.5.4.4 TransportID for initiator ports using SCSI over IEEE 1394

An IEEE 1394 TransportID (see table 285) specifies an SBP-3 initiator port based on the EUI-64 initiator port name belonging to that initiator port.

Table 285 — IEEE 1394 TransportID format

Bit / Byte	7	6	5	4	3	2	1	0
0	FORMAT CODE (00b)		Reserved		PROTOCOL IDENTIFIER (3h)			
1	Reserved							
7								
8	EUI-64 NAME							
15								
16	Reserved							
23								

The EUI-64 NAME field shall contain the EUI-64 IEEE 1394 node unique identifier (see SBP-3) for an initiator port.

7.5.4.5 TransportID for initiator ports using SCSI over an RDMA interface

A RDMA TransportID (see table 286) specifies an SRP initiator port based on the world wide unique initiator port name belonging to that initiator port.

Table 286 — RDMA TransportID format

Bit / Byte	7	6	5	4	3	2	1	0
0	FORMAT CODE (00b)		Reserved		PROTOCOL IDENTIFIER (4h)			
1	Reserved							
7								
8	INITIATOR PORT IDENTIFIER							
23								

The INITIATOR PORT IDENTIFIER field shall contain an SRP initiator port identifier (see SRP).

7.5.4.6 TransportID for initiator ports using SCSI over iSCSI

An iSCSI TransportID specifies an iSCSI initiator port using one of the TransportID formats listed in table 287.

Table 287 — iSCSI TransportID formats

Format code	Description
00b	Initiator port is identified using the world wide unique initiator device name of the iSCSI initiator device containing the initiator port (see table 288).
01b	Initiator port is identified using the world wide unique initiator port identifier (see table 289).
10b - 11b	Reserved

iSCSI TransportIDs with a format code of 00b may be rejected. iSCSI TransportIDs with a format code of 01b should not be rejected.

A iSCSI TransportID with the format code set to 00b (see table 288) specifies an iSCSI initiator port based on the world wide unique initiator device name of the iSCSI initiator device containing the initiator port.

Table 288 — iSCSI initiator device TransportID format

Bit / Byte	7	6	5	4	3	2	1	0
0	FORMAT CODE (00b)		Reserved		PROTOCOL IDENTIFIER (5h)			
1	Reserved							
2	(MSB)	ADDITIONAL LENGTH (m-3)						
3								(LSB)
4	(MSB)	ISCSI NAME						
m								(LSB)

The ADDITIONAL LENGTH field specifies the number of bytes that follow in the TransportID. The additional length shall be at least 20 and shall be a multiple of four.

The null-terminated, null-padded (see 4.4.2) ISCSI NAME field shall contain the iSCSI name of an iSCSI initiator node (see RFC 3720). The first ISCSI NAME field byte containing an ASCII null character terminates the ISCSI NAME field without regard for the specified length of the iSCSI TransportID or the contents of the ADDITIONAL LENGTH field.

NOTE 53 - The maximum length of the iSCSI TransportID is 228 bytes because the iSCSI name length does not exceed 223 bytes.

If a iSCSI TransportID with the format code set to 00b appears in a PERSISTENT RESERVE OUT parameter list (see 6.12.3), all initiator ports known to the device server with an iSCSI node name matching the one in the TransportID shall be registered.

If a iSCSI TransportID with the format code set to 00b appears in an ACE access identifier (see 8.3.1.3.2), the logical units listed in the ACE shall be accessible to any initiator port with an iSCSI node name matching the value in the TransportID. The access controls coordinator shall reject any command that attempts to define more than one ACEs with an iSCSI TransportID access identifier containing the same iSCSI name. The command shall be terminated with CHECK CONDITION status, with the sense key ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

A iSCSI TransportID with the format code set to 01b (see table 289) specifies an iSCSI initiator port based on its world wide unique initiator port identifier.

Table 289 — iSCSI initiator port TransportID format

Bit / Byte	7	6	5	4	3	2	1	0
0	FORMAT CODE (01b)		Reserved		PROTOCOL IDENTIFIER (5h)			
1	Reserved							
2	(MSB)							
3	ADDITIONAL LENGTH (m-3)							(LSB)
4	(MSB)							
n-1	ISCSI NAME							(LSB)
n	(MSB)							
n+4	SEPARATOR (2C 692C 3078h)							(LSB)
n+5	(MSB)							
m	ISCSI INITIATOR SESSION ID							(LSB)

The ADDITIONAL LENGTH field specifies the number of bytes that follow in the TransportID encompassing the ISCSI NAME, SEPARATOR, and ISCSI INITIATOR SESSION ID fields. The additional length shall be at least 20 and shall be a multiple of four.

The ISCSI NAME field shall contain the iSCSI name of an iSCSI initiator node (see RFC 3720). The ISCSI NAME field shall not be null-terminated (see 4.4.2) and shall not be padded.

The SEPARATOR field shall contain the five ASCII characters ",i,0x".

NOTE 54 - The notation used to define the SEPARATOR field is described in 3.6.1.

The null-terminated, null-padded ISCSI INITIATOR SESSION ID field shall contain the iSCSI initiator session identifier (see RFC 3720) in the form of ASCII characters that are the hexadecimal digits converted from the binary iSCSI initiator session identifier value. The first ISCSI INITIATOR SESSION ID field byte containing an ASCII null character terminates the ISCSI INITIATOR SESSION ID field without regard for the specified length of the iSCSI TransportID or the contents of the ADDITIONAL LENGTH field.

7.5.4.7 TransportID for initiator ports using SCSI over SAS Serial SCSI Protocol

A SAS Serial SCSI Protocol (SSP) TransportID (see table 290) specifies a SAS initiator port that is communicating via SSP using the SAS address belonging to that initiator port.

Table 290 — SAS Serial SCSI Protocol TransportID format

Bit / Byte	7	6	5	4	3	2	1	0
0	FORMAT CODE (00b)		Reserved		PROTOCOL IDENTIFIER (6h)			
1	Reserved							
3								
4	SAS ADDRESS							
11								
12	Reserved							
23								

The SAS ADDRESS field specifies the SAS address of the initiator port.

7.6 Vital product data parameters

7.6.1 Vital product data parameters overview and page codes

This subclause describes the vital product data (VPD) page structure and the VPD pages (see table 291) that are applicable to all SCSI devices. These VPD pages are returned by an INQUIRY command with the EVPD bit set to one (see 6.4) and contain vendor specific product information about a logical unit and SCSI target device. The vital product data may include vendor identification, product identification, unit serial numbers, device operating definitions, manufacturing data, field replaceable unit information, and other vendor specific information. This standard defines the structure of the vital product data, but not the contents.

Table 291 — Vital product data page codes

Page code	VPD Page Name	Reference	Support Requirements
01h - 7Fh	ASCII Information	7.6.2	Optional
83h	Device Identification	7.6.3	Mandatory
86h	Extended INQUIRY Data	7.6.4	Optional
85h	Management Network Addresses	7.6.5	Optional
87h	Mode Page Policy	7.6.6	Optional
81h	Obsolete	3.3.7	
82h	Obsolete	3.3.7	
88h	SCSI Ports	7.6.7	Optional
84h	Software Interface Identification	7.6.8	Optional
00h	Supported VPD Pages	7.6.9	Mandatory
80h	Unit Serial Number	7.6.10	Optional
89h - AFh	Reserved		
B0h - BFh	(See specific device type)		
C0h - FFh	Vendor specific		
Annex D contains a listing of VPD page codes in numeric order.			

7.6.2 ASCII Information VPD page

The ASCII Information VPD page (see table 292) contains information for the field replaceable unit code returned in the sense data (see 4.5).

Table 292 — ASCII Information VPD page

Bit / Byte	7	6	5	4	3	2	1	0
0	PERIPHERAL QUALIFIER			PERIPHERAL DEVICE TYPE				
1	PAGE CODE (01h - 7Fh)							
2	Reserved							
3	PAGE LENGTH (n-3)							
4	ASCII LENGTH (m-4)							
5	(MSB)			ASCII INFORMATION				
m								(LSB)
m+1	Vendor specific information							
n								

The PERIPHERAL QUALIFIER field and the PERIPHERAL DEVICE TYPE field are defined in 6.4.2.

The PAGE CODE field contains the same value as in the PAGE OR OPERATION CODE field of the INQUIRY CDB (see 6.4) and is associated with the FIELD REPLACEABLE UNIT CODE field returned in the sense data.

NOTE 55 - The FIELD REPLACEABLE UNIT CODE field in the sense data provides for 255 possible codes, while the PAGE CODE field provides for only 127 possible codes. For that reason it is not possible to return ASCII Information VPD pages for the upper code values.

The PAGE LENGTH field specifies the length of the following VPD page data. The relationship between the PAGE LENGTH field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6.

The ASCII LENGTH field specifies the length in bytes of the ASCII INFORMATION field that follows. A value of zero in this field indicates that no ASCII information is available for the specified page code. The relationship between the ASCII LENGTH field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6.

The ASCII INFORMATION field contains ASCII information concerning the field replaceable unit identified by the page code. The data in this field shall be formatted in one or more character string lines. Each line shall contain only graphic codes (i.e., code values 20h through 7Eh) and shall be terminated with a NULL (00h) character.

The contents of the vendor specific information field is not defined in this standard.

7.6.3 Device Identification VPD page

7.6.3.1 Device Identification VPD page overview

The Device Identification VPD page (see table 293) provides the means to retrieve identification descriptors applying to the logical unit. Logical units may have more than one identification descriptor (e.g., if several types or associations of identifier are supported). Device identifiers consist of one or more of the following:

a) Logical unit names;
b) SCSI target port identifiers;
c) SCSI target port names;
d) SCSI target device names;
e) Relative target port identifiers;
f) SCSI target port group number; or
g) Logical unit group number.

Identification descriptors shall be assigned to the peripheral device (e.g., a disk drive) and not to the currently mounted media, in the case of removable media devices. Operating systems are expected to use the identification

descriptors during system configuration activities to determine whether alternate paths exist for the same peripheral device.

Table 293 — Device Identification VPD page

Bit / Byte	7	6	5	4	3	2	1	0
0	PERIPHERAL QUALIFIER			PERIPHERAL DEVICE TYPE				
1	PAGE CODE (83h)							
2	(MSB)			PAGE LENGTH (n-3)				
3								(LSB)
	Identification descriptor list							
4	Identification descriptor (first)							
	⋮							
n	Identification descriptor (last)							

The PERIPHERAL QUALIFIER field and the PERIPHERAL DEVICE TYPE field in table 293 are as defined in 6.4.2.

The PAGE LENGTH field indicates the length of the identification descriptor list. The relationship between the PAGE LENGTH field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6.

Each identification descriptor (see table 294) contains information identifying the logical unit, SCSI target device containing the logical unit, or access path (i.e., target port) used by the command and returned parameter data. The Device Identification VPD page shall contain the identification descriptors enumerated in 7.6.3.2.

Table 294 — Identification descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	PROTOCOL IDENTIFIER				CODE SET			
1	PIV	Reserved	ASSOCIATION		IDENTIFIER TYPE			
2	Reserved							
3	IDENTIFIER LENGTH (n-3)							
4	IDENTIFIER							
n								

The PROTOCOL IDENTIFIER field may indicate the SCSI transport protocol to which the identification descriptor applies. If the ASSOCIATION field contains a value other than 01b (i.e., target port) or 10b (i.e., SCSI target device) or the PIV bit is set to zero, then the PROTOCOL IDENTIFIER field contents are reserved. If the ASSOCIATION field contains a value of 01b or 10b and the PIV bit is set to one, then the PROTOCOL IDENTIFIER field shall contain one of the values shown in table 261 (see 7.5.1) to indicate the SCSI transport protocol to which the identification descriptor applies.

The CODE SET field indicates the code set used for the IDENTIFIER field, as described in table 295. This field is intended to be an aid to software that displays the IDENTIFIER field.

Table 295 — CODE SET field

Code	Description
0h	Reserved
1h	The IDENTIFIER field shall contain binary values.
2h	The IDENTIFIER field shall contain ASCII printable characters (i.e., code values 20h through 7Eh)
3h	The IDENTIFIER field shall contain ISO/IEC 10646-1 (UTF-8) codes
4h - Fh	Reserved

A protocol identifier valid (PIV) bit set to zero indicates the PROTOCOL IDENTIFIER field contents are reserved. If the ASSOCIATION field contains a value of 01b or 10b, then a PIV bit set to one indicates the PROTOCOL IDENTIFIER field contains a valid protocol identifier selected from the values shown in table 261 (see 7.5.1). If the ASSOCIATION field contains a value other than 01b or 10b, then the PIV bit contents are reserved.

The ASSOCIATION field indicates the entity with which the IDENTIFIER field is associated, as described in table 296. If a logical unit returns an Identification descriptor with the ASSOCIATION field set to 00b or 10b, it shall return the same descriptor when it is accessed through any other I_T nexus.

Table 296 — ASSOCIATION field

Code	Description
00b	The IDENTIFIER field is associated with the addressed logical unit.
01b	The IDENTIFIER field is associated with the target port that received the request.
10b	The IDENTIFIER field is associated with the SCSI target device that contains the addressed logical unit.
11b	Reserved

The IDENTIFIER TYPE field (see table 297) indicates the format and assignment authority for the identifier.

Table 297 — IDENTIFIER TYPE field

Code	Description	Reference
0h	Vendor specific	7.6.3.3
1h	T10 vendor ID based	7.6.3.4
2h	EUI-64 based	7.6.3.5
3h	NAA	7.6.3.6
4h	Relative target port identifier	7.6.3.7
5h	Target port group	7.6.3.8
6h	Logical unit group	7.6.3.9
7h	MD5 logical unit identifier	7.6.3.10
8h	SCSI name string	7.6.3.11
9h - Fh	Reserved	

The IDENTIFIER LENGTH field indicates the length in bytes of the IDENTIFIER field. The relationship between the IDENTIFIER LENGTH field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6.

The IDENTIFIER field contains the identifier as described by the ASSOCIATION, IDENTIFIER TYPE, CODE SET, and IDENTIFIER LENGTH fields.

7.6.3.2 Device identification descriptor requirements

7.6.3.2.1 Identification descriptors for logical units other than well known logical units

For each logical unit that is not a well known logical unit, the Device Identification VPD page shall include at least one identification descriptor in which a logical unit name (see SAM-3) is indicated. The identification descriptor shall have the ASSOCIATION field set to 00b (i.e., logical unit) and the IDENTIFIER TYPE field set to:

a) 1h (i.e., T10 vendor ID based);
b) 2h (i.e., EUI-64-based);
c) 3h (i.e., NAA); or
d) 8h (i.e., SCSI name string).

At least one identification descriptor should have the IDENTIFIER TYPE field set to:

a) 2h (i.e., EUI-64-based);
b) 3h (i.e., NAA); or
c) 8h (i.e., SCSI name string).

In the case of virtual logical units (e.g., volume sets as defined by SCC-2), identification descriptors should contain an IDENTIFIER TYPE field set to:

a) 2h (i.e., EUI-64-based);
b) 3h (i.e., NAA); or
c) 8h (i.e., SCSI name string).

In the case of virtual logical units that have an EUI-64 based identification descriptor (see 7.6.3.5) the IDENTIFIER LENGTH field should be set to:

a) 0Ch (i.e., EUI-64-based 12-byte identifier); or
b) 10h (i.e., EUI-64-based 16-byte identifier).

In the case of virtual logical units that have an NAA identification descriptor (see 7.6.3.6) the NAA field should be set to 6h (i.e., IEEE Registered Extended).

The Device Identification VPD page shall contain the same set of identification descriptors with the ASSOCIATION field set to 00b (i.e., logical unit) regardless of the I_T nexus being used to retrieve the identification descriptors.

For logical units that are not well known logical units, the requirements for SCSI target device identification descriptors are defined in 7.6.3.2.4 and the requirements for SCSI target port identification descriptors are defined in 7.6.3.2.3.

7.6.3.2.2 Identification descriptors for well known logical units

Well known logical units shall not return any identification descriptors with the ASSOCIATION field set to 00b (i.e., logical unit).

The Device Identification VPD page shall contain the same set of identification descriptors with the ASSOCIATION field set to 10b (i.e., SCSI target device) regardless of the I_T nexus being used to retrieve the identification descriptors.

7.6.3.2.3 Identification descriptors for SCSI target ports

7.6.3.2.3.1 Relative target port identifiers

For the target port through which the Device Identification VPD page is accessed, the Device Identification VPD page should include one identification descriptor with the ASSOCIATION field set to 01b (i.e., target port) and the

IDENTIFIER TYPE field set to 4h (i.e., relative target port identifier) identifying the target port being used to retrieve the identification descriptors.

7.6.3.2.3.2 Target port names or identifiers

For the SCSI target port through which the Device Identification VPD page is accessed, the Device Identification VPD page should include one identification descriptor in which the target port name or identifier (see SAM-3) is indicated. The identification descriptor, if any, shall have the ASSOCIATION field set to 01b (i.e., target port) and the IDENTIFIER TYPE field set to:

a) 2h (i.e., EUI-64-based);
b) 3h (i.e., NAA); or
c) 8h (i.e., SCSI name string).

If the SCSI transport protocol standard (see 3.1.102) for the target port defines target port names, the identification descriptor, if any, shall contain the target port name. If the SCSI transport protocol for the target port does not define target port names, the identification descriptor, if any, shall contain the target port identifier.

7.6.3.2.4 Identification descriptors for SCSI target devices

If the SCSI target device contains a well known logical unit, the Device Identification VPD page shall have one or more identification descriptors for the SCSI target device. If the SCSI target device does not contain a well known logical unit, the Device Identification VPD page should have one or more identification descriptors for the SCSI target device.

Each SCSI target device identification descriptor, if any, shall have the ASSOCIATION field set to 10b (i.e., SCSI target device) and the IDENTIFIER TYPE field set to:

a) 2h (i.e., EUI-64-based);
b) 3h (i.e., NAA); or
c) 8h (i.e., SCSI name string).

The Device Identification VPD page shall contain identification descriptors, if any, for all the SCSI target device names for all the SCSI transport protocols supported by the SCSI target device.

7.6.3.3 Vendor specific identifier format

If the identifier type is 0h (i.e., vendor specific), no assignment authority was used and there is no guarantee that the identifier is globally unique (i.e., the identifier is vendor specific). Table 298 defines the IDENTIFIER field format.

Table 298 — Vendor specific IDENTIFIER field format

Bit / Byte	7	6	5	4	3	2	1	0
0	VENDOR SPECIFIC IDENTIFIER							
n								

7.6.3.4 T10 vendor ID based format

If the identifier type is 1h (i.e., T10 vendor ID based), the IDENTIFIER field has the format shown in table 299.

Table 299 — T10 vendor ID based IDENTIFIER field format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
7				T10 VENDOR IDENTIFICATION				(LSB)
8								
n				VENDOR SPECIFIC IDENTIFIER				

The T10 VENDOR IDENTIFICATION field contains eight bytes of left-aligned ASCII data (see 4.4.1) identifying the vendor of the product. The data shall be left aligned within this field.The T10 vendor identification shall be one assigned by INCITS. A list of assigned T10 vendor identifications is in Annex E and on the T10 web site (http://www.T10.org).

NOTE 56 - The T10 web site (http://www.t10.org) provides a convenient means to request an identification code.

The organization associated with the T10 vendor identification is responsible for ensuring that the VENDOR SPECIFIC IDENTIFIER field is unique in a way that makes the entire IDENTIFIER field unique. A recommended method of constructing a unique IDENTIFIER field is to concatenate the PRODUCT IDENTIFICATION field from the standard INQUIRY data (see 6.4.2) and the PRODUCT SERIAL NUMBER field from the Unit Serial Number VPD page (see 7.6.10).

7.6.3.5 EUI-64 based identifier format

7.6.3.5.1 EUI-64 based identifier format overview

If the identifier type is 2h (i.e., EUI-64 based identifier), the IDENTIFIER LENGTH field (see table 300) indicates the format of the identification descriptor.

Table 300 — EUI-64 based identifier lengths

Identifier Length	Description	Reference
08h	EUI-64 identifier	7.6.3.5.2
0Ch	EUI-64 based 12-byte identifier	7.6.3.5.3
10h	EUI-64 based 16-byte identifier	7.6.3.5.4
All other values	Reserved	

7.6.3.5.2 EUI-64 identifier format

If the identifier type is 2h (i.e., EUI-64 based identifier) and the IDENTIFIER LENGTH field is set to 08h, the IDENTIFIER field has the format shown in table 301. The CODE SET field shall be set to 1h (i.e., binary).

Table 301 — EUI-64 IDENTIFIER field format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)			IEEE COMPANY_ID				
2								(LSB)
3	(MSB)			VENDOR SPECIFIC EXTENSION IDENTIFIER				
7								(LSB)

The IEEE COMPANY_ID field contains a 24 bit OUI (see 3.1.74) assigned by the IEEE.

The VENDOR SPECIFIC EXTENSION IDENTIFIER field contains a 40 bit numeric value that is uniquely assigned by the organization associated with the IEEE company_id as required by the IEEE definition of EUI-64 (see 3.1.37).

7.6.3.5.3 EUI-64 based 12-byte identifier format

If the identifier type is 2h (i.e., EUI-64 based identifier) and the IDENTIFIER LENGTH field is set to 0Ch, the IDENTIFIER field has the format shown in table 302. The CODE SET field shall be set to 1h (i.e., binary).

Table 302 — EUI-64 based 12-byte IDENTIFIER field format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)			IEEE COMPANY_ID				
2								(LSB)
3	(MSB)			VENDOR SPECIFIC EXTENSION IDENTIFIER				
7								(LSB)
8	(MSB)			DIRECTORY ID				
11								(LSB)

The IEEE COMPANY_ID field and VENDOR SPECIFIC EXTENSION IDENTIFIER field are defined in 7.6.3.5.2.

The DIRECTORY ID field contains a directory identifier, as specified by ISO/IEC 13213:1994.

NOTE 57 - The EUI-64 based 12 byte format may be used to report IEEE 1394 target port identifiers (see SBP-3).

7.6.3.5.4 EUI-64 based 16-byte identifier format

If the identifier type is 2h (i.e., EUI-64 based identifier) and the IDENTIFIER LENGTH field is set to 10h, the IDENTIFIER field has the format shown in table 303. The CODE SET field shall be set to 1h (i.e., binary).

Table 303 — EUI-64 based 16-byte IDENTIFIER field format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)			IDENTIFIER EXTENSION				
7								(LSB)
8	(MSB)			IEEE COMPANY_ID				
10								(LSB)
11	(MSB)			VENDOR SPECIFIC EXTENSION IDENTIFIER				
15								(LSB)

The IDENTIFIER EXTENSION field contains a 64 bit numeric value.

The IEEE COMPANY_ID field and VENDOR SPECIFIC EXTENSION IDENTIFIER field are defined in 7.6.3.5.2.

NOTE 58 - The EUI-64 based 16-byte format may be used to report SCSI over RDMA target port identifiers (see SRP).

7.6.3.6 NAA identifier format

7.6.3.6.1 NAA identifier basic format

If the identifier type is 3h (i.e., NAA identifier), the IDENTIFIER field has the format shown in table 304. This format is compatible with the Name_Identifier format defined in FC-FS.

Table 304 — NAA IDENTIFIER field format

Bit / Byte	7	6	5	4	3	2	1	0
0	NAA							
1	NAA specific data							
n								

The Name Address Authority (NAA) field (see table 305) defines the format of the NAA specific data in the identifier.

Table 305 — Name Address Authority (NAA) field

Code	Description	Reference
2h	IEEE Extended	7.6.3.6.2
5h	IEEE Registered	7.6.3.6.3
6h	IEEE Registered Extended	7.6.3.6.4
0h - 1h	Reserved	
3h - 4h	Reserved	
7h - Fh	Reserved	

7.6.3.6.2 NAA IEEE Extended identifier format

If NAA is 2h (i.e., IEEE Extended), the eight byte fixed length IDENTIFIER field shall have the format shown in table 306. The CODE SET field shall be set to 1h (i.e., binary) and the IDENTIFIER LENGTH field shall be set to 08h.

Table 306 — NAA IEEE Extended IDENTIFIER field format

Bit / Byte	7	6	5	4	3	2	1	0
0	NAA (2h)				(MSB)			
1	VENDOR SPECIFIC IDENTIFIER A							(LSB)
2	(MSB)							
4	IEEE COMPANY_ID							(LSB)
5	(MSB)							
7	VENDOR SPECIFIC IDENTIFIER B							(LSB)

The IEEE COMPANY_ID field contains a 24 bit canonical form OUI (see 3.1.74) assigned by the IEEE.

The VENDOR SPECIFIC IDENTIFIER A contains a 12 bit numeric value that is uniquely assigned by the organization associated with the IEEE company_id.

The VENDOR SPECIFIC IDENTIFIER B contains a 24 bit numeric value that is uniquely assigned by the organization associated with the IEEE company_id.

NOTE 59 - The EUI-64 identifier format includes a 40 bit vendor specific identifier. The IEEE Extended identifier format includes 36 bits vendor specific identifier in two fields.

7.6.3.6.3 NAA IEEE Registered identifier format

If NAA is 5h (i.e., IEEE Registered), the eight byte fixed length IDENTIFIER field shall have the format shown in table 307. The CODE SET field shall be set to 1h (i.e., binary) and the IDENTIFIER LENGTH field shall be set to 08h.

Table 307 — NAA IEEE Registered IDENTIFIER field format

Bit / Byte	7	6	5	4	3	2	1	0
0	NAA (5h)				(MSB)			
1	IEEE COMPANY_ID							
2								
3				(LSB)	(MSB)			
4	VENDOR SPECIFIC IDENTIFIER							
7								(LSB)

The IEEE COMPANY_ID field contains a 24 bit canonical form OUI (see 3.1.74) assigned by the IEEE.

The VENDOR SPECIFIC IDENTIFIER a 36 bit numeric value that is uniquely assigned by the organization associated with the IEEE company_id.

NOTE 60 - The EUI-64 identifier format includes a 40 bit vendor specific identifier. The IEEE Registered identifier format includes a 36 bit vendor specific identifier.

7.6.3.6.4 NAA IEEE Registered Extended identifier format

If NAA is 6h (i.e., IEEE Registered Extended), the sixteen byte fixed length IDENTIFIER field shall have the format shown in table 308. The CODE SET field shall be set to 1h (i.e., binary) and the IDENTIFIER LENGTH field shall be set to 10h.

Table 308 — NAA IEEE Registered Extended IDENTIFIER field format

Bit / Byte	7	6	5	4	3	2	1	0
0	NAA (6h)				(MSB)			
1	IEEE COMPANY_ID							
2								
3				(LSB)	(MSB)			
4	VENDOR SPECIFIC IDENTIFIER							
7								(LSB)
8	(MSB)							
15	VENDOR SPECIFIC IDENTIFIER EXTENSION							(LSB)

The IEEE COMPANY_ID field contains a 24 bit canonical form OUI (see 3.1.74) assigned by the IEEE.

The VENDOR SPECIFIC IDENTIFIER a 36 bit numeric value that is uniquely assigned by the organization associated with the IEEE company_id.

NOTE 61 - The EUI-64 identifier format includes a 40 bit vendor specific identifier. The IEEE Registered Extended identifier format includes a 36 bit vendor specific identifier.

The VENDOR SPECIFIC IDENTIFIER EXTENSION a 64 bit numeric value that is assigned to make the IDENTIFIER field unique.

7.6.3.7 Relative target port identifier format

If the identifier type is 4h (i.e., relative target port identifier) and the ASSOCIATION field contains 01b (i.e., target port), then the IDENTIFIER field shall have the format shown in table 309. The CODE SET field shall be set to 1h (i.e., binary) and the IDENTIFIER LENGTH field shall be set to 04h. If the ASSOCIATION field does not contain 01b, use of this identifier type is reserved.

Table 309 — Relative target port IDENTIFIER field format

Bit / Byte	7	6	5	4	3	2	1	0
0	Obsolete							
1								
2	(MSB)							
3	RELATIVE TARGET PORT IDENTIFIER							(LSB)

The RELATIVE TARGET PORT IDENTIFIER field (see table 310) contains the relative port identifier (see 3.1.88) of the target port on which the INQUIRY command was received.

Table 310 — RELATIVE TARGET PORT IDENTIFIER field

Code	Description
0h	Reserved
1h	Relative port 1, historically known as port A
2h	Relative port 2, historically known as port B
3h - FFFFh	Relative port 3 through 65 535

7.6.3.8 Target port group identifier format

If the identifier type is 5h (i.e., target port group) and the ASSOCIATION value is 01b (i.e., target port), the four byte fixed length IDENTIFIER field shall have the format shown in table 311. The CODE SET field shall be set to 1h (i.e., binary) and the IDENTIFIER LENGTH field shall be set to 04h. If the ASSOCIATION field does not contain 01b, use of this identifier type is reserved.

Table 311 — Target port group IDENTIFIER field format

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved							
1								
2	(MSB)	TARGET PORT GROUP						
3								(LSB)

The TARGET PORT GROUP field indicates the target port group to which the target port is a member (see 5.8).

7.6.3.9 Logical unit group identifier format

A logical unit group is a group of logical units that share the same target port group (see 5.8) definitions. The target port groups maintain the same target port group asymmetric access states for all logical units in the same logical unit group. A logical unit shall be in no more than one logical unit group.

If the identifier type is 6h (i.e., logical unit group) and the ASSOCIATION value is 00b (i.e., logical unit), the four byte fixed length IDENTIFIER field shall have the format shown in table 312. The CODE SET field shall be set to 1h (i.e., binary) and the IDENTIFIER LENGTH field shall be set to 04h. If the ASSOCIATION field does not contain 00b, use of this identifier type is reserved.

Table 312 — Logical unit group IDENTIFIER field format

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved							
1								
2	(MSB)	LOGICAL UNIT GROUP						
3								(LSB)

The LOGICAL UNIT GROUP field indicates the logical unit group to which the logical unit is a member.

7.6.3.10 MD5 logical unit identifier format

If the identifier type is 7h (i.e., MD5 logical unit identifier) and the ASSOCIATION value is 00b (i.e., logical unit), the IDENTIFIER field has the format shown in table 313. The CODE SET field shall be set to 1h (i.e., binary). The MD5 logical unit identifier shall not be used if a logical unit provides unique identification using identifier types 2h (i.e., EUI-64 based identifier), 3h (i.e., NAA identifier), or 8h (i.e., SCSI name string). A bridge device may return a MD5 logical unit identifier type for that logical unit that does not support the Device Identification VPD page (see 7.6.3).

If the ASSOCIATION field does not contain 00b, use of this identifier type is reserved.

Table 313 — MD5 logical unit IDENTIFIER field format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
	MD5 LOGICAL UNIT IDENTIFIER							
15								(LSB)

The MD5 LOGICAL UNIT IDENTIFIER field contains the message digest of the supplied message input. The message digest shall be generated using the MD5 message-digest algorithm as specified in RFC 1321 (see 2.3) with the following information as message input:

1) The contents of the T10 VENDOR IDENTIFICATION field in the standard INQUIRY data (see 6.4.2);
2) The contents of the PRODUCT IDENTIFICATION field in the standard INQUIRY data;
3) The contents of the PRODUCT SERIAL NUMBER field in the Unit Serial Number VPD page (see 7.6.10);
4) The contents of a vendor specific IDENTIFIER field (type 0h) from the Device Identification VPD page; and
5) The contents of a T10 vendor ID based IDENTIFIER field (type 1h) from the Device Identification VPD page.

If a field or page is not available, the message input for that field or page shall be 8 bytes of ASCII space characters (i.e., 20h).

The uniqueness of the MD5 logical unit identifier is dependent upon the relative degree of randomness (i.e., the entropy) of the message input. If it is found that two or more logical units have the same MD5 logical unit identifier, the application client should determine in a vendor specific manner whether the logical units are the same entities.

The MD5 logical unit identifier example described in this paragraph and shown in table 314 and table 315 is not a normative part of this standard. The data available for input to the MD5 algorithm for this example is shown in table 314.

Table 314 — MD5 logical unit identifier example available data

MD5 message input	Available	Contents
T10 VENDOR IDENTIFICATION field	Yes	T10
PRODUCT IDENTIFICATION field	Yes	MD5 Logical Unit
PRODUCT SERIAL NUMBER field	Yes	01234567
vendor specific IDENTIFIER field	No	
T10 vendor ID based IDENTIFIER field	No	

The concatenation of the fields in table 314 to form input to the MD5 algorithm is shown in table 315.

Table 315 — Example MD5 input for computation of a logical unit identifier

Bytes	Hexadecimal values				ASCII values
00 – 15	54 31 30 20	20 20 20 20	4D 44 35 20	4C 6F 67 69	T10 MD5 Logi
16 – 31	63 61 6C 20	55 6E 69 74	30 31 32 33	34 35 36 37	cal Unit01234567
32 – 47	20 20 20 20	20 20 20 20	20 20 20 20	20 20 20 20	
NOTE Non-printing ASCII characters are shown as '.'.					

Based on the example inputs shown in table 314 and the concatenation of the inputs shown in table 315, the MD5 base 16 algorithm described in RFC 1321 produces the value 8FAC A22A 0AC0 3839 1255 25F2 0EFE 2E7Eh.

7.6.3.11 SCSI name string identifier format

If the identifier type is 8h (i.e., SCSI name string), the IDENTIFIER field has the format shown in table 316. The CODE SET field shall be set to 3h (i.e., UTF-8).

Table 316 — SCSI name string IDENTIFIER field format

Bit / Byte	7	6	5	4	3	2	1	0
0	SCSI NAME STRING							
n								

The null-terminated, null-padded (see 4.4.2) SCSI NAME STRING field contains a UTF-8 format string. The number of bytes in the SCSI NAME STRING field (i.e., the value in the IDENTIFIER LENGTH field) shall be no larger than 256 and shall be a multiple of four.

The SCSI NAME STRING field starts with either:

a) The four UTF-8 characters "eui." concatenated with 16, 24, or 32 hexadecimal digits (i.e., the UTF-8 characters 0 through 9 and A through F) for an EUI-64 based identifier (see 7.6.3.5). The first hexadecimal digit shall be the most significant four bits of the first byte (i.e., most significant byte) of the EUI-64 based identifier;
b) The four UTF-8 characters "naa." concatenated with 16 or 32 hexadecimal digits for an NAA identifier (see 7.6.3.6). The first hexadecimal digit shall be the most significant four bits of the first byte (i.e., most significant byte) of the NAA identifier; or
c) The four UTF-8 characters "iqn." concatenated with an iSCSI name for an iSCSI-name based identifier (see iSCSI).

If the ASSOCIATION field is set to 00b (i.e., logical unit) and the SCSI NAME STRING field starts with the four UTF-8 characters "iqn.", the SCSI NAME STRING field ends with the five UTF-8 characters ",L,0x" concatenated with 16 hexadecimal digits for the logical unit name extension. The logical unit name extension is a UTF-8 string containing no more than 16 hexadecimal digits. The logical unit name extension is assigned by the SCSI target device vendor and shall be assigned so the logical unit name is worldwide unique.

If the ASSOCIATION field is set to 01b (i.e., target port), the SCSI NAME STRING field ends with the five UTF-8 characters ",t,0x" concatenated with two or more hexadecimal digits as specified in the applicable SCSI transport protocol standard (see 3.1.102).

If the ASSOCIATION field is set to 10b (i.e., SCSI target device), the SCSI NAME STRING field has no additional characters.

NOTE 62 - The notation used in this subclause to specify exact UTF-8 character strings is described in 3.6.1.

7.6.4 Extended INQUIRY Data VPD page

The Extended INQUIRY Data VPD page (see table 317) provides the application client with a means to obtain information about the logical unit.

Table 317 — Extended INQUIRY Data VPD page

Bit / Byte	7	6	5	4	3	2	1	0
0	PERIPHERAL QUALIFIER			PERIPHERAL DEVICE TYPE				
1	PAGE CODE (86h)							
2	Reserved							
3	PAGE LENGTH (3Ch)							
4	Reserved				RTO	GRD_CHK	APP_CHK	REF_CHK
5	Reserved			GROUP_SUP	PRIOR_SUP	HEADSUP	ORDSUP	SIMPSUP
6	Reserved						NV_SUP	V_SUP
7	Reserved							
63								

The PERIPHERAL QUALIFIER field and the PERIPHERAL DEVICE TYPE field are as defined in 6.4.2.

The PAGE LENGTH field specifies the length of the following VPD page data and shall be set to 60. The relationship between the PAGE LENGTH field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6.

A reference tag ownership (RTO) bit set to zero indicates that the logical unit does not support application client ownership of the LOGICAL BLOCK REFERENCE TAG field in the protection information (see SBC-2), if any. A RTO bit set to one indicates that the logical unit supports application client ownership of the LOGICAL BLOCK REFERENCE TAG field.

A guard check (GRD_CHK) bit set to zero indicates that the device server does not check the LOGICAL BLOCK GUARD field in the protection information (see SBC-2), if any. A GRD_CHK bit set to one indicates that the device server checks the LOGICAL BLOCK GUARD field in the protection information, if any.

An application tag check (APP_CHK) bit set to zero indicates that the device server does not check the LOGICAL BLOCK APPLICATION TAG field in the protection information (see SBC-2), if any. An APP_CHK bit set to one indicates that the device server checks the LOGICAL BLOCK APPLICATION TAG field in the protection information, if any.

A reference tag check (REF_CHK) bit set to zero indicates that the device server does not check the LOGICAL BLOCK REFERENCE TAG field in the protection information (see SBC-2), if any. A REF_CHK bit set to one indicates that the device server checks the LOGICAL BLOCK REFERENCE TAG field in the protection information, if any.

A grouping function supported (GROUP_SUP) bit set to one indicates that the grouping function (see SBC-2) is supported by the device server. A GROUP_SUP bit set to zero indicates that the grouping function is not supported.

A priority supported (PRIOR_SUP) bit set to one indicates that task priority (see SAM-3) is supported by the logical unit. A PRIOR_SUP bit set to zero indicates that task priority is not supported.

A head of queue supported (HEADSUP) bit set to one indicates that the HEAD OF QUEUE task attribute (see SAM-3) is supported by the logical unit. A HEADSUP bit set to zero indicates that the HEAD OF QUEUE task attribute is not supported. If the HEADSUP bit is set to zero, application clients should not specify the HEAD OF QUEUE task attribute as an Execute Command (see 4.2) procedure call argument.

An ordered supported (ORDSUP) bit set to one indicates that the ORDERED task attribute (see SAM-3) is supported by the logical unit. An ORDSUP bit set to zero indicates that the ORDERED task attribute is not supported. If the

ORDSUP bit is set to zero, application clients should not specify the ORDERED task attribute as an Execute Command procedure call argument.

A simple supported (SIMPSUP) bit set to one indicates that the SIMPLE task attribute (see SAM-3) is supported by the logical unit. Logical units that support the full task management model (see SAM-3) shall set the SIMPSUP bit to one. A SIMPSUP bit set to zero indicates that the SIMPLE task attribute is not supported. If the SIMPSUP bit is set to zero, application clients should not specify the SIMPLE task attribute as an Execute Command procedure call argument.

SAM-3 defines how unsupported task attributes are processed.

An NV_SUP bit set to one indicates that the device server supports a non-volatile cache (see 3.1.70) and that the applicable command standard (see 3.1.18) defines features using this cache (e.g., the FUA_NV bit in SBC-2). An NV_SUP bit set to zero indicates that the device server may or may not support a non-volatile cache.

A V_SUP bit set to one indicates that the device server supports a volatile cache (see 3.1.126) and that the applicable command standard (see 3.1.18) defines features using this cache (e.g., the FUA bit in SBC-2). An V_SUP bit set to zero indicates that the device server may or may not support a volatile cache.

7.6.5 Management Network Addresses VPD page

The Management Network Addresses VPD page (see table 318) provides a list of network addresses of management services associated with a SCSI target device, target port, or logical unit.

Table 318 — Management Network Addresses VPD page

Bit / Byte	7	6	5	4	3	2	1	0
0	PERIPHERAL QUALIFIER			PERIPHERAL DEVICE TYPE				
1	PAGE CODE (85h)							
2	(MSB)			PAGE LENGTH (n-3)				
3								(LSB)
	Network services descriptor list							
4				Network services descriptor (first)				
	⋮							
n	Network services descriptor (last)							

The PERIPHERAL QUALIFIER field and the PERIPHERAL DEVICE TYPE field are as defined in 6.4.2.

The PAGE LENGTH field specifies the length of the network services descriptor list. The relationship between the PAGE LENGTH field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6.

Each network service descriptor (see table 319) contains information about one management service.

Table 319 — Network service descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved	ASSOCIATION		SERVICE TYPE				
1	Reserved							
2	(MSB)	NETWORK ADDRESS LENGTH (n-3)						
3								(LSB)
4	NETWORK ADDRESS							
n								

The ASSOCIATION field (see table 296 in 7.6.3.1) specifies the entity (i.e., SCSI target device, target port, or logical unit) with which the service is associated.

The SERVICE TYPE field (see table 320) allows differentiation of multiple services with the same protocol running at different port numbers or paths.

NOTE 63 - A SCSI target device may provide separate HTTP services for configuration and diagnostics. One of these services may use the standard HTTP port 80 (see 3.1.119) and the other service may use a different port (e.g., 8080).

Table 320 — Network services type

Type	Description
00h	Unspecified
01h	Storage Configuration Service
02h	Diagnostics
03h	Status
04h	Logging
05h	Code Download
06h - 1Fh	Reserved

The NETWORK ADDRESS LENGTH field contains the length in bytes of the NETWORK ADDRESS field. The network address length shall be a multiple of four.

The null-terminated, null-padded NETWORK ADDRESS field contains the URL form of a URI as defined in RFC 2396.

7.6.6 Mode Page Policy VPD page

The Mode Page Policy VPD page (see table 321) indicates which mode page policy (see 6.7) is in effect for each mode page supported by the logical unit.

Table 321 — Mode Page Policy VPD page

Bit / Byte	7	6	5	4	3	2	1	0
0	PERIPHERAL QUALIFIER			PERIPHERAL DEVICE TYPE				
1	PAGE CODE (87h)							
2	(MSB)			PAGE LENGTH (n-3)				
3								(LSB)
	Mode page policy descriptor list							
4	Mode page policy descriptor (first)							
7								
	⋮							
n-3	Mode page policy descriptor (last)							
n								

The PERIPHERAL QUALIFIER field and the PERIPHERAL DEVICE TYPE field are as defined in 6.4.2.

The PAGE LENGTH field specifies the length of the mode page policy descriptor list. The relationship between the PAGE LENGTH field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6.

Each mode page policy descriptor (see table 322) contains information describing the mode page policy for one or more mode pages or subpages (see 7.4.5). The information in the mode page policy descriptors in this VPD page shall describe the mode page policy for every mode page and subpage supported by the logical unit.

Table 322 — Mode page policy descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved		POLICY PAGE CODE					
1	POLICY SUBPAGE CODE							
2	MLUS	Reserved						MODE PAGE POLICY
3	Reserved							

The POLICY PAGE CODE field and POLICY SUBPAGE CODE field indicate the mode page and subpage to which the descriptor applies.

If the first mode page policy descriptor in the list contains a POLICY PAGE CODE field set to 3Fh and a POLICY SUBPAGE CODE field set to FFh, then the descriptor applies to all mode pages and subpages not described by other mode page policy descriptors. The POLICY PAGE CODE field shall be set to 3Fh and the POLICY SUBPAGE CODE field shall be set to FFh only in the first mode page policy descriptor in the list.

If the POLICY PAGE CODE field contains a value other than 3Fh and a POLICY SUBPAGE CODE field contains a value other than FFh, then the POLICY PAGE CODE field and the POLICY SUBPAGE CODE field indicate a single mode page and subpage to which the descriptor applies.

If the POLICY PAGE CODE field contains a value other than 3Fh, then POLICY SUBPAGE CODE field shall contain a value other than FFh. If the POLICY SUBPAGE CODE field contains a value other than FFh, then POLICY PAGE CODE field shall contain a value other than 3Fh.

If the SCSI target device has more than one logical unit, a multiple logical units share (MLUS) bit set to one indicates the mode page and subpage identified by the POLICY PAGE CODE field and POLICY SUBPAGE CODE field is shared by more than one logical unit. A MLUS bit set to zero indicates the logical unit maintains its own copy of the mode page and subpage identified by the POLICY PAGE CODE field and POLICY SUBPAGE CODE field.

The MLUS bit is set to one in the mode page policy descriptors or descriptor that indicates the mode page policy for the:

a) Disconnect-Reconnect mode page (see 7.4.8); and
b) Protocol Specific Logical Unit mode page (see 7.4.13).

The MODE PAGE POLICY field (see table 323) indicates the mode page policy for the mode page and subpage identified by the POLICY PAGE CODE field and POLICY SUBPAGE CODE field. The mode page policies are described in table 94 (see 6.7).

Table 323 — MODE PAGE POLICY field

Code	Description
00b	Shared
01b	Per target port
10b	Per initiator port
11b	Per I_T nexus

7.6.7 SCSI Ports VPD page

The SCSI Ports VPD page (see table 324) provides a means to retrieve identification descriptors for all the SCSI ports in a SCSI target device or SCSI target/initiator device.

Table 324 — SCSI Ports VPD page

Bit / Byte	7	6	5	4	3	2	1	0
0	PERIPHERAL QUALIFIER			PERIPHERAL DEVICE TYPE				
1	PAGE CODE (88h)							
2	(MSB)			PAGE LENGTH (n-3)				
3								(LSB)
	Identification descriptor list							
4	First SCSI port identification descriptor (see table 325)							
	⋮							
n	Last SCSI port identification descriptor (see table 325)							

The SCSI Ports VPD page only reports information on SCSI ports known to the device server processing the INQUIRY command. The REPORT LUNS well-known logical unit (see 8.2) may be used to return information on all SCSI ports in the SCSI device (i.e., all target ports and all initiator ports).

If the device server detects that a SCSI port is added or removed from the SCSI device and the SCSI port identification descriptor list changes, it shall establish a unit attention condition (see SAM-3), with the additional sense code set to INQUIRY DATA HAS CHANGED.

The PERIPHERAL QUALIFIER field and the PERIPHERAL DEVICE TYPE field are as defined in 6.4.2.

The PAGE LENGTH field specifies the length of the SCSI port identification descriptor list. The relationship between the PAGE LENGTH field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6.

Each SCSI Port identification descriptor (see table 325) identifies a SCSI port. The SCSI Port identification descriptors may be returned in any order.

Table 325 — SCSI port identification descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved							
1								
2	(MSB)	RELATIVE PORT IDENTIFIER						
3								(LSB)
4	Reserved							
5								
6	(MSB)	INITIATOR PORT TRANSPORTID LENGTH (k - 7)						
7								(LSB)
8	INITIATOR PORT TRANSPORTID, if any							
k								
k+1	Reserved							
k+2								
k+3	(MSB)	TARGET PORT DESCRIPTORS LENGTH (n - (k+4))						
k+4								(LSB)
	Target port descriptor list							
k+5	First target port descriptor (see table 327)							
	⋮							
n	Last target port descriptor (see table 327)							

The RELATIVE PORT IDENTIFIER field (see table 326) contains the relative port identifier (see 3.1.88) of the SCSI port to which the SCSI port identification descriptor applies.

Table 326 — RELATIVE PORT IDENTIFIER field

Code	Description
0h	Reserved
1h	Relative port 1, historically known as port A
2h	Relative port 2, historically known as port B
3h - FFFFh	Relative port 3 through 65 535

The INITIATOR PORT TRANSPORTID LENGTH field contains the length of the INITIATOR PORT TRANSPORTID field. An INITIATOR PORT TRANSPORTID LENGTH field set to zero indicates no INITIATOR PORT TRANSPORTID field is present (i.e., the SCSI port is not an initiator port and not a target/initiator port).

If the INITIATOR PORT TRANSPORTID LENGTH field contains a non-zero value, the INITIATOR PORT TRANSPORTID field contains a TransportID identifying the initiator port as specified in 7.5.4.

The TARGET PORT DESCRIPTORS LENGTH field contains the length of the target port descriptors, if any. A TARGET PORT DESCRIPTORS LENGTH field set to zero indicates no target port descriptors are present (i.e., the SCSI port is not a target port and not a target/initiator port).

Each target port descriptor (see table 327) contains an identifier for the target port. The target port descriptors may be returned in any order.

Table 327 — Target port descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	PROTOCOL IDENTIFIER				CODE SET			
1	PIV (1b)	Reserved		ASSOCIATION (01b)	IDENTIFIER TYPE			
2	Reserved							
3	IDENTIFIER LENGTH (n-3)							
4	IDENTIFIER							
n								

The PROTOCOL IDENTIFIER field indicates the SCSI transport protocol to which the identification descriptor applies as described in 7.6.3.1.

The CODE SET field, PIV field, ASSOCIATION field, IDENTIFIER TYPE field, IDENTIFIER LENGTH field, and IDENTIFIER field are as defined in the Device Identification VPD page identification descriptor (see 7.6.3.1), with the following additional requirements:

a) The PIV bit shall be set to one (i.e., the PROTOCOL IDENTIFIER field always contains a SCSI transport protocol identifier); and
b) The ASSOCIATION field shall be set to 01b (i.e., the descriptor always identifies a target port).

7.6.8 Software Interface Identification VPD page

The Software Interface Identification VPD page (see table 328) provides identification of software interfaces applicable to the logical unit. Logical units may have more than one associated software interface identifier.

NOTE 64 - Application clients may use the software IDs to differentiate peripheral device function in cases where the command set (e.g., processor devices) is too generic to distinguish different software interfaces implemented.

Table 328 — Software Interface Identification VPD page

Bit / Byte	7	6	5	4	3	2	1	0
0	PERIPHERAL QUALIFIER			PERIPHERAL DEVICE TYPE				
1	PAGE CODE (84h)							
2	Reserved							
3	PAGE LENGTH (n-3)							
	Software interface identifier list							
4	Software interface identifier (first)							
	⋮							
n	Software interface identifier (last)							

The PERIPHERAL QUALIFIER field and the PERIPHERAL DEVICE TYPE field are defined in 6.4.2.

The PAGE LENGTH field specifies the length of the software interface identifier list. The relationship between the PAGE LENGTH field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6.

Each software interface identifier (see table 329) is a six-byte, fixed-length field that contains information identifying a software interface implemented by the logical unit. The contents of software interface identifier are in EUI-48 format (see 3.1.36).

Table 329 — Software interface identifier format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
2	IEEE COMPANY_ID							(LSB)
3	(MSB)							
5	VENDOR SPECIFIC EXTENSION IDENTIFIER							(LSB)

The IEEE COMPANY_ID field contains a 24 bit OUI (see 3.1.74) assigned by the IEEE.

The VENDOR SPECIFIC EXTENSION IDENTIFIER a 24 bit numeric value that is uniquely assigned by the organization associated with the OUI as required by the IEEE definition of EUI-48 (see 3.1.36). The combination of OUI and vendor specific extension identifier shall uniquely identify the document or documents that specify the supported software interface.

7.6.9 Supported VPD pages

This VPD page contains a list of the VPD page codes supported by the logical unit (see table 330). If a device server supports any VPD pages, it also shall support this VPD page.

Table 330 — Supported VPD pages

Bit / Byte	7	6	5	4	3	2	1	0
0	PERIPHERAL QUALIFIER			PERIPHERAL DEVICE TYPE				
1	PAGE CODE (00h)							
2	Reserved							
3	PAGE LENGTH (n-3)							
4	Supported VPD page list							
n								

The PERIPHERAL QUALIFIER field and the PERIPHERAL DEVICE TYPE field are defined in 6.4.2.

The PAGE LENGTH field specifies the length of the supported VPD page list. The relationship between the PAGE LENGTH field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6.

The supported VPD page list shall contain a list of all VPD page codes (see 7.6) implemented by the logical unit in ascending order beginning with page code 00h.

7.6.10 Unit Serial Number VPD page

This VPD page (see table 331) provides a product serial number for the SCSI target device or logical unit.

Table 331 — Unit Serial Number VPD page

Bit / Byte	7	6	5	4	3	2	1	0
0	PERIPHERAL QUALIFIER			PERIPHERAL DEVICE TYPE				
1	PAGE CODE (80h)							
2	Reserved							
3	PAGE LENGTH (n-3)							
4	(MSB)	PRODUCT SERIAL NUMBER						
n								(LSB)

The PERIPHERAL QUALIFIER field and the PERIPHERAL DEVICE TYPE field are defined in 6.4.2.

The PAGE LENGTH field specifies the length of the product serial number. The relationship between the PAGE LENGTH field and the CDB ALLOCATION LENGTH field is defined in 4.3.4.6.

The PRODUCT SERIAL NUMBER field contains right-aligned ASCII data (see 4.4.1) that is vendor-assigned serial number. If the product serial number is not available, the device server shall return ASCII spaces (20h) in this field.

8 Well known logical units

8.1 Model for well known logical units

Well known logical units are addressed using the well known logical unit addressing method of extended logical unit addressing (see SAM-3). Each well known logical unit has a well known logical unit number (W-LUN) as shown in table 332.

Table 332 — Well known logical unit numbers

W-LUN	Description	Reference
00h	Reserved	
01h	REPORT LUNS well known logical unit	8.2
02h	ACCESS CONTROLS well known logical unit	8.3
03h	TARGET LOG PAGES well known logical unit	8.4
04h-FFh	Reserved	

If a well known logical unit is supported within a SCSI target device, then that logical unit shall support all the commands defined for it.

Access to well known logical units shall not be affected by access controls.

The SCSI target device name of the well known logical unit may be determined by issuing an INQUIRY command (see 6.4) requesting the Device Identification VPD page (see 7.6.3).

All well known logical units shall support the INQUIRY command's Device Identification VPD page as specified in 7.6.3.2.2.

8.2 REPORT LUNS well known logical unit

The REPORT LUNS well known logical unit shall only process the commands listed in table 333. If a command is received by the REPORT LUNS well know logical unit that is not listed in table 333, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID COMMAND OPERATION CODE.

Table 333 — Commands for the REPORT LUNS well known logical unit

Command name	Operation code	Type	Reference
INQUIRY	12h	M	6.4
REPORT LUNS	A0h	M	6.21
REQUEST SENSE	03h	M	6.27
TEST UNIT READY	00h	M	6.33
Key: M = Command implementation is mandatory.			

8.3 ACCESS CONTROLS well known logical unit

8.3.1 Access controls model

8.3.1.1 Access controls commands

The ACCESS CONTROLS well known logical unit shall only process the commands listed in table 334. If a command is received by the ACCESS CONTROLS well know logical unit that is not listed in table 334, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID COMMAND OPERATION CODE.

Table 334 — Commands for the ACCESS CONTROLS well known logical unit

Command name	Operation code	Type	Reference
ACCESS CONTROL IN	86h	M	8.3.2
ACCESS CONTROL OUT	87h	M	8.3.3
INQUIRY	12h	M	6.4
REQUEST SENSE	03h	M	6.27
TEST UNIT READY	00h	M	6.33
Key: M = Command implementation is mandatory.			

8.3.1.2 Access controls overview

Access controls are a SCSI target device feature that application clients may use to restrict logical unit access to specified initiator ports or groups of initiator ports.

Access controls shall not allow restrictions to be placed on access to well known logical units. Access controls shall not cause new well known logical units to be defined.

Access controls are handled in the SCSI target device by an access controls coordinator located at the ACCESS CONTROLS well known logical unit. The access controls coordinator also may be accessible via LUN 0. The access controls coordinator associates a specific LUN to a specific logical unit depending on which initiator port accesses the SCSI target device and whether the initiator port has access rights to the logical unit.

Access rights to a logical unit affects whether the logical unit appears in the parameter data returned by a REPORT LUNS command and how the logical unit responds to INQUIRY commands.

The access controls coordinator maintains the ACL as described in 8.3.1.3 to supply information about:

a) Which initiator ports are allowed access to which logical units; and
b) Which LUN value is used by a specific initiator port when accessing a specific logical unit.

The format of the ACL is vendor specific.

To support third party commands (e.g., EXTENDED COPY), the access controls coordinator may provide proxy tokens (see 8.3.1.6.2) to allow an application client to pass its access capabilities to the application client for another initiator port.

An application client manages the access controls state of the SCSI target device using the ACCESS CONTROL IN command (see 8.3.2) and the ACCESS CONTROL OUT command (see 8.3.3).

A SCSI target device has access controls disabled when it is manufactured and after successful completion of the ACCESS CONTROL OUT command with DISABLE ACCESS CONTROLS service action (see 8.3.3.3). When access controls are disabled, the ACL contains no entries and the management identifier key (see 8.3.1.8) is zero.

The first successful ACCESS CONTROL OUT command with MANAGE ACL service action (see 8.3.3.2) shall enable access controls. When access controls are enabled, all logical units, except LUN 0 and all well known logical units, shall be inaccessible to all initiator ports unless the ACL (see 8.3.1.3) allows access.

The ACL allows an initiator port access to a logical unit if the ACL contains an ACE (see 8.3.1.3) with an access identifier (see 8.3.1.3.2) associated with the initiator port and that ACE contains a LUACD (see 8.3.1.3.3) that references the logical unit.

When the ACL allows access to a logical unit, the REPORT LUNS command parameter data bytes representing that logical unit shall contain the LUN value found in the LUACD that references that logical unit and the application client for the initiator port shall use the same LUN value when sending commands to the logical unit.

An initiator port also may be allowed access to a logical unit through the use of a proxy token (see 8.3.1.6.2).

Once access controls are enabled, they shall remain enabled until:

a) Successful completion of an ACCESS CONTROL OUT command with DISABLE ACCESS CONTROLS service action; or
b) Vendor specific physical intervention.

Successful downloading of microcode (see 6.35) may result in access controls being disabled.

Once access controls are enabled, power cycles, hard resets, logical unit resets, and I_T nexus losses shall not disable them.

8.3.1.3 The access control list (ACL)

8.3.1.3.1 ACL overview

The specific access controls for a SCSI target device are instantiated by the access controls coordinator using data in an ACL. The ACL contains zero or more ACEs and zero or more proxy tokens (see 8.3.1.6.2.1).

Each ACE contains the following:

a) One access identifier (see 8.3.1.3.2) that identifies the initiator port(s) to which the ACE applies; and
b) A list of LUACDs (see 8.3.1.3.3) that identify the logical units to which the identified initiator port(s) have access rights and the LUNs used to access those logical units via those initiator port(s). Each LUACD contains the following:
   A) A vendor specific logical unit reference; and
   B) A LUN value.

Figure 6 shows the logical structure of an ACL.

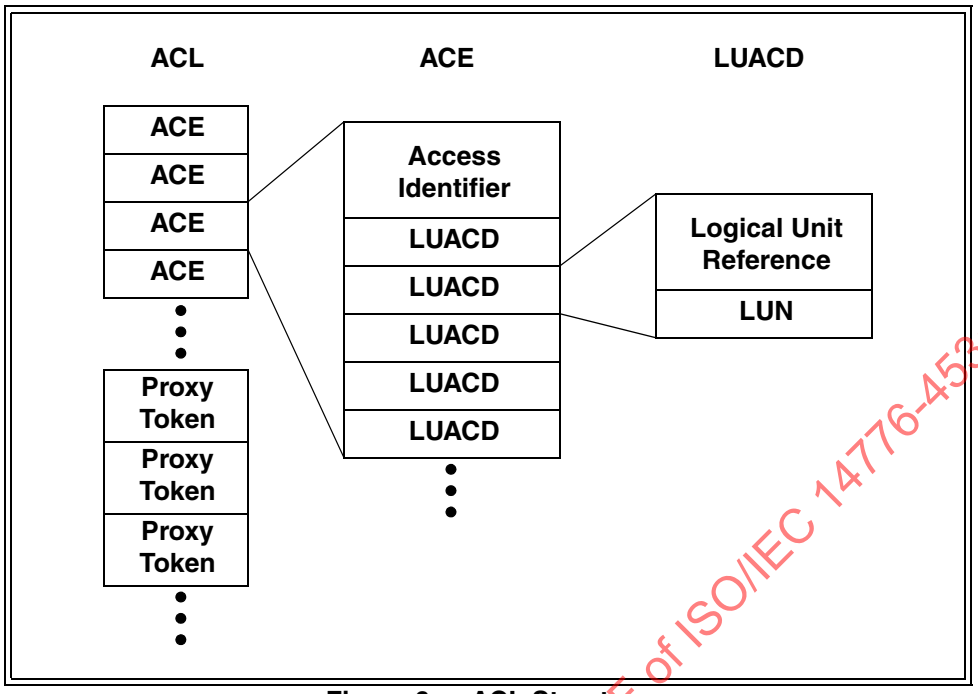

Figure 6 — ACL Structure

8.3.1.3.2 Access identifiers

Initiator ports are identified in an ACE using one of the following types of access identifiers:

a) AccessID - based on initiator port enrollment;
b) TransportID - based on protocol specific identification of initiator ports; or
c) Vendor specific access identifiers.

An initiator port is allowed access to the logical units in an ACE containing an AccessID type access identifier when that initiator port is enrolled as described in 8.3.1.5. An initiator port that has not previously enrolled uses the ACCESS CONTROL OUT command with ACCESS ID ENROLL service action to enroll including the AccessID in parameter data as specified in 8.3.3.4.

An initiator port is associated with an AccessID type access identifier if that initiator port is in the enrolled state or pending-enrolled state with respect to that AccessID (see 8.3.1.5). At any instant in time, an initiator port may be associated with at most one AccessID. All initiator ports enrolled using a specific AccessID share the same ACE and access to all the logical units its LUACDs describe.

TransportID access identifiers are SCSI transport protocol specific as described in 7.5.4.

An initiator port is allowed access to the logical units in an ACE containing a TransportID type access identifier when the identification for the initiator port matches that found in the TransportID in a way that is consistent with the TransportID definition (see 7.5.4). There is no need to process any command to obtain logical unit access based on a Transport ID because the needed information is provided by the SCSI transport protocol layer.

The formats of access identifiers are defined in 8.3.1.13.

8.3.1.3.3 Logical unit access control descriptors

Each LUACD in an ACE identifies one logical unit to which the initiator ports associated with the access identifier are allowed access and specifies the LUN value used when accessing the logical unit via those initiator ports. The format of a LUACD is vendor specific.

The identification of a logical unit in a LUACD is vendor specific. The logical unit identified by a LUACD shall not be a well known logical unit. A logical unit shall be referenced in no more than one LUACD per ACE.

The LUN value shall conform to the requirements specified in SAM-3. A specific LUN value shall appear in no more than one LUACD per ACE.

8.3.1.4 Managing the ACL

8.3.1.4.1 ACL management overview

The contents of the ACL are managed by an application client using the ACCESS CONTROL OUT command with MANAGE ACL and DISABLE ACCESS CONTROLS service actions. The ACCESS CONTROL OUT command with MANAGE ACL service action (see 8.3.3.2) is used to add, remove, or modify ACEs thus adding, revoking, or changing the allowed access of initiator ports to logical units. The ACCESS CONTROL OUT command with DISABLE ACCESS CONTROLS service action (see 8.3.3.3) disables access controls and discards the ACL.

8.3.1.4.2 Authorizing ACL management

To reduce the possibility of applications other than authorized ACL managers changing the ACL, successful completion of specific access controls service actions (e.g., ACCESS CONTROL OUT command with MANAGE ACL or DISABLE ACCESS CONTROLS service action) requires delivery of the correct management identifier key value (see 8.3.1.8) in the ACCESS CONTROL OUT parameter data. The service actions that require the correct management identifier key are shown in table 335 and table 336.

Table 335 — ACCESS CONTROL OUT management identifier key requirements

Service action name	Management Identifier Key Required	Reference
ACCESS ID ENROLL	No	8.3.3.4
ASSIGN PROXY LUN	No	8.3.3.11
CANCEL ENROLLMENT	No	8.3.3.5
CLEAR ACCESS CONTROLS LOG	Yes	8.3.3.6
DISABLE ACCESS CONTROLS	Yes	8.3.3.3
MANAGE ACL	Yes	8.3.3.2
MANAGE OVERRIDE LOCKOUT TIMER	Yes/No	8.3.3.7
OVERRIDE MGMT ID KEY	No	8.3.3.8
RELEASE PROXY LUN	No	8.3.3.12
REVOKE ALL PROXY TOKENS	No	8.3.3.10
REVOKE PROXY TOKEN	No	8.3.3.9

Table 336 — ACCESS CONTROL IN management identifier key requirements

Service action name	Management Identifier Key Required	Reference
REPORT ACCESS CONTROLS LOG	Yes	8.3.2.4
REPORT ACL	Yes	8.3.2.2
REPORT LU DESCRIPTORS	Yes	8.3.2.3
REPORT OVERRIDE LOCKOUT TIMER	Yes	8.3.2.5
REQUEST PROXY TOKEN	No	8.3.2.6

8.3.1.4.3 Identifying logical units during ACL management

The access controls coordinator shall identify every logical unit of a SCSI target device with a unique default LUN value. The default LUN values used by the access controls coordinator shall be the LUN values that would be reported by the REPORTS LUNS command if access controls were disabled.

An application client discovers the default LUN values using the ACCESS CONTROL IN command with REPORT LU DESCRIPTORS (see 8.3.2.3) or REPORT ACL (see 8.3.2.2) service action and then supplies those default LUN values to the access controls coordinator using the ACCESS CONTROL OUT command with MANAGE ACL service action.

The association between default LUN values and logical units is managed by the access controls coordinator and may change due to circumstances that are beyond the scope of this standard. To track such changes, the access controls coordinator shall maintain a generation counter value called DLgeneration as described in 8.3.1.4.4.

8.3.1.4.4 Tracking changes in logical unit identification

The access controls coordinator shall maintain a generation counter value called DLgeneration to track changes in the association between default LUN values and logical units.

When access controls are disabled DLgeneration shall be zero. When access controls are first enabled (see 8.3.1.2) DLgeneration shall be set to one. While access controls are enabled, the access controls coordinator shall increment DLgeneration by one every time the association between default LUN values and logical units changes (e.g., following the creation of a new logical unit, deletion of an existing logical unit, or removal and recreation of an existing logical unit).

The access controls coordinator shall include the current DLgeneration in the parameter data returned by an ACCESS CONTROL IN command with REPORT LU DESCRIPTORS (see 8.3.2.3) or REPORT ACL (see 8.3.2.2) service action. The application client shall supply the DLgeneration for the default LUN values it is using in the parameter data for an ACCESS CONTROL OUT command with MANAGE ACL service action (see 8.3.3.2).

Before processing the ACL change information in the parameter list provided by an ACCESS CONTROL OUT command with MANAGE ACL service action, the access controls coordinator shall verify that the DLgeneration in the parameter data matches the DLgeneration currently in use. If the DLgeneration verification finds a mismatch, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

8.3.1.5 Enrolling AccessIDs

8.3.1.5.1 Enrollment states

8.3.1.5.1.1 Summary of enrollment states

Application clients enroll an initiator port AccessID with the access controls coordinator to be allowed to access logical units listed in the ACE having the same AccessID type access identifier. The ACCESS CONTROL OUT command with ACCESS ID ENROLL service action (see 8.3.3.4) is used to enroll an AccessID. An initiator port shall be in one of three states with respect to such an enrollment:

a) **Not-enrolled**: The state for an initiator port before the first ACCESS CONTROL OUT command with ACCESS ID ENROLL service action is sent to the access controls coordinator. Also the state for an initiator port following successful completion of an ACCESS CONTROL OUT command with CANCEL ENROLLMENT service action (see 8.3.3.5);
b) **Enrolled**: The state for an initiator port following successful completion of an ACCESS CONTROL OUT command with ACCESS ID ENROLL service action; or
c) **Pending-enrolled**: The state for an enrolled initiator port following:
   A) Events in the service delivery subsystem described in 8.3.1.12; or
   B) Successful completion of an ACCESS CONTROL OUT command with MANAGE ACL service action from any initiator port with the FLUSH bit set to one (see 8.3.3.2).

8.3.1.5.1.2 Not-enrolled state

The access controls coordinator shall place an initiator port in the not-enrolled state when it first detects the receipt of a SCSI command or task management function from that initiator port. The initiator port shall remain in the not-enrolled state until successful completion of an ACCESS CONTROL OUT command with ACCESS ID ENROLL service action (see 8.3.3.5).

When in the not-enrolled state, an initiator port shall only have access to logical units on the basis of a TransportID (see 8.3.1.3.2) or on the basis of proxy tokens (see 8.3.1.6.2.1).

The access controls coordinator changes an initiator port from the enrolled state or pending-enrolled state to the not-enrolled state in response to the following events:

a) Successful completion of the ACCESS CONTROL OUT command with CANCEL ENROLLMENT service action (see 8.3.3.5) shall change the state to not-enrolled; or
b) Successful completion of an ACCESS CONTROL OUT command with MANAGE ACL service action (see 8.3.3.2) that replaces the ACL entry for the enrolled AccessID as follows:
   A) If the NOCNCL bit (see 8.3.3.2.2) is set to zero in the ACCESS CONTROL OUT command with MANAGE ACL service action parameter data, the state shall change to not-enrolled; or
   B) If the NOCNCL bit is set to one, the state may change to not-enrolled based on vendor specific criteria.

An application client for an enrolled initiator port may discover that the initiator port has transitioned to the not-enrolled state as a result of actions taken by a third party (e.g., an ACCESS CONTROL OUT command with MANAGE ACL service action performed by another initiator port or a logical unit reset).

Placing an enrolled initiator in the not-enrolled state indicates that the ACE defining that initiator port's logical unit access has changed (e.g., previous relationships between logical units and LUN values may no longer apply).

If an application client detects this loss of enrollment on an initiator port, it may take recovery actions. However, such actions may be disruptive for the SCSI initiator device and may not be required. Use of the not-enrolled state is avoidable if the application client that sends the ACCESS CONTROL OUT command with MANAGE ACL service action determines that its requested changes to the ACL do not alter the existing relationships between logical units and LUN values in any existing ACEs with AccessID type access identifiers and sets the NOCNCL bit to one, recommending that initiator ports be left in their current enrollment state.

The access controls coordinator selects from the following options for responding to a NOCNCL bit set to one in a vendor specific manner:

a) Honor the recommendation, causing the minimum effects on SCSI initiator devices and requiring no extra actions on the part of the access controls coordinator;
b) Ignore the recommendation and always place initiator ports in the non-enrolled state, causing the maximum disruption for SCSI initiator devices, but requiring no extra resources on the part of the access controls coordinator; or
c) Ignore the recommendation and examine the current and new ACEs to determine if an initiator port should be placed in the non-enrolled state.

If the application client that sends the ACCESS CONTROL OUT command with MANAGE ACL service action is unable to determine whether the ACE logical unit relationships are altered as a result of processing the command, then it should set the NOCNCL bit to zero and it should coordinate the ACL change with the application clients for affected initiator ports to ensure proper data integrity. Such coordination is beyond the scope of this standard.

8.3.1.5.1.3 Enrolled state

The access controls coordinator shall place an initiator port in the enrolled state (i.e., enroll the initiator port) following successful completion of the ACCESS CONTROL OUT command with ACCESS ID ENROLL service action (see 8.3.3.4). The ACCESS CONTROL OUT command with ACCESS ID ENROLL service action is successful only if:

a) The initiator port was in the not-enrolled state and the AccessID in the ACCESS CONTROL OUT command with ACCESS ID ENROLL service action parameter data matches the access identifier in an ACE. This results in the initiator port being enrolled and allowed access to the logical units specified in the LUACDs in the ACE (see 8.3.1.3); or
b) The initiator port was in the enrolled state or pending-enrolled state and the AccessID in the ACCESS CONTROL OUT command with ACCESS ID ENROLL service action parameter data matches the current enrolled AccessID for the initiator port.

If the initiator port was in the enrolled state or pending-enrolled state and the AccessID in the ACCESS CONTROL OUT command with ACCESS ID ENROLL service action parameter data does not match the current enrolled AccessID for the initiator port, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to ACCESS DENIED - ENROLLMENT CONFLICT. If the initiator port was in the enrolled state, it shall be transitioned to the pending-enrolled state.

Transitions from the enrolled state to the not-enrolled state are described in 8.3.1.5.1.2. Transitions from the enrolled state to the pending-enrolled state are described in 8.3.1.5.1.4.

NOTE 65 - This standard does not preclude implicit enrollments through mechanisms in the service delivery subsystem. Such mechanisms should perform implicit enrollments after identification by TransportID and should fail in the case where there are ACL conflicts as described in 8.3.1.5.2.

8.3.1.5.1.4 Pending-enrolled state

The access controls coordinator shall place an initiator port in the pending-enrolled state if that initiator port currently is in the enrolled state, and in response to the following:

a) A logical unit reset;
b) An I_T nexus loss associated with that initiator port; or
c) Successful completion of an ACCESS CONTROL OUT command with MANAGE ACL service action where the FLUSH bit is set to one in the parameter data.

While in the pending-enrolled state, the initiator port's access to logical units is limited as described in 8.3.1.7.

8.3.1.5.2 ACL LUN conflict resolution

ACL LUN conflicts may occur if:

a) An application client for an initiator port in the not-enrolled state attempts to enroll an AccessID using the ACCESS CONTROL OUT command with ACCESS ID ENROLL service action (see 8.3.3.4); or
b) An ACCESS CONTROL OUT command with MANAGE ACL service action (see 8.3.3.2) attempts to change the ACL with the result that it conflicts with existing enrollments (see 8.3.1.5) or proxy LUN assignments (see 8.3.1.6.2.2).

Three types of ACL LUN conflicts may occur:

a) The TransportID ACE (see 8.3.1.3) and the AccessID ACE for the initiator port each contain a LUACD with the same LUN value but with different logical unit references;
b) The TransportID ACE and the AccessID ACE for the initiator port each contain a LUACD with the different LUN values but with the same logical unit references; or
c) The enrolling initiator port has proxy access rights to a logical unit addressed with a LUN value that equals a LUN value in a LUACD in the AccessID ACE for the initiator port.

If an ACL LUN conflict occurs during the processing of an ACCESS CONTROL OUT command with MANAGE ACL service action, the command shall be terminated with CHECK CONDITION status (see 8.3.3.2.2).

If an ACL LUN conflict occurs during the processing of an ACCESS CONTROL OUT command with ACCESS ID ENROLL service action, the following actions shall be taken as part of the handling of the enrollment function:

a) The ACCESS CONTROL OUT command with ACCESS ID ENROLL service action shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, the additional sense code set to ACCESS DENIED - ACL LUN CONFLICT;
b) The initiator port shall remain in the not-enrolled state; and
c) If the ACL LUN conflict is not the result of proxy access rights, the access controls coordinator shall record the event in the access controls log as described in 8.3.1.10.

8.3.1.6 Granting and revoking access rights

8.3.1.6.1 Non-proxy access rights

The ACCESS CONTROL OUT command with MANAGE ACL service action (see 8.3.3.2) adds or replaces ACEs in the ACL (see 8.3.1.3). One ACE describes the logical unit access allowed by one access identifier (see 8.3.1.3.2) and the LUN values to be used in addressing the accessible logical units. The access identifier specifies the initiator port(s) permitted access to the logical units described by the ACE.

With the exception of proxy access rights (see 8.3.1.6.2), access rights are granted by:

a) Adding a new ACE to the ACL; or
b) Replacing an existing ACE with an ACE that includes additional LUACDs.

With the exception of proxy access rights, access rights are revoked by:

a) Removing an ACE from the ACL; or
b) Replacing an existing ACE with an ACE that removes one or more LUACDs.

When an ACE is added or replaced the requirements stated in 8.3.1.5.1.2 and 8.3.1.11 apply.

8.3.1.6.2 Proxy access

8.3.1.6.2.1 Proxy tokens

An application client with access rights to a logical unit via an initiator port on the basis of an ACE in the ACL (see 8.3.1.6.1) may temporarily share that access with third parties using the proxy access mechanism. The application client uses the ACCESS CONTROL IN command with REQUEST PROXY TOKEN service action (see 8.3.2.6) to request that the access control coordinator generate a proxy token for the logical unit specified by the LUN value in the CDB.

The access controls coordinator generates the proxy token in a vendor specific manner. For a specific SCSI target device, all active proxy token values should be unique. Proxy token values should be reused as infrequently as possible to prevent proxy tokens that have been used and released from being given unintended meaning.

Power cycles, hard resets, logical unit resets, and I_T nexus losses shall not affect the validity and proxy access rights of proxy tokens (see 8.3.1.12). A proxy token shall remain valid and retain the same proxy access rights until one of the following occurs:

a) An application client with access rights to a logical unit via an initiator port based on an ACE in the ACL revokes the proxy token using:
   A) The ACCESS CONTROL OUT command with REVOKE PROXY TOKEN service action (see 8.3.3.9); or
   B) The ACCESS CONTROL OUT command with REVOKE ALL PROXY TOKENS service action (see 8.3.3.10);

or

b) An application client issues the ACCESS CONTROL OUT command with MANAGE ACL service action (see 8.3.3.2) with parameter data containing the Revoke Proxy Token ACE page (see 8.3.3.2.4) or Revoke All Proxy Tokens ACE page (see 8.3.3.2.5).

8.3.1.6.2.2 Proxy LUNs

To extend proxy access rights to a third party, an application client forwards a proxy token (see 8.3.1.6.2.2) to the third party (e.g., in a target descriptor in the parameter data of the EXTENDED COPY command).

The third party sends the access controls coordinator an ACCESS CONTROL OUT command with ASSIGN PROXY LUN service action (see 8.3.3.11) specifying the proxy token to request creation of a proxy access right to the referenced logical unit. The access controls coordinator determines the referenced logical unit from the proxy token value. The third party is unaware of the exact logical unit to which it is requesting access.

The parameter data for the ACCESS CONTROL OUT command with ASSIGN PROXY LUN service action includes the LUN value that the third party intends to use when accessing the referenced logical unit. The resulting LUN value is called a proxy LUN. If the ACCESS CONTROL OUT command with ASSIGN PROXY LUN service action is successful, the proxy LUN becomes the third party's mechanism for accessing the logical unit by proxy.

Once assigned, a proxy LUN shall remain valid until one of the following occurs:

a) The third party releases the proxy LUN value using the ACCESS CONTROL OUT command with RELEASE PROXY LUN service action (see 8.3.3.12);
b) The proxy token is made invalid as described in 8.3.1.6.2.1; or
c) A logical unit reset or I_T nexus loss of the I_T nexus used to assign the proxy LUN (see 8.3.1.12).

The third party may reissue the ACCESS CONTROL OUT command with ASSIGN PROXY LUN service action in an attempt to re-establish its proxy access rights. If the cause of the proxy token becoming invalid was temporary, the reissued command should succeed. The access controls coordinator shall process the request as described in 8.3.1.6.2.1 without reference to any previous assignment of the proxy LUN value.

8.3.1.7 Verifying access rights

When access controls are enabled (see 8.3.1.2), access rights for an initiator port shall be validated as described in this subclause.

All the linked commands in a single task shall be processed based on the ACL that is in effect when the task first enters the task enabled state. Relationships between access controls and tasks in a task set are described in 8.3.1.11.1.

All commands shall be processed as if access controls were not present if the ACL (see 8.3.1.3) allows the initiator port access to the addressed logical unit as a result of one of the following conditions:

a) The ACL contains an ACE containing a TransportID type access identifier (see 8.3.1.3.2) for the initiator port and that ACE includes a LUACD with LUN value matching the addressed LUN;
b) The initiator port is in the enrolled state (see 8.3.1.5.1.3) under an AccessID, the ACL contains an ACE containing that AccessID as an access identifier, and that ACE includes a LUACD with LUN value matching the addressed LUN; or
c) The addressed LUN matches a proxy LUN value (see 8.3.1.6.2.2) assigned using the ACCESS CONTROL OUT command with ASSIGN PROXY LUN service action (see 8.3.3.11) and the proxy token (see 8.3.1.6.2.1) used to assign the proxy LUN value is still valid.

If the initiator port is in the pending-enrolled state (see 8.3.1.5.1.4) under an AccessID, the ACL contains an ACE containing that AccessID as an access identifier, and that ACE includes a LUACD with LUN value matching the addressed LUN, then commands shall be processed as follows:

a) INQUIRY, REPORT LUNS, ACCESS CONTROL OUT and ACCESS CONTROL IN commands shall be processed as if access controls were not present;
b) A REQUEST SENSE command shall be processed as if access controls were not present as described in 6.27, except in the case where a sense key set to NO SENSE would be returned. In this case, the REQUEST SENSE command shall return the sense key set to ILLEGAL REQUEST and the additional sense code set to ACCESS DENIED - INITIATOR PENDING-ENROLLED; and
c) Any other command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to ACCESS DENIED - INITIATOR PENDING-ENROLLED.

An application client should respond to the ACCESS DENIED - INITIATOR PENDING-ENROLLED additional sense code by sending an ACCESS CONTROL OUT command with ACCESS ID ENROLL service action. If the command succeeds, the application client may retry the terminated command.

If an INQUIRY command is addressed to a LUN for which there is no matching LUN value in any LUACD in any ACE allowing the initiator port logical unit access rights, the standard INQUIRY data (see 6.4.2) PERIPHERAL DEVICE TYPE field shall be set to 1Fh and the PERIPHERAL QUALIFIER field shall be set to 011b (i.e., the device server is not capable of supporting a device at this logical unit).

The parameter data returned in response to a REPORT LUNS command addressed to LUN 0 or to the REPORT LUNS well known logical unit shall return only the list of LUN values that are associated to accessible logical units according to the following criteria:

a) If the initiator port is in the enrolled state or pending-enrolled state, the REPORT LUNS parameter data shall include any LUN values found in LUACDs in the ACE containing the AccessID enrolled by the initiator port;
b) If the initiator port, in any enrollment state has a TransportID found in the access identifier of an ACE, then the REPORT LUNS parameter data shall include any LUN values found in LUACDs in that ACE; and
c) If the initiator port, in any enrollment state has access to any proxy LUNs (see 8.3.1.6.2.2), then those LUN values shall be included in the REPORT LUNS parameter data.

The parameter data returned in response to a REPORT LUNS command that describes well known logical units shall not be affected by access controls.

If the initiator port is in the not-enrolled state and is not allowed access to any logical unit as result of its TransportID or as a result of a proxy LUN assignment, then the REPORT LUNS parameter data shall include only LUN 0 and well known logical units, as specified in 6.21.

Except when access controls are disabled, all cases not described previously in this subclause shall result in termination of the command with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to LOGICAL UNIT NOT SUPPORTED.

8.3.1.8 The management identifier key

8.3.1.8.1 Management identifier key usage

The management identifier key identifies the application that is responsible for managing access controls for a SCSI target device. This identification occurs when the application client specifies a new management identifier key value in each ACCESS CONTROL OUT command with the MANAGE ACL service action (see 8.3.3.2), and when the last specified management identifier key value appears in ACCESS CONTROL IN and ACCESS CONTROL OUT service actions as required in 8.3.1.4.2.

To allow for failure scenarios where the management identifier key value has been lost, an override procedure involving a timer is described in 8.3.1.8.2.

Use of the management identifier key has the following features:

a) Management of access controls is associated with those application clients that provide the correct management identifier key without regard for the initiator port from which the command was received; and
b) Only an application client that has knowledge of the management identifier key may change the ACL, allowing the management of access controls to be limited to specific applications and application clients.

8.3.1.8.2 Overriding the management identifier key

8.3.1.8.2.1 The OVERRIDE MGMT ID KEY service action

If the management identifier key needs to be replaced and the current management identifier key is not available, then the ACCESS CONTROL OUT command with OVERRIDE MGMT ID KEY service action (see 8.3.3.8) may be used to force the management identifier key to a known value.

The ACCESS CONTROL OUT command with OVERRIDE MGMT ID KEY service action should be used only for failure recovery. If failure recovery is not required, the ACCESS CONTROL OUT command with MANAGE ACL service action should be used.

To protect the management identifier key from unauthorized overrides, the access controls coordinator shall restrict use of the ACCESS CONTROL OUT command with OVERRIDE MGMT ID KEY service action based on the value of the override lockout timer (see 8.3.1.8.2.2).

When the override lockout timer is not zero, an ACCESS CONTROL OUT command with OVERRIDE MGMT ID KEY service action shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

When the override lockout timer is zero, an ACCESS CONTROL OUT command with OVERRIDE MGMT ID KEY service action shall be processed as described in 8.3.3.8.

The access controls coordinator shall log the receipt of each ACCESS CONTROL OUT command with OVERRIDE MGMT ID KEY service action and its success or failure as described in 8.3.1.10.

8.3.1.8.2.2 The override lockout timer

The access controls coordinator shall maintain the override lockout timer capable of counting up to 65 535 seconds. When the override lockout timer is not zero it shall be decreased by one nominally once per second but no more frequently than once every 800 milliseconds until the value reaches zero. When the override lockout timer is zero, it shall not be changed except as the result of commands sent by an application client.

The ACCESS CONTROL OUT command with MANAGE OVERRIDE LOCKOUT TIMER service action manages the state of the override lockout timer (see 8.3.3.7), performing one of the following functions:

a) If the incorrect management identifier key is supplied or if no parameter data is sent, the access controls coordinator shall reset the override lockout timer to the last received initial override lockout timer value; or
b) If the correct management identifier key is supplied, then the access controls coordinator shall do the following:
    1) Save the initial override lockout timer value supplied in the parameter data; and
    2) Reset the override lockout timer to the new initial value.

Setting the initial override lockout timer value to zero disables the override lockout timer and allows the ACCESS CONTROL OUT command with OVERRIDE MGMT KEY service action to succeed at any time.

Any application that knows the management identifier key may establish an initial override lockout timer value of sufficient duration (i.e., up to about 18 hours). Maintaining a non-zero override lockout timer value may be accomplished without knowing the management identifier key or transporting the management identifier key on the service delivery subsystem. Attempts to establish a zero initial override lockout timer value that are not accom-

panied by the correct management identifier key result in decreasing the probability that a subsequent ACCESS CONTROL OUT command with OVERRIDE MGMT ID KEY service action is able to succeed by resetting the override lockout timer.

After a logical unit reset, the override lock timer shall be set to the initial override lockout timer value within ten seconds of the non-volatile memory containing the initial override lockout timer value becoming available.

The ACCESS CONTROL IN command with REPORT OVERRIDE LOCKOUT TIMER may be used to discover the state of the override lockout timer.

8.3.1.9 Reporting access control information

Specific service actions of the ACCESS CONTROL IN command may be used by an application client to request a report from the access controls coordinator about its access controls data and state.

The ACCESS CONTROL IN command with REPORT ACL service action (see 8.3.2.2) returns the ACL (see 8.3.1.3). The information reported includes the following:

a) The list of access identifiers (see 8.3.1.3.2) and the associated LUACDs (see 8.3.1.3.3) currently in effect; and
b) The list of proxy tokens (see 8.3.1.6.2.1) currently in effect.

The ACCESS CONTROL IN command with REPORT ACCESS CONTROLS LOG service action (see 8.3.2.4) returns the contents of the access controls log (see 8.3.1.10).

The ACCESS CONTROL IN command with REPORT OVERRIDE LOCKOUT TIMER service action (see 8.3.2.5) reports on the state of the override lockout timer (see 8.3.1.8.2.2).

8.3.1.10 Access controls log

The access controls log is a record of events maintained by the access controls coordinator.

The access controls log has three portions, each recording a different class of events:

a) **invalid key events**: mismatches between the management identifier key (see 8.3.1.8) specified by a service action and the current value maintained by the access controls coordinator;
b) **key override events**: attempts to override the management identifier key (see 8.3.1.8.2.1), whether the attempt fails or succeeds; and
c) **ACL LUN conflict events**: (see 8.3.1.5.2).

Each portion of the log is required to contain a counter of the events. When a SCSI target device is manufactured, all counters shall be set to zero. When access controls are disabled, all counters except the key override events counter shall be set to zero. Each counter shall be incremented by one whenever the relevant event occurs.

Each log portion may contain additional records with more specific information about each event. When the resources for additional log records are exhausted, the access controls coordinator shall preserve the most recently added log records in preference to older log records.

Log records contain a TIME STAMP field whose contents are vendor specific. If the access controls coordinator has no time stamp resources the TIME STAMP field shall be set to zero. If time stamp values are provided, the same timing clock and time stamp format shall be used for all access controls log entries.

Invalid key events occur whenever an access controls command requires the checking of an application client supplied management identifier key against the current management identifier key saved by the access controls coordinator and the two values fail to match. When such an event occurs, the access controls coordinator shall increment the invalid keys counter by one. If the log has additional resources to record event details, the access controls coordinator shall add an invalid keys log record (containing the information defined in 8.3.2.4.2.3) describing the event.

Key override events occur when the access controls coordinator receives the ACCESS CONTROL OUT command with OVERRIDE MGMT KEY service action (see 8.3.3.8). When such an event occurs, the access controls coordinator shall increment the key overrides counter by one without regard for whether the command succeeds or fails. If the log has additional resources to record event details, the access controls coordinator shall add an key overrides log record (containing the information defined in 8.3.2.4.2.2) describing the event.

ACL LUN conflict events occur as specified in 8.3.1.5.2. When such an event occurs, the access controls coordinator shall increment the ACL LUN conflicts counter by one. If the log has additional resources to record event details, the access controls coordinator shall add an ACL LUN conflicts log record (containing the information defined in 8.3.2.4.2.4) describing the event.

Selected portions of the access controls log may be requested by an application client using the ACCESS CONTROL IN command with REPORT ACCESS CONTROLS LOG service action (see 8.3.2.4). With the exception of the key overrides portion, selected portions of the log may be cleared and the counters reset to zero using the ACCESS CONTROL OUT command with CLEAR ACCESS CONTROLS LOG service action (see 8.3.3.6).

8.3.1.11 Interactions of access controls and other features

8.3.1.11.1 Task set management and access controls

Upon successful completion of an ACCESS CONTROL OUT command with MANAGE ACL service action (see 8.3.3.2), the specified ACL (see 8.3.1.3) shall apply to all tasks that subsequently enter the task enabled state. Tasks that have modified SCSI target device state information (e.g., media, mode pages, and log pages) shall not be affected by an ACCESS CONTROL OUT command that subsequently enters the task enabled state. Tasks in the task enabled state that have not modified SCSI target device state information may or may not be affected by an ACCESS CONTROL OUT command that subsequently enters the task enabled state. The ACL in effect prior to when the ACCESS CONTROL OUT command with MANAGE ACL or DISABLE ACCESS CONTROLS service action entered the task enabled state shall apply to all tasks that are not affected by the ACCESS CONTROL OUT command.

All the operations performed by a task shall complete under the control of a single ACL, either the state in effect prior to processing of the ACCESS CONTROL OUT command or the state in effect following processing of the ACCESS CONTROL OUT command. After a task enters the task enabled state for the first time changing the access control state from disabled to enabled (see 8.3.1.2) shall have no effect on the task.

Multiple access control commands, both ACCESS CONTROL IN and ACCESS CONTROL OUT, may be in the task set concurrently. The order of processing of such commands is defined by the task set management requirements (see SAM-3), but each command shall be processed as a single indivisible command without any interleaving of actions that may be required by other access control commands.

8.3.1.11.2 Existing reservations and ACL changes

If a logical unit is reserved by one I_T nexus and that logical unit becomes accessible to another I_T nexus as a result of an access control command, then there shall be no changes in the reservation of that logical unit.

If a logical unit is reserved by an I_T nexus and that logical unit becomes inaccessible to that I_T nexus as a result of an access control command or other access control related event, then there shall be no changes in the reservation. Existing persistent reservations mechanisms allow for other SCSI initiator devices with access to that logical unit to clear the reservation.

8.3.1.12 Access controls information persistence and memory usage requirements

If a SCSI target device supports access controls, then the SCSI target device shall contain an access controls coordinator that shall maintain the following information in nonvolatile memory:

a) Whether access controls are enabled or disabled; and
b) The access controls data that table 337 and table 338 require to persistent across power cycles, hard resets, and logical unit resets.

If the access control coordinator's nonvolatile memory is not ready and the access controls coordinator is unable to determine that access controls are disabled, then the device servers for all logical units shall terminate all commands except INQUIRY and REQUEST SENSE commands with CHECK CONDITION status, with the sense key set to NOT READY, and the additional sense code set as described in table 184 (see 6.33).

Following an I_T nexus loss, a previously enrolled initiator port shall be placed in the pending-enrolled state, if that initiator port was associated with the lost I_T nexus. Following a logical unit reset, all previously enrolled initiator ports shall be placed in the pending-enrolled state.

The information shown in table 337 shall be maintained by the access controls coordinator.

Table 337 — Mandatory access controls resources

Information Description	**Size** (in bits)	**Persistent Across Power Cycles, Hard Resets, and Logical Unit Resets**
One ACL (see 8.3.1.3) containing at least one ACE containing one access identifier (see 8.3.1.3.2), and at least one LUACD (see 8.3.1.3.3)	VS	Yes
The Enrollment State for each initiator port (see 8.3.1.5.1)	VS	Yes
Management Identifier Key (see 8.3.1.8)	64	Yes
Default LUNs Generation (DLgeneration, see 8.3.1.4.4)	32	Yes
Override Lockout Timer (see 8.3.1.8.2.2)	16	No
Initial Override Lockout Timer value (see 8.3.1.8.2.2)	16	Yes
Access Controls Log Event Counters (see 8.3.1.10) containing at least the following: a) Key Overrides Counter; b) Invalid Keys Counter; and c) ACL LUN Conflicts Counter	 16 16 16	Yes Yes Yes Yes