Working Draft
American National
Standard

Project
T10/1683-D

Revision 13
27 September 2007

Information technology - SCSI Architecture Model - 4 (SAM-4)

T10 Technical Editor: George Penokie
IBM Corporation
MS: 49C
3605 Highway 52 N
Rochester, MN 55901
USA

Telephone: 507-253-5308
Email: gop@us.ibm.com

Reference number
ISO/IEC 14776-*:200x**
ANSI INCITS *-200x**

Summary of Comments on SCSI Architecture Model - 4

Page: i

Author: George Penokie Subject: Note Date: 4/8/2008 10:21:03 AM
Original Comment Numbers:
001 - Brocade
017 - Emulex
208 - HPQ
001 - Network Appliance
021 - Quantum
096 - Western Digital Corporation

344

Author: George Penokie Subject: Note Date: 11/2/2007 10:52:53 AM
Key to PDF comments:
- A comment that has not been looked at yet in the current run through of the comments has no Checkmark.
- A comment that has been looked at but not responded to yet (i.e., editor has no clue what to do about it) is not marked.
- Comment with Status of Accepted has been incorporated into SAM-4.
- Comment with Status of Completed has been incorporated but needs to be talked about in a working group meeting.
- Comment with Status of Rejected has been rejected or the wording has changed from the commenters suggestion.
- Comment with reply indicates a the actual text the was incorporated if it is different than the proposed text or the reason for the comment being rejected.

Author: relliott Subject: Note Date: 10/10/2007 1:51:20 PM
move right-justified text on page i right by .2 inches to line up with the horizontal lines

Status
George Penokie Accepted 10/29/2007 3:42:03 PM

Page: iii

Author: relliott Subject: Highlight Date: 10/27/2007 1:11:19 PM
SCSI Architecture Model - 4
s/b
SCSI Architecture Model - 4 (SAM-4)

Status
George Penokie Accepted 10/29/2007 3:43:52 PM

American National ~~Standard~~
~~for~~ Information Technology

SCSI Architecture Model - 4

Secretariat
Information Technology Industry Council

Approved mm.dd.yy

American National Standards Institute, Inc.

ABSTRACT

This standard specifies the SCSI Architecture Model. The purpose of the architecture is to provide a common basis for the coordination of SCSI standards and to specify those aspects of SCSI I/O system behavior that are independent of a particular technology and common to all implementations.

Introduction

The SCSI Architecture Model - 4 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 overall SCSI architectural model.
Clause 5 describes the SCSI command model element of the SCSI architecture.
Clause 6 describes the events that may be detected by a SCSI device.
Clause 7 describes the task management functions common to SCSI devices.
Clause 8 describes the task set management capabilities common to SCSI devices.
Annex A summarizes the identifier and name definitions of the SCSI transport protocols.
Annex B summarizes the SCSI Initiator Port attributes and SCSI Target Port attributes supported by SCSI transport protocols.
Annex C ~~Lists~~ the terminology differences between SAM-3 and this standard.

Page: xviii

Author: relliott Subject: Highlight Date: 10/27/2007 1:08:39 PM
The SCSI Architecture Model - 4 standard

s/b
This standard

Status
George Penokie Accepted 10/29/2007 3:44:50 PM

Author: relliott Subject: Highlight Date: 10/27/2007 1:09:01 PM
Lists
s/b
lists

Status
George Penokie Accepted 10/29/2007 3:46:16 PM

Author: relliott Subject: Highlight Date: 10/27/2007 1:16:32 PM
SAM-3 and this standard
s/b
this standard and previous versions of this standard

Status
George Penokie Accepted 10/29/2007 3:45:44 PM

Author: suhlerp Subject: Replacement Text Date: 10/26/2007 10:11:58 AM
lists [don't capitalize, for consistency]

Status
George Penokie Accepted 10/26/2007 2:31:13 PM

American National Standard
for Information Technology -

SCSI Architecture Model - 4 (SAM-4)

1 Scope

1.1 Introduction

The set of SCSI (Small Computer System Interface) standards consists of this standard and the SCSI implementation standards described in 1.3. This standard defines a reference model that specifies common behaviors for SCSI devices, and an abstract structure that is generic to all SCSI I/O system implementations.

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.

The following architecture model concepts from previous versions of this standard are made obsolete by this standard:

a) Support for the SPI-5 SCSI transport protocol;
b) Contingent Allegiance;
c) The TARGET RESET task management function;
d) Basic task management model;
e) Untagged tasks; and
f) Linked command function.

1.2 Requirements precedence

This standard defines generic requirements that pertain to SCSI implementation standards and implementation requirements. An implementation requirement specifies behavior in terms of measurable or observable parameters that apply to an implementation. Examples of implementation requirements defined in this document are the status values to be returned upon command completion and the service responses to be returned upon task management function completion.

Generic requirements are transformed to implementation requirements by an implementation standard. An example of a generic requirement is the hard reset behavior specified in 6.3.2.

Page: 1

Author: relliott Subject: Highlight Date: 10/28/2007 10:48:46 PM
architecture model concepts
s/b
concepts

Status
George Penokie Accepted 10/29/2007 3:47:28 PM

Author: relliott Subject: Highlight Date: 10/28/2007 10:47:53 PM
SCSI Architecture Model - 4
s/b
SCSI Architecture Model - 4 (SAM-4)

(page 1 footer differs from the other pages)

Status
George Penokie Accepted 10/29/2007 3:50:02 PM

.

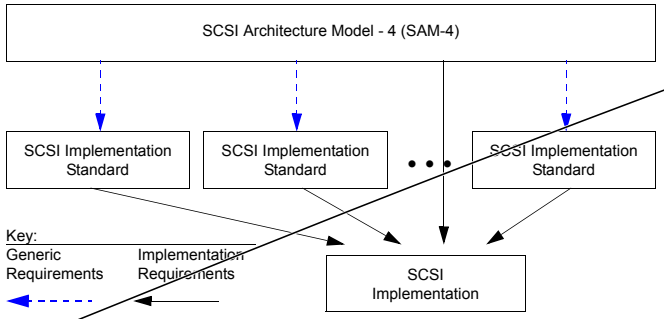


Figure 1 — Requirements precedence

As shown in figure 1, all SCSI implementation standards shall reflect the generic requirements defined herein. In addition, an implementation claiming SCSI compliance shall conform to the applicable implementation requirements defined in this standard and the appropriate SCSI implementation standards. In the event of a conflict between this document and other SCSI standards under the jurisdiction of technical committee T10, the requirements of this standard shall apply.

1.3 SCSI standards family

Figure 2 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.

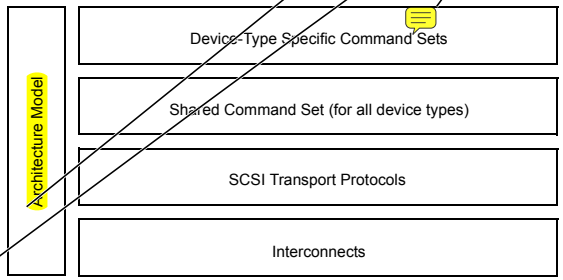


Figure 2 — SCSI document structure

The roadmap in figure 2 is intended to show the general applicability of the documents to one another. Figure 2 is not intended to imply a relationship such as a hierarchy, protocol stack, or system architecture.

The functional areas identified in figure 2 characterize the scope of standards within a group as follows:

Page: 2

Author: Mark Evans, WDC Subject: Highlight Date: 10/25/2007 1:01:53 PM
Since text takes precedence in this standard, and text should introduce figures, tables, etc., I recommend moving the following paragraph above figure 1.

As shown in figure 1, all SCSI implementation standards shall reflect the generic requirements defined herein. In addition, an implementation claiming SCSI compliance shall conform to the applicable implementation requirements defined in this standard and the appropriate SCSI implementation standards. In the event of a conflict between this document and other SCSI standards under the jurisdiction of technical committee T10, the requirements of this standard shall apply.

Status
George Penokie Accepted 10/26/2007 2:31:33 PM
Author: Mark Evans, WDC Subject: Note Date: 10/25/2007 1:03:15 PM
I recommend that at least one "e.g." be added either in the four rows in figure 2, in the descriptive text that follows, or both. See SBC-3 for an example.

Status
George Penokie Rejected 10/29/2007 4:06:27 PM
Author: George Penokie Subject: Note Date: 10/26/2007 2:33:21 PM
Copied the SBC-3 figure into SAM-4.

Author: relliott Subject: Highlight Date: 10/28/2007 10:45:42 PM
Architecture Model
s/b
SCSI Architecture Model

Status
George Penokie Accepted 10/29/2007 3:50:40 PM
Author: Emulex Subject: Highlight Date: 10/30/2007 1:16:34 PM
Emulex-001
Page: 2 first sentence below Figure 2 - "roadmap" s/b "document structure"

Status
George Penokie Accepted 10/30/2007 2:17:02 PM
Author: George Penokie Subject: Note Date: 10/30/2007 2:16:57 PM
Changed to SCSI document structure

Architecture Model: Defines ~~the SCSI systems model, the~~ functional partitioning of the SCSI standard set and requirements applicable to all SCSI implementations and implementation standards.

Device-Type Specific Command Sets: Implementation standards that define specific device types including a device model for each device type. These standards specify the required commands and behavior ~~that~~ is specific to a given device type and prescribe the requirements to be followed by a SCSI initiator device when sending commands to a SCSI target device having the specific device type. The commands and behaviors for a specific device type may include by reference commands and behaviors that are shared by all SCSI devices.

Shared Command Set: An implementation standard that defines a model for all SCSI device types. This standard specifies the required commands and behavior that is common to all SCSI devices, regardless of device type, and prescribes the requirements to be followed by a SCSI initiator device when sending commands to any SCSI target device.

SCSI Transport Protocols: Implementation standards that define the requirements for exchanging information so that different SCSI devices are capable of communicating.

Interconnects: Implementation standards that define the communications mechanism employed by the SCSI transport protocols. These standards may describe the electrical and signaling requirements essential for SCSI devices to interoperate over a given interconnect. Interconnect standards may allow the interconnection of devices other than SCSI devices in ways that are outside the scope of this standard.

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

Page: 3

Author: relliott Subject: Highlight Date: 10/28/2007 10:46:07 PM
Architecture Model
s/b
SCSI Architecture Model

Status
George Penokie Accepted 10/29/2007 3:51:15 PM

Author: Emulex Subject: Highlight Date: 10/30/2007 1:19:49 PM
Emulex-002
Page: 3 Device-Type Specific Command Sets: - second sentence "is" s/b "are"

Status
George Penokie Rejected 10/30/2007 2:19:45 PM
Author: George Penokie Subject: Note Date: 10/30/2007 2:19:22 PM
Changed to "behaviors that are"

2 Normative references

2.1 Normative references

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.

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.

2.2 Approved references

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

ISO/IEC 14776-232, *Serial Bus Protocol - 3 (SBP-3)* [ANSI INCITS 375-2004]

ISO/IEC 14776-241, *SCSI RDMA Protocol (SRP)* [ANSI INCITS 365-2002]

2.3 References under development

At the time of publication, the following referenced standards were still under development. For information on the current status of the document, or regarding availability, contact the relevant standards body or other organization as indicated.

ISO/IEC 14776-454, *SCSI Primary Commands - 4 (SPC-4)* [T10/1731-D]

ISO/IEC 14776-323, *SCSI Block Commands - 3 (SBC-3)* [T10/1799-D]

ISO/IEC 14776-357, *Automation/Drive Interface - Commands - 2 (ADC-2)* [T10/1741-D]

ISO/IEC 14776-223, *Fibre Channel Protocol for SCSI - 4 (FCP-4)* [T10/1828-D]

ISO/IEC 14776-152, *Serial Attached SCSI - 2 (SAS-2)* [T10/1760-D]

ISO/IEC 14776-192, *Automation/Drive Interface - Transport Protocol - 2 (ADT-2)* [T10/1742-D]

2.4 Other References

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

NOTE 1 - Copies of IETF standards may be obtained through the Internet Engineering Task Force (IETF) at http://www.ietf.org.

OMG Unified Modeling Language (UML) Specification Version 1.5, March 2003

NOTE 2 - For more information on the UML specification, contact the Object Modeling Group at http://www.omg.org.

Page: 4

Author: Mark Evans, WDC Subject: Highlight Date: 10/5/2007 11:35:26 AM
I'm not sure that "The following standards.." is correct because I thought the OMG document was a specification -- not a standard. If this is true, then this should be changed to, "The following standards and specifications..."

Status
George Penokie Accepted 10/26/2007 2:33:45 PM

Author: Mark Evans, WDC Subject: Highlight Date: 10/24/2007 1:31:16 PM
"Copies of the following documents..."
I think that some of the following documents are not available from ANSI (e.g., the IETF and OMG documents as noted below), therefore, I think this...
s/b
"Except where noted in the following subclauses, copies of the following documents...".

Status
George Penokie Rejected 10/29/2007 4:07:37 PM

Author: George Penokie Subject: Note Date: 11/8/2007 7:24:00 PM -06'00'
Changed to "Copies of the following documents may be obtained from ANSI:
a) approved ANSI standards;
b) approved and draft international and regional standards (ISO, IEC); and
c) approved and draft foreign standards (including JIS, and DIN).

Author: relliott Subject: Cross-Out Date: 10/9/2007 4:58:55 PM
Delete "IEC 60027-2:2000, Letter symbols to be used in electrical technology - Part 2: Telecommunications and electronics"

This is the reference that defines Ki, Mi, etc. prefixes for powers-of-two units to avoid misusing the SI powers-of-ten units. These prefixes are not used in this standard.

Status
George Penokie Accepted 11/8/2007 7:26:09 PM -06'00'

Author: suhlerp Subject: Sticky Note Date: 10/23/2007 11:24:43 AM
ADC-2 will probably have completed INCITS approval by the time SAM-4 finishes LB comment resolution.

Status
George Penokie Accepted 10/26/2007 2:44:42 PM

Author: George Penokie Subject: Note Date: 10/26/2007 2:44:37 PM
ADC-2 moved to approved references

Author: relliott Subject: Highlight Date: 10/8/2007 5:22:38 PM
References
s/b
references

Status
George Penokie Accepted 10/29/2007 3:52:19 PM

3 Definitions, symbols, abbreviations, and conventions

3.1 Definitions

3.1.1 ACA command: A command ~~performed by a task with the~~ ACA task attribute (see 3.1.8, and 8.6.5).

3.1.2 additional sense code: A combination of the ADDITIONAL SENSE CODE field and the ADDITIONAL SENSE CODE QUALIFIER field in the sense data (see 3.1.109 and SPC-3).

3.1.3 aggregation: When referring to classes (see 3.1.13), a form of association that defines a whole-part relationship between the whole (i.e., aggregate) and its parts.

3.1.4 application client: A class whose objects are, or an object that is, the source of commands and task management function requests. See 4.5.10.

3.1.5 argument: A datum provided as input to or output from a procedure call (see 3.1.80).

3.1.6 association: When referring to classes (see 3.1.13), a relationship between two or more classes that specifies connections among their objects (i.e., a relationship that specifies that objects of one class are connected to objects of another class).

3.1.7 attribute: When referring to classes (see 3.1.13), a named property of a class that describes the range of values that the class or its objects may hold. When referring to objects (see 3.1.71), a named property of the object.

3.1.8 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. See 5.8.2.

3.1.9 background operation: An operation started by a command that continues processing after the task containing the command is no longer in the task set. See 5.5.

3.1.10 blocked task state: When in this state a task is prevented from completing due to an ACA condition.

3.1.11 blocking boundary: A task set boundary denoting a set of conditions that inhibit tasks outside the boundary from entering the enabled task state.

3.1.12 byte: An 8-bit construct.

3.1.13 class: A description of a set of objects that share the same attributes, operations, relationships (e.g., aggregation, association, generalization, and dependency), and semantics. Classes may have attributes and may support operations.

3.1.14 class diagram: Shows a set of classes and their relationships. Class diagrams are used to illustrate the static design view of a system.

3.1.15 client-server: A relationship established between a pair of distributed entities wherein one (the client) requests the other (the server) to perform some operation or unit of work on the client's behalf.

3.1.16 client: An entity that requests a service from a server. This standard defines one client, the application client.

3.1.17 code value: A defined numeric value, possibly a member of a series of defined numeric values, representing an identified and described instance or condition. Code values are defined to be used in a specific field (see 3.1.42), in a procedure call input argument (see 3.6.4), in a procedure call output argument, or in a procedure call result.

Page: 5

Author: Mark Evans, WDC Subject: Note Date: 10/29/2007 1:19:47 PM
The definitions (i.e., those with the numbers 3.1.x) are all left and right justified, and several have hyphens added by FrameMaker to split words across lines. All of the other clauses are left justified and right ragged with no hyphens added to split words. At a quick glance, SPC-4 is left and right justified with hyphens added throughout. SBC-3 and SAS- 2 are left justified and right ragged with no hyphens added throughout. Interesting to me, the style guide (i.e., 05-085r7) has the same odd combination as SAM-4. I wonder why that is? One way or the other, I think that all the clauses should have the same format -- editor's choice.

Status
George Penokie Accepted 10/29/2007 8:17:48 AM
Author: George Penokie Subject: Note Date: 10/26/2007 3:09:39 PM
There was one paragraph style that had justification and hyphenation set. That has been corrected.

Author: relliott Subject: Cross-Out Date: 10/29/2007 10:38:22 AM
performed by a task

Status
George Penokie Accepted 11/8/2007 7:27:38 PM -06'00'
Author: relliott Subject: Highlight Date: 10/8/2007 6:28:38 PM
ACA task attribute
s/b ACA (smallcaps lowercase) task attribute

Status
George Penokie Accepted 10/29/2007 3:55:20 PM
Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:20:00 PM
see 3.1.8: This cross reference hot link is broken. When I click on it, nothing happens. This is true for almost all of the "3.1.x" cross references in this subclause and for some of them in other clauses.

Status
George Penokie Accepted 10/29/2007 8:17:59 AM
Author: George Penokie Subject: Note Date: 10/26/2007 4:20:09 PM
This was a pdf generation issue. To make it work in the pdf setup link tab the Create named destinations for all paragraphs box must be checked.

Author: Mark Evans, WDC Subject: Note Date: 10/24/2007 1:32:48 PM
Add "(see 8.5.3)" at the end of 3.1.10.

Status
George Penokie Accepted 10/29/2007 8:26:50 AM
Author: George Penokie Subject: Note Date: 10/29/2007 8:25:46 AM
Added "See 8.5.3."

Author: Emulex Subject: Highlight Date: 10/30/2007 1:21:18 PM
Page: 5 3.1.10 "When in this state" s/b "A state in which"

Status
George Penokie Accepted 10/30/2007 2:21:03 PM
Author: Mark Evans, WDC Subject: Note Date: 10/24/2007 1:37:33 PM
Add "(see 8.9.1)" at the end of 3.1.11.

Status
George Penokie Accepted 10/29/2007 8:27:03 AM
Author: George Penokie Subject: Note Date: 10/29/2007 8:26:42 AM
Added "See 8.9." as that is a more accurate reference

Author: relliott Subject: Note Date: 10/16/2007 7:28:28 PM
At the end of 3.1.14 class diagram, add "See 3.6.2."

Status
George Penokie Accepted 10/29/2007 3:57:09 PM
Author: Mark Evans, WDC Subject: Note Date: 10/24/2007 3:02:08 PM
Add "(4.3)" at the end of 3.1.15.

Status
George Penokie Accepted 10/29/2007 8:28:42 AM
Author: George Penokie Subject: Note Date: 10/29/2007 8:28:36 AM
Added "See 4.3."

Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:22:04 PM
"...possibly a member of a series of defined numeric values,"
This phrase tells me nothing. Remove the "ly" adverb and add an "e.g.", and then it tells me something.
s/b
"...sometimes a member of a series of defined numeric values (e.g., an additional sense code),"

Status
George Penokie Rejected 11/8/2007 7:34:03 PM -06'00'
Author: George Penokie Subject: Note Date: 11/8/2007 7:33:39 PM -06'00'
Deleted the entire glossary entery

~~**3.1.10 command:** A request describing a unit of work to be performed by a device server.~~

3.1.19 command descriptor block (CDB): A structure used to communicate a command from an application client to a device server. A CDB may have a fixed length of 6 bytes, 10 bytes, 12 bytes, or 16 bytes, or a variable length of between 12 and 260 bytes. See 5.2 and SPC-4.

3.1.20 command standard: A SCSI standard that defines the model, commands, and parameter data for a device type (e.g., SPC-4, SBC-3). See clause 1.

3.1.21 completed command: A command that has ended by returning a status and service response of TASK COMPLETE.

~~**3.1.22 completed task:** A task that has ended by returning a status and service response of TASK COMPLETE.~~

3.1.23 confirmation: A response returned to an application client or device server that signals the completion of a service request.

3.1.24 confirmed SCSI transport protocol service: A service available at the SCSI transport protocol service interface that includes a confirmation of completion. See 4.9.

3.1.25 constraint: When referring to classes (see 3.1.13) and objects (see 3.1.71), a mechanism for specifying semantics or conditions that are maintained as true between entities (e.g., a required condition between associations).

3.1.26 current task: A task that has a data transfer SCSI transport protocol service request in progress (see 5.4.3) or is in the process of sending command status. Each SCSI transport protocol standard may define the SCSI transport protocol specific conditions under which a task is considered a current task.

3.1.27 deferred error: An error generated by a background operation (see SPC-4).

3.1.28 dependency: A relationship between two elements in which a change to one element (e.g., the supplier) may affect or supply information needed by the other element (e.g., the client).

3.1.29 dependent logical unit: A logical unit that is addressed via some other logical unit(s) in a hierarchical logical unit structure (see 3.1.45), also a logical unit that is at a higher numbered level in the hierarchy than the referenced logical unit (see 4.5.19.4).

3.1.30 device model: The description of a type of SCSI target device (e.g., block, stream).

3.1.31 device server: A class whose objects process, or an object that processes, SCSI tasks according to the requirements for task management described in clause 8. See 4.5.20.

3.1.32 device service request: A request submitted by an application client conveying a command to a device server.

3.1.33 device service response: The response returned to an application client by a device server on completion of a command.

3.1.34 domain: An I/O system consisting of a set of SCSI devices and a service delivery subsystem, where the SCSI devices interact with one another by means of a service delivery subsystem.

3.1.35 dormant task state: When in this state a task is prevented from entering the enabled task state (see 3.1.36) due to the presence of certain other tasks in the task set.

3.1.36 enabled task state: When in this state a task may complete at any time.

Page: 6

Author: relliott Subject: Note Date: 10/29/2007 10:19:32 AM
The relationship of command and task is unclear now that linked commands are gone. The standard uses a mix of the terms with no apparent reason (if read without remembering the history).

Status
George Penokie Accepted 12/12/2007 9:46:28 AM -06'00'
Author: George Penokie Subject: Note Date: 11/16/2007 2:51:49 PM -06'00'
Change task to command in all places were the two are really referring to one entity. Make sure to change task identifier to command identifier.

Author: relliott Subject: Highlight Date: 10/29/2007 10:15:30 AM
returning a status and service response of TASK COMPLETE
s/b
returning a service response of TASK COMPLETE

The status is of secondary importance and doesn't need to be mentioned. As worded, it sounds like TASK COMPLETE could be a status value.

Status
George Penokie Accepted 12/12/2007 9:48:04 AM -06'00'
Author: relliott Subject: Cross-Out Date: 10/29/2007 10:17:20 AM
3.1.22 completed task: A task that has ended by returning a status and service response of TASK COMPLETE.

Delete this definition and replace all uses of "completed task" with "completed command." Their definitions are identical now that task=command.

Status
George Penokie Accepted 12/14/2007 10:19:57 AM -06'00'
Author: Emulex Subject: Highlight Date: 10/30/2007 1:31:54 PM
Emulex-004
Page: 6 3.1.28 "supplier" s/b "server"

Status
George Penokie Accepted 10/30/2007 2:23:01 PM
Author: relliott Subject: Highlight Date: 10/28/2007 10:50:27 PM
(see 3.1.45), also
s/b
(see 3.1.45). Also,

Status
George Penokie Rejected 10/29/2007 4:03:23 PM
Author: George Penokie Subject: Note Date: 10/29/2007 4:03:15 PM
Changed to "(see 3.1.45) and that is...

Author: Mark Evans, WDC Subject: Highlight Date: 10/24/2007 3:05:27 PM
"(e.g., block, stream)."
s/b
"(e.g., a block device or a stream device)."

Status
George Penokie Accepted 10/29/2007 8:33:02 AM
Author: Mark Evans, WDC Subject: Highlight Date: 10/24/2007 3:06:21 PM
"..a service delivery subsystem."
s/b
"..the service delivery subsystem."

Status
George Penokie Accepted 10/29/2007 8:33:47 AM
Author: Mark Evans, WDC Subject: Note Date: 10/24/2007 3:08:39 PM
Add "(see 8.5.4)" at the end of 3.1.35.

Status
George Penokie Accepted 10/29/2007 8:36:07 AM
Author: George Penokie Subject: Note Date: 10/29/2007 8:35:47 AM
Added "See 8.5.4."

Author: Emulex Subject: Highlight Date: 10/30/2007 1:32:04 PM
Emulex-005
Page: 6 3.1.35 and 3.1.36 "When in this state" s/b "A state in which"

Status
George Penokie Accepted 10/30/2007 2:22:08 PM
Author: Mark Evans, WDC Subject: Note Date: 10/24/2007 3:10:36 PM
Add "(see 8.5.2)" at the end of 3.1.36.

Status
George Penokie Accepted 10/29/2007 8:36:00 AM
Author: George Penokie Subject: Note Date: 10/29/2007 8:35:56 AM
Added "See 8.5.2."

Author: Emulex Subject: Highlight Date: 10/30/2007 1:37:09 PM
Emulex-005
Page: 6 3.1.35 and 3.1.36 "When in this state" s/b "A state in which"

Status
George Penokie Accepted 10/30/2007 2:22:19 PM

Comments from page 6 continued on next page

3.1.57 instance: A concrete manifestation of an abstraction to which a set of operations may be applied and which may have a state that stores the effects of the operation (e.g., an object is an instance of a class).

3.1.58 in transit: Information that has been delivered to a service delivery subsystem for transmission, but not yet received.

3.1.59 implicit head of queue: An optional processing model for specified commands wherein a task may be treated as if it had been received with a HEAD OF QUEUE task attribute. See 8.2.

3.1.60 layer: A subdivision of the architecture constituted by SCSI initiator device and SCSI target device elements at the same level relative to the interconnect.

3.1.61 link: An individual connection between two objects in an object diagram. Represents an instance of an association.

3.1.62 logical unit: A class whose objects implement, or an object that implements a device model and manages tasks to process commands sent by an application client. See 4.5.19.

3.1.63 logical unit reset: A condition resulting from a hard reset condition or a logical unit reset event in which the logical unit performs the logical unit reset operations described in 6.3.3, SPC-4, and the appropriate command standards.

3.1.64 logical unit reset event: An event that results in a logical unit reset condition as described in 6.3.3.

3.1.65 logical unit inventory: The list of the logical unit numbers reported by a REPORT LUNS command (see SPC-4)

3.1.66 logical unit number (LUN): A 64-bit or 16-bit identifier for a logical unit. See 4.6.

3.1.67 multiplicity: When referring to classes (see 3.1.13), an indication of the range of allowable instances that a class or an attribute may have.

3.1.68 name: A label of an object that is unique within a specified context and should never change ~~(e.g., the terms name and world wide identifier (WWID) may be interchangeable)~~.

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 4.7.

3.1.70 non-faulted I_T nexus: An I_T nexus that is not a faulted I_T nexus (see 3.1.38).

3.1.71 object: An entity with a well-defined boundary and identity that encapsulates state and behavior. All objects are instances of classes (see 3.1.57).

3.1.72 object diagram: ~~shows~~ a set of objects and their relationships at a point in time. Object diagrams are used to illustrate static shapshots of instances of the things found in class diagrams.

3.1.73 operation: A service that may be requested from any object of the class in order to effect behavior. Operations describe what a class is allowed to do and may be a request or a query. A request may change the state of the object but a query should not.

3.1.74 peer entities: Entities within the same layer (see 3.1.60).

~~3.1.75 pending command:~~ From the point of view of the application client, the description of command between the time that the application client calls the **Send SCSI Command** SCSI transport protocol service and the time one of the SCSI target device responses described in 5.5 is received.

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

Page: 8

Author: Mark Evans, WDC Subject: Highlight Date: 10/24/2007 3:12:12 PM
"...for transmission, but not yet received."
s/b
"...for transmission, but has not yet arrived at the intended recipient."

Status
George Penokie Accepted 12/12/2007 9:55:03 AM -06'00'

Author: Mark Evans, WDC Subject: Highlight Date: 10/24/2007 3:13:23 PM
"...in an object diagram. Represents an instance..."
s/b
"...in an object diagram representing an instance..."

Status
George Penokie Accepted 10/29/2007 8:40:29 AM

Author: suhlerp Subject: Sticky Note Date: 10/25/2007 3:31:38 PM
Should there be a definition for "logical unit name" (see 4.5.19.3)? A designator can be associated with a logical unit.

3.1.x logical unit name: A name (see 3.1.68) of a logical unit that is world wide unique within the SCSI transport protocol of a SCSI domain in which the SCSI device containing the logical unit has SCSI ports (see 4.5.4.2). The logical unit name may be made available to other SCSI devices or SCSI ports in SCSI transport protocol specific ways.

Status
George Penokie Accepted 12/12/2007 9:55:41 AM -06'00'

Author: relliott Subject: Note Date: 10/29/2007 10:33:39 AM
The phrase "logical unit number" is used many times in the standard where the acronym LUN could/should be used instead.

Status
George Penokie Accepted 12/12/2007 9:58:42 AM -06'00'
Author: George Penokie Subject: Note Date: 11/16/2007 3:16:53 PM -06'00'
In addition to making LUN the choice W-LUN was replaces well-known logical unit

Author: Mark Evans, WDC Subject: Cross-Out Date: 10/29/2007 1:21:32 PM
Delete "(e.g., the terms name and world wide identifier (WWID) may be interchangeable)", as neither the terms "world wide identifier" or "WWID" are used anywhere else in this document. Alternately, those terms could be defined, but that would be a little odd since they are only used in this definition.

Status
George Penokie Accepted 12/12/2007 10:02:58 AM -06'00'

Author: suhlerp Subject: Replacement Text Date: 10/23/2007 12:43:01 PM
Shows [capitalize]

Status
George Penokie Accepted 10/29/2007 8:49:00 AM

Author: relliott Subject: Note Date: 10/16/2007 7:28:48 PM
At the end of 3.1.72 object diagram, add "See 3.6.3."

Status
George Penokie Accepted 10/30/2007 11:00:00 AM

Author: George Penokie Subject: Cross-Out Date: 12/12/2007 9:59:36 AM -06'00'
Delete this term from SAM-4 as it is dated and no longer needed - 3.1.75 pending command:

Status
George Penokie Accepted 12/14/2007 10:20:30 AM -06'00'

Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:22:01 PM
"From the point of view of the application client, the description of command between the time that the application client calls the Send SCSI Command SCSI transport protocol service and the time one of the SCSI target device responses described in 5.5 is received."
s/b
"From the point of view of the application client, a command from the time that the application client calls the Send SCSI Command SCSI transport protocol service until the application client receives a response for the command from the SCSI target device (see 5.5)."

Status
George Penokie Rejected 11/16/2007 3:20:32 PM -06'00'
Author: George Penokie Subject: Note Date: 11/16/2007 3:20:27 PM -06'00'
Changed to "A command for which an application client task (see 4.5.10) exists."

~~**3.1.77 power on:** A condition resulting from a power on event in which the SCSI device performs the power on operations described in 6.3.1, SPC-4, and the appropriate command standards.~~

3.1.78 power on event: Power being applied to a SCSI device, resulting in a power on condition as described in 6.3.1.

3.1.79 procedure: An operation that is invoked through an external calling interface.

3.1.80 procedure call: The model used by this standard for the interfaces involving both the SAL (see 3.1.91) and STPL (see 3.1.102), having the appearance of a programming language function call.

3.1.81 protocol: A specification and/or implementation of the requirements governing the content and exchange of information passed between distributed entities through a service delivery subsystem.

3.1.82 queue: The arrangement of tasks within a task set (see 3.1.128), usually according to the ~~temporal order~~ in which they were created.

3.1.83 receiver: A client or server that is the recipient of a service delivery transaction.

3.1.84 reference model: A standard model used to specify system requirements in an implementation-independent manner.

3.1.85 relative port identifier: An identifier for a SCSI port that is unique within a SCSI device. See 4.5.5.2.

3.1.86 request: A transaction invoking a service.

3.1.87 request-response transaction: An interaction between a pair of distributed, cooperating entities, consisting of a request for service submitted to an entity followed by a response conveying the result.

3.1.88 reset event: A SCSI transport protocol specific event that results in a hard reset condition as described in 6.3.2.

3.1.89 response: A transaction conveying the result of a request.

3.1.90 role: When referring to classes (see 3.1.13) and objects (see 3.1.71), a label at the end of an association or aggregation that defines a relationship to the class on the other side of the association or aggregation.

3.1.91 SCSI application layer (SAL): The protocols and procedures that implement or issue commands and task management functions by using services provided by a STPL (see 3.1.102).

3.1.92 SCSI device: A class whose objects are, or an object that is, connected to a service delivery subsystem and supports a SCSI application protocol. See 4.5.4.

3.1.93 SCSI device name: A name (see 3.1.68) of a SCSI device that is world wide unique within the SCSI transport protocol of a SCSI domain in which the SCSI device has SCSI ports (see 4.5.4.2). The SCSI device name may be made available to other SCSI devices or SCSI ports in SCSI transport protocol specific ways.

3.1.94 SCSI event: A condition defined by this standard (e.g., logical unit reset) that is detected by SCSI device and that requires notification of its occurrence within the SCSI device. See clause 6.

3.1.95 SCSI I/O system: An I/O system, consisting of two or more SCSI devices, a SCSI interconnect and a SCSI transport protocol that collectively interact to perform SCSI I/O operations.

3.1.96 SCSI initiator device: A class whose objects originate, or an object that originates, device service and task management requests to be processed by a SCSI target device and receives device service and task management responses from SCSI target devices. ~~When used this term refers to SCSI initiator devices.~~

Page: 9

Author: relliott Subject: Note Date: 10/27/2007 1:40:45 PM
Add:
3.1.xx power loss expected: A condition resulting from a power loss expected event in which the logical unit performs the power loss expected operations described in 6.3.5, SPC-4, and the appropriate transport protocol and command standards.

Status
George Penokie Accepted 12/12/2007 10:06:12 AM -06'00'

Author: relliott Subject: Note Date: 10/27/2007 1:41:44 PM
Add:
3.1.xx power loss expected event: An event that results in a power loss expected condition (see 3.1.xx) as described in 6.3.5.

Status
George Penokie Accepted 12/12/2007 10:06:42 AM -06'00'

Author: relliott Subject: Note Date: 10/16/2007 7:29:08 PM
At the end of 3.1.80 procedure call, add "See 3.6.4."

Status
George Penokie Accepted 10/30/2007 11:07:05 AM

Author: Mark Evans, WDC Subject: Highlight Date: 10/24/2007 3:18:46 PM
"...usually according to the temporal order..."
s/b
"...most often according to the temporal order..."

Status
George Penokie Accepted 10/29/2007 9:40:54 AM
Author: George Penokie Subject: Note Date: 11/16/2007 3:24:19 PM -06'00'
Deleted ', most often according to the temporal order in which they were created.'

Author: relliott Subject: Cross-Out Date: 1/7/2008 4:30:39 PM -06'00'
Delete << When used this term refers to SCSI initiator devices. >>

Status
George Penokie Accepted 1/7/2008 4:31:45 PM -06'00'

Author: relliott Subject: Note Date: 1/7/2008 6:00:11 PM -06'00'
In the definition for the SCSI initiator device it states << objects originate, or an object that originates, >>. In the definition for the SCSI target device it states << processing and sends device service and task management responses >>. Only one term should be used. Either both devices << originate >> or both << send >>.

Status
George Penokie Rejected 1/7/2008 6:01:53 PM -06'00'
Author: George Penokie Subject: Note Date: 1/7/2008 6:01:47 PM -06'00'
Initiators originate stuff and targets send stuff. Targets never originate. I believe having two terms here is correct.

3.1.97 SCSI initiator port: A class whose objects act, or an object that acts, the connection between application clients and a service delivery subsystem through which requests, indications, responses, and confirmations are routed. In all cases when this term is used it refers to a SCSI initiator port. See 4.5.7

3.1.98 SCSI port: A class whose objects connect, or an object that connects, the application client, device server or task manager to a service delivery subsystem through which requests and responses are routed. SCSI port is synonymous with port. A SCSI port is one of: a SCSI initiator port (see 3.1.97) or a SCSI target port (see 3.1.101). See 4.5.5.

3.1.99 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 3.1.55) or a target port identifier (see 3.1.117).

3.1.100 SCSI target device: A class whose objects receive, or an object that receives, device service and task management requests for processing and sends device service and task management responses to SCSI initiator devices. When used this term refers to SCSI target devices. See 4.5.14.

3.1.101 SCSI target port: A class whose objects contain, or an object that contains, a task router and acts as the connection between device servers and task managers and a service delivery subsystem through which indications and responses are routed. When this term is used it refers to a SCSI target port. See 4.5.6.

3.1.102 SCSI transport protocol layer (STPL): The protocol and services used by a SAL (see 3.1.91) to transport data representing a SCSI application protocol transaction.

3.1.103 SCSI transport protocol service confirmation: A procedure call from the STPL notifying the SAL that a SCSI transport protocol service request has completed.

3.1.104 SCSI transport protocol service indication: A procedure call from the STPL notifying the SAL that a SCSI transport protocol transaction has occurred.

3.1.105 SCSI transport protocol service request: A procedure call to the STPL to begin a SCSI transport protocol service transaction.

3.1.106 SCSI transport protocol service response: A procedure call to the STPL containing a reply from the SAL in response to a SCSI transport protocol service indication.

3.1.107 SCSI transport protocol specific: Implementation of the referenced item is defined by a SCSI transport protocol standard (see 1.3).

3.1.108 sender: A client or server that originates a service delivery transaction.

3.1.109 sense data: Data returned to an application client in the same I_T_L_Q nexus transaction (see 3.1.51) as a CHECK CONDITION status (see 5.8.6). Fields in the sense data are referenced by name in this standard. See SPC-4 for a complete sense data format definition. Sense data may also be retrieved using the REQUEST SENSE command (see SPC-4).

3.1.110 sense key: The SENSE KEY field in the sense data (see 3.1.109 and SPC-4).

3.1.111 server: An entity that performs a service on behalf of a client.

3.1.112 service: Any operation or function performed by a SCSI object that is invoked by other SCSI objects.

3.1.113 service delivery failure: Any non-recoverable error causing the corruption or loss of one or more service delivery transactions while in transit.

3.1.114 service delivery subsystem: A class whose objects are, or an object that is, 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 4.5.3.

Page: 10

Author: relliott Subject: Highlight Date: 1/7/2008 5:21:27 PM -06'00'
This << oject that acts, the connection between >> should be << bject that acts, as the connection between >>

Status
George Penokie Accepted 1/7/2008 5:08:57 PM -06'00'
Author: relliott Subject: Highlight Date: 1/7/2008 5:38:12 PM -06'00'
This << through which requests, indications, responses, and confirmations are routed. >> should be << through which requests and confirmations are routed. >> as the initiator side does not have any indications or responses defined.

Status
George Penokie Completed 1/7/2008 5:34:26 PM -06'00'
Author: relliott Subject: Cross-Out Date: 1/7/2008 4:31:08 PM -06'00'
Delete << In all cases when this term is used it refers to a SCSI initiator port. >>

Status
George Penokie Accepted 1/7/2008 4:31:52 PM -06'00'
Author: George Penokie Subject: Highlight Date: 1/7/2008 5:40:52 PM -06'00'
This << through which requests and responses are routed. >> should be <<through which requests, indications, responses, and confirmations are routed >>

Status
George Penokie Completed 1/7/2008 5:40:49 PM -06'00'
Author: relliott Subject: Cross-Out Date: 1/7/2008 4:30:55 PM -06'00'
Delete << When used this term refers to SCSI target devices. >>

Status
George Penokie Accepted 1/7/2008 4:31:38 PM -06'00'
Author: relliott Subject: Highlight Date: 1/7/2008 5:21:11 PM -06'00'
This << A class whose objects contain, or an object that contains, a task router and acts as the connection between device servers >> should be << class whose objects act, or an object that acts, as the connection between device servers >>

Status
George Penokie Accepted 1/7/2008 5:20:36 PM -06'00'
Author: relliott Subject: Highlight Date: 1/7/2008 5:37:50 PM -06'00'
This << a service delivery subsystem through which indications and responses are routed. >> should be << a service delivery subsystem through which requests, indications, responses, and confirmations are routed. >>

Status
George Penokie Completed 1/7/2008 5:37:12 PM -06'00'
Author: relliott Subject: Cross-Out Date: 1/7/2008 4:31:25 PM -06'00'
Delete << When this term is used it refers to a SCSI target port. >>

Status
George Penokie Accepted 1/7/2008 4:31:33 PM -06'00'

3.1.115 service delivery transaction: A request or response sent through a service delivery subsystem.

3.1.116 standard INQUIRY data: Data returned to an application client as a result of an INQUIRY command with the EVPD bit set to zero. Fields in the standard INQUIRY data are referenced by name in this standard and SPC-4 contains a complete definition of the standard INQUIRY data format.

3.1.117 target port identifier: A value by which a SCSI target port is referenced within a domain. See 4.5.6.

3.1.118 target port name: A name (see 3.1.68) of a SCSI target port that is world wide unique within the SCSI transport protocol of the SCSI domain of that SCSI target port (see 4.5.5). The name may be made available to other SCSI devices or SCSI ports in that SCSI domain in SCSI transport protocol specific ways. See 4.5.6.

3.1.119 task: A class whose objects are, or an object that is, within the logical unit representing the work associated with a command. See 4.5.19.

3.1.120 task attribute: An attribute of a task (see 3.1.119) that specifies the processing relationship of a task with regard to other tasks in the task set (see 3.1.128). See 8.6.

3.1.121 task identifier: The portion (i.e, Q) of an I_T_L nexus (see 3.1.49) in a task set (see 3.1.128). See 4.7.2.

3.1.122 task priority: The relative scheduling importance of a task having the SIMPLE task attribute among the set of tasks having the SIMPLE task attribute already in the task set. See 8.7.

3.1.123 task management function: A task management service capable of being requested by an application client to affect the processing of one or more tasks.

3.1.124 task management request: A request submitted by an application client, invoking a task management function to be processed by a task manager.

3.1.125 task management response: The response returned to an application client by a task manager on completion of a task management request.

3.1.126 task manager: A class whose objects are, or an object that is, within a logical unit that controls the sequencing of one or more tasks and processes task management functions. See 4.5.21.

3.1.127 task router: A class whose objects are, or an object that is, within a SCSI target port that routes commands and task management functions between a service delivery subsystem (see 3.1.114) and the appropriate task manager(s). See 4.5.8.

3.1.128 task set: A class whose objects are, or an object that is, a group of tasks within a logical unit, whose interaction is dependent on the task management (e.g., queuing) and ACA requirements. See 4.5.22.

3.1.129 task tag: A term used by previous versions of this standard (see Annex C). See 3.1.121.

3.1.130 **third-party command:** A command that requires a logical unit within a SCSI target device to assume the SCSI initiator device role and send command(s) to another SCSI target device.

3.1.131 transaction: A cooperative interaction between two entities, involving the exchange of information or the processing of some request by one entity on behalf of the other.

3.1.132 unconfirmed SCSI transport protocol service: A service available at the SCSI transport protocol service interface that does not result in a completion confirmation. See 4.9.

3.1.133 well known logical unit: A class whose objects are, or an object that is, a logical unit that only performs specific functions. Well known logical units allow an application client to issue requests to receive and manage specific information relating to a SCSI target device. See 4.5.25.

Page: 11

Author: relliott Subject: Highlight Date: 12/10/2007 3:59:48 PM -06'00'
INQUIRY command
s/b
INQUIRY command (see SPC-4)

then delete the second sentence altogether

Status
George Penokie Rejected 12/10/2007 4:02:35 PM -06'00'
Author: George Penokie Subject: Note Date: 12/10/2007 4:02:29 PM -06'00'
Only deleted the part of the sentence <<and SPC-4 contains a complete definition of the standard INQUIRY data format. >>

Author: relliott Subject: Highlight Date: 10/28/2007 10:55:10 PM
whose objects are, or an object that is, within the logical unit representing
s/b
within the logical unit whose objects represent, or an object that represents,

Status
George Penokie Rejected 12/10/2007 4:16:56 PM -06'00'
Author: George Penokie Subject: Note Date: 12/10/2007 4:16:53 PM -06'00'
Changed the definition to << Synonymous with command (see 3.1.17 and Annex C). >>

Author: relliott Subject: Highlight Date: 9/28/2007 6:49:07 PM
(global)

task identifier
s/b
task tag

There is no good justification for making this change from SAM-3 to SAM-4. Every transport protocol uses the name "tag" now and will have to unnecessarily change. This is reminiscent of changing "queue" to "task set" from SCSI-2 to SCSI-3.

This helps make the ingredients in I_T_L_Q nexus have similar names, but "logical unit number" is not being renamed to "logical unit identifier" to make them all consistent.

Status
George Penokie Rejected 10/30/2007 11:40:38 AM
Author: George Penokie Subject: Note Date: 10/30/2007 11:40:30 AM
The old term "task tag" is just confusing and not consistent. The change is justified as it does not have heavy usage. It is only used 2 times in SAS-2. FCP-4 is it only used 1 time. SPC-4 is only used 2 times. SBC-3 has none.

Author: Mark Evans, WDC Subject: Highlight Date: 10/24/2007 3:20:19 PM
"The portion (i.e, Q) of an I_T_L nexus (see 3.1.49) in a task set (see 3.1.128). See 4.7.2."
s/b
"The portion of an I_T_L_Q nexus (i.e., the Q) that is the numerical identifier of the task in the nexus (see 3.1.49) in a task set (see 3.1.128). See 4.7.2."

Status
George Penokie Rejected 12/12/2007 10:15:36 AM -06'00'
Author: George Penokie Subject: Note Date: 10/29/2007 9:44:56 AM
Changed to "The portion of an I_T_L_Q nexus (i.e., the Q) that is the numerical identifier of the task (see 3.1.49). See 4.7.2."

Author: Emulex Subject: Highlight Date: 10/30/2007 1:38:35 PM
Emulex-006
Page: 11 3.1.121 This definition is a partial sentence. It needs to specify:
The portion (i.e., Q) of an I_T_L nexus (see 3.1.49) in a task set that uniquely identifies each task.

Status
George Penokie Rejected 10/30/2007 2:25:41 PM
Author: George Penokie Subject: Note Date: 10/30/2007 2:25:36 PM
Sentence now reads "The portion of an I_T_L_Q nexus (i.e., the Q) that is the numerical identifier of the task (see 3.1.122) in a task set (see 3.1.131)."

Author: George Penokie Subject: Note Date: 11/8/2007 6:22:53 PM -06'00'
Sentence now reads "The portion of an I_T_L_Q nexus (i.e., the Q) that is the identifier of the task (see 3.1.122) within an I_T_L nexus. See 4.7.2."

Author: Mark Evans, WDC Subject: Note Date: 10/24/2007 3:21:44 PM
Add "See clause 7." at the end of 3.1.123.

Status
George Penokie Accepted 10/29/2007 9:48:32 AM
Author: relliott Subject: Highlight Date: 10/28/2007 10:57:31 PM
whose objects are, or an object that is, within a logical unit that controls
s/b
within a logical unit whose objects control, or an object that controls

Status
George Penokie Accepted 10/30/2007 11:44:53 AM
Author: relliott Subject: Highlight Date: 10/29/2007 12:26:07 AM

Comments from page 11 continued on next page

3.1.115 service delivery transaction: A request or response sent through a service delivery subsystem.

3.1.116 standard INQUIRY data: Data returned to an application client as a result of an INQUIRY command with the EVPD bit set to zero. Fields in the standard INQUIRY data are referenced by name in this standard and SPC-4 contains a complete definition of the standard INQUIRY data format.

3.1.117 target port identifier: A value by which a SCSI target port is referenced within a domain. See 4.5.6.

3.1.118 target port name: A name (see 3.1.68) of a SCSI target port that is world wide unique within the SCSI transport protocol of the SCSI domain of that SCSI target port (see 4.5.5). The name may be made available to other SCSI devices or SCSI ports in that SCSI domain in SCSI transport protocol specific ways. See 4.5.6.

3.1.119 task: A class whose objects are, or an object that is, within the logical unit representing the work associated with a command. See 4.5.19.

3.1.120 task attribute: An attribute of a task (see 3.1.119) that specifies the processing relationship of a task with regard to other tasks in the task set (see 3.1.128). See 8.6.

3.1.121 task identifier: The portion (i.e, Q) of an I_T_L nexus (see 3.1.49) in a task set (see 3.1.128). See 4.7.2.

3.1.122 task priority: The relative scheduling importance of a task having the SIMPLE task attribute among the set of tasks having the SIMPLE task attribute already in the task set. See 8.7.

3.1.123 task management function: A task manager service capable of being requested by an application client to affect the processing of one or more tasks.

3.1.124 task management request: A request submitted by an application client, invoking a task management function to be processed by a task manager.

3.1.125 task management response: The response returned to an application client by a task manager on completion of a task management request.

3.1.126 task manager: A class whose objects are, or an object that is, within a logical unit that controls the sequencing of one or more tasks and processes task management functions. See 4.5.21.

3.1.127 task router: A class whose objects are, or an object that is, within a SCSI target port that routes commands and task management functions between a service delivery subsystem (see 3.1.114) and the appropriate task manager(s). See 4.5.8.

3.1.128 task set: A class whose objects are, or an object that is, a group of tasks within a logical unit, whose interaction is dependent on the task management (e.g., queuing) and ACA requirements. See 4.5.22.

3.1.129 task tag: A term used by previous versions of this standard (see Annex C). See 3.1.121.

3.1.130 **third-party command:** A command that requires a logical unit within a SCSI target device to assume the SCSI initiator device role and send command(s) to another SCSI target device.

3.1.131 transaction: A cooperative interaction between two entities involving the exchange of information or the processing of some request by one entity on behalf of the other.

3.1.132 unconfirmed SCSI transport protocol service: A service available at the SCSI transport protocol service interface that does not result in a completion confirmation. See 4.9.

3.1.133 well known logical unit: A class whose objects are, or an object that is, a logical unit that only performs specific functions. Well known logical units allow an application client to issue requests to receive and manage specific information relating to a SCSI target device. See 4.5.25.

whose objects are, or an object that is, within a SCSI target port that routes
s/b
within a SCSI target port whose objects route, or an object that routes

Status
George Penokie Accepted 10/30/2007 11:45:51 AM
Author: relliott Subject: Highlight Date: 10/16/2007 7:30:44 PM
"assume the SCSI initiator device role"

The logical unit doesn't become an SCSI initiator device itself. The command forces the SCSI device containing that logical unit to assume the SCSI initiator device role.

Status
George Penokie Rejected 12/12/2007 10:19:20 AM -06'00'
Author: George Penokie Subject: Note Date: 12/12/2007 10:19:16 AM -06'00'
Deleted term as it is not used in this standard.

Author: Mark Evans, WDC Subject: Highlight Date: 10/24/2007 3:23:41 PM
"...and send command(s) to another SCSI target device."
s/b
"...and send one or more commands to another SCSI target device."

Status
George Penokie Rejected 12/12/2007 10:18:44 AM -06'00'
Author: George Penokie Subject: Note Date: 11/16/2007 3:25:57 PM -06'00'
Deleted term as it is not used in this standard.

Author: relliott Subject: Highlight Date: 10/29/2007 12:25:16 AM
are
s/b
are each

to match "is" later

Status
George Penokie Accepted 10/30/2007 11:47:23 AM

3.1.134 well known logical unit number (W-LUN): The logical unit number that identifies a well known logical unit. See 4.6.11.

3.2 Acronyms

ACA	Auto Contingent Allegiance (see 3.1.8)
ADC-2	Automation/Drive Interface - Commands - 2 (see 1.3)
CDB	Command Descriptor Block (see 3.1.19)
CRN	Command Reference Number
FCP-4	SCSI Fibre Channel Protocol -4 (see 1.3)
iSCSI	Internet SCSI (see RFC 3720, http://www.ietf.org/rfc/rfc3720.txt)
ISO	Organization for International Standards
LUN	Logical Unit Number (see 3.1.66)
n/a	Not Applicable
RAID	Redundant Array of Independent Disks
SAL	SCSI application layer (see 3.1.91)
SAS-2	Serial Attached SCSI-2 (see 1.3)
SBC-3	SCSI Block Commands-3 (see 1.3)
SBP-3	Serial Bus Protocol -3 (see 1.3)
SCSI	The architecture defined by the family of standards described in 1.3
SPC-4	SCSI Primary Commands -3 (see 1.3)
SRP	SCSI RDMA Protocol (see 1.3)
STPL	SCSI transport protocol layer (see 3.1.102)
VPD	Vital Product Data (see SPC-4)
W-LUN	Well known logical unit number (see 3.1.134)
UML	Unified Modeling Language

3.3 Keywords

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

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

3.3.3 may: A keyword that indicates flexibility of choice with no implied preference (synonymous with may or may not).

3.3.4 may not: A keyword that indicates flexibility of choice with no implied preference (synonymous with may or may not).

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

3.3.6 option, optional: Keywords that describe 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.

3.3.7 prohibited: A keyword used to describe a feature, function, or coded value that is defined in a a non-SCSI standard (i.e., a standard that is not a member of the SCSI family of standards) to which this standard makes a normative reference where the use of said feature, function, or coded value is not allowed for implementations of this standard.

Page: 12

Author: suhlerp Subject: Sticky Note Date: 10/23/2007 1:00:37 PM
ADT-2 Automation/Drive Interface Transport Protocol - 2 (see 1.3)
[used in table A.3]

Status
George Penokie Accepted 10/29/2007 9:53:12 AM

Author: Emulex Subject: Note Date: 10/30/2007 1:41:47 PM
Emulex-007
Page: 12 ADC-2, FCP-4, SAS-2, SBC-3, SBP-3 and SPC-4 make hyphenation consistent in all full standard names. Some have a space before hyphen some do not. ADC-2 has a space after the hyphen.

Status
George Penokie Accepted 10/30/2007 2:28:56 PM
Author: George Penokie Subject: Note Date: 10/30/2007 2:28:51 PM
Added space before and after hyphen when not already there.

Author: Mark Evans, WDC Subject: Highlight Date: 10/24/2007 3:51:51 PM
"...preference (synonymous with may or may not)."
s/b
"...preference. May is synonymous with the phrase "may or may not"."

Status
George Penokie Accepted 10/29/2007 10:17:51 AM

Author: Mark Evans, WDC Subject: Highlight Date: 10/24/2007 3:53:03 PM
"...preference (synonymous with may or may not)."
s/b
"...preference. May not is synonymous with the phrase "may or may not"."

Status
George Penokie Accepted 10/29/2007 10:17:57 AM

Author: Mark Evans, WDC Subject: Cross-Out Date: 10/5/2007 12:38:32 PM
3.3.7 prohibited: this definition should be deleted as it is not used in this standard.

Status
George Penokie Rejected 12/14/2007 10:21:45 AM -06'00'
Author: George Penokie Subject: Note Date: 10/29/2007 2:12:02 PM
Although not in the standard it is a keyword.

3.3.8 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 error.

3.3.9 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.

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

3.3.11 vendor specific: Specification of the referenced item is determined by the SCSI device vendor.

3.4 Editorial conventions

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

Upper case is used when referring to the name of a numeric value defined in this specification or a formal attribute possessed by an entity. When necessary for clarity, names of objects, procedure calls, arguments or discrete states are capitalized or set in bold type. Names of fields are identified using small capital letters (e.g., NACA bit).

Names of procedure calls are identified by a name in bold type, such as **Execute Command** (see clause 5). Names of arguments are denoted by capitalizing each word in the name. For instance, Sense Data is the name of an argument in the **Execute Command** procedure call.

Quantities having a defined numeric value are identified by large capital letters. CHECK CONDITION, for example, refers to the numeric quantity defined in table 25 (see 5.3.1). Quantities having a discrete but unspecified value are identified using small capital letters. As an example, TASK COMPLETE, indicates a quantity returned by the **Execute Command** procedure call (see clause 5). Such quantities are associated with an event or indication whose observable behavior or value is specific to a given implementation standard.

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

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

Notes do not constitute any requirements for implementors.

3.5 Numeric conventions

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

Page: 13

Author: Emulex Subject: Highlight Date: 11/8/2007 6:33:08 PM -06'00'
Emulex-008
Page: 13 3.3.8 last sentence: For backward compatibility in future standards, shouldn't this be "Recipients shall not check ..."?

Status
George Penokie Rejected 11/8/2007 6:32:27 PM -06'00'
Author: George Penokie Subject: Note Date: 11/8/2007 6:31:50 PM -06'00'
This is the way it's been as long as SCSI has been around.

Author: relliott Subject: Highlight Date: 10/27/2007 1:18:29 PM
as error
s/b
as an error

Status
George Penokie Accepted 10/30/2007 11:49:21 AM
Author: Mark Evans, WDC Subject: Highlight Date: 10/25/2007 1:12:30 PM
The paragraph that begins, "Lists sequenced by letters...",
s/b
...replaced by something more complete (e.g., how lists are described in the SCSI style guide).

Status
George Penokie Rejected 12/14/2007 10:28:34 AM -06'00'
Author: George Penokie Subject: Note Date: 10/29/2007 10:23:17 AM
If we do this then we would have to add in the style guide as a referenced document. I do not think that is a good idea. I think the statement is adequate.

Author: George Penokie Subject: Note Date: 12/14/2007 10:28:31 AM -06'00'
Changed to << Lists sequenced by letters (e.g., a) red, b) blue, c) green) show no ordering relationship between the listed items. Lists sequenced by numbers (e.g., 1) red, 2) blue, 3) green) show an ordering relationship between the listed items. >>

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

3.6 Notation conventions

3.6.1 Notation conventions overview

This standard uses class diagrams and object diagrams with notation that is based on the Unified Modeling Language (UML).

See 3.6.2 for the conventions used for class diagrams.

See 3.6.3 for the conventions used for object diagrams.

Within class diagrams and object diagrams there may be constraints which specify requirements and notes which are informative.

A constraint is specified as text encapsulated with a { } notation within a box. See figure 5 for an example of a constraint.

A note is specified as text within a box (i.e., no { }). See figure 6 for an example of a note.

Page: 14

Author: relliott Subject: Highlight Date: 10/8/2007 7:11:07 PM
Within class diagrams and object diagrams there may be constraints which specify requirements and notes which are informative.
s/b
Class diagrams and object diagrams may include constraints, which specify requirements, and notes, which are informative.

Status
George Penokie Accepted 10/30/2007 11:52:29 AM

Solid lines with arrowheads ~~(see figure 4) are the notation~~ used to describe the association relationship between classes in class diagrams. Multiplicity notation occurs at each end of the solid line.

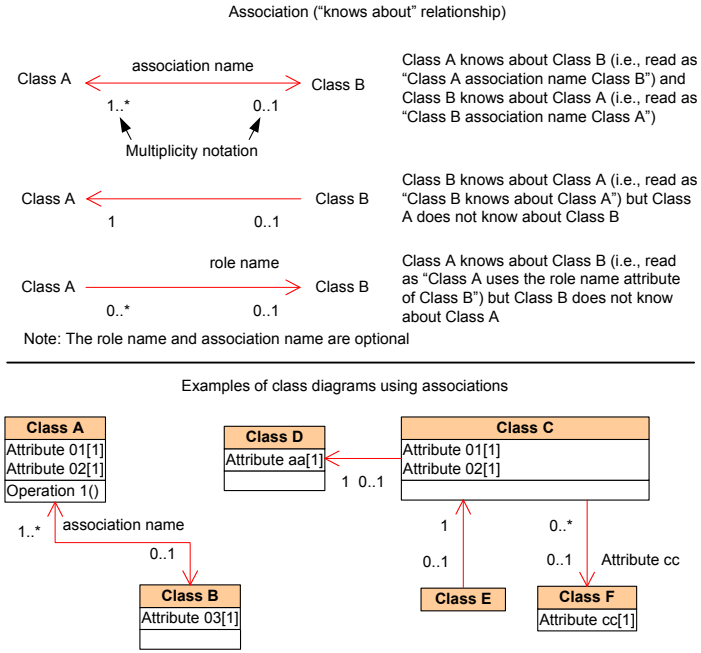


Figure 4 — Notation for association relationships for class diagrams

Page: 17

Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:19:21 PM
"Solid lines with arrowheads (see figure 4) are the notation..."
s/b
"...is the notation..." ["notation" meaning, "...any particular system of characters or symbols used to briefly express elements...", and this is my first choice]
or
"...are the notations..." ["notations" meaning, "...the characters or symbols used in such a system...]
One way or the other, the verb has to agree with the object (i.e., either "is the notation" or "are the notations").

Status
George Penokie Accepted 10/29/2007 10:25:26 AM
Author: George Penokie Subject: Note Date: 10/29/2007 10:25:18 AM
Changed to "is the notation"

Solid lines with diamonds ~~(see figure 5) are the notation~~ used to describe the aggregation relationship between classes in class diagrams. Multiplicity notation occurs at each end of the solid line.

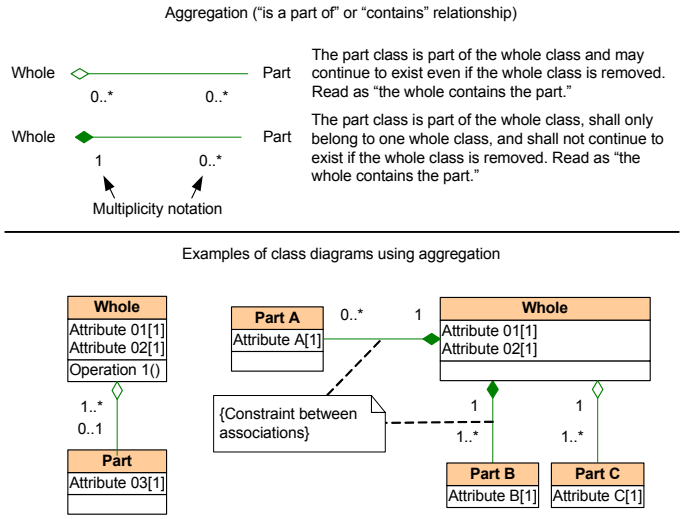


Figure 5 — Notation for aggregation relationships for class diagrams

Page: 18

Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:18:53 PM
"Solid lines with diamonds (see figure 5) are the notation..."
s/b
"...is the notation..." [see my earlier comment]

Status
George Penokie Accepted 10/29/2007 10:26:15 AM

Solid lines with triangles (see figure 6) are the notation used to describe the generalization relationship between classes in class diagrams.

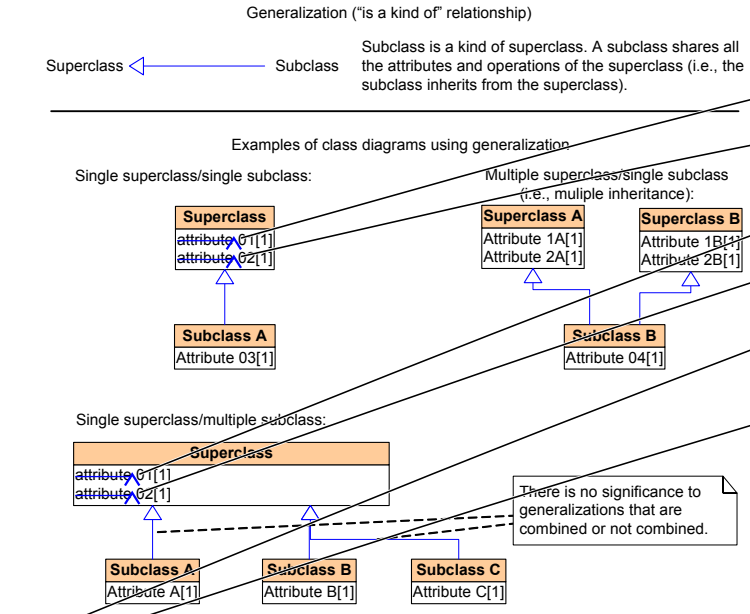


Figure 6 — Notation for generalization relationships for class diagrams

Dashed lines with arrowheads (see figure 7) are the notation used to describe the dependency relationship between classes in class diagrams.

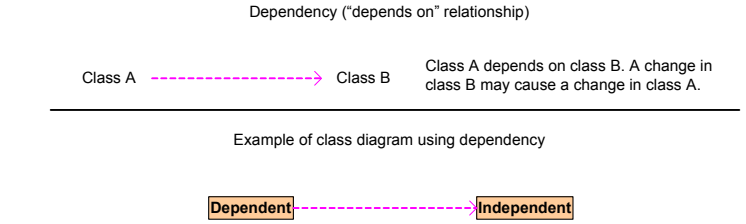


Figure 7 — Notation for dependency relationships for class diagrams

Page: 19

Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:18:31 PM
~~Solid lines with triangles (see figure 6) are the notation...~~"
s/b
"...is the notation..." [see my earlier comment]

Status
George Penokie Accepted 10/29/2007 10:26:34 AM
Author: suhlerp Subject: Replacement Text Date: 10/23/2007 1:08:48 PM
Attribute

Status
George Penokie Accepted 10/29/2007 10:28:16 AM
Author: suhlerp Subject: Replacement Text Date: 10/23/2007 1:08:57 PM
Attribute

Status
George Penokie Accepted 10/29/2007 10:28:21 AM
Author: suhlerp Subject: Replacement Text Date: 10/23/2007 1:09:18 PM
Attribute

Status
George Penokie Accepted 10/29/2007 10:28:26 AM
Author: suhlerp Subject: Replacement Text Date: 10/23/2007 1:09:23 PM
Attribute

Status
George Penokie Accepted 10/29/2007 10:28:34 AM
Author: suhlerp Subject: Sticky Note Date: 10/23/2007 1:12:43 PM
Page break before this paragraph. All the other notational elements start on a new page.

Status
George Penokie Rejected 10/29/2007 10:30:33 AM
Author: George Penokie Subject: Note Date: 10/29/2007 10:30:28 AM
The others are that way because it keeps the text with the figure. That is no need to do that as it fits on one pape.

Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:21:41 PM
~~Dashed lines with arrowheads (see figure 7) are the notation...~~
s/b
...is the notation... [see earlier comment]

Status
George Penokie Accepted 10/29/2007 10:31:17 AM

Page: 20

Author: suhlerp Subject: Sticky Note Date: 10/23/2007 1:19:54 PM
Does this mean that the value of the attribute is the set {x,y,z} ? Maybe an example would help.

Author: George Penokie Subject: Note Date: 10/29/2007 3:30:04 PM
Having multiple values for an attribute is not valid for an instance of a class. It has been deleted from the figure. Note this change was also made to the style guide.

Status
George Penokie Completed 10/29/2007 3:30:04 PM

Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:21:29 PM
"Solid lines (see figure 9) are the notation..."
s/b
"...is the notation..." [see my earlier comment]

Status
George Penokie Accepted 10/29/2007 3:30:47 PM

3.6.3 Object diagram conventions

Figure 8 shows the notation used for objects in object diagrams.

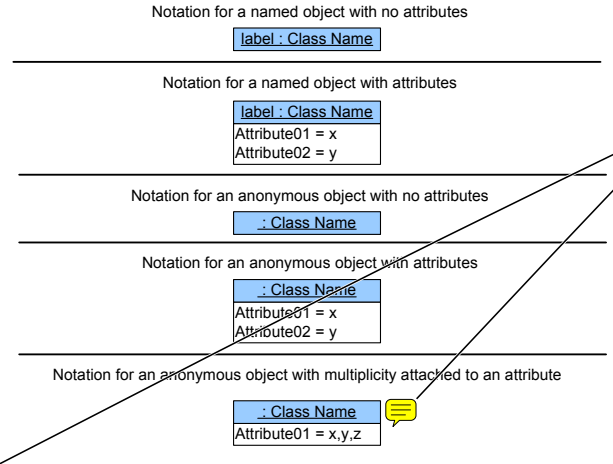


Figure 8 — Object diagram conventions

Solid lines (see figure 9) are the notation used to describe the link relationship between objects in object diagrams.

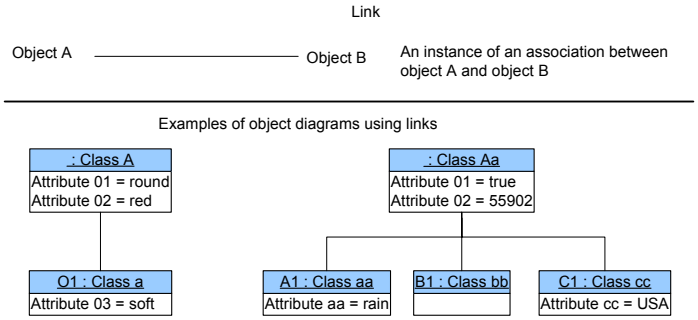


Figure 9 — Notation for link relationships for object diagrams

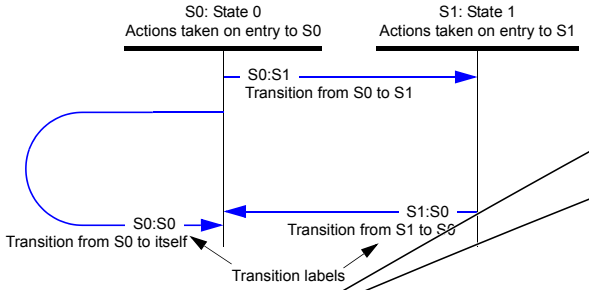


Figure 10 — Example state diagram

The state diagram is followed by a list of the state transitions, using the transition labels. Each transition is described in the list with particular attention to the conditions that cause the transition to occur and special conditions related to the transition. Using figure 10 as an example, the transition list might read as follows:

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

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

Transition S0:S0: This transition occurs when state S0 transitions to itself. The reason for a transition from S0 to itself is to specify that the actions taken whenever state S0 is entered are repeated every time the transition occurs.

A system specified in this manner has the following properties:

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

Page: 22

Author: Mark Evans, WDC Subject: Highlight Date: 10/24/2007 4:08:43 PM
"...transitions, using..."
s/b
"...transitions using..."

Status
George Penokie Accepted 10/29/2007 3:31:40 PM

Author: Mark Evans, WDC Subject: Highlight Date: 10/24/2007 4:09:35 PM
"Using figure 10 as an example, the transition list might read as follows:"
s/b
"Using figure 10 as an example, the transition list reads as follows:"

Status
George Penokie Accepted 10/29/2007 3:32:19 PM

Author: Mark Evans, WDC Subject: Highlight Date: 10/24/2007 5:14:51 PM
"State transitions are logically instantaneous;"
s/b
"Transitions from one state to another are instantaneous;"

Status
George Penokie Rejected 12/14/2007 10:35:31 AM -06'00'

Author: George Penokie Subject: Note Date: 12/14/2007 10:35:26 AM -06'00'
Change to << State transitions are instantaneous; >>

4 SCSI architecture model

4.1 Introduction

The purpose of the SCSI architecture model is to:

a) Provide a basis for the coordination of SCSI standards development that allows each standard to be placed into perspective within the overall SCSI architecture model;
b) Establish a layered model in which standards may be developed;
c) Provide a common reference for maintaining consistency among related standards; and
d) Provide the foundation for application compatibility across all SCSI interconnect and SCSI transport protocol environments by specifying generic requirements that apply uniformly to all implementation standards within each functional area.

The development of this standard is assisted by the use of an abstract model. To specify the external behavior of a SCSI system, elements in a system are replaced by functionally equivalent components within this model. Only externally observable behavior is retained as the standard of behavior. The description of internal behavior in this standard is provided only to support the definition of the observable aspects of the model. Those aspects are limited to the generic properties and characteristics needed for host applications to interoperate with SCSI devices in any SCSI interconnect and SCSI transport protocol environment. The model does not address other requirements that may be essential to some I/O system implementations (e.g., the mapping from SCSI device addresses to network addresses, the procedure for discovering SCSI devices on a network, and the definition of network authentication policies for SCSI initiator devices or SCSI target devices). These considerations are outside the scope of this standard.

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.

The SCSI architecture model is described in terms of classes (see 3.1.13), protocol layers, and service interfaces between classes. As used in this standard, classes are abstractions, encapsulating a set of related functions (i.e., attributes), operations, data types, and other classes. Certain classes are defined by SCSI (e.g., an interconnect), while others are needed to understand the functioning of SCSI but have implementation definitions outside the scope of SCSI (e.g., a task). These classes exhibit well-defined and observable behaviors, but they do not exist as separate physical elements. A class may contain a single attribute ~~(e.g., a task identifier)~~ or be a complex entity that may:

a) contain multiple attributes; or
b) perform a set of operations or services on behalf of another class.

Service interfaces are defined between distributed classes and protocol layers. The template for a distributed service interface is the client-server model described in 4.2. The structure of a SCSI I/O system is specified in 4.4 by defining the relationship among classes. The set of distributed services to be provided are specified in clause 5 and clause 7.

Requirements that apply to each SCSI transport protocol standard are specified in the SCSI transport protocol service model described in 5.4, 6.4, and 7.12. The model describes required behavior in terms of layers, classes within layers and SCSI transport protocol service transactions between layers.

4.2 The SCSI distributed service model

Service interfaces between distributed classes are represented by the client-server model shown in figure 11. Dashed horizontal lines with arrowheads denote a single request-response transaction as it appears to the client and server. The solid lines with arrowheads indicate the actual transaction path through a service delivery subsystem. In such a model, each client or server is a single thread of processing that runs concurrently with all other clients or servers.

Page: 23

Author: relliott Subject: Highlight Date: 10/27/2007 1:24:37 PM
The model does not address other requirements that may be essential to some I/O system implementations
s/b
The model does not address other requirements that are essential to some I/O system implementations

Status
George Penokie Rejected 12/14/2007 10:38:58 AM -06'00'
Author: George Penokie Subject: Note Date: 12/14/2007 10:43:05 AM -06'00'
The change fro a << may be >> to a << are >> implies a requirment where none should be.

Author: relliott Subject: Cross-Out Date: 10/8/2007 6:40:56 PM
Delete "(e.g., a task identifier)"

None of the classes include that as their single attribute.

Status
George Penokie Accepted 10/30/2007 12:03:00 PM

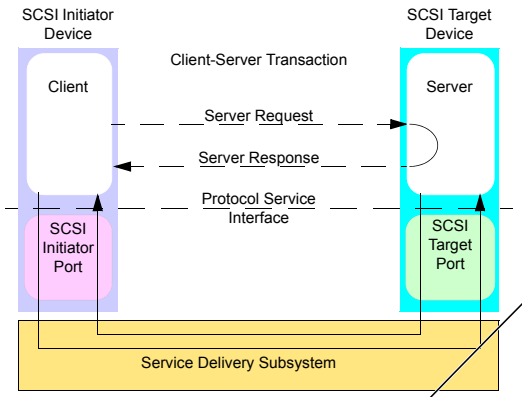


Figure 11 — Client-Server model

A client-server transaction is represented as a procedure call with inputs supplied by the caller (i.e., the client). The procedure call is processed by the server and returns outputs and a procedure call status. A client directs requests to a remote server via the SCSI initiator port and service delivery subsystem and receives a completion response or a failure notification. The request identifies the server and the service to be performed and includes the input data. The response conveys the output data and request status. A failure notification indicates that a condition has been detected (e.g., a reset or service delivery failure) that precludes request completion.

As seen by the client, a request becomes pending when it is passed to the SCSI initiator port for transmission. The request is complete when the server response is received or when a failure notification is sent. As seen by the server, the request becomes pending upon receipt and completes when the response is passed to the SCSI target port for return to the client. As a result there may be a time skew between the server and client's perception of request status and server state. ~~All references to a pending command or task management function in this standard are from the application client's point of view (see 6.5 and 7.11).~~

Client-server relationships are not symmetrical. A client may only originate requests for service. A server may only respond to such requests.

The client requests an operation provided by a server located in another SCSI device and waits for completion, which includes transmission of the request to and response from the remote server. From the client's point of view, the behavior of a service requested from another SCSI device is indistinguishable from a request processed in the same SCSI device. In this model, confirmation of successful request or response delivery by the sender is not required. The model assumes that delivery failures are detected by the SCSI initiator port or within a service delivery subsystem.

4.3 The SCSI client-server model

4.3.1 SCSI client-server model overview

As shown in figure 12, each SCSI target device provides services performed by device servers and task management functions performed by task managers. A logical unit is a class that implements one of the device functional models described in the SCSI command standards and processes commands (e.g., reading from or writing to the media). Each ~~pending~~ command defines a unit of work to be performed by the logical unit. Each

Page: 24

Author: George Penokie Subject: Cross-Out Date: 12/14/2007 10:15:39 AM -06'00'
All references to a pending command or task management function in this standard are from the application client's point of view (see 5.5 and 7.11).

Status
George Penokie Accepted 12/14/2007 10:17:57 AM -06'00'

Author: George Penokie Subject: Cross-Out Date: 12/11/2007 3:48:34 PM -06'00'
Delete the term << pending >> from SAM-4 as it is dated and no longer needed

Status
George Penokie Accepted 12/14/2007 10:18:21 AM -06'00'

Author: relliott Subject: Highlight Date: 10/8/2007 7:12:38 PM
A client may only originate requests for service. A server may only respond to such requests.
s/b
A client only originates requests for service. A server only responds to such requests.

Status
George Penokie Accepted 10/30/2007 12:04:58 PM

Author: George Penokie Subject: Cross-Out Date: 12/11/2007 3:49:02 PM -06'00'
Delete the term << pending >> from SAM-4 as it is dated and no longer needed

Status
George Penokie Accepted 12/14/2007 10:18:31 AM -06'00'

unit of work is represented within the SCSI target device by a task that may be externally referenced and controlled through requests issued to the task manager.

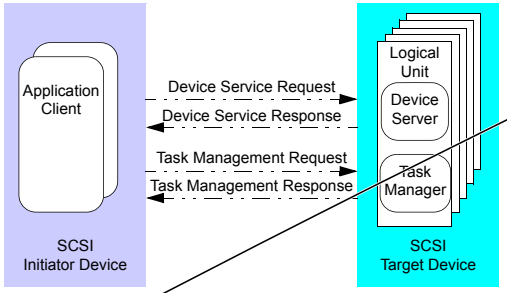


Figure 12 — SCSI client-server model

All requests originate from application clients residing within a SCSI initiator device. An application client is independent of the interconnect and SCSI transport protocol (e.g., an application client may correspond to the device driver and any other code within the operating system that is capable of managing I/O requests without requiring knowledge of the interconnect or SCSI transport protocol).

As described in 4.2, each request takes the form of a procedure call with arguments and a status to be returned. An application client may request processing of a command through a request directed to the device server within a logical unit. Each device service request contains a CDB defining the operation to be performed along with a list of command specific inputs and other parameters specifying how the command is to be processed.

4.3.2 Synchronizing client and server states

One way a client is informed of changes in server state is through the arrival of server responses. Such state changes occur after the server has sent the associated response and possibly before the response has been received by the SCSI initiator device (e.g., the SCSI target device changes state upon processing the **Send Command Complete** procedure call (see 5.4.2), but the SCSI initiator device is not informed of the state change until the **Command Complete Received** SCSI transport protocol service confirmation arrives).

SCSI transport protocols may require the SCSI target device to verify that the response has been received successfully before completing a state change. State changes controlled in this manner are said to be synchronized. Since synchronized state changes are not assumed or required by the architecture model, there may be a time lag between the occurrence of a state change within the SCSI target device and the SCSI initiator device's awareness of that change.

This standard assumes that state synchronization, if required by a SCSI transport protocol standard, is enforced by a service delivery subsystem transparently to the server (i.e., whenever the server invokes a SCSI transport protocol service to return a response as described in 7.12 and 5.4. It is assumed that the SCSI port for such a SCSI transport protocol does not return control to the server until the response has been successfully delivered to the SCSI initiator device).

4.3.3 Request/Response ordering

Request or response transactions are said to be in order if, relative to a given pair of sending and receiving SCSI ports, transactions are delivered in the order they were sent.

A sender may require control over the order in which its requests or responses are presented to the receiver (e.g., the sequence in which requests are received is often important whenever a SCSI initiator device issues a

Page: 25

Author: relliott Subject: Note Date: 10/28/2007 10:37:45 PM
The last paragraph in 4.3.1 discusses commands and device server requests. It should also discuss TMFs and task management requests, to cover everything shown in figure 12.

Author: George Penokie Subject: Note Date: 12/14/2007 10:55:26 AM -06'00'
An application client may request processing of a command or a task management function through a request directed to the device server within a logical unit. Device service requests are used to request the processing of commands (see clause 5) and task manager requests are used to request the processing of task management functions (see clause 7).

Author: relliott Subject: Highlight Date: 10/28/2007 10:40:39 PM
"SCSI initiator device" is not quite right.

The SCSI initiator device might deduce that a command was received by the target by noticing data transfer requests for that command. Some protocols explicitly mention that "implicit ACK."

It might be better to word this sentence with "application client", since although it is part of the initiator device it is not involved in the data transfer protocol services.

Status
George Penokie Completed 10/30/2007 12:09:15 PM

Author: Mark Evans, WDC Subject: Highlight Date: 10/24/2007 5:15:53 PM
"...the response has been received successfully..."
s/b
"...the response has been received without error..."

Status
George Penokie Accepted 10/30/2007 2:31:15 PM

Author: relliott Subject: Highlight Date: 10/28/2007 10:41:08 PM
architecture model
s/b
SCSI architecture model

Status
George Penokie Accepted 10/30/2007 2:32:19 PM

Author: Mark Evans, WDC Subject: Highlight Date: 10/24/2007 5:17:28 PM
"...until the response has been successfully delivered..."
s/b
"...until the response has been delivered without error..."

Status
George Penokie Accepted 10/30/2007 2:31:50 PM

series of commands with the ORDERED task attribute to a logical unit as described in clause 8). In this case, the order in which these commands are completed, and hence the final state of the logical unit, may depend on the order in which these commands are received. The SCSI initiator device may develop knowledge about the state of ~~pending~~ commands and task management functions and may take action based on the nature and sequence of SCSI target device responses (e.g., a SCSI initiator device should be aware that further responses are possible from an aborted command because the command completion response may be delivered out of order with respect to the abort response).

The manner in which ordering constraints are established is vendor specific. An implementation may delegate this responsibility to the application client (e.g., the device driver). In-order delivery may be an intrinsic property of a service delivery subsystem or a requirement established by the SCSI transport protocol standard.

The order in which task management requests are processed is not specified by the SCSI architecture model. The SCSI architecture model does not require in-order delivery of such requests or processing by the task manager in the order received. To guarantee the processing order of task management requests referencing a specific logical unit, an application client should not have more than one such request pending to that logical unit.

To simplify the description of behavior, the SCSI architecture model assumes in-order delivery of requests or responses to be a property of a service delivery subsystem. This assumption does not constitute a requirement. The SCSI architecture model makes no assumption about and places no requirement on the ordering of requests or responses for different I_T nexuses.

4.4 The SCSI structural model

The SCSI structural model represents a view of the classes in a SCSI I/O system as seen by the application clients interacting with the system. As shown in figure 13, the fundamental class is the SCSI domain that represents an I/O system. A SCSI domain is made up of SCSI devices and a service delivery subsystem that transports commands, data, task management functions, and related information. A SCSI device contains clients or servers or both and the infrastructure to support them.

Page: 26

Author: George Penokie Subject: Cross-Out Date: 12/11/2007 3:49:29 PM -06'00'
Delete the term << pending >> from SAM-4 as it is dated and no longer needed

Status
George Penokie Accepted 12/14/2007 10:18:49 AM -06'00'

Author: relliott Subject: Highlight Date: 10/8/2007 6:56:09 PM
SCSI domain
s/b
SCSI Domain class

Status
George Penokie Accepted 10/30/2007 2:33:08 PM

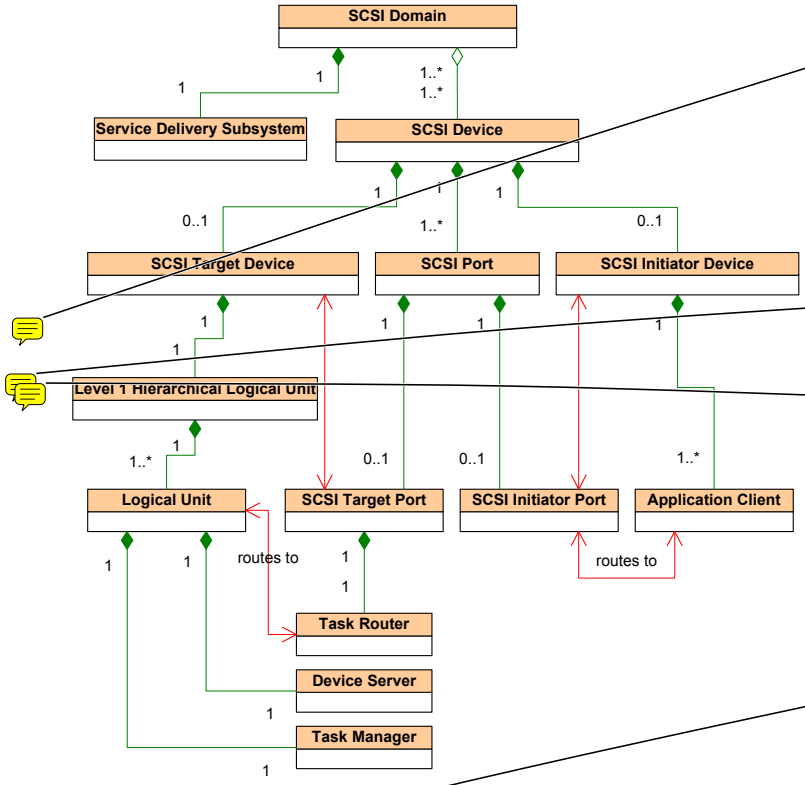


Figure 14 — SCSI Domain class diagram overview

4.5.2 SCSI Domain class

The SCSI Domain class (figure 15) contains the:

a) Service Delivery Subsystem class (see 4.5.3); and
b) SCSI Device class (see 4.5.4).

Page: 28

Author: relliott Subject: Note Date: 10/29/2007 9:34:12 AM
The implication that a logical unit is contained within only one SCSI target device may be too tight.

1. The way hierarchical logical units are currently modeled, a logical unit is part of its real SCSI target device and also part of each of the SCSI target devices that route to it. (A separate comment suggests removing hierarchical logical units from the model)

2. With virtualization (e.g., RAID), some logical units (e.g., the physical disk drives) are used by a "higher level" logical unit (e.g., a RAID-5 volume). There is some interaction between the states of the higher and lower level logical units.

3. With remote replication, a logical unit can be in two different places at the same time. The media is essentially synchronized (writes to New York are immediately picked up by reads in Los Angeles). However, the task set states are not (an ABORT TASK SET in New York doesn't abort commands pending in Los Angeles).

To acknowledge these oddities, perhaps add a statement like "The medium accessed by a logical unit may not be exclusively accessible through that logical unit."

An optional containment relationship from logical unit to logical unit might represent the more complex interactions.

Status
George Penokie Rejected 12/14/2007 11:25:59 AM -06'00'
Author: George Penokie Subject: Note Date: 12/14/2007 11:25:53 AM -06'00'
Place in section 4.5.19.1 << Data contained within a logical unit may be duplicated in whole or part on a different logical unit. The synchronization of that data between multiple logical units is outside the scope of this standard. >>

Author: suhlerp Subject: Sticky Note Date: 10/23/2007 4:54:35 PM
[Technical]
Both ADC-2 and SSC-2 & -3 include a physical device as part of the SCSI target device. If you wish, I could provide a proposal to add this. Otherwise, it could wait for SAM-5.

Status
George Penokie Rejected 10/30/2007 2:35:33 PM
Author: George Penokie Subject: Note Date: 10/30/2007 2:35:28 PM
That should be a SAM-5 discusion.

Author: relliott Subject: Note Date: 10/29/2007 9:56:10 AM
The SCSI Device to SCSI Target Device/SCSI Initiator Device relationship should be an inheritance relationship, not an aggregation relationship. Same for SCSI Port to SCSI Initiator Port/SCSI Target Port. This would be a proper use of multiple inheritance.

Comment from someone experienced with UML at HP:
"When they start showing that ports "contain" target and initiator ports, it seems to me that they really are describing inheritance. If that is true, they are mixing inheritance (a port shouldn't really be a SCSI device so much as a Network device anyway so that SCSI can go over any topology) and containment concepts in the same UML by overloading the aggregation symbol to include inheritance.

It is probably going to be hard for UML people to decipher. It is not at all consistent with UML for SCSI management (read SMI-S here), so if they are planning on representing topologies or developing a data model for management or as part of the protocol with this then I am really concerned."

Status
George Penokie Rejected 10/30/2007 2:48:26 PM
Author: George Penokie Subject: Note Date: 10/30/2007 2:48:19 PM
The problem with making the SCSI target device and the SCSI initiator device a children of the SCSI device is that you would now have to create another class called a SCSI initiator/target device. By using containment that is not required. We went to great lengths to eliminate initiator/target things and I have no intention of putting them back in. The same is true for the SCSI ports. This is the model that represents SCSI as it it today, shifting it around to make it look like some other interconnect is not a good idea. As far as the SMI-S is concerned, I have seen what they call UML and I would suggest what they call UML is not very closely related to what is in the UML standards.

Author: relliott Subject: Highlight Date: 10/8/2007 6:53:52 PM
SCSI Domain class (figure 15)
s/b
SCSI Domain class (see figure 15)

Status
George Penokie Accepted 10/30/2007 3:42:44 PM

Page: 29

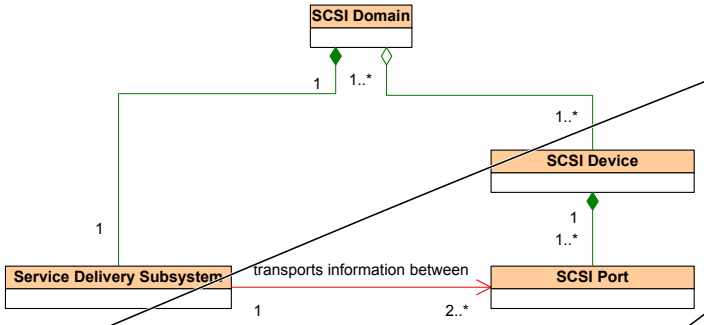


Figure 15 — SCSI Domain class diagram

Each instance of a SCSI Domain class shall contain the following objects:

a) one service delivery subsystem;
b) one or more SCSI devices; and
c) one or more SCSI ports.

See figure 16 for the instantiation of the minimum set of objects that make up a valid SCSI domain.

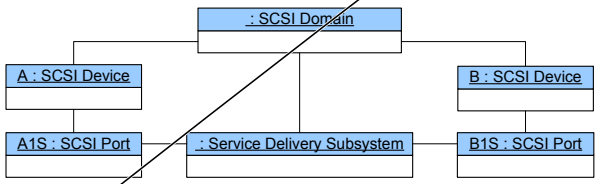


Figure 16 — SCSI domain object diagram

The boundaries of a SCSI domain are established by the system implementor, within the constraints of a specific SCSI transport protocol and associated interconnect standards.

4.5.3 Service Delivery Subsystem class

A Service Delivery Subsystem class (see figure 15) connects all the SCSI ports (see 3.1.98) in the SCSI domain, providing a mechanism through which application clients communicate with device servers and task managers.

A service delivery subsystem is composed of one or more interconnects that appear to a client or server as a single path for the transfer of requests and responses between SCSI devices.

A service delivery subsystem is assumed to provide error-free transmission of requests and responses between client and server. Although a device driver in a SCSI implementation may perform these transfers through

Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:21:01 PM
"Each instance of a SCSI Domain class shall contain the following objects:
a) one service delivery subsystem;
b) one or more SCSI devices; and
c) one or more SCSI ports."
I think there shall be two or more SCSI devices and two or more SCSI ports, as shown in figure 16.

Status
George Penokie Rejected 10/30/2007 3:47:54 PM
Author: George Penokie Subject: Note Date: 10/30/2007 3:47:47 PM
That's not correct. A SCSI device is allowed to contain both an initiator and a target. Also a SCSI port is allowed to contain an initiator port and a target port. So given that it's one or more SCSI devices and one or more SCSI ports. The instance diagram is only one example of what it could look like. The could be an instance diagram that showed only one SCSI device and one SCSI port but that would not be very interesting.

Author: relliott Subject: Highlight Date: 10/8/2007 6:54:20 PM
A Service Delivery Subsystem class
s/b
The Service Delivery Subsystem class

Status
George Penokie Accepted 10/30/2007 3:48:33 PM

several interactions with its STPL, the architecture model ~~portrays each operation~~ from the viewpoint of the application client, as occurring in one discrete step. The request or response is:

a) considered sent by the sender when the sender passes it to the SCSI port for transmission;
b) in transit until delivered; and
c) considered received by the receiver when it has been forwarded to the receiver via the destination SCSI device's SCSI port.

4.5.4 SCSI Device class

4.5.4.1 SCSI Device class ~~overview~~

See figure 17 for the SCSI Device class diagram.

The SCSI Device class contains the:

a) SCSI Port class (see 4.5.5); and
b) SCSI Initiator Device class (see 4.5.9), the SCSI Target Device class (see 4.5.14), or both.

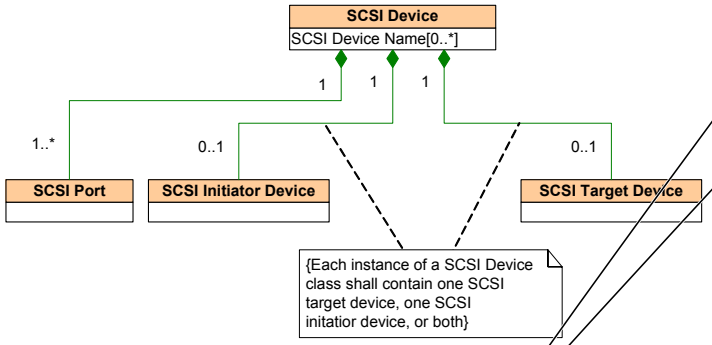


Figure 17 — SCSI Device class diagram

Each instance of a SCSI Device class shall contain:

a) one or more SCSI ports; and
b) one SCSI target device, one SCSI initiator device, or both.

4.5.4.2 SCSI Device Name attribute

The SCSI Device Name attribute contains a name (see 3.1.68) for a SCSI device that is world wide unique within the SCSI transport protocol of each SCSI domain in which the SCSI device has SCSI ports. For each supported SCSI transport protocol, a SCSI device shall have no more than one (i.e., zero or one) SCSI Device Name attribute that is not in the SCSI name string format (see SPC-4). A SCSI device shall have no more than one (i.e., zero or one) SCSI Device Name attribute in the SCSI name string format regardless of the number of SCSI transport protocols supported by the SCSI device. If a SCSI device has a SCSI Device Name attribute in the SCSI name string format then the SCSI device should have only one SCSI Device Name attribute. A SCSI device name shall never change and may be used to ~~persistently~~ identify a SCSI device in contexts where specific references to port names or port identifiers is not required.

A SCSI transport protocol standard may require that a SCSI device include a SCSI Device Name attribute if the SCSI device has SCSI ports in a SCSI domain of that SCSI transport protocol. The SCSI Device Name attribute

Page: 30

Author: relliott Subject: Highlight Date: 10/28/2007 10:44:44 PM
architecture model
s/b
SCSI architecture model

Status
George Penokie Accepted 10/30/2007 3:49:10 PM

Author: relliott Subject: Highlight Date: 10/8/2007 6:53:35 PM
"See figure 17 for the SCSI Device class diagram.
The SCSI Device class"
s/b
"The SCSI Device class (see figure 17)"

Status
George Penokie Accepted 10/30/2007 3:51:18 PM

Author: Mark Evans, WDC Subject: Cross-Out Date: 10/24/2007 5:18:54 PM
Delete "persistently".

Status
George Penokie Rejected 12/14/2007 11:36:18 AM -06'00'

Author: George Penokie Subject: Note Date: 12/14/2007 11:34:46 AM -06'00'
A SCSI device name shall never change and may be used for persistent identification of a SCSI device in contexts where specific references to port names or port identifiers is not required.

Author: relliott Subject: Note Date: 12/19/2007 4:13:14 PM -06'00'
Add this section the following << The SCSI device name for a SCSI target device may be reported in a target device name designation descriptor in the Device Identification VPD page (see SPC-4). The SCSI device name for a SCSI initiator device is reported by methods outside the scope of this standard. >>

Status
George Penokie Accepted 12/19/2007 4:12:44 PM -06'00'

target device may assign relative port identifiers to its SCSI target ports and any SCSI initiator ports. If relative port identifiers are assigned, the SCSI target device shall assign each of its SCSI target ports and any SCSI initiator ports a unique relative port identifier from 1 to 65 535. SCSI target ports and SCSI initiator ports share the same number space.

Relative port identifiers may be retrieved through the Device Identification VPD page (see SPC-4) and the SCSI Ports VPD page (see SPC-4).

The relative port identifiers are not required to be contiguous. The relative port identifier for a SCSI port shall not change once assigned unless physical reconfiguration of the SCSI target device occurs.

4.5.6 SCSI Target Port class

4.5.6.1 SCSI Target Port class overview

The SCSI Target Port class (see figure 18) contains the:

a) Task Router class (see 4.5.8);

The SCSI Target Port class connects SCSI target devices to a service delivery subsystem.

The SCSI Target Port class processes the:

a) Send Data-in operation (see 5.4.3.2.1) to send data to the service delivery subsystem;
b) Receive Data-out operation (see 5.4.3.3.1) to receive data from the service delivery subsystem;
c) Terminate Data Transfer operation (see 5.4.3.4) to terminate data transfers;
d) Send Command Complete operation (see 5.4.2.4) to transmit a command complete indication to the service delivery subsystem; and
e) Task Management Function Executed operation (see 7.12.4) to transmit a task management function executed indication to the service delivery subsystem.

4.5.6.2 Target Port Identifier attribute

The Target Port Identifier attribute contains a target port identifier (see 3.1.117) for a SCSI target port. The target port identifier is a value by which a SCSI target port is referenced within a domain.

4.5.6.3 Target Port Name attribute

A Target Port Name attribute contains an optional name (see 3.1.68) of a SCSI target port that is world wide unique within the SCSI transport protocol of the SCSI domain of that SCSI target port. A SCSI target port may have at most one name. A SCSI target port name shall never change and may be used to persistently identify the SCSI target port.

A SCSI transport protocol standard may require that a SCSI target port include a SCSI target port name if the SCSI target port is in a SCSI domain of that SCSI transport protocol. The SCSI target port name may be made available to other SCSI devices or SCSI ports in the given SCSI domain in SCSI transport protocol specific ways.

4.5.7 SCSI Initiator Port class

4.5.7.1 SCSI Initiator Port class overview

The SCSI Initiator Port class connects SCSI initiator devices to a service delivery subsystem.

The SCSI Initiator Port (see figure 18) class processes the:

a) Send SCSI Command operation (see 5.4.2.2) to send a SCSI command to the service delivery subsystem; and
b) Send Task Management Request operation (see 7.12.2) to send a task management request to the service delivery subsystem.

Page: 32

Author: relliott Subject: Highlight Date: 12/14/2007 11:38:05 AM -06'00'
Relative port identifiers may be retrieved through the Device Identification VPD page (see SPC-4) and the SCSI Ports VPD page (see SPC-4).
s/b
The Device Identification VPD page (see SPC-4) and the SCSI Ports VPD page (see SPC-4) report relative port identifiers.

Status
George Penokie Accepted 12/14/2007 11:38:41 AM -06'00'

Author: relliott Subject: Highlight Date: 10/8/2007 6:57:01 PM
; s/b .

Status
George Penokie Accepted 10/30/2007 3:54:55 PM

Author: relliott Subject: Note Date: 12/19/2007 4:03:37 PM -06'00'
Add to this section the following << The target port identifer may be reported in a target port name designation descriptor in the Device Identification VPD page (see SPC-4). If a SCSI target port has a target port identifier and a target port name see SPC-4 to determine which is reported. >>

Status
George Penokie Accepted 12/19/2007 4:01:26 PM -06'00'

Author: relliott Subject: Highlight Date: 10/8/2007 7:14:45 PM
A SCSI target port may have at most one name.
s/b
A SCSI target port shall have at most one name.

Status
George Penokie Rejected 12/14/2007 11:47:52 AM -06'00'

Author: George Penokie Subject: Note Date: 12/14/2007 11:47:43 AM -06'00'
Changed to << may be used for persistent identification of a
>>

Author: Mark Evans, WDC Subject: Cross-Out Date: 10/24/2007 5:19:08 PM
Delete "persistently".

Author: George Penokie Subject: Note Date: 12/14/2007 11:50:56 AM -06'00'
Changed to << may be used for persistent identification of a
>>

Author: relliott Subject: Note Date: 12/19/2007 4:03:45 PM -06'00'
Add to this section the following << The target port name may be reported in a target port name designation descriptor in the Device Identification VPD page (see SPC-4). If a SCSI target port has a target port identifier and a target port name see SPC-4 to determine which is reported.

Status
George Penokie Accepted 12/19/2007 4:02:45 PM -06'00'

Author: relliott Subject: Highlight Date: 10/8/2007 6:50:37 PM
SCSI Initiator Port class
s/b
SCSI Initiator Port class (see figure 18)

Status
George Penokie Accepted 10/30/2007 3:57:59 PM

Author: relliott Subject: Highlight Date: 10/8/2007 6:50:59 PM
SCSI Initiator Port (see figure 18) class
s/b
SCSI Initiator Port class

Status
George Penokie Accepted 10/30/2007 3:58:10 PM

4.5.7.2 Initiator Port Identifier attribute

The Initiator Port Identifier attribute contains the initiator port identifier for a SCSI initiator port. The initiator port identifier is a value by which a SCSI initiator port is referenced within a domain.

4.5.7.3 Initiator Port Name attribute

A Initiator Port Name attribute contains an optional name (see 3.1.68) of a SCSI initiator port that is world wide unique within the SCSI transport protocol of the SCSI domain of that SCSI initiator port. A SCSI initiator port may have at most one name. A SCSI initiator port name shall never change and may be used to ~~persistently~~ identify the SCSI initiator port.

A SCSI transport protocol standard may require that a SCSI initiator port include a SCSI initiator port name if the SCSI initiator port is in a SCSI domain of that SCSI transport protocol. The SCSI initiator port name may be made available to other SCSI devices or SCSI ports in the given SCSI domain in SCSI transport protocol specific ways.

4.5.8 Task Router class

The Task Router class (see figure 18) routes:

a) task management functions between a task manager and a service delivery subsystem by processing the Route Task operation; and
b) commands between a logical unit task manager and a service delivery subsystem by processing the Route Task operation.

The task router routes commands and task management functions as follows:

a) commands addressed to a valid logical unit are routed to the task manager in the specified logical unit;
b) commands addressed to an incorrect logical unit are handled as described in 5.8.4;
c) task management functions with I_T_L nexus scope (e.g., ABORT TASK SET, CLEAR TASK SET, CLEAR ACA, LOGICAL UNIT RESET, QUERY TASK SET, and QUERY UNIT ATTENTION) or I_T_L_Q nexus scope (e.g., ABORT TASK and QUERY TASK) addressed to a valid logical unit are routed to the task manager in the specified logical unit;
d) task management functions with an I_T nexus scope (e.g., I_T NEXUS RESET) are routed to the task manager in each logical unit about which the task router knows; and
e) task management functions with I_T_L nexus scope or I_T_L_Q nexus scope addressed to an incorrect logical unit are handled as described in 7.12.

In some transport protocols, the task router may check for overlapped task identifiers on commands (see 5.8.3).

4.5.9 SCSI Initiator Device class

A SCSI Initiator Device class (see figure 19) is a SCSI Device class that contains the:

a) Application Client class (see 4.5.10).

Page: 33

Author: relliott Subject: Note Date: 12/19/2007 4:17:26 PM -06'00'
Add to this section the following << The initiator port identifier is reported by methods outside the scope of this standard. >>

Status
George Penokie Accepted 12/19/2007 4:16:58 PM -06'00'
Author: relliott Subject: Highlight Date: 10/8/2007 7:15:16 PM
A SCSI initiator port may have at most one name.
s/b
A SCSI initiator port shall have at most one name.

Status
George Penokie Completed 10/30/2007 3:58:44 PM
Author: Mark Evans, WDC Subject: Cross-Out Date: 10/24/2007 5:19:14 PM
Delete "persistently".

Status
George Penokie Accepted 12/14/2007 11:48:28 AM -06'00'
Author: George Penokie Subject: Note Date: 12/14/2007 11:48:24 AM -06'00'
Changed to << may be used for persistent identification of a
>>

Author: relliott Subject: Note Date: 12/19/2007 4:18:35 PM -06'00'
Add to this section the following << The initiator port name is reported by methods outside the scope of this standard. >>

Status
George Penokie Accepted 12/19/2007 4:18:17 PM -06'00'
Author: suhlerp Subject: Sticky Note Date: 10/23/2007 5:18:11 PM
Why "logical unit task manager"? Why not "task manager," as in a) above? This is the only use of "logical unit task manager."

Status
George Penokie Accepted 10/30/2007 4:00:31 PM
Author: George Penokie Subject: Note Date: 10/30/2007 4:00:28 PM
Deleted "logical unit"

Author: relliott Subject: Highlight Date: 10/29/2007 11:15:01 AM
incorrect logical unit

There is no such thing as an incorrect logical unit, just incorrect logical unit numbers.

Status
George Penokie Rejected 12/26/2007 10:56:06 AM -06'00'
Author: George Penokie Subject: Note Date: 12/26/2007 10:56:00 AM -06'00'
Add to glossary: Incorrect logical unit number and Incorrect logical unit:

Status
George Penokie Rejected 12/26/2007 10:56:06 AM -06'00'
Author: relliott Subject: Highlight Date: 10/29/2007 11:15:51 AM
incorrect logical unit

There is no such thing as an incorrect logical unit, just incorrect logical unit numbers.

Status
George Penokie Rejected 12/26/2007 10:56:31 AM -06'00'
Author: George Penokie Subject: Note Date: 12/26/2007 10:56:24 AM -06'00'
Add to glossary: Incorrect logical unit number and Incorrect logical unit:

Page: 34

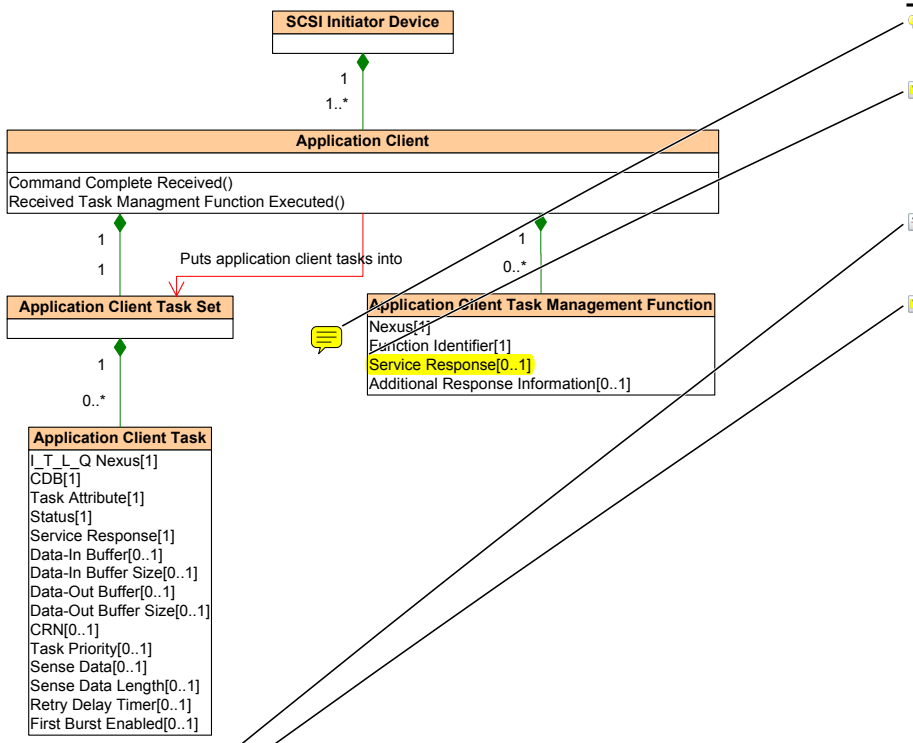


Figure 13 — SCSI Initiator Device class diagram

Each instance of a SCSI Initiator Device class shall contain the following objects:

a) one or more application clients that contain:
   A) zero or more application client task management functions; and
   B) one application client task set.

4.5.10 Application Client class

An Application Client class (see figure 19) contains the:

originates commands by issuing **Send SCSI Command** requests (see 5.4.2).

a) Application Client Task Management Function class (see 4.5.11); and
b) Application Client Task Set class (see 4.5.12).

Author: relliott Subject: Note Date: 10/8/2007 6:53:04 PM
Nexus and Function Identifier should be swapped in figure 19 and in sections 4.5.11.2 and 4.5.11.3 to match the SCSI Target Device side in figure 22 and following sections. (or the target side should be swapped to follow this)

Status
George Penokie Accepted 10/30/2007 4:10:10 PM

Author: relliott Subject: Highlight Date: 10/27/2007 2:12:55 PM
Service Response[0..1]
s/b
Service Response[1]

like in Application Client Task

(also see comment on text removing "if any")

Status
George Penokie Accepted 10/30/2007 4:10:22 PM

Author: Mark Evans, WDC Subject: Cross-Out Date: 10/24/2007 5:20:44 PM
Delete "originates commands by issuing Send SCSI Command requests (see 5.4.2)."
s/b
I don't know where this goes, but it doesn't go here.

Status
George Penokie Accepted 10/30/2007 4:12:48 PM

Author: Emulex Subject: Highlight Date: 10/30/2007 1:44:24 PM
Emulex-009
Page: 34 4.5.10 second paragraph: This sentence fragment seems to be out of place.

Status
George Penokie Accepted 10/30/2007 4:13:09 PM

Author: George Penokie Subject: Note Date: 10/30/2007 4:13:06 PM
It has been deleted

The Application Client class processes the:

a) Command Complete Received operation (see 5.4.2.5) to determine when a requested command has completed; and
b) Received Task Management Function Executed operation (see 7.12.5) to determine when a requested task management function has completed.

The Application Client class originates a command by issuing a Send SCSI Command request (see 5.4.2.2). Issuing the Send SCSI Command request causes the Initiator Port class to create an Application Client Task object that is placed into the application client task set. The Application Client Task object remains in the application client task set until the application client determines when a command that it has originated completes using the command lifetime (see 5.5), including the processing of a Command Complete Received operation.

The Application Client class originates a task management request by issuing a Send Task Management request (see 7.12.2). An Application Client class determines when a task management request that it has originated completes using the task management function lifetime information (see 7.11), including the processing of a Received Task Management Function Executed operation.

The application client may request processing of a task management function for:

a) a logical unit through a request directed to the task manager within the logical unit; or
b) all logical units known by a task router through a request directed to the task router within the target port.

The interactions between the task manager, or a task router, and application client when a task management request is processed are shown in 7.13.

4.5.11 Application Client Task Management Function class

4.5.11.1 Application Client Task Management Function class overview

The Application Client Task Management Function class represents a SCSI task management function (see clause 7).

4.5.11.2 Nexus attribute

The Nexus attribute contains the nexus affected by the task management function (see 4.7).

4.5.11.3 Function Identifier attribute

The Function Identifier attribute contains function identifier (see clause 7.12).

4.5.11.4 Service Response attribute

The Service Response attribute, if any, contains the service response (see clause 7).

4.5.11.5 Additional Response Information attribute

The Additional Response Information attribute, if any, contains any additional response information for the task management function (see clause 7).

4.5.12 Application Client Task Set class

The Application Client Task Set class (see figure 19) contains the:

a) Application Client Task class (see 4.5.13).

Each instance of an Application Client Task Set class shall contain the following objects:

a) zero or more application client tasks.

Page: 35

Author: Emulex Subject: Highlight Date: 10/30/2007 1:45:39 PM
Emulex-010
Page: 35 Paragraph after second a-b list "The interactions between the task manager, or a task router, &" remove the first comma.

Status
George Penokie Accepted 10/30/2007 4:13:54 PM

Author: relliott Subject: Highlight Date: 10/8/2007 6:49:43 PM
Application Client Task Management Function class
s/b
Application Client Task Management Function class (see figure 19)

Status
George Penokie Accepted 10/30/2007 4:15:02 PM

Author: Emulex Subject: Highlight Date: 10/30/2007 4:15:16 PM
Emulex-011
Page: 35 4.5.11.3 "The Function Identifier attribute contains function identifier" s/b "The Function Identifier attribute contains a function identifier"

Status
George Penokie Accepted 10/30/2007 4:17:01 PM

Author: relliott Subject: Highlight Date: 10/27/2007 2:12:42 PM
Service Response attribute, if any,
s/b
Service Response attribute

(also see comment on table changing [0..1] to [1])

Status
George Penokie Accepted 10/30/2007 4:17:36 PM

The interactions among the application client tasks in an application client task set are not specified in this standard.

4.5.13 Application Client Task class

4.5.13.1 Application Client Task class overview

The Application Client Task class represents the work associated with a command (see clause 5). A new command causes the creation of an application client task. The application client task persists until a task complete response is sent or until the task is ended by a task management function or exception condition. For an example of the processing for a command see 5.7.

4.5.13.2 I_T_L_Q Nexus attribute

The I_T_L_Q Nexus attribute contains the I_T_L_Q nexus of the task (see 4.7).

4.5.13.3 CDB attribute

The CDB attribute contains a CDB (see 5.2 and SPC-3) that defines the work to be performed by a logical unit.

4.5.13.4 Task Attribute attribute

The Task Attribute attribute (see 8.6) contains the task attribute (e.g., SIMPLE task attribute, ORDERED task attribute, HEAD OF QUEUE task attribute, ACA task attribute) of a command.

4.5.13.5 Status attribute

The Status attribute contains the status of the completed command (see 5.3).

4.5.13.6 Service Response attribute

The Service Response attribute contains the service response for the completed command (see 5.4.2.5).

4.5.13.7 Data-In Buffer attribute

The Data-In Buffer attribute, if any, contains the Data-In Buffer argument from an Execute Command procedure call (see 5.1).

4.5.13.8 Data-In Buffer Size attribute

The Data-In Buffer Size attribute, if any, contains the Data-In Buffer Size argument from an Execute Command procedure call (see 5.1).

4.5.13.9 Data-Out Buffer attribute

The Data-Out Buffer attribute, if any, contains the Data-Out Buffer argument from an Execute Command procedure call (see 5.1).

4.5.13.10 Data-Out Buffer size attribute

The Data-Out Buffer Size attribute, if any, contains the Data-Out Buffer Size argument from an Execute Command procedure call (see 5.1).

4.5.13.11 CRN attribute

The CRN attribute, if any, contains the CRN of the command (see 5.4.2.2).

Page: 36

Author: relliott Subject: Highlight Date: 10/8/2007 6:49:27 PM
Application Client Task class
s/b
Application Client Task class (see figure 19)

Status
George Penokie Accepted 10/30/2007 4:18:14 PM
Author: Mark Evans, WDC Subject: Highlight Date: 10/25/2007 9:50:14 AM
"The application client task persists until a task complete response is sent..."
s/b
"The application client task persists until a task complete response is received..."

Status
George Penokie Accepted 10/30/2007 4:23:43 PM
Author: relliott Subject: Note Date: 10/30/2007 4:26:20 PM
All the "output" attribute sections 4.5.13.2 to 4.5.13.12 and 4.5.13.19 should cross reference 5.4.2.2 (Send SCSI Command), just like the "input" attribute sections (4.5.13.13 to 4.5.13.16) reference 5.4.2.5 (Command Complete Received).

Status
George Penokie Accepted 10/30/2007 4:44:57 PM
Author: relliott Subject: Highlight Date: 10/29/2007 10:13:03 AM
SPC-3
s/b
SPC-4

or delete this reference and just refer to 5.2 alone

Status
George Penokie Accepted 10/30/2007 4:47:29 PM
Author: George Penokie Subject: Note Date: 10/30/2007 4:47:26 PM
Changed to SPC-4.

Author: relliott Subject: Highlight Date: 10/27/2007 1:27:49 PM
(e.g., SIMPLE task attribute, ORDERED task
attribute, HEAD OF QUEUE task attribute, ACA task attribute)
s/b
(e.g., SIMPLE, ORDERED, HEAD OF QUEUE, or ACA)

Status
George Penokie Rejected 10/30/2007 4:49:48 PM
Author: George Penokie Subject: Note Date: 10/30/2007 4:49:44 PM
The correct name as used in the rest of the standard is xxx task attribute. So no change made.

4.5.13.12 Task Priority attribute

The Task Priority attribute, if any, contains the priority of the command (see 8.7).

4.5.13.13 Sense Data attribute

The Sense Data attribute, if any, contains the sense data for the completed command (see 5.4.2.5).

4.5.13.14 Sense Data Length attribute

The Sense Data Length attribute, if any, contains the length of the sense data for the completed command (see 5.4.2.5).

4.5.13.15 Retry Delay Timer attribute

The Retry Delay Timer attribute, if any, contains the retry delay time for the completed command (see 5.4.2.5).

4.5.13.16 First Burst Enabled attribute

The First Burst Enabled attribute, if any, specifies that first burst for the command is enabled (see 5.4.2.2).

4.5.14 SCSI Target Device class

The SCSI Target Device class (see figure 20) is a SCSI Device class that contains the:

a) Level 1 Hierarchical Logical Unit class (see 4.5.15).

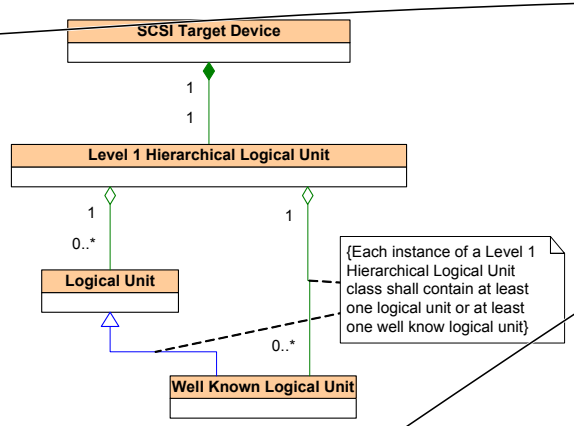


Figure 20 — SCSI Target Device class diagram

Each instance of the SCSI Target Device class shall contain the following objects:

a) one level 1 hierarchical logical unit that contains:
   A) at least one logical unit or well known logical unit;
   B) zero or more logical units; and

Page: 37

Author: relliott Subject: Highlight Date: 10/29/2007 9:59:30 AM
command (see 5.4.2.5).
command (see 5.8.6 and 5.4.2.5).

Status
George Penokie Accepted 10/30/2007 4:52:02 PM
Author: George Penokie Subject: Note Date: 10/30/2007 4:51:57 PM
Reference placed right after the term "sense data"

Author: relliott Subject: Highlight Date: 10/29/2007 9:59:10 AM
command (see 5.4.2.5).
s/b
command (see 5.8.6 and 5.4.2.5)

Status
George Penokie Accepted 10/30/2007 4:52:40 PM
Author: George Penokie Subject: Note Date: 10/30/2007 4:52:23 PM
Reference placed right after the term "sense data"

Author: relliott Subject: Highlight Date: 10/29/2007 9:58:14 AM
the retry delay time for the completed command (see 5.4.2.5)

s/b
the additional status information for the completed command (see 5.3.2 and 5.4.2.5).

Status
George Penokie Accepted 12/14/2007 4:27:53 PM -06'00'
Author: George Penokie Subject: Note Date: 10/30/2007 4:53:40 PM
What???

Author: relliott Subject: Note Date: 10/29/2007 10:01:22 AM
Move 4.5.13.16 First Burst Enabled attribute up after 4.5.13.12 Task Priority attribute so the outputs are all ahead of the inputs. Also move it higher in the UML diagram attribute list.

Status
George Penokie Accepted 10/30/2007 5:00:24 PM
Author: relliott Subject: Note Date: 10/29/2007 12:47:45 AM
Delete the hierarchical logical unit classes from the UML model.

UML should just model the logical units that are contained in the SCSI target device (i.e., the level 1 hierarchical logical units). Some LUN values address those logical units; others address logical units in other SCSI target devices. They should not be considered part of the same SCSI target device.

The Task Router class should own the rules about parsing a LUN field (e.g. deciding where to send a task or TMF - send it to a logical unit in this target, or forward it elsewhere)

Status
George Penokie Rejected 1/16/2008 2:52:45 PM -06'00'
Author: George Penokie Subject: Note Date: 1/16/2008 2:52:38 PM -06'00'
This is too much change for SAM-4 letter ballot. If this change is desired a proposal should be written for SAM-5.

Author: relliott Subject: Highlight Date: 10/8/2007 6:57:27 PM
; s/b :

Status
George Penokie Accepted 10/30/2007 5:01:32 PM

All logical units and well known logical units contained within level 3 hierarchical logical unit shall have a Dependent Logical Unit attribute (see 4.5.19.4).

4.5.18 Level 4 Hierarchical Logical Unit class

The Level 4 Hierarchical Logical Unit class (see figure 21) contains the:

a) Logical Unit class; and
b) Well Known Logical Unit class.

The Level 4 Hierarchical Logical Unit class is a Hierarchical Logical Unit class placed at level 4 within the hierarchical logical unit structure.

All logical units and well known logical units contained within level 4 hierarchical logical unit shall have a Dependent Logical Unit attribute (see 4.5.19.4).

4.5.19 Logical Unit class

4.5.19.1 Logical Unit class overview

The Logical Unit class (see figure 22) contains the:

a) Device Server class (see 4.5.20);
b) Task Manager class (see 4.5.21);
c) Task Management Function class (see 4.5.24); and
d) Task Set class (see 4.5.22).

The Logical Unit class (see figure 22) may be substituted with the:

a) Well Known Logical Unit class (see 4.5.19.1); or
b) Hierarchical Logical Unit class.

Page: 40

Author: relliott Subject: Highlight Date: 10/8/2007 7:19:23 PM
Hierarchical Logical Unit class

There is no class with that name, and it doesn't appear in figure 22.

Status
George Penokie Accepted 10/30/2007 5:05:22 PM
Author: George Penokie Subject: Note Date: 10/30/2007 5:05:06 PM
Deleted the entry as it was a hold over from a previous version of the UML.

Page: 41

Author: relliott Subject: Highlight Date: 10/8/2007 6:47:24 PM
I_T_L_Q nexus
s/b
I_T_L_Q Nexus

Status
George Penokie Accepted 10/30/2007 5:06:13 PM

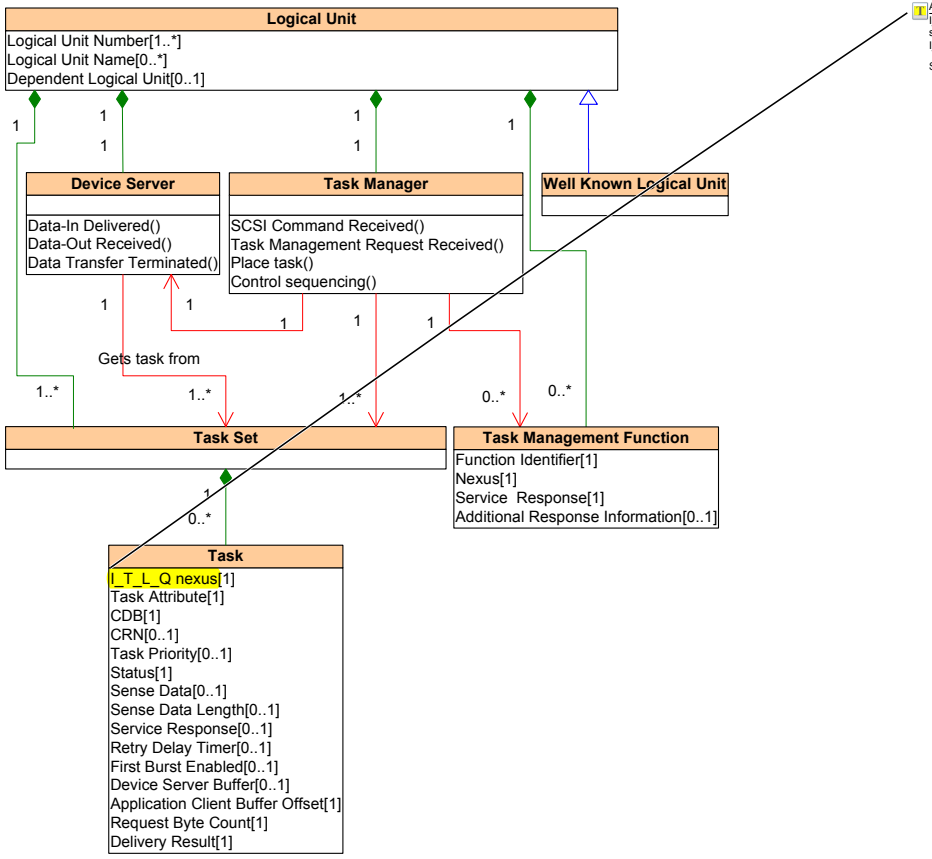


Figure 22 — Logical Unit class diagram

Each instance of a Logical Unit class shall contain the following objects:

a) one device server;
b) one task manager;
c) zero or more task management functions; and
d) one or more task sets.

The logical unit is the class to which commands are sent. One of the logical units within the SCSI target device shall be accessed using the logical unit number zero or the REPORT LUNS well-known logical unit number.

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 5.8.7) for the initiator port associated with every I_T nexus, with the additional sense code set to REPORTED LUNS DATA HAS CHANGED.

4.5.19.2 Logical Unit Number attribute

The Logical Unit Number attribute identifies the logical unit within a SCSI target device when accessed by a SCSI target port. If any logical unit within the scope of a SCSI target device includes one or more dependent logical units (see 4.5.19.4) in its composition, then all logical unit numbers within the scope of the SCSI target device shall have the format described in 4.6.6. If there are no dependent logical units within the scope of the SCSI target device, the logical unit numbers should have the format described in 4.6.5.

The 64-bit or 16-bit quantity called a LUN is the Logical Unit Number attribute defined by this standard. The fields containing the acronym LUN that compose the Logical Unit Number attribute are historical nomenclature anomalies, not Logical Unit Number attributes. Logical Unit Number attributes having different values represent different logical units, regardless of any implications to the contrary in 4.6 (e.g., LUN 00000000 00000000h is a different logical unit from LUN 40000000 00000000h and LUN 40FF0000 00000000h is a different logical unit from LUN 40FF0000 00000000h).

Logical unit number(s) are required as follows:

a) If access controls (see SPC-4) are not in effect, one logical unit number per logical unit; or
b) If access controls are in effect, one logical unit number per SCSI initiator port that has access rights plus one default logical unit number per logical unit.

See 4.6 for a definition of the construction of logical unit numbers to be used by SCSI target devices. Application clients should use only those logical unit numbers returned by a REPORT LUNS command. The task router shall respond to logical unit numbers other than those returned by a REPORT LUNS command (i.e., incorrect logical unit numbers) as specified in 5.8.4 and 7.12.

4.5.19.3 Logical Unit Name attribute

The Logical Unit Name attribute identifies a name (see 3.1.68) for a logical unit that is not a well known logical unit. A logical unit name shall be world wide unique. A logical unit name shall never change and may be used to ~~persistently~~ identify a logical unit.

Logical unit name(s) are required as follows:

a) one or more logical unit names if the logical unit is not a well-known logical unit; or
b) zero logical unit names in the logical unit is a well-known logical unit.

4.5.19.4 Dependent Logical Unit attribute

The Dependent Logical Unit attribute identifies a logical unit that is addressed via a hierarchical logical unit that resides at a lower numbered level in the hierarchy (i.e., no logical unit within level 1 contains a Dependent Logical Unit attribute while all logical units within level 2, level 3, and level 4 do contain a Dependent Logical Unit attribute).

Any instance of a Logical Unit class that contains Dependent Logical Unit attribute shall utilize the hierarchical logical unit number structure defined in 4.6.6. If any logical unit within a SCSI target device includes Dependent Logical Unit attribute:

a) all logical units within the SCSI target device shall format all logical unit numbers as described in 4.6.6; and
b) logical unit number zero or the REPORT LUNS well-known logical unit (see SPC-4) shall set the HISUP bit to one in the standard INQUIRY data.

Page: 42

Author: Mark Evans, WDC Subject: Cross-Out Date: 10/25/2007 9:51:02 AM
Delete "persistently".

Status
George Penokie Accepted 12/14/2007 11:48:54 AM -06'00'
Author: George Penokie Subject: Note Date: 12/14/2007 11:48:46 AM -06'00'
Changed to << may be used for persistent identification of a
>>

Author: relliott Subject: Note Date: 12/19/2007 3:48:01 PM -06'00'
Add this to this section << The name used to identify the logical unit is the logical unit name designation descriptor in the Device Identification VPD page (see SPC-4). >>

Status
George Penokie Accepted 12/19/2007 3:48:51 PM -06'00'
Author: relliott Subject: Highlight Date: 10/8/2007 7:18:17 PM
in the logical unit
s/b
if the logical unit

Status
George Penokie Accepted 10/30/2007 5:06:51 PM

4.5.20 Device Server class

The Device Server class (see figure 22) processes the:

a) Data-In Delivered operation (see 5.4.3.2.2) to determine ~~when data~~ requested to be sent has been sent;
b) Data-Out Received operation ~~(see 5.4.3.3.2)~~ to determine when data requested to be received has been ~~received~~;
c) Data Transfer Terminated operation (see 5.4.3.4.3) to determines when a requested termination of a data transfer has been terminated; and
d) commands.

4.5.21 Task Manager class

The Task Manager class (see figure 22) processes the:

a) SCSI Command Received operation (see 5.4.2.3) to determine when a task has been received;
b) Place Task operation to place tasks into a task set;
a) Control Sequencing operation to control the sequencing of one or more tasks within a logical unit;
b) Task Management Request Received operation (see 7.12.3) to determine when a task management function has been received; and
c) task management functions.

4.5.22 Task Set class

The Task Set class (see figure 22) contains the:

a) Task class (see 4.5.23).

Each instance of a Task Set class shall contain the following objects:

a) zero or more tasks.

The interactions among the tasks in a task set are determined by the requirements for task set management specified in clause 8 and the ACA requirements specified in 5.8.1. The number of task sets per logical unit and the boundaries between task sets are governed by the TST field in the Control mode page (see SPC-4).

4.5.23 Task class

4.5.23.1 Task class overview

The Task class represents the work associated with a command.

The task persists until a **Send Command Complete** transport protocol service response is sent or until the task is ended by a task management function or exception condition. For an example of the processing for a command see 5.7.

4.5.23.2 I_T_L_Q Nexus attribute

The I_T_L_Q Nexus attribute contains the I_T_L_Q nexus of the task (see 4.7).

4.5.23.3 Task Attribute attribute

A Task Attribute attribute (see 8.6) contains the task attribute (e.g., SIMPLE task attribute, ORDERED task attribute, HEAD OF QUEUE task attribute, ACA task attribute) of a command.

4.5.23.4 CDB attribute

The CDB attribute contains a CDB (see 5.2 and SPC-3) that defines the work to be performed by a logical unit.

Page: 43

Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:20:11 PM
"c) Data Transfer Terminated operation (see 5.4.3.4.3) to determines..."
s/b
"c) Data Transfer Terminated operation (see 5.4.3.4.3) to determine..."

Status
George Penokie Accepted 10/30/2007 5:07:29 PM

Author: relliott Subject: Highlight Date: 10/8/2007 6:46:55 PM
Task class
s/b
Task class (see figure 22)

Status
George Penokie Accepted 10/30/2007 5:08:13 PM

Author: George Penokie Subject: Note Date: 1/7/2008 5:51:49 PM -06'00'
Global
The term << transport protocol service >> is used several times instead of << SCSI transport protocol service >>. The term << SCSI >> should be added in all cases were it is missing.

Author: relliott Subject: Highlight Date: 10/27/2007 1:29:14 PM
(see 8.6) contains

Delete the blue underline after the ")" and before "contains"

Status
George Penokie Accepted 10/30/2007 5:09:25 PM

Author: suhlerp Subject: Sticky Note Date: 10/23/2007 5:52:53 PM
SPC-4 ?

Status
George Penokie Accepted 10/30/2007 5:10:07 PM
Author: George Penokie Subject: Note Date: 10/30/2007 5:10:04 PM
Changed to SPC-4.

Author: relliott Subject: Highlight Date: 10/29/2007 10:13:13 AM
SPC-3
s/b
SPC-4

or delete this reference and just refer to 5.2 alone

Status
George Penokie Accepted 10/30/2007 5:10:19 PM
Author: George Penokie Subject: Note Date: 10/30/2007 5:10:16 PM
Changed to SPC-4.

4.5.24 Task Management Function class

4.5.24.1 Task Management Function class overview

The Task Management Function class (see figure 22) represents a SCSI task management function (see clause 7).

4.5.24.2 Nexus attribute

The Nexus attribute identifies the nexus affected by the task management function (see 4.7).

4.5.24.3 Function Identifier attribute

The Function Identifier attribute contains the function identifier (see clause 7).

4.5.24.4 Service Response attribute

The Service Response attribute contains the service response (see 7.12.4).

4.5.24.5 Additional Response Information attribute

The Additional Response Information attribute, if any, contains any additional response information for the task management function (see clause 7).

4.5.25 Well Known Logical Unit class

The Well Known Logical Unit class (see figure 22) is a Logical Unit class (see 4.5.19.1) with the additional characteristics defined in this subclause.

Well known logical units are addressed using the well known logical unit addressing method (see 4.6.11) of extended logical unit addressing (see 4.6.10). Each well known logical unit has a well known logical unit number (W-LUN). W-LUN values are defined in SPC-4.

If a SCSI target port receives a W-LUN and the well known logical unit specified by the W-LUN does not exist, the task router shall follow the rules for selection of incorrect logical units described in 5.8.4 and 7.12.

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.

All well known logical units:

a) Shall not have logical unit names; and
b) Shall identify themselves using the SCSI device names of the SCSI device in which they are contained.

NOTE 3 - A SCSI target device may have multiple SCSI device names if the SCSI target device supports multiple SCSI transport protocols (see 4.5.14).

The name of the well known logical unit may be determined by issuing an INQUIRY command requesting the Device Identification VPD page (see SPC-4).

4.6 Logical unit numbers

4.6.1 Introduction

Subclause 4.6 defines the construction of logical unit numbers to be used by SCSI target devices. Application clients should use only those logical unit numbers returned by a REPORT LUNS command. The task router shall

Page: 45

Author: relliott Subject: Highlight Date: 10/8/2007 6:46:29 PM
4.5.24.3 should be swapped with 4.5.24.2 to match the order the attributes are listed in figure 22.

Status
George Penokie Accepted 10/30/2007 5:11:36 PM
Author: suhlerp Subject: Sticky Note Date: 10/25/2007 6:21:35 PM
Should this say "receives a task or a task management function specifying a W-LUN" ?

Status
George Penokie Accepted 10/30/2007 5:13:52 PM
Author: George Penokie Subject: Note Date: 10/30/2007 5:13:49 PM
Yes

Author: relliott Subject: Highlight Date: 10/29/2007 11:16:28 AM
selection of incorrect logical units
s/b
incorrect logical unit numbers

Status
George Penokie Completed 12/26/2007 11:07:34 AM -06'00'
Author: George Penokie Subject: Note Date: 12/26/2007 11:16:12 AM -06'00'
Changed to << addressing an incorrect logical unit >> Changed globally. Also, changed all << incorrect LUN >> to << incorrect logical unit number >>.

Author: relliott Subject: Highlight Date: 10/16/2007 7:15:48 PM
"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."
s/b
"A well known logical unit shall support all the commands defined for it."

Status
George Penokie Accepted 12/19/2007 3:22:15 PM -06'00'
Author: relliott Subject: Highlight Date: 10/8/2007 7:20:28 PM
The name of the well known logical unit may be determined by issuing an INQUIRY command requesting the Device Identification VPD page (see SPC-4).
s/b
The Device Identification VPD page (see SPC-4) reports the names of the SCSI target device (i.e., the names of the well-known logical unit).

Status
George Penokie Rejected 12/19/2007 3:45:26 PM -06'00'
Author: George Penokie Subject: Note Date: 12/19/2007 3:44:58 PM -06'00'
Changed to << The name used to identify the well known logical unit is the SCSI target device name designation descriptor in the Device Identification VPD page (see SPC-4). >>

respond to logical unit numbers other than those returned by a REPORT LUNS command (i.e., incorrect logical unit numbers) as specified in 5.8.4 and 7.12.

4.6.2 Logical unit representation format

When an application client displays or otherwise makes a 64-bit LUN value visible to a user, it should display it in hexadecimal format with byte 0 first (i.e., on the left) and byte 7 last (i.e., on the right), regardless of the internal representation of the LUN value (e.g., a single level LUN with an ADDRESS METHOD field set to 01b (i.e., flat space addressing) and a FLAT SPACE LUN field set to 0001h should be displayed as 40 01 00 00 00 00 00 00h, not 00 00 00 00 00 00 01 40h). A separator (e.g., space, dash, or colon) may be included between each byte, each two bytes (e.g., 4001-0000-0000-0000h), or each four bytes (e.g., 40010000 00000000h).

When displaying a single level 64-bit LUN value, an application client may display it as a single 2-byte value representing only the first level LUN (e.g., 40 01h). A separator (e.g., space, dash, or colon) may be included between each byte.

When displaying a 16-bit LUN value, an application client should display it as a single 2-byte value (e.g., 40 01h). A separator (e.g., space, dash, or colon) may be included between each byte.

4.6.3 Logical unit numbers overview

All logical unit number formats described in this standard are hierarchical in structure even when only a single level in that hierarchy is used. The HISUP bit shall be set to one in the standard INQUIRY data (see SPC-4) when any logical unit number format described in this standard is used. Non-hierarchical formats are outside the scope of this standard.

A logical unit number shall contain 64 bits or 16 bits, with the size being defined by the SCSI transport protocol. For SCSI transport protocols that define 16-bit logical unit numbers, the two bytes shall be formatted as described for the FIRST LEVEL ADDRESSING field (see table 7 in 4.6.6).

4.6.4 Minimum LUN addressing requirements

All SCSI devices shall support LUN 0 (i.e., 00000000 00000000h) or the REPORT LUNS well-known logical unit. For SCSI devices that support the hierarchical addressing model the LUN 0 or the REPORT LUNS well-known logical unit shall be the logical unit that an application client addresses to determine information about the SCSI target device and the logical units contained within the SCSI target device.

The responses to commands sent to unsupported logical units are defined in 5.8.4. The response to task management functions sent to unsupported logical units is defined in 7.1.

4.6.5 Single level logical unit number structure

Table 3 describes a single level subset of the format described in 4.6.6 for SCSI target devices that contain 256 or fewer logical units.

Page: 46

Author: Mark Evans, WDC Subject: Highlight Date: 10/25/2007 10:21:32 AM
"When an application client displays or otherwise makes a 64-bit LUN value visible to a user, it should display it in hexadecimal format..."
s/b
"When an application client displays or otherwise makes a 64-bit LUN value visible, the application client should display the value in hexadecimal format..."

Status
George Penokie Accepted 10/31/2007 10:48:33 AM

Author: Mark Evans, WDC Subject: Highlight Date: 10/25/2007 10:22:44 AM
"...an application client may display it as a single..."
s/b
"...an application client may display the value as a single..."

Status
George Penokie Accepted 10/31/2007 10:49:50 AM

Author: Mark Evans, WDC Subject: Highlight Date: 10/25/2007 10:22:58 AM
"...an application client should display it as a single..."
s/b
"...an application client should display the value as a single..."

Status
George Penokie Accepted 10/31/2007 10:49:56 AM

Author: suhlerp Subject: Sticky Note Date: 10/23/2007 6:02:47 PM
Is the use of "SCSI device" rather than "SCSI target device" in the first two sentences intentional? If so, then is the implication that a SCSI initiator device shall be able to issue commands to LUN 0?

Or should both sentences be changed to "SCSI target device"?

Status
George Penokie Completed 10/31/2007 11:25:46 AM

Author: George Penokie Subject: Note Date: 10/31/2007 10:58:18 AM
Both were changed to "SCSI target device".

Author: George Penokie Subject: Note Date: 10/31/2007 11:27:20 AM
There are several cases thoughtout section 4.6 that has this problem. All have been fixed.

Table 3 — Single level logical unit number structure using peripheral device addressing method

Bit / Byte	7	6	5	4	3	2	1	0
0	ADDRESS METHOD (00b)				BUS IDENTIFIER (00h)			
1	TARGET OR LUN							
2	Null second level LUN (0000h)							
3								
4	Null third level LUN (0000h)							
5								
6	Null fourth level LUN (0000h)							
7								

All logical unit number structure fields beyond byte 1 shall be zero (see table 3). The value in the TARGET OR LUN field shall address a single level logical unit and be between 0 and 255, inclusive. The 00b in the ADDRESS METHOD field specifies peripheral device addressing (see 4.6.6) and the 00h in the BUS IDENTIFIER field specifies the current level (see 4.6.7).

Table 4 describes a single level subset of the format described in 4.6.6 for SCSI target devices that contain 16 384 or fewer logical units.

Table 4 — Single level logical unit number structure using flat space addressing method

Bit / Byte	7	6	5	4	3	2	1	0
0	ADDRESS METHOD (01b)		(MSB)					
1	FLAT SPACE LUN							(LSB)
2	Null second level LUN (0000h)							
3								
4	Null third level LUN (0000h)							
5								
6	Null fourth level LUN (0000h)							
7								

All logical unit number structure fields beyond byte 1 shall be zero (see table 4). The value in the FLAT SPACE LUN field shall be between 0 and 16 383, inclusive. The 01b in the ADDRESS METHOD field specifies flat space addressing (see 4.6.8) at the current level.

Table 5 describes a single level subset of the format described in 4.6.6 for SCSI target devices that contain more than 16 384 logical units.

Page: 47

Author: Mark Evans, WDC Subject: Highlight Date: 10/25/2007 10:24:13 AM
"All logical unit number structure fields beyond byte 1 shall be zero (see table 3)."
s/b
"Byte 2 through byte 7 in an 8-byte single level logical unit number structure using the peripheral device addressing method shall contain 00h (see table 3)."

Status
George Penokie Accepted 10/31/2007 11:00:34 AM

Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:21:05 PM
"The 00b in the ADDRESS METHOD field specifies peripheral device addressing (see 4.6.6) and the 00h in the BUS IDENTIFIER field specifies the current level (see 4.6.7)."
s/b
"A value of 00b in the ADDRESS METHOD field specifies peripheral device addressing (see 4.6.6). A value of 00h in the in the BUS IDENTIFIER field specifies the current level (see 4.6.7)."

Status
George Penokie Accepted 10/31/2007 11:01:50 AM

Author: Mark Evans, WDC Subject: Highlight Date: 10/25/2007 10:26:04 AM
"All logical unit number structure fields beyond byte 1 shall be zero (see table 4)."
s/b
"Byte 2 through byte 7 in an 8-byte single level logical unit number structure using the flat space addressing method shall contain 00h (see table 4)."

Status
George Penokie Accepted 10/31/2007 11:02:54 AM

Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:21:58 PM
"The 01b in the ADDRESS METHOD field specifies flat space addressing (see 4.6.8) at the current level."
s/b
"A value of 01b in the ADDRESS METHOD field specifies flat space addressing (see 4.6.8) at the current level."

Status
George Penokie Accepted 10/31/2007 11:03:25 AM

Table 5 — Single level logical unit number structure using extended flat space addressing method

Bit / Byte	7	6	5	4	3	2	1	0
0	ADDRESS METHOD (11b)		LENGTH (01b)		EXTENDED ADDRESS METHOD (2h)			
1	(MSB)							
2	EXTENDED FLAT SPACE ADDRESS							
3								(LSB)
4	Null second level LUN (0000h)							
5								
6	Null third level LUN (0000h)							
7								

All logical unit number structure fields beyond byte 3 shall be zero (see table 5). The value in the EXTENDED FLAT SPACE ADDRESS field shall be between 0 and 16 777 215, inclusive. The 11b in the ADDRESS METHOD field with a 2h in the EXTENDED ADDRESS METHOD field specifies extended flat space addressing (see 4.6.12) at the current level. The 01b in the LENGTH field specifies that the LUN specified in the EXTENDED FLAT SPACE ADDRESS field is three bytes in length.

The presence of well-known logical units shall not affect the requirements defined within this subclause.

If a SCSI target device contains 256 or fewer logical units, none of which are dependent logical units (see 4.5.19.4), then the SCSI target device's logical unit numbers:

a) should have the format shown in table 3 (i.e., peripheral device addressing);
b) may have the format shown in table 4 (i.e., flat space addressing); or
c) may have the format shown in table 5 (i.e., extended flat space addressing).

If a SCSI target device contains more than 256 logical units and 16 384 or fewer logical units, none of which are dependent logical units (see 4.5.19.4), then the SCSI target device's logical unit numbers:

a) should have the format shown in table 4 (i.e., flat space addressing);
b) may have the format shown in table 5 (i.e., extended flat space addressing); or
c) may have the format shown in table 3 (i.e., peripheral device addressing) for up to 256 of the logical units within SCSI target device.

If a SCSI target device contains more than 16 384 logical units, none of which are dependent logical units (see 4.5.19.4), then the SCSI target device's logical unit numbers:

a) should have the format shown in table 5 (i.e., extended flat space addressing);
b) may have the format shown in table 4 (i.e., flat space addressing) for up to 16 384 of the logical units within SCSI target device; or
c) may have the format shown in table 3 (i.e., peripheral device addressing) for up to 256 of the logical units within SCSI target device.

4.6.6 Eight byte logical unit number structure

The eight byte logical unit number structure (see table 7) contains four levels of addressing fields. Each level shall use byte 0 and byte 1 to define the address and location of the SCSI device to be addressed on that level.

If the logical unit number specifies that the command is to be relayed to the next level then the current level shall use byte 0 and byte 1 of the eight byte logical unit number structure to determine the address of the SCSI device to which the command is to be sent. When the command is sent to the SCSI target device the eight byte logical

Page: 48

Author: relliott Subject: Highlight Date: 10/29/2007 10:25:05 AM
8 9 10 11
s/b
4 5 6 7

Status
George Penokie Accepted 10/31/2007 11:04:43 AM
Author: Mark Evans, WDC Subject: Highlight Date: 10/25/2007 10:28:12 AM
"All logical unit number structure fields beyond byte 3 shall be zero (see table 5)."
s/b
"Byte 4 through byte 11 in an 12-byte single level logical unit number structure using the extended flat space addressing method shall contain 00h (see table 3)."

Status
George Penokie Accepted 10/31/2007 11:05:51 AM
Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:20:59 PM
"The 11b in the ADDRESS METHOD field with a 2h in the EXTENDED ADDRESS METHOD field specifies extended flat space addressing (see 4.6.12) at the current level. The 01b in the LENGTH field specifies that the LUN specified in the EXTENDED FLAT SPACE ADDRESS field is three bytes in length."
s/b
"A value of 11b in the ADDRESS METHOD field with a value of 2h in the EXTENDED ADDRESS METHOD field specifies extended flat space addressing (see 4.6.12) at the current level. A value of 01b in the LENGTH field specifies that the LUN specified in the EXTENDED FLAT SPACE ADDRESS field is three bytes in length."

Status
George Penokie Accepted 10/31/2007 11:06:53 AM
Author: relliott Subject: Highlight Date: 12/10/2007 4:32:24 PM -06'00'
command
s/b
command or TMF throughout this section, since TMFs are also sent to a LUN

Status
George Penokie Accepted 12/19/2007 4:21:19 PM -06'00'

unit number structure that was received shall be adjusted to create a new eight byte logical unit number structure (see table 6 and figure 23).

SCSI devices shall keep track of the addressing information necessary to transmit information back through all intervening levels to the task's originating SCSI initiator port.

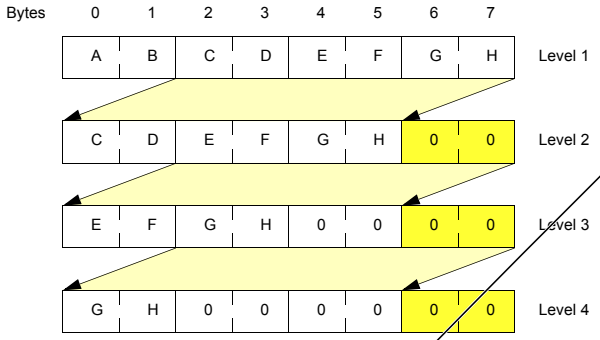


Figure 23 — Eight byte logical unit number structure adjustments

Table 6 — Eight byte logical unit number structure adjustments

Byte position		
Old		**New**
0 & 1	Moves to	Not Used
2 & 3	Moves to	0 & 1
4 & 5	Moves to	2 & 3
6 & 7	Moves to	4 & 5
N/A	zero fill	6 & 7

The eight byte logical unit number structure requirements as viewed from the application client are shown in table 7.

Page: 49

Author: Mark Evans, WDC Subject: Highlight Date: 10/25/2007 10:29:44 AM
"N/A"
s/b
"n/a"

Status
George Penokie Accepted 10/31/2007 11:07:18 AM

Page: 50

Author: relliott Subject: Note Date: 10/29/2007 12:56:01 AM
The field name is ADDRESS METHOD, but the descriptions all use "addressing method"

Status
George Penokie Rejected 10/31/2007 11:11:18 AM
Author: George Penokie Subject: Note Date: 10/31/2007 11:11:11 AM
And what's wrong with that. I didn't know we had a rule that the name of a field had to use exactly the same wordings as the description of the field.

Table 7 — Eight byte logical unit number structure

Bit Byte	7	6	5	4	3	2	1	0
0	FIRST LEVEL ADDRESSING (see table 8)							
1								
2	SECOND LEVEL ADDRESSING (see table 8)							
3								
4	THIRD LEVEL ADDRESSING (see table 8)							
5								
6	FOURTH LEVEL ADDRESSING (see table 8)							
7								

The FIRST LEVEL ADDRESSING field specifies the first level address of a SCSI device. See table 8 for a definition of the FIRST LEVEL ADDRESSING field.

The SECOND LEVEL ADDRESSING field specifies the second level address of a SCSI device. See table 8 for a definition of the SECOND LEVEL ADDRESSING field.

The THIRD LEVEL ADDRESSING field specifies the third level address of a SCSI device. See table 8 for a definition of the THIRD LEVEL ADDRESSING field.

The FOURTH LEVEL ADDRESSING field specifies the fourth level address of a SCSI device. See table 8 for a definition of the FOURTH LEVEL ADDRESSING field.

Table 8 — Format of addressing fields

Bit Byte	7	6	5	4	3	2	1	0
n-1	ADDRESS METHOD							
n	ADDRESS METHOD SPECIFIC							

The ADDRESS METHOD field defines the contents of the ADDRESS METHOD SPECIFIC field. See table 9 for the address methods defined for the ADDRESS METHOD field. The ADDRESS METHOD field only defines address methods for entities that are directly addressable by an application client.

Table 9 — ADDRESS METHOD field

Code	Description	Reference
00b	Peripheral device addressing method	4.6.7
01b	Flat space addressing method	4.6.8
10b	Logical unit addressing method	4.6.9
11b	Extended logical unit addressing method	4.6.10

4.6.7 Peripheral device addressing method

If the peripheral device addressing method (see table 10) is selected, the SCSI device should relay the received command or task management function to the addressed dependent logical unit.

If the SCSI device does not relay any commands or task management functions to the addressed dependent logical unit, it shall follow the rules for selection of incorrect logical units described in 5.8.4 and 7.12.

If the SCSI device does relay some commands and task management functions to the addressed dependent logical unit, it shall:

a) terminate any command that is not relayed with CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID COMMAND OPERATION CODE; and
b) terminate a task management function that is not relayed with a service response of SERVICE DELIVERY OR TARGET FAILURE.

NOTE 4 - A SCSI device may filter (i.e., not relay) commands or task management functions to prevent operations with deleterious effects from reaching a dependent logical unit (e.g., a WRITE command directed to a logical unit that is participating in a RAID volume).

Table 10 — Peripheral device addressing format

Bit / Byte	7	6	5	4	3	2	1	0
n-1	ADDRESS METHOD (00b)		BUS IDENTIFIER					
n	TARGET OR LUN							

The BUS IDENTIFIER field identifies the bus or path that the SCSI device shall use to relay the received command or task management function. The BUS IDENTIFIER field may use the same value encoding as the BUS NUMBER field (see 4.6.9) with the most significant bits set to zero. However, bus identifier zero shall specify that the command or task management function is to be relayed to a logical unit within the SCSI device at the current level.

The TARGET OR LUN field specifies the address of the peripheral device (e.g., a SCSI device at the next level) to which the SCSI device shall relay the received command or task management function. The meaning and usage of the TARGET OR LUN field depends on whether the BUS IDENTIFIER field contains zero.

A BUS IDENTIFIER field of zero specifies a logical unit at the current level. This representation of a logical unit may be used either when the SCSI device at the current level does not use hierarchical addressing for assigning LUNs to entities or when the SCSI device at the current level includes entities that are assigned LUNs but are not attached to SCSI buses. When the BUS IDENTIFIER field contains zero, the command or task management

Page: 51

Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:20:52 PM
"If the SCSI device does not relay any commands or task management functions to the addressed dependent logical unit, it shall follow the rules for selection of incorrect logical units described in 5.8.4 and 7.12."
s/b
"If the SCSI device does not relay any commands or task management functions to the addressed dependent logical unit, then the SCSI device shall follow the rules for selection of incorrect logical units described in 5.8.4 and 7.12."

Status
George Penokie Accepted 10/31/2007 11:29:27 AM

Author: relliott Subject: Highlight Date: 10/29/2007 11:16:56 AM
selection of incorrect logical units
s/b
incorrect logical unit numbers

Status
George Penokie Completed 12/26/2007 11:08:03 AM -06'00'

Author: George Penokie Subject: Note Date: 12/26/2007 11:07:57 AM -06'00'
Changed to << addressing an incorrect logical unit >>

Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:20:55 PM
"If the SCSI device does relay some commands and task management functions to the addressed dependent logical unit, it shall:"
s/b
"If the SCSI device does relay some commands and task management functions to the addressed dependent logical unit, then the SCSI device shall:"

Status
George Penokie Accepted 10/31/2007 11:30:37 AM

Author: relliott Subject: Note Date: 10/29/2007 11:21:20 AM
In 4.6.7 Peripheral device addressing method and 4.6.9 Logical unit addressing method, add a figure showing one level of hierarchy to illustrate the relay concept and how those addressing methods parse the fields.

Status
George Penokie Accepted 2/27/2008 10:10:55 AM -06'00'

Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:21:49 PM
"However, bus identifier zero shall specify that the command or task management function is to be relayed to a logical unit within the SCSI device at the current level."
s/b
"However, if the BUS IDENTIFIER field is set to 00h, then the command or task management function shall be relayed to a logical unit within the SCSI device at the current level."

Status
George Penokie Accepted 10/31/2007 11:32:45 AM

function shall be relayed to the current level logical unit specified by the TARGET OR LUN field within or joined to the current level SCSI device.

A BUS IDENTIFIER field greater than zero represents a SCSI domain that connects a group of SCSI devices to the current level SCSI device. Each SCSI domain shall be assigned a unique bus identifier number ~~from~~ 1 to 63. These bus identifiers shall be used in the BUS IDENTIFIER field when assigning ~~addresses~~ to peripheral devices attached to the SCSI domains. When the BUS IDENTIFIER field is greater than zero, the command or task management function shall be relayed to the logical unit with the logical unit number zero within the SCSI target device specified in the TARGET OR LUN field located in the SCSI domain specified by the BUS IDENTIFIER field. The SCSI target device information in the TARGET OR LUN field is a ~~mapped representation of a~~ target port identifier.

The SCSI device located within the current level may be addressed by a ~~BUS IDENTIFIER field~~ and a TARGET OR LUN field of all zeros, also known as LUN 0 (see 4.6.4).

4.6.8 Flat space addressing method

The flat space addressing method (see table 11) specifies a logical unit at the current level.

The contents of all hierarchical structure addressing fields following a flat space addressing method addressing field shall be ignored.

Table 11 — Flat space addressing format

Bit Byte	7	6	5	4	3	2	1	0
n-1	ADDRESS METHOD (01b)		(MSB)					
n	FLAT SPACE LUN							(LSB)

The FLAT SPACE LUN field specifies the current level logical unit.

4.6.9 Logical unit addressing method

If the logical unit addressing method (see table 12) is selected, the SCSI device should relay the received command or task management function to the addressed dependent logical unit.

If the SCSI device does ~~not relay any~~ commands or task management functions to the addressed dependent logical unit, ~~it shall~~ follow the rules for selection of incorrect logical units described in 5.8.4 and 7.12.

If the SCSI device does relay some commands and task management functions to the addressed dependent logical unit, it shall:

a) terminate any command that is not relayed with CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID COMMAND OPERATION CODE; and
b) terminate a task management function that is not relayed with a service response of SERVICE DELIVERY OR TARGET FAILURE.

NOTE 5 - A SCSI device may filter (i.e., not relay) commands or task management functions to prevent operations with deleterious effects from reaching a dependent logical unit (e.g., a WRITE command directed to a logical unit that is participating in a RAID volume).

The contents of all hierarchical structure addressing fields following a logical unit addressing method addressing field shall be ignored.

Page: 52

Author: George Penokie Subject: Highlight Date: 2/27/2008 10:16:18 AM -06'00'
with the logical unit number zero within the SCSI target device specified in the TARGET OR LUN field located in the SCSI domain specified by the BUS IDENTIFIER field

Status
George Penokie Accepted 2/27/2008 12:05:08 PM -06'00'
Author: George Penokie Subject: Sticky Note Date: 2/27/2008 12:05:04 PM -06'00'
Change to << within the SCSI target device specified in the target or lun field located in the SCSI domain specified by the bus identifier field with the LUN being set to the contents of the received LUN shifted by two bytes as described in 4.6.6. >>

Author: relliott Subject: Highlight Date: 10/8/2007 7:22:14 PM
may be addressed
s/b
is addressed

Status
George Penokie Accepted 10/31/2007 11:37:58 AM

Author: relliott Subject: Highlight Date: 10/8/2007 7:21:53 PM
by a BUS IDENTIFIER field
d/n
nu a BUS IDENTIFIER field of zero

Status
George Penokie Rejected 10/31/2007 11:39:02 AM
Author: George Penokie Subject: Note Date: 10/31/2007 11:38:55 AM
Changed to "The SCSI target device located within the current level is addressed when the bus identifier field is set to zero and the target or lun field is set to zero, also known as LUN 0 (see 4.6.4)."

Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:21:38 PM
"If the SCSI device does not relay any commands or task management functions to the addressed dependent logical unit, it shall follow the rules for selection of incorrect logical units described in 5.8.4 and 7.12."
s/b
"If the SCSI device does not relay any commands or task management functions to the addressed dependent logical unit, then the SCSI device shall follow the rules for selection of incorrect logical units described in 5.8.4 and 7.12."

Status
George Penokie Accepted 10/31/2007 11:48:06 AM

Author: relliott Subject: Highlight Date: 10/29/2007 11:17:13 AM
selection of incorrect logical units
s/b
incorrect logical unit numbers

Status
George Penokie Completed 12/26/2007 11:08:27 AM -06'00'
Author: George Penokie Subject: Note Date: 12/26/2007 11:08:21 AM -06'00'
Changed to << addressing an incorrect logical unit >>

Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:21:43 PM
"If the SCSI device does relay some commands and task management functions to the addressed dependent logical unit, it shall:"
s/b
"If the SCSI device does relay some commands and task management functions to the addressed dependent logical unit, then the SCSI device shall:"

Status
George Penokie Accepted 10/31/2007 11:48:18 AM

Page: 53

Author: relliott Subject: Highlight Date: 10/29/2007 12:57:43 AM
m
should be two rows :
n+1
m

sharing the EXTENDED ADDRESS METHOD SPEIFIC field

Status
George Penokie Accepted 10/31/2007 2:21:05 PM

Table 12 — Logical unit addressing format

Bit / Byte	7	6	5	4	3	2	1	0
n-1	ADDRESS METHOD (10b)		TARGET					
n	BUS NUMBER			LUN				

The TARGET field, BUS NUMBER field, and LUN field address the logical unit to which the received command or task management function shall be relayed. The command or task management function shall be relayed to the logical unit specified by the LUN field within the SCSI target device specified by the TARGET field located on the bus specified by the BUS NUMBER field. The value in the LUN field shall be placed in the least significant bits of the TARGET OR LUN field in a single level logical unit number structure for logical unit numbers 255 and below (see 4.6.5). The TARGET field contains a mapped representation of a target port identifier.

4.6.10 Extended logical unit addressing

Extended logical unit addressing (see table 13) specifies a logical unit at the current level.

Extended logical unit addressing builds on the formats defined for dependent logical units (see 4.5.19.4) but may be used by SCSI devices having single level logical unit structure. In dependent logical unit addressing, the logical unit information at each level fits in exactly two bytes. Extended logical unit addresses have sizes of two bytes, four bytes, six bytes, or eight bytes.

The contents of all hierarchical structure addressing fields following an extended logical unit addressing method addressing field shall be ignored.

Extended logical units are identified by the ADDRESS METHOD field (see table 9 in 4.6.6) in the same manner as is the case for dependent logical units. An ADDRESS METHOD field value of 11b specifies the extended logical unit addressing method.

Table 13 — Extended logical unit addressing format

Bit / Byte	7	6	5	4	3	2	1	0
n	ADDRESS METHOD (11b)		LENGTH		EXTENDED ADDRESS METHOD			
m	EXTENDED ADDRESS METHOD SPECIFIC							

The LENGTH field (see table 14) specifies the length of the EXTENDED ADDRESS METHOD SPECIFIC field.

Table 14 — LENGTH **field and related sizes**

Code	Size in bytes of		Reference
	EXTENDED ADDRESS METHOD SPECIFIC **field**	**Extended logical unit addressing format**	
00b	1	2	table 15
01b	3	4	table 16
10b	5	6	table 17
11b	7	8	table 18

Table 15, table 16, table 17, and table 18 show the four extended logical unit addressing formats.

Table 15 — Two byte extended logical unit addressing format

Bit Byte	7	6	5	4	3	2	1	0
n	ADDRESS METHOD (11b)		LENGTH (00b)		EXTENDED ADDRESS METHOD			
n+1	EXTENDED ADDRESS METHOD SPECIFIC							

Table 16 — Four byte extended logical unit addressing format

Bit Byte	7	6	5	4	3	2	1	0
n	ADDRESS METHOD (11b)		LENGTH (01b)		EXTENDED ADDRESS METHOD			
n+1	(MSB)							
	EXTENDED ADDRESS METHOD SPECIFIC							
n+3								(LSB)

Table 17 — Six byte extended logical unit addressing format

Bit Byte	7	6	5	4	3	2	1	0
n	ADDRESS METHOD (11b)		LENGTH (10b)		EXTENDED ADDRESS METHOD			
n+1	(MSB)							
	EXTENDED ADDRESS METHOD SPECIFIC							
n+5								(LSB)

Page: 54

Author: relliott Subject: Note Date: 10/29/2007 10:55:07 AM
A LUN that includes a LENGTH field value that goes beyond the LUN field length supported by the transport protocol (2 bytes or 8 bytes) is invalid.

For example, if the protocol supports 8-byte LUNs, and a LUN contains:
bytes 0-1: logical unit addressing format
byte 2: address method 11b, length 11b
bytes 3-7: ...
that L:UN must be treated as an invalid LUN (two bytes are being truncated).

For example, if the protocol only supports 2-byte LUNs, then a LUN containing anything longer must be considered invalid:
byte 0: address method 11b, length 01b, 10b, or 11b
byte 1: ...

Status
George Penokie Completed 10/31/2007 2:35:45 PM
Author: George Penokie Subject: Note Date: 10/31/2007 2:34:18 PM
Added the following statement: "A LUN that includes a LENGTH field value that goes beyond the LUN field length supported by the transport protocol is invalid shall follow the rules for selection of incorrect logical units described in 5.8.4."

Author: relliott Subject: Note Date: 10/29/2007 1:01:48 AM
Earlier addressing format tables used n-1 to n. Tables 13, 15, 16, 17, 20, 21 should end in n, not start with n.

Status
George Penokie Completed 10/31/2007 3:01:57 PM
Author: George Penokie Subject: Note Date: 10/31/2007 3:01:52 PM
Changed the table that used the "n-1 to n " to (n to n+1).

Author: relliott Subject: Cross-Out Date: 10/29/2007 10:26:51 AM
Delete (MSB)/(LSB) from the EXTENDED ADDRESS METHOD SPECIFIC field in table 16, 17, 18

Status
George Penokie Accepted 10/31/2007 3:03:22 PM
Author: relliott Subject: Cross-Out Date: 10/29/2007 10:27:14 AM
Delete (MSB)/(LSB) from the EXTENDED ADDRESS METHOD SPECIFIC field in table 16, 17, 18

Status
George Penokie Accepted 10/31/2007 3:03:14 PM

Page: 55

Table 18 — Eight byte extended logical unit addressing format

Bit / Byte	7	6	5	4	3	2	1	0
0	ADDRESS METHOD (11b)		LENGTH (11b)		EXTENDED ADDRESS METHOD			
1	~~(MSB)~~							
	EXTENDED ADDRESS METHOD SPECIFIC							
7								(LSB)

The EXTENDED ADDRESS METHOD field combined with the LENGTH field (see table 19) specifies the type and size of extended logical unit address found in the EXTENDED ADDRESS METHOD SPECIFIC field.

Table 19 — Logical unit extended addressing

EXTENDED ADDRESS METHOD **Code(s)**	LENGTH **Code(s)**	**Description**	**Reference**
0h	00b - 11b	Reserved	
1h	00b	Well known logical unit	4.6.11
1h	01b - 11b	Reserved	
2h	01b	Extended flat space addressing	4.6.12
2h	00b, 10b, 11b	Reserved	
3h - Eh	00b - 11b	Reserved	
Fh	00b - 10b	Reserved	
Fh	11b	Logical unit not specified	4.6.13

4.6.11 Well known logical unit addressing

~~A SCSI target device may support zero or more well known logical units (see 4.5.25).~~ A single SCSI target device shall only support one instance of each supported well known logical unit. All well known logical units within a SCSI target device shall be accessible from all SCSI target ports contained within the SCSI target device.

Well known logical units are addressed using the well known logical unit extended address format (see table 20).

Table 20 — Well known logical unit extended addressing format

Bit / Byte	7	6	5	4	3	2	1	0
n	ADDRESS METHOD (11b)		LENGTH (00b)		EXTENDED ADDRESS METHOD (1h)			
n+1	W-LUN							

The W-LUN field specifies the well known logical unit to be addressed (see SPC-4).

Author: relliott Subject: Cross-Out Date: 10/29/2007 10:27:18 AM
Delete (MSB)/(LSB) from the EXTENDED ADDRESS METHOD SPECIFIC field in table 16, 17, 18

Status
George Penokie Accepted 10/31/2007 3:03:31 PM

Author: relliott Subject: Note Date: 12/19/2007 4:50:10 PM -06'00'
Code Fh length 00b
Code Fh length 01b
Code Fh length 10b
should each also be a variant of "Logical unit not specified", used for hierarchical LUN situations where the lowest level logical unit receives one of these incoming LUN values, not all FFs:
FFFF0000_00000000h
FFFFFFFF_00000000h
FFFFFFFF_FFFF0000h

Status
George Penokie Rejected 12/19/2007 4:53:28 PM -06'00'
Author: George Penokie Subject: Note Date: 12/19/2007 4:38:09 PM -06'00'
Changed table 22 Logical unit not specified extended addressing format as follows. Changed bytes 3 - 7 from FFh to ignored.

Author: relliott Subject: Cross-Out Date: 10/8/2007 7:24:52 PM
Delete "A SCSI target device may support zero or more well known logical units (see 4.5.25)."

Since this allows 0 through infinity, it is not stating a requirement or allowance. Could replace with ""A SCSI target device supports zero or more well known logical units (see 4.5.25)."

Status
George Penokie Rejected 12/19/2007 5:04:02 PM -06'00'
Author: George Penokie Subject: Note Date: 12/19/2007 5:03:53 PM -06'00'
Although technically this statement is a duplicate of the UML the described in the SCSI target device class diagram. W-LUNs are still new enough that allowing more than any number of W-LUNs in a target could be missed.

c) I_T_L_Q nexus; and
d) I_T_L_x nexus.

Table 23 defines the types of nexuses and the identifiers used to construct each of them.

Table 23 — Nexus

Nexus [a]	Identifiers used to construct nexuses	Reference
I_T nexus	Initiator port identifier Target port identifier	4.5.7.2 4.5.6.2
I_T_L nexus	Initiator port identifier Target port identifier Logical unit number	4.5.7.2 4.5.6.2 4.5.19.2
I_T_L_Q nexus	Initiator port identifier Target port identifier Logical unit number Task identifier	4.5.7.2 4.5.6.2 4.5.19.2 4.7.2

[a] I_T_L_x nexus specifies either an I_T_L nexus or an I_T_L_Q nexus.

4.7.2 Task identifier

The task identifier (i.e., the Q in an I_T_L_Q nexus) represents a task, allowing many uniquely identified tasks to be outstanding at once. Each SCSI transport protocol defines the size of the task identifier, up to a maximum of 64 bytes, to be used by SCSI ports that support that SCSI transport protocol.

The SCSI initiator device assigns a task identifier value for each I_T_L_Q nexus in a way that ensures that the nexus uniqueness requirements stated in this subclause are met. SCSI transport protocols may define additional restrictions on task identifier assignments (e.g., requiring task identifier to be unique per I_T nexus or per I_T_L nexus, or sharing task identifier values with other uses such as task management functions).

4.7.3 Nexus usage rules

An I_T_L_Q nexus that is in use (i.e., during the interval bounded by the events specified in 5.5) shall be unique as seen by the SCSI initiator port originating the command and the logical unit to which the command was addressed, otherwise an overlapped command condition exists (see 5.8.3). An I_T_L_Q nexus is unique if one or more of its components is unique within the specified time interval.

The SCSI initiator device shall not create more than one task from a specific SCSI initiator port having identical values for the target port identifier, logical unit number, and task identifier.

4.8 SCSI ports

4.8.1 SCSI port configurations

A SCSI device may contain only SCSI target ports, only SCSI initiator ports, or any combination of ports. Some of the port configurations possible for a SCSI device are shown in figure 24.

Page: 57

Author: Brocade Subject: Highlight Date: 10/31/2007 3:09:20 PM
Brocade-001
The text in SAM-4 rev 13, clause 4.7.2
"The task identifier (i.e., the Q in an I_T_L_Q nexus) represents a task, allowing many uniquely identified tasks to be outstanding at once. Each SCSI transport protocol defines the size of the task identifier, up to a maximum of 64 bytes, to be used by SCSI ports that support that SCSI transport protocol."
Has been changed to read something like:
"The Task Identifier (i.e., the Q in an I_T_L_Q nexus) uniquely identifies a task,..."
I do not believe that is precisely correct. It only identifies the task uniquely within the context of a particular I_T_L nexus. As an example, see SPI-3's use of the Message Out and Message In to provide the Q value, which is only valid for a particular I_T_L nexus. As a second example, consider FCP, that uses X_ID between a single initiator and target as the identifier, but where the same X_ID may appear on other commands from a different initiator to the same
target.
As a result, the proper wording would be something like:
"The Task Identifier (i.e., the Q in an I_T_L_Q nexus) uniquely identifies a task in the context of a particular I_T_L nexus, ..."

Status
George Penokie Accepted 12/17/2007 10:59:10 AM -06'00'
Author: George Penokie Subject: Note Date: 12/17/2007 11:24:26 AM -06'00'
Changed to <<A command identifier (i.e., the Q in an I_T_L_Q nexus) is assigned by a SCSI initiator device to uniquely identify one command in the context of a particular I_T_L nexus, allowing more than one command to be outstanding for that I_T_L nexus at the same time. >>

Author: Seagate Subject: Rectangle Date: 3/24/2008 2:06:14 PM
This << 64 bytes >> should be << 64 bits >>

Status
George Penokie Accepted 3/24/2008 2:06:12 PM
Author: relliott Subject: Highlight Date: 10/8/2007 7:25:41 PM
A SCSI device may contain only SCSI target ports, only SCSI initiator ports, or any combination of ports.
s/b
A SCSI device shall contain only SCSI target ports, only SCSI initiator ports, or any combination of ports.

Status
George Penokie Rejected 12/21/2007 10:21:44 AM -06'00'
Author: George Penokie Subject: Note Date: 12/21/2007 10:21:25 AM -06'00'
Changed to << A SCSI device contains only the following combinations of SCSI ports:
all SCSI target ports;
all SCSI initiator ports; or
any combination of SCSI target ports and SCSI initiator ports. >>

Author: George Penokie Subject: Note Date: 12/21/2007 10:20:09 AM -06'00'
Change the term << port >> to << SCSI port >>

Status
George Penokie Accepted 12/21/2007 10:20:01 AM -06'00'

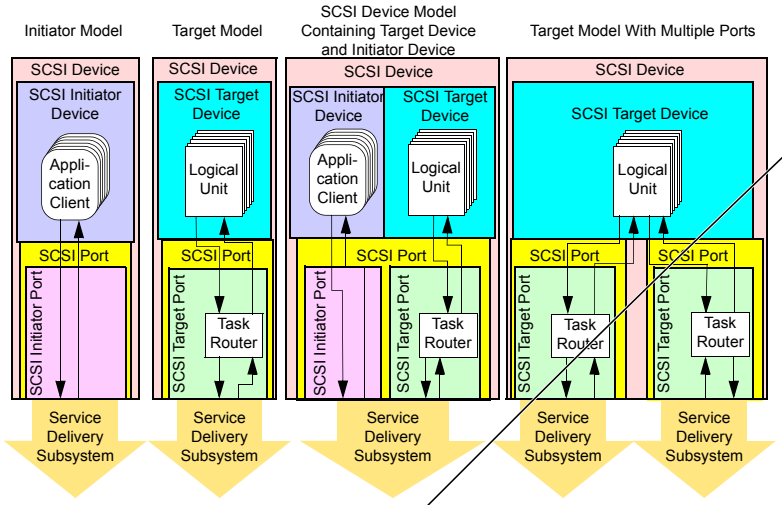


Figure 24 — SCSI device functional models

4.8.2 SCSI devices with multiple ports

The model for a SCSI device with multiple ports is a single:

a) SCSI target device (see 4.5.14) with multiple SCSI target ports;
b) SCSI initiator device (see 4.5.9) with multiple SCSI initiator ports; or
c) SCSI device containing a SCSI initiator device and a SCSI target device, and multiple SCSI ports.

The identifiers representing the SCSI ports shall meet the requirements for initiator port identifiers (see 4.5.9) or target port identifiers (see 4.5.14). How a multiple port SCSI device is viewed by counterpart SCSI devices in the SCSI domain also depends on whether a SCSI initiator port is examining a SCSI target port, or a SCSI target port is servicing a SCSI initiator port.

4.8.3 Multiple port SCSI target device structure

Figure 25 shows the structure of a SCSI target device with multiple SCSI ports each containing a SCSI target port. Each SCSI target port contains a task router that is shared by a collection of logical units. Each logical unit contains a single task manager and a single device server.

Page: 58

Author: relliott Subject: Highlight Date: 10/8/2007 7:26:20 PM
Move "a single" into each of a) b) and c) to improve readability

Status
George Penokie Rejected 12/19/2007 5:08:07 PM -06'00'

Page: 59

Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:21:23 PM
"Two-way communications shall be possible between all logical units and all SCSI target ports, however, communications between any logical unit and any SCSI target port may be inactive. Two-way communications shall be available between each task manager and all task routers. Each SCSI target port shall accept commands sent to LUN 0 or the REPORT LUNS well-known logical unit and the task router shall route them to a device server for processing. REPORT LUNS commands (see SPC-4) shall be accepted by the logical unit with the logical unit number zero or the REPORT LUNS well-known logical unit from any SCSI target port and shall return the logical unit inventory available via that SCSI target port. The availability of the same logical unit through multiple SCSI target ports is discovered by matching logical unit name values in the INQUIRY command Device Identification VPD page (see SPC-4)."
s/b
"Two-way communications shall be possible between all logical units and all SCSI target ports in a SCSI device. However, communications between any logical unit and any SCSI target port in a SCSI device may be inactive. Two-way communications shall be available between each task manager and all task routers in the SCSI target ports in the SCSI device. Each SCSI target port in a SCSI device shall accept commands sent to LUN 0 or the REPORT LUNS well-known logical unit, and the task router in that SCSI target port shall route the commands to a device server in a logical unit in the SCSI device for processing. REPORT LUNS commands (see SPC-4) shall be accepted by the logical unit with the logical unit number zero or the REPORT LUNS well-known logical unit from any SCSI target port in the SCSI device, and the logical unit shall return the logical unit inventory available via that SCSI target port. An application client determines the availability of the same logical unit through multiple SCSI target ports in a SCSI device by matching logical unit name values in the Device Identification VPD page (see SPC-4)."

Status
George Penokie Accepted 10/31/2007 3:19:50 PM

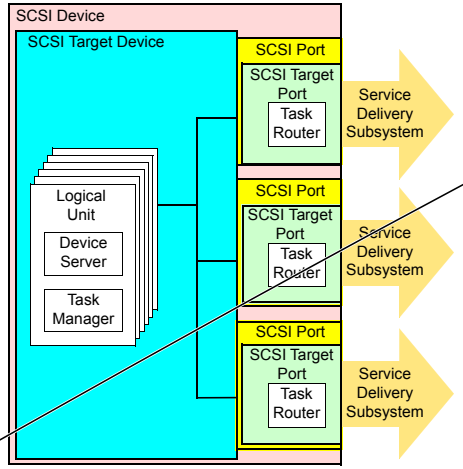


Figure 25 — Multiple port target SCSI device structure model

Two-way communications shall be possible between all logical units and all SCSI target ports, however, communications between any logical unit and any SCSI target port may be inactive. Two-way communications shall be available between each task manager and all task routers. Each SCSI target port shall accept commands sent to LUN 0 or the REPORT LUNS well-known logical unit and the task router shall route them to a device server for processing. REPORT LUNS commands (see SPC-4) shall be accepted by the logical unit with the logical unit number zero or the REPORT LUNS well-known logical unit from any SCSI target port and shall return the logical unit inventory available via that SCSI target port. The availability of the same logical unit through multiple SCSI target ports is discovered by matching logical unit name values in the INQUIRY command Device Identification VPD page (see SPC-4).

4.8.4 Multiple port SCSI initiator device structure

Figure 26 shows the structure of a SCSI initiator device with multiple SCSI ports each containing a SCSI initiator port. Each SCSI initiator port is shared by a collection of application clients.

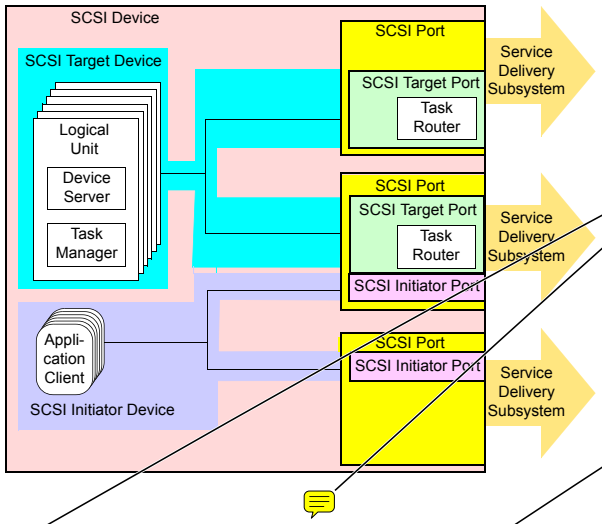


Figure 27 — Multiple port SCSI device structure model

Two-way communications shall be possible between all logical units and all SCSI target ports, however, communications between any logical unit and any SCSI target port may be inactive. Two-way communications shall be possible between an application client and its associated SCSI initiator port. Each SCSI target port shall accept commands sent to LUN 0 or the REPORT LUNS well-known logical unit and the task router shall route them to a device server for processing. REPORT LUNS commands (see SPC-4) shall be accepted by the logical unit with the logical unit number zero or the REPORT LUNS well-known logical unit from any SCSI target port and shall return the logical unit inventory available via that SCSI target port. The availability of the same logical unit through multiple SCSI target ports is discovered by matching logical unit name values in the INQUIRY command Device Identification VPD page (see SPC-4).

This standard does not specify or require the definition of any mechanisms by which a SCSI target device would have the ability to discover that it is communicating with multiple SCSI ports that also contain a SCSI initiator port on a single SCSI device. In those SCSI transport protocols where such mechanisms are defined, they shall not have any effect on how commands are processed (e.g., reservations shall be handled as if no such mechanisms exist).

4.8.6 SCSI initiator device view of a multiple port SCSI target device

A SCSI target device may be connected to multiple SCSI domains such that a SCSI initiator port is only able to communicate with its logical units using a single SCSI target port. However, SCSI target devices with multiple SCSI ports may be configured where application clients have the ability to discover that one or more logical units are accessible via multiple SCSI target ports. Figure 28 and figure 29 show two examples of such configurations.

Figure 28 shows a SCSI target device with multiple SCSI ports each containing a SCSI target port participating in a single SCSI domain with two SCSI initiator devices. There are three SCSI devices, one of which has two SCSI

Page: 61

Author: suhlerp Subject: Sticky Note Date: 10/23/2007 6:19:34 PM
Too much white space here

Status
George Penokie Accepted 10/31/2007 3:20:18 PM

Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:21:34 PM
"Two-way communications shall be possible between all logical units and all SCSI target ports, however, communications between any logical unit and any SCSI target port may be inactive. Two-way communications shall be available between each task manager and all task routers. Each SCSI target port shall accept commands sent to LUN 0 or the REPORT LUNS well-known logical unit and the task router shall route them to a device server for processing. REPORT LUNS commands (see SPC-4) shall be accepted by the logical unit with the logical unit number zero or the REPORT LUNS well-known logical unit from any SCSI target port and shall return the logical unit inventory available via that SCSI target port. The availability of the same logical unit through multiple SCSI target ports is discovered by matching logical unit name values in the INQUIRY command Device Identification VPD page (see SPC-4)."
s/b
"Two-way communications shall be possible between all logical units and all SCSI target ports in a SCSI device. However, communications between any logical unit and any SCSI target port in a SCSI device may be inactive. Two-way communications shall be available between each task manager and all task routers in the SCSI target ports in the SCSI device. Each SCSI target port in a SCSI device shall accept commands sent to LUN 0 or the REPORT LUNS well-known logical unit, and the task router in that SCSI target port shall route the commands to a device server in a logical unit in the SCSI device for processing. REPORT LUNS commands (see SPC-4) shall be accepted by the logical unit with the logical unit number zero or the REPORT LUNS well-known logical unit from any SCSI target port in the SCSI device, and the logical unit shall return the logical unit inventory available via that SCSI target port. An application client determines the availability of the same logical unit through multiple SCSI target ports in a SCSI device by matching logical unit name values in the Device Identification VPD page (see SPC-4)."

Status
George Penokie Accepted 10/31/2007 3:24:05 PM

Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:21:56 PM
"A SCSI target device may be connected to multiple SCSI domains such that a SCSI initiator port is only able to communicate with its logical units using a single SCSI target port."
s/b
"A SCSI target device may have SCSI target ports connected to different SCSI domains such that a SCSI initiator port is only able to communicate with the logical units in the SCSI target device using the SCSI target ports in a single SCSI domain."

Status
George Penokie Accepted 10/31/2007 3:26:34 PM

target ports, and two of which have one SCSI initiator port each. There are two target port identifiers and two initiator port identifiers in this SCSI domain. Using the INQUIRY command Device Identification VPD page (see SPC-4), the application clients in each of the SCSI initiator devices have the ability to discover if the logical units in the SCSI target devices are accessible via multiple SCSI target ports and map the configuration of the SCSI target device.

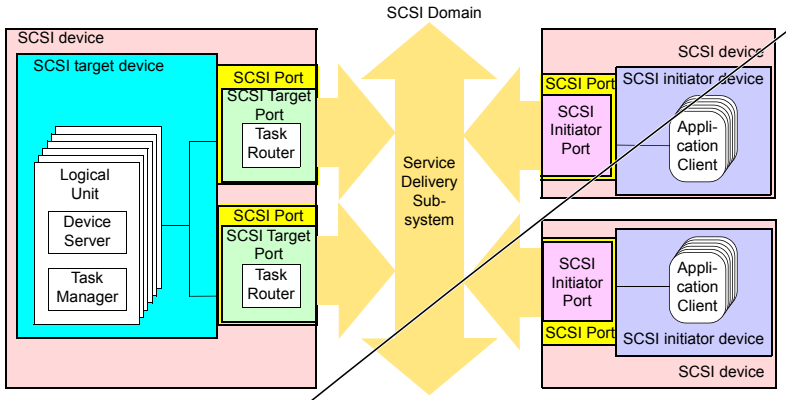


Figure 28 — SCSI target device configured in a single SCSI domain

Figure 29 shows a SCSI target device with multiple SCSI ports each containing a SCSI target port participating in two SCSI domains and a SCSI initiator device with multiple SCSI ports each containing a SCSI initiator port participating in the same two SCSI domains. There is one SCSI target device with two SCSI target ports and one SCSI initiator device with two SCSI initiator ports. There is one target port identifier and one initiator port identifier in each of the two SCSI domains. Using the INQUIRY command Device Identification VPD page (see SPC-4), the application clients in the SCSI initiator device have the ability to discover that logical units in the SCSI target device are accessible via multiple SCSI initiator ports and multiple SCSI target ports and map the configuration. However, application clients may not be able to distinguish between the configuration shown in figure 29 and the configuration shown in figure 30.

Page: 62

Author: relliott Subject: Highlight Date: 10/8/2007 7:27:49 PM
application clients may not be able to distinguish between
s/b
application clients are not required to be able to distinguish between

Status
George Penokie Rejected 12/19/2007 5:10:29 PM -06'00'

Page: 63

Author: relliott Subject: Note Date: 10/29/2007 1:04:52 AM
In figure 30 and its predecessors, add color to at least the logical unit boxes.

Status
George Penokie Accepted 10/31/2007 3:53:47 PM

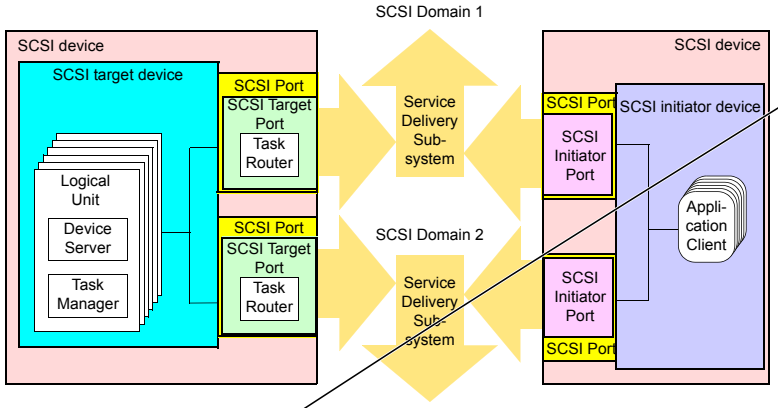


Figure 29 — SCSI target device configured in multiple SCSI domains

Figure 30 shows the same configuration as figure 29 except that the two SCSI domains have been replaced by a single SCSI domain.

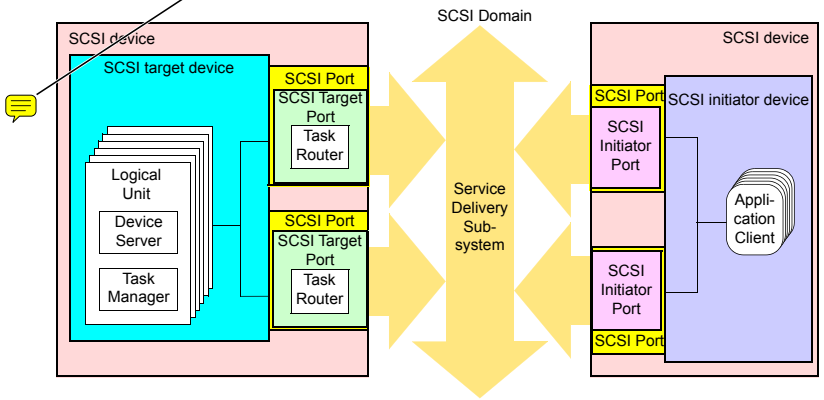


Figure 30 — SCSI target device and SCSI initiator device configured in a single SCSI domain

This model for application client determination of multiple SCSI target port configurations relies on information that is available only to the application clients via commands.

4.8.7 SCSI target device view of a multiple port SCSI initiator device

This standard does not require a SCSI target device to have the ability to detect the presence of a SCSI initiator device with multiple SCSI initiator ports. Therefore, a SCSI target device handles a SCSI initiator device with multiple SCSI initiator ports exactly as it would handle multiple separate SCSI initiator devices (e.g., a SCSI target device handles the configurations shown in figure 29 and figure 30 in exactly the same way it handles the configuration shown in figure 28).

NOTE 6 - The implications of this view of a SCSI initiator device are more far reaching than are immediately apparent (e.g., after a SCSI initiator device makes a persistent exclusive access reservation via one SCSI initiator port, access is denied to the other SCSI initiator port(s) on that same SCSI initiator device).

4.9 The SCSI model for distributed communications

The SCSI model for communications between distributed objects is based on the technique of layering as shown in figure 31.

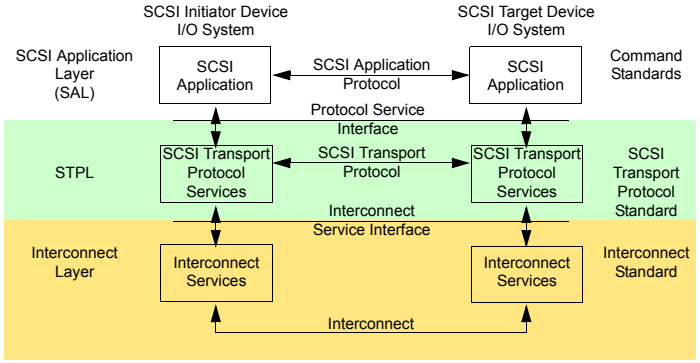


Figure 31 — Protocol service reference model

The layers in this model and the specifications defining the functionality of each layer are denoted by horizontal sequences. A layer consists of peer entities that communicate with one another by means of a protocol. Except for the interconnect layer, such communication is accomplished by invoking services provided by the adjacent layer. The following layers are defined:

a) **SAL:** Clients and servers that originate and process SCSI I/O operations by means of a SCSI application protocol;
b) **STPL:** Services and protocols through which clients and servers communicate; and
c) **Interconnect layer:** Services, signaling mechanism and interconnect subsystem used for the physical transfer of data from sender to receiver. In the SCSI model, the interconnect layer is known as a service delivery subsystem.

The set of SCSI transport protocol services implemented by a service delivery subsystem identify external behavioral requirements that apply to SCSI transport protocol standards. While these SCSI transport protocol services may serve as a guide for designing reusable software or firmware that is adaptable to different SCSI transport protocols, there is no requirement for an implementation to provide the service interfaces specified in this standard.

Page: 64

Author: Mark Evans, WDC Subject: Highlight Date: 10/25/2007 11:10:07 AM
"This standard does not require a SCSI target device to have the ability to detect the presence of a SCSI initiator device with multiple SCSI initiator ports. Therefore, a SCSI target device handles a SCSI initiator device with multiple SCSI initiator ports exactly as it would handle multiple separate SCSI initiator devices (e.g., a SCSI target device handles the configurations shown in figure 29 and figure 30 in exactly the same way it handles the configuration shown in figure 28)."
s/b
"This standard does not require a SCSI target device to be able to detect that a SCSI initiator device contains more than one SCSI initiator port. In the cases where a SCSI target device does not detect that a SCSI initiator device contains more than one SCSI initiator port, the SCSI target device interacts with the SCSI initiator device as if each SCSI initiator port was contained in a separate SCSI initiator device (e.g., a SCSI target device operates in the configurations shown in figure 29 and figure 30 in the same way it operates in the configuration shown in figure 28)."

Status
George Penokie Accepted 10/31/2007 3:57:13 PM

5 SCSI command model

5.1 The Execute Command procedure call

An application client requests the processing of a command by invoking the SCSI transport protocol services described in 5.4, the collective operation of which is modeled in the following procedure call:

Service Response = Execute Command (IN (I_T_L_Q Nexus, CDB, Task Attribute, [Data-In Buffer Size], [Data-Out Buffer], [Data-Out Buffer Size], [CRN], [Task Priority]), OUT ([Data-In Buffer], [Sense Data], [Sense Data Length], Status, [Retry Delay Timer]))

Input arguments:

I_T_L_Q Nexus: The I_T_L_Q nexus identifying the task (see 4.7).

CDB: Command descriptor block (see 5.2).

Task Attribute: A value specifying one of the task attributes defined in 8.6.

Data-In Buffer Size: The number of bytes available for data transfers to the Data-In Buffer (see 5.4.3). SCSI transport protocols may interpret the Data-In Buffer Size to include both the size and the location of the Data-In Buffer.

Data-Out Buffer: A buffer containing command specific information to be sent to the logical unit (e.g., data or parameter lists needed to process the command). The buffer size is indicated by the Data-Out Buffer Size argument. The content of the buffer shall not change during the lifetime of the command (see 5.5) as viewed by the application client.

Data-Out Buffer Size: The number of bytes available for data transfers from the Data-Out Buffer (see 5.4.3).

CRN: When the CRN is used, all sequential commands of an I_T_L nexus shall include a CRN argument that is incremented by one. The CRN shall be set to one for each I_T_L nexus involving the SCSI port after the SCSI port receives a hard reset or detects I_T nexus loss. The CRN shall be set to one after it reaches the maximum CRN value supported by the protocol. The CRN value zero shall be reserved for use as defined by the SCSI transport protocol. It is not an error for the application client to provide a CRN when CRN is not supported by the SCSI transport protocol or logical unit.

Task Priority: The priority assigned to the task (see 8.7).

Page: 69

Author: relliott Subject: Highlight Date: 10/29/2007 10:06:17 AM
A buffer containing

This sentence should reference 5.4.3, where the buffer is described in more detail.

Status
George Penokie Accepted 10/31/2007 4:03:32 PM
Author: relliott Subject: Highlight Date: 12/10/2007 4:44:00 PM -06'00'
sequential commands
s/b
commands sent on an

("sequential" sounds like SSC)

Status
George Penokie Accepted 12/21/2007 10:28:24 AM -06'00'
Author: George Penokie Subject: Highlight Date: 12/21/2007 11:19:17 AM -06'00'
Add the following to the CRN definition << See the SCSI transport protocol standards for rules regarding CRN checking.
>>

Status
George Penokie Accepted 12/21/2007 11:19:34 AM -06'00'
Author: relliott Subject: Note Date: 10/27/2007 1:55:00 PM
Does power loss expected have any impact on CRN?

Status
George Penokie Rejected 12/21/2007 10:31:15 AM -06'00'
Author: George Penokie Subject: Note Date: 12/21/2007 10:31:09 AM -06'00'
It should be the same thing that happens if a CLEAR QUEUE is issued. Which has no impact so therefore CRN should not be impacted by a power loss expected.

Output arguments:

Data-In Buffer: A buffer to contain command specific information returned by the logical unit by the time of command completion. The **Execute Command** procedure call shall not return a status of GOOD or CONDITION MET unless the buffer contents are valid. The application client shall treat the buffer contents as invalid unless the command completes with a status of GOOD or CONDITION MET. While some valid data may be present for other values of status, the application client should rely on additional information from the logical unit (e.g., sense data) to determine the state of the buffer contents. If the command ends with a service response of SERVICE DELIVERY OR TARGET FAILURE, the application client shall consider the buffer to be undefined.

Sense Data: A buffer containing sense data returned in the same I_T_L_Q nexus transaction (see 3.1.51) as a CHECK CONDITION status (see 5.5.6). The buffer length is indicated by the Sense Data Length argument. If the command ends with a service response of SERVICE DELIVERY OR TARGET FAILURE, the application client shall consider the sense data to be undefined.

Sense Data Length: The length in bytes of the Sense Data.

Status: A one-byte field containing command completion status (see 5.3). If the command ends with a service response of SERVICE DELIVERY OR TARGET FAILURE, the application client shall consider status to be undefined.

Retry Delay Timer: Additional information about the indicated status code (see 5.3.2).

Service Response assumes one of the following values:

TASK COMPLETE: A logical unit response indicating that the task has ended. The Status argument shall have one of the values specified in 5.3.

SERVICE DELIVERY OR TARGET FAILURE: The command has been ended due to a service delivery failure (see 3.1.113) or SCSI target device malfunction. All output parameters are invalid.

The SCSI transport protocol events corresponding to a response of TASK COMPLETE or SERVICE DELIVERY OR TARGET FAILURE shall be specified in each SCSI transport protocol standard.

5.2 Command descriptor block (CDB)

The CDB defines the operation to be performed by the device server.

For all commands, if the logical unit detects an invalid parameter in the CDB, then the logical unit shall not process the command.

All CDBs shall have an OPERATION CODE as the first byte.

Some operation codes provide for modification of their operation based on a service action. In such cases, the combination of operation code value and service action code value may be modeled as a single, unique command determinate. The location of the SERVICE ACTION field in the CDB varies depending on the operation code value.

All CDBs shall contain a CONTROL byte (see table 24). The location of the CONTROL byte within a CDB depends on the CDB format (see SPC-4).

Page: 70

Author: relliott Subject: Highlight Date: 10/29/2007 10:06:33 AM
A buffer to contain

This sentence should reference 5.4.3, where the buffer is described in more detail.

Status
George Penokie Accepted 10/31/2007 4:05:24 PM
Author: relliott Subject: Highlight Date: 10/29/2007 10:08:03 AM
Sense Data
s/b
Sense Data (see 5.8.6)

The Data-In Buffer Size field description points to 5.4.3, so the Sense Data Length field description should point to 5.8.6.

Status
George Penokie Accepted 10/31/2007 4:06:37 PM
Author: relliott Subject: Highlight Date: 10/10/2007 2:09:40 PM
output parameters
s/b
output arguments

Status
George Penokie Accepted 10/31/2007 4:07:49 PM
Author: relliott Subject: Note Date: 10/10/2007 4:17:43 PM
After first sentence in 5.2, add "CDB formats are defined in SPC-4."

Status
George Penokie Completed 10/31/2007 4:10:55 PM
Author: George Penokie Subject: Note Date: 10/31/2007 4:10:45 PM
Changed "The CDB defines the operation to be performed by the device server." to "The CDB defines the operation to be performed by the device server. See SPC-4 for the CDB formats."

Author: relliott Subject: Highlight Date: 10/29/2007 11:13:55 AM
parameter
s/b
field

Status
George Penokie Accepted 10/31/2007 4:12:04 PM
Author: relliott Subject: Highlight Date: 10/10/2007 7:58:20 PM
OPERATION CODE
s/b
OPERATION CODE field

Status
George Penokie Accepted 10/31/2007 4:12:09 PM
Author: suhlerp Subject: Cross-Out Date: 10/25/2007 6:59:49 PM

Status
George Penokie Accepted 10/31/2007 4:14:08 PM
Author: suhlerp Subject: Inserted Text Date: 10/25/2007 7:00:02 PM
determinant

Status
George Penokie Completed 10/31/2007 4:14:43 PM
Author: George Penokie Subject: Note Date: 10/31/2007 4:14:39 PM
Deleted the term "determinate"

Table 24 — CONTROL byte

Bit	7	6	5	4	3	2	1	0
	Vendor specific		Reserved			NACA	Obsolete	Obsolete

All SCSI transport protocol standards shall define as mandatory the functionality needed for a logical unit to implement the NACA bit.

The NACA (Normal ACA) bit specifies whether an auto contingent allegiance (ACA) is established if the command returns with CHECK CONDITION status. An NACA bit set to one specifies that an ACA shall be established. An NACA bit set to zero specifies that an ACA shall not be established. The actions for ACA are specified in 5.8.2. Actions that may be required when an ACA is not established are described in 5.8.1. All logical units shall implement support for the NACA value of zero and may support the NACA value of one (i.e., ACA). The ability to support a NACA value of one is indicated with the NORMACA bit in the standard INQUIRY data (see SPC-4).

If the NACA bit is set to one but the logical unit does not support ACA, 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.

5.3 Status

5.3.1 Status codes

The status codes are specified in table 25. Status shall be sent from the device server to the application client whenever a command ends with a service response of TASK COMPLETE.

Table 25 — Status codes

Status Code	Status	Task Ended	Service Response
00h	GOOD	Yes	TASK COMPLETE
02h	CHECK CONDITION	Yes	TASK COMPLETE
04h	CONDITION MET	Yes	TASK COMPLETE
08h	BUSY	Yes	TASK COMPLETE
10h	Obsolete		
14h	Obsolete		
18h	RESERVATION CONFLICT	Yes	TASK COMPLETE
22h	Obsolete		
28h	TASK SET FULL	Yes	TASK COMPLETE
30h	ACA ACTIVE	Yes	TASK COMPLETE
40h	TASK ABORTED	Yes	TASK COMPLETE
All other codes	Reserved		

Definitions for each status code are as follows:

GOOD. This status indicates that the device server has successfully completed the task.

CHECK CONDITION. This status indicates that sense data has been delivered in the buffer defined by the Sense Data argument to the **Execute Command** procedure call (see 5.8.6). Additional actions that are required when CHECK CONDITION status is returned are described in 5.8.1.

Page: 71

Author: relliott Subject: Highlight Date: 12/11/2007 11:08:08 AM -06'00'
returns with CHECK CONDITION status
s/b
is terminated with CHECK CONDITION status

or
ends with CHECK CONDITION status

Status
George Penokie Rejected 12/11/2007 11:09:12 AM -06'00'
Author: George Penokie Subject: Note Date: 12/26/2007 1:38:39 PM -06'00'
Changed to << terminates with CHECK CONDITION status >> Do this globally. The term << terminates >> is used in all cases in which a command or task management function abnormatly ends. The term << completes >> is used in any case in which a command or task management function successfully completes.

Author: relliott Subject: Highlight Date: 10/8/2007 7:06:00 PM
An NACA bit
s/b
a NACA bit

(last line of this paragraph uses "a naca" so it must be pronounced "a nak-ka")

Status
George Penokie Accepted 12/26/2007 1:46:22 PM -06'00'
Author: George Penokie Subject: Note Date: 12/21/2007 10:38:22 AM -06'00'
Make this change globally

Author: relliott Subject: Highlight Date: 10/8/2007 7:06:21 PM
An NACA bit
s/b
A NACA bit

Status
George Penokie Accepted 12/26/2007 1:46:34 PM -06'00'
Author: Mark Evans, WDC Subject: Highlight Date: 10/25/2007 11:11:08 AM
"This status indicates that the device server has successfully completed the task."
s/b
"This status indicates that the device server has completed the task without error."

Status
George Penokie Accepted 10/31/2007 4:16:37 PM

CONDITION MET. The use of this status is limited to commands for which it is specified (see the PRE-FETCH commands in SBC-3).

BUSY. This status indicates that the logical unit is busy. This status shall be returned whenever a logical unit is temporarily unable to accept a command. The recommended application client recovery action is to issue the command again at a later time.

If the UA_INTLCK_CTRL field in the Control mode page contains 11b (see SPC-4), termination of a command with BUSY status shall cause a unit attention condition to be established for the SCSI initiator port that sent the command with an additional sense code set to PREVIOUS BUSY STATUS unless a PREVIOUS BUSY STATUS unit attention condition already exists.

Retry delay timer, when supported by a protocol, may provide the SCSI initiator port with more information on when the command should be retransmitted (see table 26).

RESERVATION CONFLICT. This status shall be returned whenever a command attempts to access a logical unit in a way that conflicts with an existing reservation. (See the PERSISTENT RESERVE OUT command and PERSISTENT RESERVE IN command in SPC-4.)

If the UA_INTLCK_CTRL field in the Control mode page contains 11b (see SPC-4), termination of a command with RESERVATION CONFLICT status shall cause a unit attention condition to be established for the SCSI initiator port that sent the command with an additional sense code set to PREVIOUS RESERVATION CONFLICT STATUS unless a PREVIOUS RESERVATION CONFLICT STATUS unit attention condition already exists.

TASK SET FULL. When the logical unit has at least one task in the task set for an I_T nexus and a lack of task set resources prevents accepting a received task from that I_T nexus into the task set, TASK SET FULL status shall be returned. When the logical unit has no task in the task set for an I_T nexus and a lack of task set resources prevents accepting a received task from that I_T nexus into the task set, BUSY status should be returned.

The logical unit should allow at least one command in the task set for each supported I_T nexus (i.e., for each SCSI target port, allow at least one command from each SCSI initiator port that has identified itself to the SCSI target port in a SCSI transport protocol specific manner (e.g., login), or by the successful transmission of a c command).

Retry delay timer, when supported by a protocol, may provide the SCSI initiator port with more information on when the command should be retransmitted (see table 26).

If the UA_INTLCK_CTRL field in the Control mode page contains 11b (see SPC-4), termination of a command with TASK SET FULL status shall cause a unit attention condition to be established for the SCSI initiator port that sent the command with an additional sense code set to PREVIOUS TASK SET FULL STATUS unless a PREVIOUS TASK SET FULL STATUS unit attention condition already exists.

ACA ACTIVE. This status shall be returned as described in 5.8.2.2 and 5.8.2.3 when an ACA exists within a task set. The application client may reissue the command on the same I_T nexus after the ACA condition has been cleared.

TASK ABORTED. This status shall be returned when a task is aborted by a command or task management function on another I_T nexus and the Control mode page TAS bit is set to one (see 5.6).

5.3.2 Retry delay timer codes

The retry delay timer codes are specified in table 26 and provide additional information about the reason for the status code.

Page: 72

Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:20:04 PM
"If the UA_INTLCK_CTRL field in the Control mode page contains 11b (see SPC-4), termination of a command with BUSY status shall cause a unit attention condition to be established for the SCSI initiator port that sent the command with an additional sense code set to PREVIOUS BUSY STATUS unless a PREVIOUS BUSY STATUS unit attention condition already exists."
s/b
"If the UA_INTLCK_CTRL field in the Control mode page contains 11b (see SPC-4), then completion of a command with BUSY status shall cause a unit attention condition to be established for the I_T nexus on which the command was received with an additional sense code set to PREVIOUS BUSY STATUS unless a PREVIOUS BUSY STATUS unit attention condition already exists."

Status
George Penokie Rejected 12/21/2007 10:48:19 AM -06'00'
Author: George Penokie Subject: Note Date: 12/21/2007 10:48:13 AM -06'00'
Changed to << If the ua_intlck_ctrl field in the Control mode page contains 11b (see SPC-4), completion of a command with BUSY status shall cause a unit attention condition to be established for the SCSI initiator port on the I_T nexus that sent the command with an additional sense code set to PREVIOUS BUSY STATUS unless a PREVIOUS BUSY STATUS unit attention condition already exists. >>

Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:20:19 PM
"Retry delay timer,"
s/b
"The retry delay timer,"

Status
George Penokie Accepted 10/31/2007 4:18:31 PM
Author: relliott Subject: Highlight Date: 10/8/2007 7:29:55 PM
Retry delay timer, when supported by a protocol, may provide the SCSI initiator port with more information on when the command should be retransmitted (see table 26).
s/b
Retry delay timer, when supported by a SCSI transport protocol, provides the SCSI initiator port with more information about when the command should be retransmitted (see table 26).

(same comment on both BUSY and TASK SET FULL descriptions)

Status
George Penokie Accepted 10/31/2007 4:21:42 PM
Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:20:39 PM
"This status shall be returned whenever a command attempts to access a logical unit in a way that conflicts with an existing reservation."
s/b
"This status shall be returned whenever a command is directed by an application client to access a logical unit in a way that conflicts with an existing reservation."

Status
George Penokie Rejected 12/21/2007 10:51:57 AM -06'00'
Author: George Penokie Subject: Note Date: 12/11/2007 11:14:48 AM -06'00'
Rob does not like this wording I suggest << is sent by an application client to a logical unit in a way that conflicts with an >>

Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:20:46 PM
"If the UA_INTLCK_CTRL field in the Control mode page contains 11b (see SPC-4), termination of a command with RESERVATION CONFLICT status shall cause a unit attention condition to be established for the SCSI initiator port that sent the command with an additional sense code set to PREVIOUS RESERVATION CONFLICT STATUS unless a PREVIOUS RESERVATION CONFLICT STATUS unit attention condition already exists."
s/b
"If the UA_INTLCK_CTRL field in the Control mode page contains 11b (see SPC-4), then completion of a command with RESERVATION CONFLICT status shall cause a unit attention condition to be established for the I_T nexus on which the command was received with an additional sense code set to PREVIOUS RESERVATION CONFLICT STATUS unless a PREVIOUS RESERVATION CONFLICT STATUS unit attention condition already exists."

Status
George Penokie Rejected 12/21/2007 10:49:16 AM -06'00'
Author: George Penokie Subject: Note Date: 12/21/2007 10:49:09 AM -06'00'
Changed to << If the ua_intlck_ctrl field in the Control mode page contains 11b (see SPC-4), completion of a command with RESERVATION CONFLICT status shall cause a unit attention condition to be established for the SCSI initiator port on the I_T nexus that sent the command with an additional sense code set to PREVIOUS RESERVATION CONFLICT STATUS unless a PREVIOUS RESERVATION CONFLICT STATUS unit attention condition already exists. >>

Author: Mark Evans, WDC Subject: Highlight Date: 10/25/2007 11:17:08 AM
"...prevents accepting a received task from that I_T nexus into the task set,"
s/b
"...prevents the logical unit from accepting an additional task received from that I_T nexus into the task set,"

Status
George Penokie Accepted 10/31/2007 4:29:08 PM
Author: Emulex Subject: Highlight Date: 10/30/2007 1:58:47 PM
Emulex-012
Page: 72 task set full end of second paragraph there is an extraneous "c" in sentence

Status
George Penokie Accepted 10/31/2007 4:30:15 PM
Author: Mark Evans, WDC Subject: Highlight Date: 10/25/2007 11:19:05 AM
"(i.e., for each SCSI target port, allow at least one command from each SCSI initiator port that has identified itself to the SCSI target port in a SCSI transport protocol specific manner (e.g., login), or by the successful transmission of a c command)."
s/b
"(i.e., a logical unit should allow at least one command into the task set for any I_T nexus that has been identified in a SCSI transport protocol specific manner (e.g., a login), or by the successful reception of a command)."

Status
George Penokie Accepted 12/21/2007 10:55:20 AM -06'00'
Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:21:27 PM
"Retry delay timer,"
s/b

Comments from page 72 continued on next page

CONDITION MET. The use of this status is limited to commands for which it is specified (see the PRE-FETCH commands in SBC-3).

BUSY. This status indicates that the logical unit is busy. This status shall be returned whenever a logical unit is temporarily unable to accept a command. The recommended application client recovery action is to issue the command again at a later time.

If the UA_INTLCK_CTRL field in the Control mode page contains 11b (see SPC-4), termination of a command with BUSY status shall cause a unit attention condition to be established for the SCSI initiator port that sent the command with an additional sense code set to PREVIOUS BUSY STATUS unless a PREVIOUS BUSY STATUS unit attention condition already exists.

Retry delay timer, when supported by a protocol, may provide the SCSI initiator port with more information on when the command should be retransmitted (see table 26).

RESERVATION CONFLICT. This status shall be returned whenever a command attempts to access a logical unit in a way that conflicts with an existing reservation. (See the PERSISTENT RESERVE OUT command and PERSISTENT RESERVE IN command in SPC-4.)

If the UA_INTLCK_CTRL field in the Control mode page contains 11b (see SPC-4), termination of a command with RESERVATION CONFLICT status shall cause a unit attention condition to be established for the SCSI initiator port that sent the command with an additional sense code set to PREVIOUS RESERVATION CONFLICT STATUS unless a PREVIOUS RESERVATION CONFLICT STATUS unit attention condition already exists.

TASK SET FULL. When the logical unit has at least one task in the task set for an I_T nexus and a lack of task set resources prevents accepting a received task from that I_T nexus into the task set, TASK SET FULL status shall be returned. When the logical unit has no task in the task set for an I_T nexus and a lack of task set resources prevents accepting a received task from that I_T nexus into the task set, BUSY status should be returned.

The logical unit should allow at least one command in the task set for each supported I_T nexus (i.e., for each SCSI target port, allow at least one command from each SCSI initiator port that has identified itself to the SCSI target port in a SCSI transport protocol specific manner (e.g., login), or by the successful transmission of a c command).

Retry delay timer, when supported by a protocol, may provide the SCSI initiator port with more information on when the command should be retransmitted (see table 26).

If the UA_INTLCK_CTRL field in the Control mode page contains 11b (see SPC-4), termination of a command with TASK SET FULL status shall cause a unit attention condition to be established for the SCSI initiator port that sent the command with an additional sense code set to PREVIOUS TASK SET FULL STATUS unless a PREVIOUS TASK SET FULL STATUS unit attention condition already exists.

ACA ACTIVE. This status shall be returned as described in 5.8.2.2 and 5.8.2.3 when an ACA exists within a task set. The application client may reissue the command on the same I_T nexus after the ACA condition has been cleared.

TASK ABORTED. This status shall be returned when a task is aborted by a command or task management function on another I_T nexus and the Control mode page TAS bit is set to one (see 5.6).

5.3.2 Retry delay timer codes

The retry delay timer codes are specified in table 26 and provide additional information about the reason for the status code.

"The retry delay timer,"

Status
George Penokie Accepted 10/31/2007 4:31:03 PM
Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:21:36 PM
"If the UA_INTLCK_CTRL field in the Control mode page contains 11b (see SPC-4), termination of a command with TASK SET FULL status shall cause a unit attention condition to be established for the SCSI initiator port that sent the command with an additional sense code set to PREVIOUS TASK SET FULL STATUS unless a PREVIOUS TASK SET FULL STATUS unit attention condition already exists."
s/b
"If the UA_INTLCK_CTRL field in the Control mode page contains 11b (see SPC-4), then completion of a command with TASK SET FULL status shall cause a unit attention condition to be established for the I_T nexus on which the command was received with an additional sense code set to PREVIOUS TASK SET FULL STATUS unless a PREVIOUS TASK SET FULL STATUS unit attention condition already exists."

Status
George Penokie Rejected 12/21/2007 10:57:09 AM -06'00'
Author: George Penokie Subject: Note Date: 12/21/2007 10:57:05 AM -06'00'
Changed to << If the ua_intlck_ctrl field in the Control mode page contains 11b (see SPC-4), completion of a command with TASK SET FULL status shall cause a unit attention condition to be established for the SCSI initiator port on gthe I_T nexus that sent the command with an additional sense code set to PREVIOUS TASK SET FULL STATUS unless a PREVIOUS TASK SET FULL STATUS unit attention condition already exists. >>

Page: 73

Table 26 — Retry delay timer

Status code	Retry delay timer code	Description
BUSY	0000h	No addition information (i.e., the same as normal busy)
	0001h - FFEFh	The number of 100 milliseconds increments the application client should wait before sending another command to the logical unit on any I_T nexus.
	FFF0h - FFFDh	Reserved
	FFFEh	The application client should stop sending commands on this I_T_L nexus.
	FFFFh	The logical unit is not able to accept the command because it is servicing too many other I_T nexus.
TASK SET FULL	0000h	No addition information (i.e., the same as normal task set full)
	0001h - FFEFh	The application client should wait before sending another command to the logical unit on any I_T nexus until: a) at least the number of 100 milliseconds increments indicated in the RETRY DELAY TIMER CODE field have elapsed; or b) a command addressed to the logical unit on any I_T nexus completes.
	FFF0h - FFFFh	Reserved
GOOD	0000h - FFFFh	Reserved
CHECK CONDITION	0000h - FFFFh	Reserved
CONDITION MET	0000h - FFFFh	Reserved
RESERVATION CONFLICT	0000h - FFFFh	Reserved
ACA ACTIVE	0000h - FFFFh	Reserved
TASK ABORTED	0000h - FFFFh	Reserved

5.3.3 Status precedence

If a device server detects that more than one of the following conditions applies to a completed task, it shall select the condition to report based on the following precedence:

1) An ACA ACTIVE status;
2) A CHECK CONDITION status for any of the following unit attention conditions (i.e., with a sense key set to UNIT ATTENTION and one of the following additional sense codes):
   A) POWER ON, RESET, OR BUS DEVICE RESET OCCURRED;
   B) POWER ON OCCURRED;
   C) SCSI BUS RESET OCCURRED;
   D) MICROCODE HAS BEEN CHANGED;
   E) BUS DEVICE RESET FUNCTION OCCURRED;
   F) DEVICE INTERNAL RESET; or
   G) I_T NEXUS LOSS OCCURRED;
3) A RESERVATION CONFLICT status;
   and
4) A status of:
   A) CHECK CONDITION, for any reason not listed in 2);

Author: relliott Subject: Highlight Date: 10/27/2007 1:23:42 PM
Retry delay timer
s/b
either
a) Retry delay time. Reason: the timer is the entity initialized to this value, not the value itself.

or
b) Additional status code. Reason: it is likely that this field will have a different meaning if used by any of the other status codes. A more generic name would be clearer. There can still be a "retry delay timer" that uses this code value for BUSY and TASK SET FULL.

Status
George Penokie Completed 12/14/2007 4:31:59 PM -06'00'
Author: George Penokie Subject: Note Date: 12/17/2007 10:41:56 AM -06'00'
Changed all << retry delay timer >> to << additional status qualifier >>.

Author: relliott Subject: Highlight Date: 10/29/2007 9:41:10 AM
busy
s/b
BUSY

Status
George Penokie Accepted 10/31/2007 4:33:03 PM
Author: relliott Subject: Note Date: 10/29/2007 9:48:15 AM
Some designs cannot return this information on a per I_T_L basis, but can return it on a per I_T basis. The target device should be able to return whichever scope it wants (perhaps with a "should" preferring the I_T_L scope).

Either:
a) Add a bit indicating scope (logical unit, target port, target device). This requires changing the transport protocols.

b) redefine the code values:

0001h - 4FFFh wait for this logical unit (any I, any T, this L)
5000h - 9FFFh wait for this target port (any I, this T, any L)
A000h - EFFFh wait for this target device (any I, any T, any L)
F000h - FFEFh reserved

The current maximum of FFEFh is 65519, so the current field supports 6551.9 seconds (109 minutes). Reducing that range by a third shouldn't overload a fabric with retries.

Lack of results on a google search hints that this has not been widely implemented yet, so a change may still be viable.

Status
George Penokie Rejected 1/17/2008 7:12:54 PM -06'00'
Author: George Penokie Subject: Note Date: 1/17/2008 1:53:52 PM -06'00'
Make this with a 2 bit field that would be the 2 MSBs of the retry delay timer. Where 00 - logical unit , 01 - target port, 10 - target device, 11 - Reserved. Look into using the LSB for legacy reasons if necessary.

Author: relliott Subject: Highlight Date: 10/29/2007 9:40:40 AM
addition
s/b
additional

Status
George Penokie Accepted 10/31/2007 4:34:31 PM
Author: relliott Subject: Highlight Date: 10/29/2007 9:41:19 AM
task set full
s/b
TASK SET FULL

Status
George Penokie Accepted 10/31/2007 4:34:06 PM
Author: relliott Subject: Note Date: 10/16/2007 6:57:03 PM
Replace the GOOD through TASK ABORTED rows with:

All others 0000h - FFFFh Reserved

That covers all the reserved status codes (table 25 defines 256 total codes).

Status
George Penokie Rejected 12/21/2007 11:02:14 AM -06'00'
Author: George Penokie Subject: Note Date: 12/21/2007 11:02:09 AM -06'00'
Added a row that covers all other status code and made them reserved.

B) GOOD;
C) CONDITION MET; or
D) TASK ABORTED.

NOTE 7 - The names of the unit attention conditions listed in this subclause (e.g., SCSI BUS RESET OCCURRED) are based on usage in previous versions of this standard. The use of these unit attention condition names is not to be interpreted as a description of how the unit attention conditions are represented by any given SCSI transport protocol.

A device server may report the following status codes with any level of precedence:

a) BUSY status;
b) TASK SET FULL status; or
c) CHECK CONDITION status with a sense key set to ILLEGAL REQUEST.

5.4 SCSI transport protocol services in support of Execute Command

5.4.1 Overview

The SCSI transport protocol services that support the **Execute Command** procedure call are described in 5.4. Two groups of SCSI transport protocol services are described. The SCSI transport protocol services that support the delivery of the command and status are described in 5.4.2. The SCSI transport protocol services that support the data transfers associated with processing a command are described in 5.4.3.

5.4.2 Command and Status SCSI transport protocol services

5.4.2.1 Command and Status SCSI transport protocol services overview

All SCSI transport protocol standards shall define the SCSI transport protocol specific requirements for implementing the **Send SCSI Command** request (see 5.4.2.2), the **SCSI Command Received** indication (see 5.4.2.3), the **Send Command Complete** response (see 5.4.2.4), and the **Command Complete Received** confirmation (see 5.4.2.5) SCSI transport protocol services.

All SCSI initiator devices shall implement the **Send SCSI Command** request and the **Command Complete Received** confirmation SCSI transport protocol services as defined in the applicable SCSI transport protocol standards. All SCSI target devices shall implement the **SCSI Command Received** indication and the **Send Command Complete** response SCSI transport protocol services as defined in the applicable SCSI transport protocol standards.

5.4.2.2 Send SCSI Command transport protocol service request

An application client uses the Send SCSI Command transport protocol service request to request that a SCSI initiator port send a SCSI command.

Send SCSI Command transport protocol service request:

Send SCSI Command (IN (I_T_L_Q Nexus, CDB, Task Attribute, [Data-In Buffer Size], [Data-Out Buffer], [Data-Out Buffer Size], [CRN], [Task Priority], [First Burst Enabled]))

Page: 74

Author: Network Appliance Subject: Note Date: 10/30/2007 10:34:15 AM
From proposal 07-450 - Add the following to the end of section 5.3.3
A pending I_T nexus unit attention (e.g. REPORTED LUNS DATA HAS CHANGED) should be reported with a higher precedence than ILLEGAL REQUEST when an incorrect LUN is addressed (see 5.8.4).

Status
George Penokie Rejected 11/8/2007 7:05:19 PM -06'00'
Author: George Penokie Subject: Note Date: 11/8/2007 7:03:07 PM -06'00'
Added the following to section 5.3.3 "Any unit attention condition that was established for all logical units should be reported with a higher precedence than a CHECK CONDITION status with a sense key set to ILLEGAL REQUEST and an additional sense code set to LOGICAL UNIT NOT SUPPORTED."

Author: relliott Subject: Highlight Date: 11/20/2007 2:20:40 PM -06'00'
Combine the last 3 sentences into an a)b) list

Status
George Penokie Completed 11/20/2007 2:24:36 PM -06'00'
Author: George Penokie Subject: Note Date: 11/20/2007 2:24:31 PM -06'00'
Replaced with: The following two groups of SCSI transport protocol services are described:
The SCSI transport protocol services that support the delivery of the command and status (see 5.4.2); and
The SCSI transport protocol services that support the data transfers associated with processing a command (see 5.4.3).

Author: relliott Subject: Highlight Date: 10/8/2007 5:24:13 PM
Status
s/b
status

Status
George Penokie Accepted 10/31/2007 4:35:56 PM
Author: relliott Subject: Highlight Date: 10/8/2007 5:24:18 PM
Status
s/b
status

Status
George Penokie Accepted 10/31/2007 4:36:00 PM
Author: relliott Subject: Highlight Date: 12/11/2007 11:20:03 AM -06'00'
SCSI command
s/b
command

since "SCSI" is not used anywhere else

Status
George Penokie Accepted 12/11/2007 11:20:18 AM -06'00'

Input arguments:

I_T_L_Q Nexus: The I_T_L_Q nexus identifying the task (see 4.7).

CDB: Command descriptor block (see 5.2).

Task Attribute: A value specifying one of the task attributes defined in 8.6. For specific requirements on the Task Attribute argument see 5.1.

Data-In Buffer Size: The number of bytes available for data transfers to the Data-In Buffer (see 5.4.3). SCSI transport protocols may interpret the Data-In Buffer Size to include both the size and the location of the Data-In Buffer.

Data-Out Buffer: A buffer containing command specific information to be sent to the logical unit (e.g., data or parameter lists needed to process the command (see 5.1)). The content of the Data-Out Buffer shall not change during the lifetime of the command (see 5.5) as viewed by the application client.

Data-Out Buffer Size: The number of bytes available for data transfers from the Data-Out Buffer (see 5.4.3).

CRN: When CRN is used, all sequential commands of an I_T_L nexus shall include a CRN argument that is incremented by one (see 5.1).

Task Priority: The priority assigned to the task (see 8.7).

First Burst Enabled: An argument specifying that a SCSI transport protocol specific number of bytes from the Data-Out Buffer shall be delivered to the logical unit without waiting for the device server to invoke the **Receive Data-Out** SCSI transport protocol service.

5.4.2.3 SCSI Command Received transport protocol service indication

The task router (see 4.5.8) uses the SCSI Command Received transport protocol service indication to notify a task manager that it has received a SCSI command.

SCSI Command Received transport protocol service indication:

SCSI Command Received **(IN (I_T_L_Q Nexus, CDB, Task Attribute, [CRN], [Task Priority], [First Burst Enabled]))**

Input arguments:

I_T_L_Q Nexus: The I_T_L_Q nexus identifying the task (see 4.7).

CDB: Command descriptor block (see 5.2).

Task Attribute: A value specifying one of the task attributes defined in 8.6. For specific requirements on the Task Attribute argument see 5.1.

CRN: When a CRN argument is used, all sequential commands of an I_T_L nexus shall include a CRN argument that is incremented by one (see 5.1).

Task Priority: The priority assigned to the task (see 8.7).

First Burst Enabled: An argument specifying that a SCSI transport protocol specific number of bytes from the Data-Out Buffer are being delivered to the logical unit without waiting for the device server to invoke the **Receive Data-Out** SCSI transport protocol service.

Page: 75

Author: relliott Subject: Highlight Date: 12/11/2007 11:24:57 AM -06'00'
sequential commands
s/b
commands sent on an

("sequential" sounds like SSC)

Status
George Penokie Accepted 12/11/2007 11:25:28 AM -06'00'

Author: relliott Subject: Highlight Date: 12/11/2007 11:27:50 AM -06'00'
SCSI command
s/b
command

to match general usage

Status
George Penokie Accepted 12/11/2007 11:28:29 AM -06'00'

Author: relliott Subject: Highlight Date: 12/11/2007 11:30:40 AM -06'00'
Also, saying the CRN shall be anything isn't really an appropriate rule for SCSI Command Received(). SCSI Command Receive() delivers the CRN that the command happens to have arrived with.

If the target port is expected to check the CRN values and block them from running out of order, then it doesn't need to pass along the value to the device server - the order in which it invokes SCSI Command Received () suffices. I don't think that's the way this is supposed to work though.

Status
George Penokie Rejected 12/21/2007 11:13:32 AM -06'00'

Author: George Penokie Subject: Note Date: 12/21/2007 11:13:27 AM -06'00'
Changed to << When a CRN argument is used, all commands on an I_T_L nexus include a CRN argument (see 5.1).
>> also made the same change in the Send SCSI Commands.

Author: George Penokie Subject: Highlight Date: 12/11/2007 11:31:13 AM -06'00'
sequential commands
s/b
commands sent on an

("sequential" sounds like SSC)

Status
George Penokie Accepted 12/11/2007 11:31:13 AM -06'00'

5.4.2.4 Send Command Complete transport protocol service response

A device server uses the Send Command Complete transport protocol service response to request that a SCSI target port transmit command complete information.

Send Command Complete transport protocol service response**:**

Send Command Complete (IN (I_T_L_Q Nexus, [Sense Data], [Sense Data Length], Status, Service Response, [Retry Delay Timer]))

Input arguments:

I_T_L_Q Nexus: The I_T_L_Q nexus identifying the task (see 4.7).

Sense Data: If present, a Sense Data argument instructs the SCSI target port to return sense data to the SCSI initiator port (see 5.8.6).

Sense Data Length: The length in bytes of the sense data to be returned to the SCSI initiator port.

Status: Command completion status (see 5.1).

Service Response: Possible service response information for the command (see 5.1).

Retry Delay Timer: The Retry Delay Timer code for the command (see 5.3.2).

5.4.2.5 Command Complete Received transport protocol service confirmation

A SCSI initiator port uses the Command Complete Received transport protocol service confirmation to notify an application client that it has received command complete information.

Command Complete Received transport protocol service confirmation:

Command Complete Received (IN (I_T_L_Q Nexus, [Data-In Buffer], [Sense Data], [Sense Data Length], Status, Service Response, [Retry Delay Timer]))

Input arguments:

I_T_L_Q Nexus: The I_T_L_Q nexus identifying the task (see 4.7).

Data-In Buffer: A buffer containing command specific information returned by the logical unit on command completion (see 5.1).

Sense Data: Sense data returned in the same I_T_L_Q nexus transaction (see 3.1.51) as a CHECK CONDITION status (see 5.8.6).

Sense Data Length: The length in bytes of the received sense data.

Status: Command completion status (see 5.1).

Service Response: Service response for the command (see 5.1).

Retry Delay Timer: The Retry Delay Timer code for the command (see 5.3.2).

5.4.3 Data transfer SCSI transport protocol services

5.4.3.1 Introduction

The data transfer services described in 5.4.3 provide mechanisms for moving data to and from the SCSI initiator port in response to commands transmitted using the **Execute Command** procedure call. All SCSI transport protocol standards shall define the protocols required to implement these services.

Page: 76

Author: relliott Subject: Highlight Date: 12/11/2007 11:34:00 AM -06'00'
transmitted using the Execute Command procedure call.

commands are not transmitted using that. Commands are transmitted with Send SCSI Command () and received with SCSI Command Received().

Execute Command () is just an abstraction for the combination protocol services, including the data transfer protocol services.

Status
George Penokie Rejected 12/21/2007 11:27:28 AM -06'00'
Author: George Penokie Subject: Note Date: 12/21/2007 11:27:23 AM -06'00'
Changed to << The data transfer services described in 5.4.3 provide mechanisms for moving data to and from the SCSI initiator port while processing commands. >>

The application client's Data-In Buffer and/or Data-Out Buffer each appears to the device server as a single, logically contiguous block of memory large enough to hold all the data required by the command (see figure 38). This standard allows either unidirectional or bidirectional data transfer. The processing of a command may require the transfer of data from the application client using the Data-Out Buffer, or to the application client using the Data-In Buffer, or both to and from the application client using both the Data-In Buffer and the Data-Out Buffer.

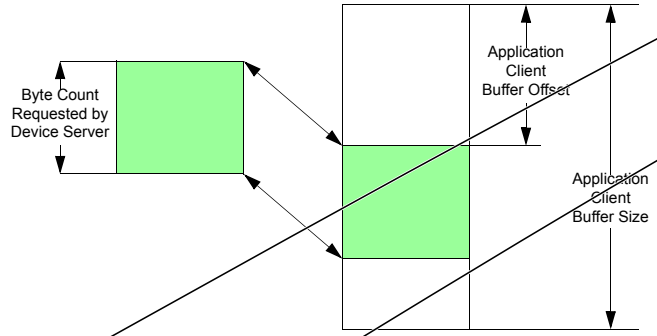


Figure 38 — Model for Data-In and Data-Out data transfers

This standard assumes that the buffering resources available to the logical unit are limited and may be less than the amount of data that is capable of being transferred in one command. Such data needs to be moved between the application client and the media in segments that are smaller than the transfer size specified in the command. The amount of data moved per segment is usually a function of the buffering resources available to the logical unit. Figure 38 shows the model for such incremental data transfers.

SCSI transport protocols may allow logical units to accept the initial portion of the Data-Out Buffer data, called the first burst, along with the command without waiting for the device server to invoke the **Receive Data-Out** SCSI transport protocol service. This is modeled using **Receive Data-Out** protocol service calls for which the SCSI transport protocol may have moved the first burst prior to the call.

SCSI transport protocols that define a first burst capability shall include the First Burst Enabled argument in their definitions for the **Send SCSI Command** and **SCSI Command Received** transport protocol services. Logical units that implement the first burst capability shall implement the FIRST BURST SIZE field in the Disconnect-Reconnect mode page (see SPC-4).

The movement of data between the application client and device server is controlled by the following arguments:

Application Client Buffer Size: The total number of bytes in the application client's buffer (i.e., equivalent to Data-In Buffer Size for the Data-In Buffer or equivalent to Data-Out Buffer Size for the Data-Out Buffer).

Application Client Buffer Offset: Offset in bytes from the beginning of the application client's buffer (Data-In or Data-Out) to the first byte of transferred data.

Byte Count Requested by Device Server: Number of bytes to be moved by the data transfer request.

Page: 77

Author: Mark Evans, WDC Subject: Highlight Date: 10/25/2007 3:24:19 PM
"This standard assumes that the buffering resources available to the logical unit are limited and may be less than the amount of data that is capable of being transferred in one command."
s/b
"This standard assumes that the buffering resources available to a logical unit are limited, and the buffer in the logical unit may not be capable of containing all of the data required to be transferred for one command."

Status
George Penokie Accepted 10/31/2007 4:38:02 PM

Author: Mark Evans, WDC Subject: Highlight Date: 10/25/2007 11:27:40 AM
"...media..."
s/b
"...logical unit..."

Status
George Penokie Accepted 12/21/2007 11:32:47 AM -06'00'

For any specific data transfer SCSI transport protocol service request, the **Byte Count Requested by Device Server** is less than or equal to the combination of **Application Client Buffer Size** minus the **Application Client Buffer Offset**.

If a SCSI transport protocol supports random buffer access, the offset and byte count specified for each data segment to be transferred may overlap. In this case the total number of bytes moved for a command is not a reliable indicator of highest byte transferred and shall not be used by a SCSI initiator device or SCSI target device implementation to determine whether all data has been transferred.

All SCSI transport protocol standards shall define support for a resolution of one byte for the Application Client Buffer Size argument.

SCSI transport protocol standards may define restrictions on the resolution of the Application Client Buffer Offset argument. SCSI transport protocol standards may define restrictions on the resolution of the Request Byte Count argument for any call to **Send Data-In** or any call to **Receive Data-Out** that does not transfer the last byte of the Application Client Buffer.

Random buffer access occurs when the device server requests data transfers to or from segments of the application client's buffer that have an arbitrary offset and byte count. Buffer access is sequential when successive transfers access a series of increasing, adjoining buffer segments. Support for random buffer access by a SCSI transport protocol standard is optional. A device server implementation designed for any SCSI transport protocol implementation should be prepared to use sequential buffer access when necessary.

The STPL confirmed services specified in 5.4.3.2 and 5.4.3.3 are used by the device server to request the transfer of data to or from the application client Data-In Buffer or Data-Out Buffer, respectively. The SCSI initiator device SCSI transport protocol service interactions are unspecified.

This standard provides only for the transfer phases to be sequential. Provision for overlapping transfer phases is outside the scope of this standard.

5.4.3.2 Data-In delivery service

5.4.3.2.1 Send Data-In transport protocol service request

A device server uses the Send Data-In transport protocol service request to request that a SCSI target port send data.

Send Data-In transport protocol service request**:**

Send Data-In (IN (I_T_L_Q Nexus, Device Server Buffer, Application Client Buffer Offset, Request Byte Count))

Input argument:

I_T_L_Q Nexus: The I_T_L_Q nexus identifying the task (see 4.7).

Device Server Buffer: The buffer in the device server from which data is to be transferred.

Application Client Buffer Offset: Offset in bytes from the beginning of the application client's buffer (i.e., the Data-In Buffer) to the first byte of transferred data.

Request Byte Count: Number of bytes to be moved by this request.

5.4.3.2.2 Data-In Delivered transport protocol service confirmation

A SCSI target port uses the Data-In Delivered transport protocol service confirmation to notify a device server that it has sent data.

Page: 78

Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:20:43 PM
"Random buffer access occurs when the device server requests data transfers to or from segments of the application client's buffer that have an arbitrary offset and byte count. Buffer access is sequential when successive transfers access a series of increasing, adjoining buffer segments. Support for random buffer access by a SCSI transport protocol standard is optional. A device server implementation designed for any SCSI transport protocol implementation should be prepared to use sequential buffer access when necessary."
s/b
Move this paragraph above the one that begins, "If a SCSI transport protocol supports random buffer access,".

Status
George Penokie Accepted 10/31/2007 4:39:44 PM

Author: relliott Subject: Highlight Date: 10/16/2007 7:11:11 PM
interactions
s/b
interactions for data transfers

Status
George Penokie Accepted 10/31/2007 4:41:49 PM

Data-In Delivered transport protocol service confirmation:

Data-In Delivered (IN (I_T_L_Q Nexus, Delivery Result))

This confirmation notifies the device server that the specified data was successfully delivered to the application client buffer, or that a service delivery subsystem error occurred while attempting to deliver the data.

Input arguments:

I_T_L_Q Nexus: The I_T_L_Q nexus identifying the task (see 4.7).

Delivery Result: an encoded value representing one of the following:
DELIVERY SUCCESSFUL: The data was delivered successfully.
DELIVERY FAILURE: A service delivery subsystem error occurred while attempting to deliver the data.

5.4.3.3 Data-Out delivery service

5.4.3.3.1 Receive Data-Out transport protocol service request

A device server uses the Receive Data-Out transport protocol service request to request that a SCSI target port receive data.

Receive Data-Out transport protocol service request:

Receive Data-Out (IN (I_T_L_Q Nexus, Application Client Buffer Offset, Request Byte Count, Device Server Buffer))

Input arguments:

I_T_L_Q Nexus: The I_T_L_Q nexus identifying the task (see 4.7).

Device Server Buffer: The buffer in the device server to which data is to be transferred.

Application Client Buffer Offset: Offset in bytes from the beginning of the application client's buffer (i.e., the Data-Out Buffer) to the first byte of transferred data.

Request Byte Count: Number of bytes to be moved by this request.

If the **SCSI Command Received** SCSI transport protocol service included a First Burst Enabled argument and random buffer access is not supported, first burst data shall be transferred to the Device Server Buffer until all first burst data has been transferred. If the **SCSI Command Received** SCSI transport protocol service included a First Burst Enabled argument and random buffer access is supported, first burst data should be transferred to the Device Server Buffer but first burst data may be re-transferred across a service delivery subsystem.

5.4.3.3.2 Data-Out Received transport protocol service confirmation

A SCSI target port uses the Data-Out Received transport protocol service confirmation to notify a device server that it has received data.

Data-Out Received transport protocol service confirmation:

Data-Out Received (IN (I_T_L_Q Nexus, Delivery Result))

This confirmation notifies the device server that the requested data has been successfully delivered to its buffer, or that a service delivery subsystem error occurred while attempting to receive the data.

Page: 79

Author: Mark Evans, WDC Subject: Highlight Date: 10/25/2007 11:53:15 AM
"DELIVERY SUCCESSFUL:The data was delivered successfully."
s/b a space after the colon.

Status
George Penokie Accepted 10/31/2007 4:43:03 PM

Author: relliott Subject: Highlight Date: 10/16/2007 6:51:18 PM
:T
s/b
: T

Status
George Penokie Accepted 10/31/2007 4:42:51 PM

Author: Mark Evans, WDC Subject: Highlight Date: 10/25/2007 11:54:18 AM
"DELIVERY FAILURE:A service delivery subsystem error occurred while attempting to deliver the data."
s/b a space after the colon.

Status
George Penokie Accepted 10/31/2007 4:44:17 PM

Author: relliott Subject: Highlight Date: 10/16/2007 6:51:11 PM
:A
s/b
: A

Status
George Penokie Accepted 10/31/2007 4:43:57 PM

Author: Emulex Subject: Note Date: 10/30/2007 1:59:59 PM
Emulex-013
Page: 79 Receive Data-Out Input argument list: Put these arguments in the same order as in the service request above.

Status
George Penokie Accepted 10/31/2007 4:45:58 PM

Input arguments:

I_T_L_Q Nexus: The I_T_L_Q nexus identifying the task (see 4.7).

Delivery Result: an encoded value ~~representing~~ one of the following:

DELIVERY SUCCESSFUL:The data was delivered ~~successfully.~~

DELIVERY FAILURE:A service delivery subsystem error occurred while attempting to receive the data.~~:~~

5.4.3.4 Terminate Data Transfer service

5.4.3.4.1 Terminate Data Transfer service overview

~~The terminate data transfer request and confirmation may be used by a task manager to terminate partially completed transfers to the Data-In Buffer or from the Data-Out Buffer.~~

The **Terminate Data Transfer** SCSI transport protocol ~~service~~ allows a device server to specify that one or more **Send Data-In** or **Receive Data-Out** SCSI transport protocol service ~~requests be~~ terminated by a SCSI target port.

5.4.3.4.2 Terminate Data Transfer transport protocol service request

A device ~~server~~ uses the Terminate Data Transfer transport protocol service request to request that a SCSI target port terminate ~~data~~ transfers.

Terminate Data Transfer transport ~~protocol~~ service request:

Terminate Data Transfer (IN (Nexus))

Input argument:

Nexus: An I_T nexus, I_T_L nexus, or I_T_L_Q nexus (see 4.7).

The SCSI target port terminates all transfer service requests for the specified nexus (e.g., if an I_T_L nexus is specified, then the SCSI target port terminates all transfer service requests from the logical unit for the specified SCSI initiator port).

5.4.3.4.3 Data Transfer Terminated transport protocol service confirmation

A SCSI target port uses the Data Transfer Terminated transport protocol service confirmation to notify a device server that it has terminated all outstanding data transfers for a specified nexus.

Data Transfer Terminated transport protocol service confirmation:

Data Transfer Terminated (IN (Nexus))

Input argument:

Nexus: An I_T nexus, I_T_L nexus, or I_T_L_Q nexus (see 4.7).

This confirmation is returned in response to a **Terminate Data Transfer** request whether or not the specified nexus existed in the SCSI target port when the request was received. After a **Data Transfer Terminated** SCSI transport protocol service confirmation has been sent in response to a **Terminate Data Transfer** SCSI transport protocol service request, **Data-In Delivered** or **Data-Out Received** SCSI transport protocol service confirmations shall not be sent for the tasks specified by the nexus.

Page: 80

Author: Mark Evans, WDC Subject: Highlight Date: 10/25/2007 11:55:08 AM
"DELIVERY SUCCESSFUL:The data was delivered successfully."
s/b a space after the colon.

Status
George Penokie Accepted 10/31/2007 4:46:31 PM
Author: relliott Subject: Highlight Date: 10/16/2007 6:51:35 PM
:T
s/b
: T

Status
George Penokie Accepted 10/31/2007 4:46:16 PM
Author: Mark Evans, WDC Subject: Highlight Date: 10/25/2007 11:54:33 AM
"DELIVERY FAILURE:A service delivery subsystem error occurred while attempting to receive the data."
s/b a space after the colon.

Status
George Penokie Accepted 10/31/2007 4:46:39 PM
Author: relliott Subject: Highlight Date: 10/16/2007 6:51:42 PM
:A
s/b
: A

Status
George Penokie Accepted 10/31/2007 4:46:34 PM
Author: relliott Subject: Cross-Out Date: 10/16/2007 7:08:32 PM
The terminate data transfer request and confirmation may be used by a task manager to terminate partially completed transfers to the Data-In Buffer or from the Data-Out Buffer."

Delete that and replace "device server" with "device server or task manager" in the next sentence

Status
George Penokie Accepted 12/21/2007 11:55:20 AM -06'00'
Author: George Penokie Subject: Note Date: 12/21/2007 11:55:12 AM -06'00'
Moved to section 5.4.3.1 and restated as << The STPL confirmed services specified in 5.4.3.4 are used by the task manager or device server to terminate partially completed transfers to the Data-In Buffer or from the Data-Out Buffer. The **Terminate Data Transfer** SCSI transport protocol service requests that one or more **Send Data-In** or **Receive Data-Out** SCSI transport protocol service requests be terminated by a SCSI target port. >> and deleted section 5.4.3.4.1.

Author: relliott Subject: Highlight Date: 10/16/2007 7:08:07 PM
device server
s/b
device server or task manager

allowing deletion of the first sentence in 5.4.3.4.1.

Status
George Penokie Accepted 12/21/2007 11:55:50 AM -06'00'
Author: relliott Subject: Highlight Date: 10/16/2007 7:09:00 PM
device server
s/b
device server or task manager

Status
George Penokie Accepted 12/21/2007 11:57:25 AM -06'00'
Author: relliott Subject: Highlight Date: 10/16/2007 7:09:13 PM
device server
s/b
device server or task manager

Status
George Penokie Accepted 12/21/2007 11:58:02 AM -06'00'

5.5 Task and command lifetimes

This subclause specifies the events delimiting the beginning and end (i.e., lifetime) of a ~~task or pending~~ command from the viewpoint of the device server and application client.

The device server shall create a task upon receiving a ~~**SCSI Command Received**~~ indication.

The task shall exist until:

a) The device server sends a SCSI transport protocol service response for the task of TASK COMPLETE; or
b) The task is aborted as described in 5.6.

~~The application client maintains an application client task to interact with the task from the time the~~ **Send SCSI Command** SCSI transport protocol ~~service request is invoked until it receives one of the following SCSI target device responses:~~

a) A service response of TASK COMPLETE for that task;
b) Notification of a unit attention condition with one of the following additional sense codes;
   A) Any additional sense code whose ADDITIONAL SENSE CODE field contains 2Fh (e.g., COMMANDS CLEARED BY ANOTHER INITIATOR, or COMMANDS CLEARED BY POWER LOSS NOTIFICATION), if in reference to the task set containing the task;
   B) Any additional sense code whose ADDITIONAL SENSE CODE field contains 29h (e.g., POWER ON, RESET, OR BUS DEVICE RESET OCCURRED; POWER ON OCCURRED; SCSI BUS RESET OCCURRED; BUS DEVICE RESET FUNCTION OCCURRED; DEVICE INTERNAL RESET; or I_T NEXUS LOSS OCCURRED); or
   C) MICROCODE HAS BEEN CHANGED.
c) Notification that the task manager has detected the use of a duplicate I_T_L_Q nexus (see 5.8.3);
d) A service response of FUNCTION COMPLETE following an ABORT TASK task management function directed to the specified task;
e) A service response of FUNCTION COMPLETE following an ABORT TASK SET or a CLEAR TASK SET task management function directed to the task set containing that task;
f) A service response of FUNCTION COMPLETE following an I_T NEXUS RESET task management function delivered on the I_T nexus used to deliver that task; or
g) ~~A service response of FUNCTION COMPLETE~~ in response to a LOGICAL UNIT RESET task management function directed to the logical unit.

If a service response of SERVICE DELIVERY OR TARGET FAILURE is received for a command (e.g., when an I_T nexus loss is detected by the SCSI initiator port), the application client shall maintain an ~~application client task to~~ interact with the task until the application client has determined that the task is no longer known to the device server. An application client may determine that a task is no longer known to the device server by detecting:

a) Completion of an ABORT TASK task management function specifying that task;
b) Completion of an ABORT TASK SET or an I_T NEXUS RESET task management function on the I_T nexus used to deliver that task; or
c) ~~Completion of a CLEAR TASK SET or LOGICAL UNIT RESET~~ task management function.

NOTE 8 - The names of the unit attention conditions listed in the subclause (e.g., SCSI BUS RESET OCCURRED) are based on usage in ~~previous~~ versions of this standard. The use of these unit attention condition names is not to be interpreted as a description of how the unit attention conditions are represented by any given SCSI transport protocol.

~~To the application client, the command is pending from the time it calls the~~ **Send SCSI Command** ~~SCSI transport protocol service until one of the responses described in this subclause.~~

When a SCSI transport protocol does not require state synchronization (see 4.3.2), there may be a time skew between the completion of a device server request-response transaction as seen by the application client and device server. As a result, the lifetime of a task or command as it appears to the application client is different from the lifetime observed by the device server.

Some commands (e.g., commands with immediate bits like SEND DIAGNOSTIC, or write commands when a write cache is enabled) start background operations that operate after the task containing the command is no

Page: 81

Author: relliott Subject: Cross-Out Date: 12/11/2007 11:37:17 AM -06'00'
Delete << task or pending >>

Status
George Penokie Accepted 12/11/2007 11:37:28 AM -06'00'
Author: relliott Subject: Note Date: 12/11/2007 3:56:31 PM -06'00'
Discuss application client before device server.

app create
app end
device create
device end

or create before end:
app create
dev create
app end
dev end

Status
George Penokie Completed 12/11/2007 3:56:46 PM -06'00'
Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:20:25 PM
"The application client maintains an application client task to interact with the task from the time the Send SCSI Command SCSI transport protocol service request is invoked until it receives one of the following SCSI target device responses:"
s/b
"An application client maintains an application client task to interact with the task from the time the Send SCSI Command SCSI transport protocol service request is invoked until the application client receives one of the following SCSI target device responses:"

Status
George Penokie Accepted 10/31/2007 4:50:54 PM
Author: relliott Subject: Highlight Date: 10/16/2007 7:43:12 PM
application client task to interact with the task
s/b
application client task to represent the task

Status
George Penokie Accepted 11/1/2007 2:46:59 PM
Author: relliott Subject: Note Date: 1/17/2008 7:24:57 PM -06'00'
Items b) through g) should be qualified with knowledge that the unit attention condition or service response was reported after the task arrived at the target port. Otherwise, it might still be in flight. This is the subtle ordering assumption in 4.3.3.

Status
George Penokie Rejected 1/17/2008 7:33:28 PM -06'00'
Author: George Penokie Subject: Note Date: 1/17/2008 7:33:20 PM -06'00'
Added a note that states << Items other than a) assume in-order delivery (see 4.3.3). >>

Author: suhlerp Subject: Sticky Note Date: 10/25/2007 7:46:49 PM
[Technical]
How about ...
h) A service response of FUNCTION COMPLETE following a QUERY TASK task management function directed to the specified task; or
i) A service response of FUNCTION COMPLETE following a QUERY TASK SET task management function directed to the specified task set.

Status
George Penokie Accepted 1/17/2008 7:37:30 PM -06'00'
Author: relliott Subject: Highlight Date: 10/16/2007 7:43:37 PM
application client task to interact with the task
s/b
application client task to represent the task

Status
George Penokie Accepted 11/1/2007 2:54:36 PM
Author: suhlerp Subject: Sticky Note Date: 10/25/2007 7:47:01 PM
[Technical]
How about ...
d) Completion of a QUERY TASK task management function specifying the task with a service response of FUNCTION COMPLETE; or
e) Completion of a QUERY TASK SET task management function specifying the task set [more words needed?] with a service response of FUNCTION COMPLETE.

Status
George Penokie Rejected 1/17/2008 7:52:43 PM -06'00'
Author: George Penokie Subject: Note Date: 1/17/2008 7:52:36 PM -06'00'
The entire list was deleted as it contained no useful information.

Author: relliott Subject: Note Date: 10/16/2007 8:00:24 PM
This list is incomplete (not that it claims to be complete). Receiving unit attention condition about a reset, etc. - items b) through g) in the previous list - also apply here.

This list might have originally been worded as the application client may send these TMFs to actively make the determination, but it's now worded too much like the previous list.

Status
George Penokie Rejected 1/17/2008 7:51:52 PM -06'00'

Comments from page 81 continued on next page

5.5 Task and command lifetimes

This subclause specifies the events delimiting the beginning and end (i.e., lifetime) of a ~~task or pending~~ command from the viewpoint of the device server and application client.

The device server shall create a task upon receiving a **SCSI Command Received** indication.

The task shall exist until:

a) The device server sends a SCSI transport protocol service response for the task of TASK COMPLETE; or
b) The task is aborted as described in 5.6.

The application client maintains an application client task to interact with the task from the time the **Send SCSI Command** SCSI transport protocol service request is invoked until it receives one of the following SCSI target device responses:

a) A service response of TASK COMPLETE for that task;
b) Notification of a unit attention condition with one of the following additional sense codes;
   A) Any additional sense code whose ADDITIONAL SENSE CODE field contains 2Fh (e.g., COMMANDS CLEARED BY ANOTHER INITIATOR, or COMMANDS CLEARED BY POWER LOSS NOTIFICATION), if in reference to the task set containing the task;
   B) Any additional sense code whose ADDITIONAL SENSE CODE field contains 29h (e.g., POWER ON, RESET, OR BUS DEVICE RESET OCCURRED; POWER ON OCCURRED; SCSI BUS RESET OCCURRED; BUS DEVICE RESET FUNCTION OCCURRED; DEVICE INTERNAL RESET; or I_T NEXUS LOSS OCCURRED); or
   C) MICROCODE HAS BEEN CHANGED.
c) Notification that the task manager has detected the use of a duplicate I_T_L_Q nexus (see 5.8.3);
d) A service response of FUNCTION COMPLETE following an ABORT TASK task management function directed to the specified task;
e) A service response of FUNCTION COMPLETE following an ABORT TASK SET or a CLEAR TASK SET task management function directed to the task set containing that task;
f) A service response of FUNCTION COMPLETE following an I_T NEXUS RESET task management function delivered on the I_T nexus used to deliver that task; or
g) A service response of FUNCTION COMPLETE in response to a LOGICAL UNIT RESET task management function directed to the logical unit.

If a service response of SERVICE DELIVERY OR TARGET FAILURE is received for a command (e.g., when an I_T nexus loss is detected by the SCSI initiator port), the application client shall maintain an application client task to interact with the task until the application client has determined that the task is no longer known to the device server. An application client may determine that a task is no longer known to the device server by detecting:

a) Completion of an ABORT TASK task management function specifying that task;
b) Completion of an ABORT TASK SET or an I_T NEXUS RESET task management function on the I_T nexus used to deliver that task; or
c) Completion of a CLEAR TASK SET or LOGICAL UNIT RESET task management function.

NOTE 8 - The names of the unit attention conditions listed in the subclause (e.g., SCSI BUS RESET OCCURRED) are based on usage in previous versions of this standard. The use of these unit attention condition names is not to be interpreted as a description of how the unit attention conditions are represented by any given SCSI transport protocol.

~~To the application client, the command is pending from the time it calls the **Send SCSI Command** SCSI transport protocol service until one of the responses described in this subclause.~~

When a SCSI transport protocol does not require state synchronization (see 4.3.2), there may be a time skew between the completion of a device server request-response transaction as seen by the application client and device server. As a result, the lifetime of a task or command as it appears to the application client is different from the lifetime observed by the device server.

Some commands (e.g., commands with immediate bits like SEND DIAGNOSTIC, or write commands when a write cache is enabled) start background operations that operate after the task containing the command is no

Author: George Penokie Subject: Note Date: 1/17/2008 7:51:44 PM -06'00'
The list was deleted as it contained no useful information.

Author: relliott Subject: Cross-Out Date: 12/11/2007 3:51:20 PM -06'00'
Delete this <<To the application client, the command is pending from the time it calls the Send SCSI Command SCSI transport protocol service until one of the responses described in this subclause. >> as it is a duplicate of the 4th paragarph (i.e., the a,c list) above

Status
George Penokie Accepted 1/17/2008 7:56:57 PM -06'00'

Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:30:34 PM
"Some commands (e.g., commands with immediate bits like SEND DIAGNOSTIC, or write commands when a write cache is enabled) start background operations that operate after the task containing the command is no longer in the task set."
s/b
"Some commands initiate background operations that are processed after the task containing the command is no longer in the task set (i.e., status has been returned for the command). For examples, see the SEND DIAGNOSTIC command when used to initiate a background self-test (see SPC-4) or write commands when write cache is enabled (see SBC-3). "

Status
George Penokie Accepted 1/17/2008 8:04:26 PM -06'00'

longer in the task set. Background operations may be aborted by power on, hard resets, or logical unit resets. Background operations shall not be aborted by I_T nexus loss.

Background operations may generate deferred errors that are reported in the sense data for a subsequent completed command (see SPC-4). Information that a deferred error occurred may be cleared before it is reported (e.g., by power on, hard reset, or logical unit reset). Deferred errors should not be cleared by I_T nexus loss.

Unless a command completes with a GOOD or CONDITION MET status the degree to which the required command processing has been completed is vendor specific.

5.6 Aborting tasks

A task is aborted when a SCSI device condition (see 6.3), command, or task management function causes termination of the task prior to its completion by the device server.

See table 27 for a list of the SCSI device conditions that cause tasks to be aborted in a SCSI initiator device.

Table 27 — SCSI device conditions that abort tasks in a SCSI initiator device

SCSI device condition	Scope	Reference
Power on	All tasks in the SCSI initiator device	6.3.1
Hard reset	All tasks with an I_T nexus involving the SCSI initiator port	6.3.2
I_T nexus loss	All tasks associated with the lost I_T nexus	6.3.4
SCSI transport protocol specific conditions	As defined by the applicable SCSI transport protocol standard	

Page: 82

Author: relliott Subject: Highlight Date: 10/8/2007 7:30:56 PM
hard resets, or logical unit resets
s/b
hard reset, or logical unit reset

Status
George Penokie Accepted 11/1/2007 3:05:35 PM
Author: relliott Subject: Highlight Date: 10/27/2007 1:43:25 PM
Background operations shall not be aborted by I_T nexus loss.

Add "or power loss expected."

Status
George Penokie Accepted 1/22/2008 4:34:13 PM -06'00'
Author: Mark Evans, WDC Subject: Cross-Out Date: 10/29/2007 1:19:56 PM
Delete "completed", as the sense data is not reported for a "completed" command, but as part of the command completion process for the subsequent command.

Status
George Penokie Accepted 11/1/2007 3:09:39 PM
Author: relliott Subject: Highlight Date: 10/27/2007 1:43:48 PM
Deferred errors should not be cleared by I_T nexus loss.

Add "or power loss expected."

Status
George Penokie Accepted 1/22/2008 4:34:30 PM -06'00'
Author: Mark Evans, WDC Subject: Highlight Date: 10/25/2007 12:02:18 PM
"Unless a command completes with a GOOD or CONDITION MET status the degree to which the required command processing has been completed is vendor specific."
s/b
"Unless a command completes with GOOD status or CONDITION MET status, the degree to which the required command processing has been completed is vendor specific."

Status
George Penokie Accepted 11/1/2007 3:10:30 PM

See table 30 for a list of the command related conditions that cause tasks to be aborted.

Table 30 — Command related conditions that abort tasks

Command related conditions	Scope	Unit attention condition (see 5.8.7) additional sense code, if any [a]	TASK ABORTED status [b]	Reference
CHECK CONDITION status if: a) the QERR field is set to 01b; and b) the TST field is set to 000b (i.e., shared) in the Control mode page (see SPC-4)	All tasks in the task set	COMMANDS CLEARED BY ANOTHER INITIATOR	Yes	5.8.1.3 and 5.8.2.2
CHECK CONDITION status if: a) the QERR field is set to 01b; and b) the TST field is set to 001b (i.e., per I_T nexus) in the Control mode page (see SPC-4)	All tasks in the task set [c]	None	No	5.8.1.3 and 5.8.2.2
Completion of a command with a CHECK CONDITION status if the QERR field is set to 11b in the Control mode page (see SPC-4)	All tasks in the task set with the same I_T nexus as the command that was terminated	None	No	5.8.1.3 and 5.8.2.2
Processing of a PERSISTENT RESERVE OUT command with a PREEMPT AND ABORT service action with a reservation key that is associated with the I_T nexus on which the task was received (see SPC-4)	All tasks from all I_T nexuses with the specified reservation key	COMMANDS CLEARED BY ANOTHER INITIATOR	Yes	SPC-4
The return of an Execute Command service response of SERVICE DELIVERY OR TARGET FAILURE	The indicated task	None	No	5.1
Termination of an overlapped command	All tasks with the same I_T nexus as the command that was terminated	None	No	5.8.3

[a] If the TAS bit is set to zero in the Control mode page (see SPC-4), the device server creates this unit attention condition for each I_T nexus that had task(s) aborted other than the I_T nexus that delivered the task management function. If the TAS bit is set to one in the Control mode page (see SPC-4), the device server does not create this unit attention condition.

[b] "Yes" indicates that each task that is aborted on an I_T nexus other than the one that delivered the command is terminated with TASK ABORTED status, if the TAS bit is set to one in the Control mode page (see SPC-4). "No" indicates that no status is returned for aborted tasks.

[c] As a result of the TST field being set to 001b, there is one task set per I_T nexus, so no other I_T nexuses are affected.

If one or more tasks are cleared or aborted, the affected tasks are also cleared from the SCSI initiator ports in a manner that is outside the scope of this standard.

Page: 85

Author: relliott Subject: Highlight Date: 10/24/2007 7:24:37 PM
"that had task(s) aborted" might be incorrect.

Is the unit attention condition with COMMANDS CLEARED BY ANOTHER INITIATOR created for all I_T nexuses that were affected by the preempt, regardless of whether or not they actually had tasks aborted?

Or is it only created for I_T nexuses that had one or more tasks aborted?

Status
George Penokie Rejected 1/17/2008 8:26:13 PM -06'00'
Author: George Penokie Subject: Note Date: 1/17/2008 8:25:59 PM -06'00'
The behavior is correct as written. Because any command that is aborted as a result of an action by another initiator will have a UA established for the I_T nexus of the command that was aborted.

When a device server receives a command or task management function on an I_T nexus that causes tasks on the same I_T nexus to be aborted, the device server shall not return any notification that those tasks have been aborted other than:

a) the completion response for the command or task management function that caused the task(s) to be aborted; and
b) notification(s) associated with related effects of the command or task management function (e.g., a reset unit attention condition).

When a device server receives a command or task management function on an I_T nexus that causes tasks on other I_T nexuses to be aborted, the device server shall return notifications for those tasks based on the setting of the TAS bit in the Control mode page (see SPC-4):

a) If the TAS bit is set to zero, the device server:
   A) shall not return status for the tasks that were aborted; and
   B) shall establish a unit attention condition for the SCSI initiator port associated with each I_T nexus containing tasks that were aborted with an additional sense code set as defined in table 29 and table 30;

   or

b) If the TAS bit is set to one, the device server:
   A) shall return TASK ABORTED status for each aborted task; and
   B) shall not establish a unit attention condition for this reason.

When a logical unit is aborting one or more tasks received on an I_T nexus using the TASK ABORTED status it should complete all of those tasks before entering additional tasks received on that I_T nexus into the task set.

Page: 86

Author: Mark Evans, WDC Subject: Highlight Date: 10/25/2007 3:28:57 PM
"When a logical unit is aborting one or more tasks received on an I_T nexus using the TASK ABORTED status it should complete all of those tasks before entering additional tasks received on that I_T nexus into the task set."
s/b
"When a logical unit completes one or more tasks received on an I_T nexus with a status of TASK ABORTED, the logical unit should terminate all of the affected tasks before entering any other tasks received on that I_T nexus into the task set."

Status
George Penokie Accepted 11/1/2007 3:12:56 PM

5.7 Command processing example

A command is used to show the events associated with the processing of a single device service request (see figure 39). This example does not include error or exception conditions.

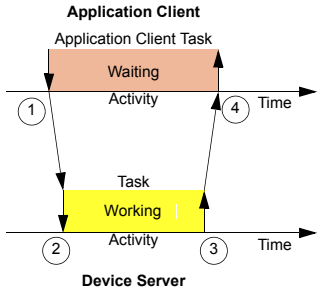


Figure 39 — Command processing events

The numbers in figure 39 identify the events described as follows:

1) The application client task performs an **Execute Command** procedure call by invoking the **Send SCSI Command** SCSI transport protocol service to send the CDB and other input parameters to the logical unit.

2) The device server is notified through a **SCSI Command Received** indication containing the CDB and command parameters. A task is created and entered into the task set. The device server may invoke the appropriate data delivery service one or more times to complete command processing.
3) The task ends upon completion of the command. On command completion, the **Send Command Complete** SCSI transport protocol service is invoked to return a status of GOOD and a service response of TASK COMPLETE.
4) A confirmation of **Command Complete Received** is passed to the application client task by the SCSI initiator port.

5.8 Command processing considerations and exception conditions

5.8.1 Commands that complete with CHECK CONDITION status

5.8.1.1 Overview

When a command completes with a CHECK CONDITION status, the application client may request that the device server alter command processing by establishing an ACA condition, using the NACA bit in the CONTROL byte of the CDB as follows:

a) If the NACA bit is set to zero, an ACA condition shall not be established; or
b) If the NACA bit is set to one, an ACA condition shall be established (see 5.8.2).

The requirements that apply when the ACA condition is not in effect are described in 5.8.1.2.

When a command completes with a CHECK CONDITION status and an ACA condition is not established, tasks other than the task for the command returning the CHECK CONDITION status may be aborted as described in 5.8.1.3.

5.8.1.2 Handling tasks when ACA is not in effect

Table 31 describes the handling of tasks when an ACA condition is not in effect for the task set. Which I_T nexuses are associated with the task set is influenced by the TST field in the Control mode page (see SPC-4).

Table 31 — Task handling when ACA is not in effect

New Task Properties		Device Server Action	ACA Established if New Task Terminates with a CHECK CONDITION status
Task attribute [a]	NACA Value [b]		
Any task attribute except ACA	0	Process the task. [c]	No
	1		Yes
ACA	0	Process an invalid task attribute condition as described in 5.8.5.	No
	1		Yes

[a] Task attributes are described in 8.6.
[b] The NACA bit is in the CONTROL byte in the CDB (see 5.2).
[c] All the conditions that affect the processing of commands (e.g., reservations) apply.

5.8.1.3 Aborting other tasks when CHECK CONDITION status is returned without establishing an ACA

When a CHECK CONDITION status is returned for a command where the NACA bit is set to zero in the command's CDB CONTROL byte (i.e., when an ACA condition is not established), tasks in the dormant or enabled task state (see 8.5) may be aborted based on the contents of the TST field and QERR field in the Control mode page (see SPC-4) as shown in table 32. The TST field specifies the type of task set in the logical unit. The QERR

Page: 87

Author: relliott Subject: Highlight Date: 12/11/2007 4:04:27 PM -06'00'
application client task
s/b
application client

Status
George Penokie Accepted 12/11/2007 4:04:44 PM -06'00'
Author: relliott Subject: Note Date: 10/16/2007 7:39:31 PM
Consider eliminating the 5.8 Command processing considerations level and upgrading each of the 5.8.x sections to 5.xx.

The Unit Attention section, for example, is as important as 5.6 Aborting tasks.

Status
George Penokie Accepted 1/22/2008 4:40:34 PM -06'00'
Author: George Penokie Subject: Note Date: 12/14/2007 3:32:14 PM -06'00'
I would rather move the out of place aborting task section under 5.8.

Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:20:49 PM
"When a command completes with a CHECK CONDITION status, the application client may request that the device server alter command processing by establishing an ACA condition, using the NACA bit in the CONTROL byte of the CDB as follows:"
s/b
"An application client uses the NACA bit in the CONTROL byte of the CDB (see 5.2) to specify whether or not the device server establishes an ACA condition when a command completes with CHECK CONDITION status. The meaning of the value in the NACA bit is as follows:"

Status
George Penokie Completed 11/1/2007 3:36:49 PM
Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:21:25 PM
"Which I_T nexuses are associated with the task set is influenced by the TST field in the Control mode page (see SPC-4)."
s/b
"Which I_T nexuses are associated with a task set is specified by the value in the TST field in the Control mode page (see SPC-4)."

Status
George Penokie Accepted 11/1/2007 3:38:19 PM
Author: relliott Subject: Highlight Date: 10/8/2007 6:22:47 PM
ACA s/b smallcaps lowercase

Status
George Penokie Completed 11/1/2007 3:42:34 PM
Author: George Penokie Subject: Note Date: 11/1/2007 3:42:29 PM
Changed to ACA task attribute

Author: relliott Subject: Highlight Date: 10/8/2007 6:22:51 PM
ACA s/b smallcaps lowercase

Status
George Penokie Completed 11/1/2007 3:42:49 PM
Author: George Penokie Subject: Note Date: 11/1/2007 3:42:42 PM
Changed to ACA task attribute.

Author: relliott Subject: Highlight Date: 12/11/2007 4:10:54 PM -06'00'
commands in the dormant
or enabled task command state

maybe s/b
dormant or enabled commands

Status
George Penokie Accepted 3/1/2008 5:05:46 PM -06'00'
Author: George Penokie Subject: Sticky Note Date: 3/1/2008 5:05:29 PM -06'00'
This was changed to the << dormant state or enabled command state >> style in this and other places.

field specifies how the device server handles blocked and dormant tasks when another task receives a CHECK CONDITION status.

Table 32 — Aborting tasks when an ACA is not established

QERR	TST	Action
00b	000b	Tasks other than the task returning CHECK CONDITION status shall not be aborted.
	001b	
01b	000b	All enabled and dormant tasks received on all I_T nexuses shall be aborted (see 5.6).
	001b	All enabled and dormant tasks received on the I_T nexus on which the CHECK CONDITION status was returned shall be aborted (see 5.6). All tasks received on other I_T nexuses shall not be aborted.
11b	000b	All enabled and dormant tasks received on the I_T nexus on which the CHECK CONDITION status was returned shall be aborted (see 5.6). All tasks received on other I_T nexuses shall not be aborted.
	001b	

5.8.2 Auto contingent allegiance (ACA)

5.8.2.1 ACA Overview

When a command completes with a CHECK CONDITION status, the application client may request that the device server alter command processing by establishing an ACA condition, using the NACA bit in the CONTROL byte of the CDB as follows:

a) If the NACA bit is set to zero, an ACA condition shall not be established (see 5.8.1.1); or
b) If the NACA bit is set to one, an ACA condition shall be established.

The steps taken by the device server to establish an ACA condition are described in 5.8.2.2. Upon establishment of the ACA condition, some tasks other than the task returning the CHECK CONDITION status may be aborted and continued processing of other tasks may be blocked as described in 5.8.2.2.

While the ACA condition is in effect and the TMF_ONLY bit is set to zero in the Control mode page (see SPC-4), new tasks received by the logical unit from the faulted I_T nexus are not allowed to enter the task set unless they have the ACA task attribute (see 8.6.5). One of the results of the ACA task attribute requirement is that commands in-flight when the CHECK CONDITION status occurs are returned unprocessed with an ACA ACTIVE status. Multiple commands may be sent one at a time using the ACA task attribute to recover from the event that resulted in the ACA condition without clearing the ACA.

While the ACA condition is in effect and the TMF_ONLY bit is set to one, no new tasks received by the logical unit from the faulted I_T nexus are allowed to enter the task set.

While the ACA condition is in effect:

a) New tasks received on the faulted I_T nexus shall be handled as described in 5.8.2.3, and
b) New tasks received on I_T nexuses other than the faulted I_T nexus shall be handled as described in 5.8.2.4.

The methods for clearing an ACA condition are described in 5.8.2.5.

5.8.2.2 Establishing an ACA

When a device server terminates a command with a CHECK CONDITION status and the NACA bit was set to one in the CONTROL byte of the faulting command, the device server shall create an ACA condition.

When an ACA condition is established, tasks in the dormant or enabled task state (see 8.5) shall either be aborted or blocked based on the contents of the TST field and QERR field in the Control mode page (see SPC-4)

Page: 88

Author: relliott Subject: Highlight Date: 12/11/2007 4:12:46 PM -06'00'
receives
s/b
is terminated with
or
completes with
or
ends with

seems to be a mix of wording...

Status
George Penokie Accepted 3/1/2008 4:45:29 PM -06'00'
Author: George Penokie Subject: Sticky Note Date: 2/27/2008 10:41:19 AM -06'00'
Change them to all be complete or a derivative there of.

Author: relliott Subject: Highlight Date: 12/11/2007 4:14:04 PM -06'00'
returning
s/b
being terminated with

(this would be a global change if it is appropriate. Maybe return is fine too and it should remain as is)

Status
George Penokie Accepted 3/1/2008 4:45:39 PM -06'00'
Author: George Penokie Subject: Sticky Note Date: 2/27/2008 10:41:47 AM -06'00'
Change them to all be complete or a derivative there of.

Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:21:10 PM
"When a command completes with a CHECK CONDITION status, the application client may request that the device server alter command processing by establishing an ACA condition, using the NACA bit in the CONTROL byte of the CDB as follows:
a) If the NACA bit is set to zero, an ACA condition shall not be established (see 5.8.1.1); or
b) If the NACA bit is set to one, an ACA condition shall be established."
s/b
"An application client specifies if an ACA condition is established when a command completes with CHECK CONDITION status (see 5.8.1.1)."
[All of the words that are recommended for replacement are in 5.8.1.1 and don't need to repeated here.]

Status
George Penokie Rejected 2/27/2008 10:57:04 AM -06'00'
Author: George Penokie Subject: Sticky Note Date: 2/27/2008 10:56:43 AM -06'00'
Deleted the a,b list and change the intro to << The application client may request that the device server alter command processing when a command terminates with a CHECK CONDITION status by establishing an ACA condition using the naca bit in the control byte (see 5.8.1).>>

Author: relliott Subject: Highlight Date: 10/8/2007 6:23:36 PM
ACA task attribute
s/b ACA (smallcaps lowercase) task attribute

Status
George Penokie Accepted 11/1/2007 3:45:11 PM
Author: relliott Subject: Highlight Date: 10/8/2007 6:23:29 PM
ACA task attribute
s/b ACA (smallcaps lowercase) task attribute

Status
George Penokie Accepted 11/1/2007 3:45:07 PM
Author: relliott Subject: Highlight Date: 10/8/2007 6:23:45 PM
ACA task attribute
s/b ACA (smallcaps lowercase) task attribute

Status
George Penokie Accepted 11/1/2007 3:45:15 PM

Page: 90

Author: relliott Subject: Highlight Date: 10/8/2007 6:24:05 PM
ACA s/b smallcaps lowercase

Status
George Penokie Completed 11/1/2007 3:47:46 PM
Author: George Penokie Subject: Note Date: 11/1/2007 3:47:42 PM
Changed to ACA task attribute.

Author: relliott Subject: Highlight Date: 10/8/2007 6:24:10 PM
ACA s/b smallcaps lowercase

Status
George Penokie Completed 11/1/2007 3:48:00 PM
Author: George Penokie Subject: Note Date: 11/1/2007 3:47:56 PM
Changed to ACA task attribute.

Author: relliott Subject: Highlight Date: 10/8/2007 6:24:32 PM
ACA task attribute
s/b
ACA (smallcaps lowercase) task attribute

Status
George Penokie Accepted 11/1/2007 3:49:55 PM

Table 34 — Handling for new tasks received on a faulted I_T nexus during ACA

New Task Properties		ACA Task Present in the Task Set	TMF_ONLY value [c]	Device Server Action	ACA Established If New Task Terminates with a CHECK CONDITION status
Task attribute [a]	NACA Value [b]				
ACA	0	No	0	Process the task. [e]	No [d]
	1	No	0		Yes [d]
	n/a	n/a	1	Terminate the task with ACA ACTIVE status.	n/a
	0 or 1	Yes	n/a		n/a
Any task attribute except ACA	0 or 1	n/a	n/a	Terminate the task with ACA ACTIVE status.	n/a

[a] Task attributes are described in 8.6.
[b] The NACA bit is in the CONTROL byte in the CDB (see 5.2).
[c] The TMF_ONLY bit is in the Control mode page (see SPC-4).
[d] If a task with the ACA task attribute terminates with a CHECK CONDITION status, the existing ACA condition shall be cleared and the value of the NACA bit shall control the establishment of a new ACA condition.
[e] All the conditions that affect the processing of commands (e.g., reservations) apply.

5.8.2.4 Handling new tasks received on non-faulted I_T nexuses when ACA is in effect

5.8.2.4.1 Command processing permitted for tasks received on non-faulted I_T nexuses during ACA

The device server shall process a PERSISTENT RESERVE OUT command with a PREEMPT AND ABORT service action (see SPC-4) while an ACA condition is established when the command is received on a non-faulted I_T nexus.

NOTE 9 - The processing of specific commands (e.g., PERSISTENT RESERVE OUT command with a PREEMPT AND ABORT service action) received on a non-faulted I_T nexus while an ACA condition is in effect provides SCSI initiator ports not associated with the faulted I_T nexus the opportunity to recover from error conditions that the initiator port associated with the faulted I_T nexus is unable to recover from itself.

5.8.2.4.2 Handling new tasks received on non-faulted I_T nexuses when ACA is in effect

The handling of tasks received on I_T nexuses other than the faulted I_T nexus depends on the value in the TST field in the Control mode page (see SPC-4).

Table 35 describes the handling of new tasks received on I_T nexuses other than the faulted I_T nexus when ACA is in effect.

Table 35 — Handling for new tasks received on non-faulted I_T nexuses during ACA

TST Field Value in Control mode page	New Task Properties		New Command Permitted During ACA [c]	Device Server Action	ACA Established If New Task Terminates with a CHECK CONDITION status
	Task attribute [a]	NACA Value [b]			
000b	ACA	n/a	n/a	Terminate the task with ACA ACTIVE status.	n/a
	Any task attribute except ACA	0	No	Terminate the task with BUSY status	n/a
		1	No	Terminate the task with ACA ACTIVE status.	n/a
		0	Yes	Process the task.	No [d]
		1	Yes		Yes [d]
001b	ACA	0	n/a	Process an invalid task attribute condition as described in 5.8.5.	No
		1			Yes
	Any task attribute except ACA	0 or 1	n/a	Process the task. [e]	See 5.8.1.2.

[a] Task attributes are described in 8.6.
[b] The NACA bit is in the CONTROL byte in the CDB (see 5.2).
[c] See 5.8.2.4.1.
[d] If a permitted command terminates with a CHECK CONDITION status, the existing ACA condition shall be cleared and the value of the NACA bit shall control the establishment of a new ACA condition.
[e] When the TST field in the Control mode page contains 001b, commands received on a non-faulted I_T nexus shall be processed as if the ACA condition does not exist (see 5.8.1.2). In this case, the logical unit shall be capable of handling concurrent ACA conditions and sense data associated with each I_T nexus.

5.8.2.5 Clearing an ACA condition

An ACA condition shall only be cleared:

a) As a result of a hard reset (see 6.3.2), logical unit reset (see 6.3.3), or I_T nexus loss (see 6.3.4);
b) By a CLEAR ACA task management function (see 7.4) received on the faulted I_T nexus;
c) By a PERSISTENT RESERVE OUT command with a PREEMPT AND ABORT service action with the ACA task attribute received on the faulted I_T nexus that clears the tasks received on the faulted I_T nexus (see SPC-4);
d) By a PERSISTENT RESERVE OUT command with a PREEMPT AND ABORT service action with a task attribute other than ACA task attribute received on a non-faulted I_T nexus that clears the tasks received on the faulted I_T nexus;
e) When a command with the ACA task attribute received on the faulted I_T nexus terminates with a CHECK CONDITION status; or

Page: 91

Author: relliott Subject: Highlight Date: 10/8/2007 6:25:00 PM
ACA
s/b smallcaps lowercase

Status
George Penokie Completed 11/1/2007 3:48:19 PM
Author: George Penokie Subject: Note Date: 11/1/2007 3:48:14 PM
Changed to ACA task attribute.

Author: relliott Subject: Highlight Date: 10/8/2007 6:24:48 PM
ACA
s/b smallcaps lowercase

Status
George Penokie Completed 11/1/2007 3:48:32 PM
Author: George Penokie Subject: Note Date: 11/1/2007 3:48:27 PM
Changed to ACA task attribute.

Author: relliott Subject: Highlight Date: 10/8/2007 6:24:55 PM
ACA
s/b smallcaps lowercase

Status
George Penokie Completed 11/1/2007 3:48:55 PM
Author: George Penokie Subject: Note Date: 11/1/2007 3:48:42 PM
Changed to ACA task attribute.

Author: relliott Subject: Highlight Date: 10/8/2007 6:25:05 PM
ACA
s/b smallcaps lowercase

Status
George Penokie Completed 11/1/2007 3:49:07 PM
Author: George Penokie Subject: Note Date: 11/1/2007 3:49:03 PM
Changed to ACA task attribute.

Author: relliott Subject: Highlight Date: 10/8/2007 6:26:04 PM
ACA task attribute
s/b ACA (smallcaps lowercase) task attribute

Status
George Penokie Accepted 11/1/2007 4:17:16 PM
Author: relliott Subject: Highlight Date: 10/8/2007 6:25:39 PM
ACA task attribute
s/b
ACA (small caps lowercase)

Status
George Penokie Accepted 11/1/2007 4:17:20 PM
Author: relliott Subject: Highlight Date: 10/8/2007 6:26:08 PM
ACA task attribute
s/b ACA (smallcaps lowercase) task attribute

Status
George Penokie Accepted 11/1/2007 4:17:25 PM

f) When a PERSISTENT RESERVE OUT command with a PREEMPT AND ABORT service action terminates in a CHECK CONDITION status.

Cases e) and f) may result in the establishment of a new ACA based on the value of the NACA bit.

When an ACA condition is cleared and no new ACA condition is established, the state of all tasks in the task set shall be modified as described in 8.8.

5.8.3 Overlapped commands

An overlapped command occurs when a task manager or a task router detects the use of a duplicate I_T_L_Q nexus (see 4.5.6) in a command before a task holding that I_T_L_Q nexus completes its task lifetime (see 5.5). Each SCSI transport protocol standard shall specify whether or not a task manager or a task router is required to detect overlapped commands.

A task manager or a task router that detects an overlapped command shall abort all tasks received on the I_T nexus on which the overlapped command was received and the device server shall return CHECK CONDITION status for the overlapped command. The sense key shall be set to ABORTED COMMAND and the additional sense code shall be set to OVERLAPPED COMMANDS ATTEMPTED.

NOTE 10 - An overlapped command may be indicative of a serious error and, if not detected, may result in corrupted data. This is considered a catastrophic failure on the part of the SCSI initiator device. Therefore, vendor specific error recovery procedures may be required to guarantee the data integrity on the medium. The SCSI target device logical unit may return additional sense data to aid in this error recovery procedure (e.g., sequential-access devices may return the residue of blocks remaining to be written or read at the time the second command was received).

5.8.4 Incorrect logical unit selection

The SCSI target device's response to a command addressed to an incorrect logical unit number is described in this subclause.

In response to a REQUEST SENSE command, a REPORT LUNS command, or an INQUIRY command the SCSI target device shall respond as defined in SPC-4.

Any command except REQUEST SENSE, REPORT LUNS, or INQUIRY:

a) Shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and with the additional sense code set to LOGICAL UNIT NOT SUPPORTED, if:
   A) The SCSI target device is not capable of supporting the logical unit (e.g., some SCSI target devices support only one peripheral device); or
   B) The SCSI target device supports the logical unit, but the peripheral device is not currently connected to the SCSI target device;

   or
b) Is responded to in a vendor specific manner, if:
   A) The SCSI target device supports the logical unit and the peripheral device is connected, but the peripheral device is not operational; or
   B) The SCSI target device supports the logical unit but is incapable of determining if the peripheral device is connected or is not operational because the peripheral device is not ready.

5.8.5 Task attribute exception conditions

If a command is received with a task attribute that is not supported or is not valid (e.g., an ACA task attribute when an ACA condition does not exist), the command shall be terminated with CHECK CONDITION status, sense key set to ILLEGAL REQUEST, and additional sense code set to INVALID MESSAGE ERROR.

NOTE 11 - The use of the INVALID MESSAGE ERROR additional sense code is based on its similar usage in previous versions of this standard. The use of the INVALID MESSAGE ERROR additional sense code is not to be interpreted as a description of how the task attributes are represented by any given SCSI transport protocol.

Page: 92

Author: relliott Subject: Note Date: 12/11/2007 4:17:37 PM -06'00'
The 5.5 command lifetime section discusses the device server's view of the lifetime.

This text hints that it should also mention that view is shared by the task manager and task router as well.

Status
George Penokie Accepted 2/27/2008 11:24:14 AM -06'00'
Author: George Penokie Subject: Sticky Note Date: 2/27/2008 11:24:10 AM -06'00'
Added this to the first paragraph of section 5.5 << The task router and task manager have the same viewpoint of the beginning and end of a command as the device server. >>

Author: relliott Subject: Highlight Date: 10/29/2007 11:17:55 AM
Incorrect logical unit selection
s/b
Incorrect logical unit numbers

"select" is an ancient parallel SCSI term, and the logical unit number is what is incorrect, not the logical unit.

Status
George Penokie Rejected 12/26/2007 10:54:28 AM -06'00'
Author: George Penokie Subject: Note Date: 12/17/2007 12:01:25 PM -06'00'
Add to glossary:

Incorrect logical unit number: The logical unit number of a logical unit that does not exist in the SCSI target device when addressed through a given I_T nexus.

Incorrect logical unit: A logical unit that does not exist in the SCSI target device when addressed by a given I_T_L nexus.

Change title to << incorrect logical unit >>

Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:21:54 PM
"...the command shall be terminated with CHECK CONDITION status, sense key set to ILLEGAL REQUEST, and additional sense code set to INVALID MESSAGE ERROR."
s/b
"...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 MESSAGE ERROR."

Status
George Penokie Accepted 11/1/2007 4:19:10 PM

Task attribute support should be reported with the Extended INQUIRY Data VPD page (see SPC-4).

5.8.6 Sense data

Sense data shall be made available by the logical unit in the event a command completes with a CHECK CONDITION status or other conditions (e.g., the processing of a REQUEST SENSE command). The format, content, and conditions under which sense data shall be prepared by the logical unit are specified in this standard, SPC-4, the applicable command standard, and the applicable SCSI transport protocol standard.

Sense data associated with an I_T nexus shall be preserved by the logical unit until:

a) The sense data is transferred;
b) A logical unit reset (see 6.3.3) occurs;
c) Power loss expected (see 6.3.5) occurs; or
d) An I_T nexus loss (see 6.3.4) occurs for the I_T nexus associated with the preserved sense data.

When a command completes with a CHECK CONDITION status, sense data shall be returned in the same I_T_L_Q nexus transaction (see 3.1.51) as the CHECK CONDITION status. After the sense data is returned, it shall be cleared except when it is associated with a unit attention condition and the UA_INTLCK_CTRL field in the Control mode page (see SPC-4) contains 10b or 11b.

The return of sense data in the same I_T_L_Q nexus transaction as a CHECK CONDITION status shall not affect ACA (see 5.8.2) or the sense data associated with a unit attention condition when the UA_INTLCK_CTRL field contains 10b or 11b.

5.8.7 Unit Attention condition

Each logical unit shall generate a unit attention condition whenever one of the following events occurs:

a) A power on (see 6.3.1), hard reset (see 6.3.2), logical unit reset (see 6.3.3), I_T nexus loss (see 6.3.4), or power loss expected (see 6.3.5) occurs;
b) A removable medium may have been changed;
c) The mode parameters associated with this I_T nexus have been changed by a task received on another I_T nexus (i.e., SCSI initiator ports share mode parameters, see SPC-4);
d) The log parameters associated with this I_T nexus have been changed by a task received on another I_T nexus (i.e., SCSI initiator ports share log parameters, see SPC-4);
e) The version or level of microcode has been changed (see SPC-4);
f) Tasks received on this I_T nexus have been cleared by a task or a task management function associated with another I_T nexus and the TAS bit was set to zero in the Control mode page associated with this I_T nexus (see 5.6);
g) INQUIRY data has been changed (see SPC-4);
h) The logical unit inventory has been changed (see 4.5.19.1);
i) The mode parameters in effect for the associated I_T nexus have been restored from non-volatile memory (see SPC-4); or
j) Any other event requiring the attention of the SCSI initiator device.

Logical units may queue unit attention conditions. After the first unit attention condition is cleared, another unit attention condition may exist (e.g., a unit attention condition with an additional sense code set to POWER ON OCCURRED may be followed by one with an additional sense code set to MICROCODE HAS BEEN CHANGED).

A unit attention condition shall persist on the logical unit for the SCSI initiator port associated with each I_T nexus until the SCSI initiator port associated with the I_T nexus clears the condition. Unit attention conditions are affected by the processing of commands as follows:

a) If an INQUIRY command enters the enabled task state, the device server shall perform the INQUIRY command and shall neither report nor clear any unit attention condition;
b) If a REPORT LUNS command enters the enabled task state, the device server shall perform the REPORT LUNS command and shall not report any unit attention condition.

Page: 93

Author: relliott Subject: Highlight Date: 10/8/2007 5:21:56 PM
Attention
s/b
attention

Status
George Penokie Accepted 11/1/2007 4:19:42 PM
Author: relliott Subject: Note Date: 10/27/2007 2:08:24 PM
Incorporate 07-459 Unit attention queuing

Status
George Penokie Accepted 2/13/2008 4:58:02 PM -06'00'
Author: relliott Subject: Highlight Date: 10/8/2007 7:31:54 PM
A removable medium may have been changed;
s/b
A removable medium has possibly been changed

Status
George Penokie Rejected 2/13/2008 3:24:42 PM -06'00'
Author: George Penokie Subject: Sticky Note Date: 2/13/2008 3:25:08 PM -06'00'
I see no problem with the current wording.

Author: relliott Subject: Note Date: 10/8/2007 6:20:00 PM
The list of unit attention conditions includes some but not all those defined in SPC-4. What is the basis for including some reasons here? Item j) does serve as a catch-all, but perhaps all the non-SAM related items should be removed.

Not covered include:
- successful completion of a SET IDENTIFYING INFORMATION command that changes identifying information saved by the logical unit (see SPC-4)
- On successful completion of a SET PRIORITY command or change to the mode page
- On successful completion of a SET TIMESTAMP command
- If the ETC bit is set to one and the result of the comparison is true (log parameters)
- block descriptor values changed
- informational exceptions

Status
George Penokie Accepted 2/27/2008 11:34:19 AM -06'00'
Author: George Penokie Subject: Sticky Note Date: 2/27/2008 11:31:59 AM -06'00'
Removed all non-SAM items from the list as it is not complete and never could be.

Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:21:46 PM
"...(e.g., a unit attention condition with an additional sense code set to POWER ON OCCURRED may be followed by one with an additional sense code set to MICROCODE HAS BEEN CHANGED)."
s/b
"...(e.g., a unit attention condition with an additional sense code set to COMMANDS CLEARED BY ANOTHER INITIATOR may be followed by a unit attention condition with an additional sense code set to MODE PARAMETERS CHANGED)."
[I think the example to be replaced is a poor example (i.e., if both a POWER ON OCCURRED and a MICROCODE HAS BEEN CHANGED occurred, most SCSI target devices would only report the POWER ON OCCURRED), and the suggested replacement is a much more likely scenario.]

Status
George Penokie Accepted 2/13/2008 3:23:03 PM -06'00'
Author: relliott Subject: Highlight Date: 10/29/2007 10:36:38 AM
perform
s/b
process

Status
George Penokie Accepted 11/1/2007 4:29:29 PM
Author: relliott Subject: Highlight Date: 10/29/2007 10:36:42 AM
perform
s/b
process

Status
George Penokie Accepted 11/1/2007 4:29:34 PM

If the UA_INTLCK_CTRL field in the Control mode page is set to 00b (see SPC-4), the SCSI target device shall clear any pending unit attention condition with an additional sense code of REPORTED LUNS DATA HAS CHANGED established for the initiator port associated with that I_T nexus in each logical unit accessible by the I_T nexus on which the REPORT LUNS command was received. Other pending unit attention conditions shall not be cleared.

If the UA_INTLCK_CTRL field in the Control mode page contains 10b or 11b, the SCSI target device shall not clear any unit attention condition(s);

c) If a REQUEST SENSE command enters the enabled task state while a unit attention condition exists for the SCSI initiator port associated with the I_T nexus on which the REQUEST SENSE command was received, then the device server shall return GOOD status and either:
   A) Report any pending sense data as parameter data and preserve all unit attention conditions on the logical unit; or
   B) Report a unit attention condition as parameter data for the REQUEST SENSE command to the SCSI initiator port associated with the I_T nexus on which the REQUEST SENSE command was received. The logical unit may discard any pending sense data and shall clear the reported unit attention condition for the SCSI initiator port associated with that I_T nexus. If the unit attention condition has an additional sense code of REPORTED LUNS DATA HAS CHANGED, the SCSI target device shall clear any pending unit attention conditions with an additional sense code of REPORTED LUNS DATA HAS CHANGED established for the I_T nexus on which the command was received in each logical unit accessible by that I_T nexus;

   If the device server has already generated the ACA condition (see 5.8.2) for a unit attention condition, the device server shall report the unit attention condition (i.e., option c)B) above);

d) if the device server supports the NOTIFY DATA TRANSFER DEVICE command (see ADC-2) and a NOTIFY DATA TRANSFER DEVICE command enters the enabled task state, then the device server shall perform the NOTIFY DATA TRANSFER DEVICE command and shall neither report nor clear any unit attention condition; and
e) If a command other than INQUIRY, REPORT LUNS, REQUEST SENSE, or NOTIFY DATA TRANSFER DEVICE enters the enabled task state while a unit attention condition exists for the SCSI initiator port associated with the I_T nexus on which the command was received, the device server shall terminate the command with a CHECK CONDITION status. The device server shall provide sense data that reports a unit attention condition for the SCSI initiator port that sent the command on the I_T nexus.

If a device server reports a unit attention condition with a CHECK CONDITION status and the UA_INTLCK_CTRL field in the Control mode page contains 00b (see SPC-4), then the device server shall clear the reported unit attention condition for the SCSI initiator port associated with that I_T nexus on the logical unit. If the unit attention condition has an additional sense code of REPORTED LUNS DATA HAS CHANGED, the SCSI target device shall clear any pending unit attention conditions with an additional sense code of REPORTED LUNS DATA HAS CHANGED established for the I_T nexus on which the command was received in each logical unit accessible by that I_T nexus. If the UA_INTLCK_CTRL field contains 10b or 11b, the device server shall not clear unit attention conditions reported with a CHECK CONDITION status.

Page: 94

Author: relliott Subject: Highlight Date: 10/29/2007 10:37:14 AM
perform
s/b
process

Status
George Penokie Accepted 11/1/2007 5:08:18 PM

6 SCSI events and event notification model

6.1 SCSI events overview

SCSI events may occur or be detected in either:

a) The SCSI device;
b) One or more SCSI ports within a SCSI device; or
c) The application client, task manager, or device server.

The detection of any event may require processing by the object that detects it.

Events that occur in the SCSI device are assumed to be detected and processed by all objects within the SCSI device.

When a SCSI port detects an event, it shall use the event notification services (see 6.4) to notify device servers, task managers, or application clients that the event has been detected.

The events detected and event notification services usage depends on whether the SCSI device is a SCSI target device (see figure 40) or a SCSI initiator device (see figure 41).

Page: 95

Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:20:28 PM
"Events that occur in the SCSI device..."
s/b
"Events that occur in a SCSI device..."

Status
George Penokie Accepted 11/1/2007 5:09:05 PM

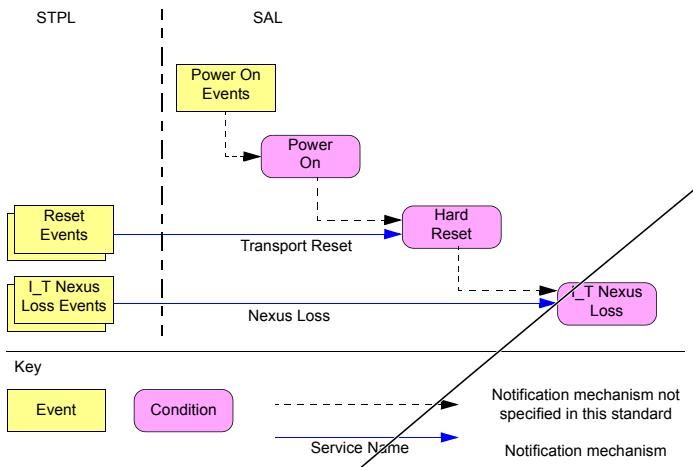


Figure 41 — Events and event notifications for SCSI initiator devices

6.2 Establishing a unit attention condition subsequent to detection of an event

Table 36 shows the additional sense code that a logical unit shall use when a unit attention (see 5.8.7) is established for each of the conditions shown in figure 40 (see 6.1). A SCSI transport protocol may define a more specific additional sense code than SCSI BUS RESET OCCURRED for reset events. The most specific condition in table 36 known to the logical unit should be used to establish the additional sense code for a unit attention.

Table 36 — Unit attention additional sense codes for events detected by SCSI target devices

Condition	Additional Sense Code	Specificity
Logical unit is unable to distinguish between the conditions	POWER ON, RESET, OR BUS DEVICE RESET OCCURRED	Lowest
Power loss expected	COMMANDS CLEARED BY POWER LOSS NOTIFICATION	
Power on	POWER ON OCCURRED or DEVICE INTERNAL RESET	
Hard reset	SCSI BUS RESET OCCURRED or MICROCODE HAS BEEN CHANGED or protocol specific	
Logical unit reset	BUS DEVICE RESET FUNCTION OCCURRED	
I_T nexus loss	I_T NEXUS LOSS OCCURRED	Highest

Page: 97

Author: relliott Subject: Note Date: 10/27/2007 1:53:42 PM
The row
"Power loss expected/COMMANDS CLEARED BY POWER LOSS NOTIFICATION"

needs to move down in table 36 to be below (higher specificity) than I_T nexus loss.

Section 6.3.5 says it simply aborts tasks; it doesn't wipe out background operations, clear deferred errors, etc. This means it has less impact than a hard reset, logical unit reset, and a set of I_T nexus losses (it has more impact than a single I_T nexus loss...). If the target device experiences hard reset, logical unit reset, or I_T nexus loss, it is not an acceptable substitute to only report COMMANDS CLEARED BY POWER LOSS NOTIFICATION, which its current position in the table endorses.

Status
George Penokie Completed 2/13/2008 3:06:21 PM -06'00'

NOTE 12 - The names of the unit attention conditions listed in the subclause (e.g., SCSI BUS RESET OCCURRED) are based on usage in previous versions of this standard. The use of these unit attention condition names is not to be interpreted as a description of how the unit attention conditions are represented by any given SCSI transport protocol.

A logical unit may use the ~~I_T NEXUS LOSS OCCURRED~~ additional sense code when establishing a unit attention condition if:

a) The SCSI initiator port to which the sense data is being delivered is the SCSI initiator port that was associated with the I_T nexus loss, and the logical unit has maintained all state information specific to that SCSI initiator port since the I_T nexus loss; and
b) The I_T nexus being used to deliver the sense data is the same I_T nexus that was lost, and the logical unit has maintained all state information specific to that I_T nexus since the I_T nexus loss.

Otherwise, the logical unit shall use one of the less specific additional sense codes (e.g., POWER ON OCCURRED) when establishing a unit attention condition.

6.3 Conditions resulting from SCSI events

6.3.1 Power on

Power on is a SCSI device condition resulting from a power on event. When a SCSI device is powered on, it shall cause a hard reset.

The power on condition applies to both SCSI initiator devices and SCSI target devices.

6.3.2 Hard reset

Hard reset is a SCSI device condition resulting from:

a) A power on condition (see 6.3.1);
b) Microcode change (see SPC-4); or
c) A reset event indicated by a **Transport Reset** event notification (see 6.4).

The definition of reset events and the notification of their detection is SCSI transport protocol specific.

Each SCSI transport protocol standard that defines reset events shall specify a SCSI target port's protocol specific actions in response to reset events. Each SCSI transport protocol standard that defines reset events should specify when those events result in the delivery of a **Transport Reset** event notification to the SCSI applications layer.

SCSI transport protocols may include reset events that have no SCSI effects (e.g., a Fibre Channel non-initializing loop initialization primitive).

The hard reset condition applies to both SCSI initiator devices and SCSI target devices.

A SCSI target port's response to a hard reset condition shall include a logical unit reset condition (see 6.3.3) for all logical units to which the SCSI target port has access. A hard reset condition shall not affect any other SCSI target ports in the SCSI target device, however, the logical unit reset condition established by a hard reset may affect tasks that are communicating via other SCSI target ports.

Although the task manager response to task management requests is subject to the presence of access restrictions, as managed by ACCESS CONTROL OUT commands (see SPC-4), a hard reset condition shall not be prevented by access controls.

When a SCSI initiator port detects a hard reset condition, it should terminate all its outstanding **Execute Command** procedure calls and all its outstanding task management procedure calls with a service response of SERVICE DELIVERY OR TARGET FAILURE. A hard reset condition shall not affect any other SCSI initiator ports in the SCSI initiator device, however, the logical unit reset condition established in a SCSI target device by a hard reset may affect tasks that are communicating via other SCSI initiator ports.

Page: 98

Author: relliott Subject: Highlight Date: 10/27/2007 2:04:54 PM
when establishing a unit
attention condition
s/b
when establishing a unit attention condition for an I_T nexus loss

Status
George Penokie Accepted 11/1/2007 5:10:57 PM

Author: relliott Subject: Highlight Date: 10/27/2007 2:04:11 PM
may use the I_T NEXUS LOSS OCCURRED additional sense code
s/b
should use ...

Status
George Penokie Accepted 2/27/2008 10:59:15 AM -06'00'

Author: relliott Subject: Highlight Date: 12/11/2007 4:22:22 PM -06'00'
tasks
s/b
commands and task management functions

Status
George Penokie Completed 12/11/2007 4:22:56 PM -06'00'
gpenokie Accepted 2/13/2008 3:01:23 PM -06'00'

Author: relliott Subject: Highlight Date: 10/16/2007 7:06:30 PM
hard reset condition
s/b
hard reset condition (see 6.3.2)

Status
George Penokie Accepted 11/1/2007 5:12:03 PM

Author: relliott Subject: Highlight Date: 12/11/2007 4:22:45 PM -06'00'
tasks

s/b
commands and task management functions

Status
George Penokie Accepted 2/13/2008 4:54:28 PM -06'00'
gpenokie Accepted 2/13/2008 3:01:37 PM -06'00'

A SCSI port's response to a hard reset condition shall include establishing an I_T nexus loss condition (see 6.3.4) for every I_T nexus associated with that SCSI port.

6.3.3 Logical unit reset

Logical unit reset is a logical unit condition resulting from:

a) A hard reset condition (see 6.3.2); or
b) A logical unit reset event indicating that a LOGICAL UNIT RESET task management request (see 7.7) has been processed.

The logical unit reset condition applies only to SCSI target devices.

When responding to a logical unit reset condition, the logical unit shall:

a) Abort all tasks as described in 5.6;
b) Clear all ACA conditions (see 5.8.2.5) in all task sets in the logical unit;
c) Establish a unit attention condition (see 5.8.7 and 6.2);
d) Initiate a logical unit reset for all dependent logical units (see 4.5.19.4); and
e) Perform any additional functions required by the applicable command standards.

6.3.4 I_T nexus loss

I_T nexus loss is a SCSI device condition resulting from:

a) A hard reset condition (see 6.3.2);
b) An I_T nexus loss event (e.g., logout) indicated by a **Nexus Loss** event notification (see 6.4); or
c) An I_T nexus loss event indicating that an I_T NEXUS RESET task management request (see 7.6) has been processed.

An I_T nexus loss event is an indication from the SCSI transport protocol to the SAL that an I_T nexus no longer exists. SCSI transport protocols may define I_T nexus loss events.

Each SCSI transport protocol standard that defines I_T nexus loss events should specify when those events result in the delivery of a **Nexus Loss** event notification to the SCSI applications layer.

The I_T nexus loss condition applies to both SCSI initiator devices and SCSI target devices.

When a SCSI target port detects an I_T nexus loss, a **Nexus Loss** event notification indication shall be delivered to each logical unit to which the I_T nexus has access. In response to the resulting I_T nexus loss condition a logical unit shall take the following actions:

a) Abort all tasks received on the I_T nexus as described in 5.6;
b) Clear all ACA conditions (see 5.8.2.5) associated with the I_T nexus;
c) Establish a unit attention condition for the SCSI initiator port associated with the I_T nexus (see 5.8.7 and 6.2); and
d) Perform any additional functions required by the applicable command standards.

If the logical unit retains state information for the I_T nexus that is lost, its response to the subsequent I_T nexus re-establishment for the logical unit should include establishing a unit attention with an additional sense code set to I_T NEXUS LOSS OCCURRED.

If the logical unit does not retain state information for the I_T nexus that is lost, it shall consider the subsequent I_T nexus re-establishment, if any, as the formation of a new I_T nexus for which there is no past history (e.g., establish a unit attention with an additional sense code set to POWER ON OCCURRED).

When a SCSI initiator port detects an I_T nexus loss, it should terminate all its outstanding **Execute Command** procedure calls and all its outstanding task management procedure calls for the SCSI target port associated with the I_T nexus with a service response of SERVICE DELIVERY OR TARGET FAILURE.

Page: 99

Author: relliott Subject: Note Date: 10/27/2007 2:06:48 PM
in 6.3.3, add "abort all task management functions;"

Status
George Penokie Accepted 2/27/2008 11:01:18 AM -06'00'
gpenokie Completed 2/13/2008 3:00:34 PM -06'00'
Author: gpenokie Subject: Sticky Note Date: 2/13/2008 3:00:30 PM -06'00'
Added << terminate all task managment functions >> into the abc list.

Author: relliott Subject: Note Date: 10/27/2007 2:06:55 PM
in 6.3.4, add "abort all task management functions received on the I_T nexus;"

Status
gpenokie Completed 2/13/2008 2:59:19 PM -06'00'
Author: gpenokie Subject: Sticky Note Date: 2/13/2008 2:59:15 PM -06'00'
Added << terminate all task managment functions >> into the abc list.

6.3.5 Power loss expected

Power loss expected is a SCSI device condition resulting from a power loss expected event indicated by a Power Loss Expected event notification (see 6.4).

A power loss expected event is an indication from the SCSI transport protocol to the SAL that power loss may occur within a protocol specific period of time. SCSI transport protocols may define power loss expected events.

Each SCSI transport protocol standard that defines power loss expected events should specify when those events result in the delivery of a Power Loss Expected event notification to the SCSI applications layer.

The power loss expected condition applies only to SCSI target devices and is equivalent to a CLEAR TASK SET task management function (see 7.5) applied to all task sets.

When a SCSI target port detects a power loss expected, a Power Loss Expected event notification indication shall be delivered to each logical unit to which the I_T nexus has access. In response to the resulting I_T power loss expected condition a logical unit shall take the following actions:

a) Abort all tasks and establish a unit attention condition as described in 5.6; and
b) Perform any additional functions required by the applicable protocol standards.

6.4 Event notification SCSI transport protocol services

The SCSI transport protocol services described in this subclause are used by a SCSI initiator port or a SCSI target port to deliver an indication to the SAL that a SCSI event has been detected.

All SCSI transport protocol standards should define the SCSI transport protocol specific requirements for implementing the **Nexus Loss** indication, the **Transport Reset** indication and the **Power Loss Expected** indication described in this subclause and when these indications are to be delivered to the SCSI applications layer.

The **Nexus Loss** indication and the **Transport Reset** indication are defined for both SCSI target devices and SCSI initiator devices.

Indication delivered to device servers, task managers, and application clients:

Nexus Loss (IN (I_T Nexus))

Argument description:

I_T Nexus: The specific I_T nexus that has been detected as lost.

Indication delivered to device servers, task managers, and application clients:

Transport Reset (IN (SCSI Port))

Argument descriptions:

SCSI Port: The specific SCSI port in the SCSI device for which a transport reset was detected.

The **Power Loss Expected** indication is defined for SCSI target devices.

Indication delivered to device servers and task managers.

Power Loss Expected (IN (SCSI Port))

Page: 100

Author: relliott Subject: Note Date: 10/27/2007 2:07:34 PM
Does power loss expected also abort task management functions? I think it should do so.

If so, then the comparison to "CLEAR TASK SET for all task sets" is incomplete, and the a) b) list needs to be expanded to include "abort all task management functions;"

If not, then rules in 7.11 about task management function lifetimes are incorrect.

Status
George Penokie Accepted 2/27/2008 11:09:21 AM -06'00'
gpenokie Completed 2/13/2008 2:55:29 PM -06'00'
Author: gpenokie Subject: Sticky Note Date: 2/13/2008 2:55:09 PM -06'00'
Added << terminate all task managment functions >> into the abc list. There is no equlivent to a clear task set for task management functions.

Author: relliott Subject: Highlight Date: 10/29/2007 10:40:17 AM
protocol standards
s/b
SCSI transport protocol standards

Status
George Penokie Accepted 11/1/2007 5:13:29 PM

7 Task management functions

7.1 Introduction

An application client requests the processing of a task management function by invoking the SCSI transport protocol services described in 7.12, the collective operation of which is modeled in the following procedure call:

Service Response = Function name (IN (nexus), OUT ([additional response information]))

The task management function names are summarized in table 37.

Table 37 — Task Management Functions

Task Management Function	Nexus	Additional Response Information argument supported	Reference
ABORT TASK	I_T_L_Q	no	7.2
ABORT TASK SET	I_T_L	no	7.3
CLEAR ACA	I_T_L	no	7.4
CLEAR TASK SET	I_T_L	no	7.5
I_T NEXUS RESET	I_T	no	7.6
LOGICAL UNIT RESET	I_T_L	no	7.7
QUERY TASK	I_T_L_Q	no	7.8
QUERY TASK SET	I_T_L	no	7.9
QUERY UNIT ATTENTION	I_T_L	yes	7.10

Input arguments:

Nexus: An I_T nexus, I_T_L nexus, or I_T_L_Q nexus (see 4.7) identifying the task or tasks affected by the task management function.

I_T Nexus: A SCSI initiator port and SCSI target port nexus (see 4.7).

I_T_L Nexus: A SCSI initiator port, SCSI target port, and logical unit nexus (see 4.7).

I_T_L_Q Nexus: A SCSI initiator port, SCSI target port, logical unit, and task identifier nexus (see 4.7).

Output arguments:

Additional Response Information: If supported by the SCSI transport protocol and the logical unit, then three bytes that are returned along with the service response for certain task management functions (e.g., QUERY UNIT ATTENTION). SCSI transport protocols may or may not support the Additional Response Information argument. A SCSI transport protocol supporting the Additional Response Information argument may or may not require that logical units accessible through a target port using that transport protocol support the Additional Response Information argument.

Page: 102

Author: relliott Subject: Highlight Date: 10/16/2007 7:01:57 PM
Introduction
s/b
Task management function procedure calls

(to parallel the section heading of 5.1 for commands)

Status
George Penokie Accepted 11/1/2007 5:14:21 PM
Author: relliott Subject: Highlight Date: 10/10/2007 2:02:44 PM
the following procedure call
s/b
a procedure call using the following format:

Status
George Penokie Accepted 11/1/2007 5:14:55 PM
Author: relliott Subject: Highlight Date: 10/27/2007 1:22:12 PM
(IN (nexus), OUT ([additional response information])
s/b
(IN (Nexus), OUT ([Additional Response Information])

Status
George Penokie Accepted 11/1/2007 5:15:43 PM
Author: relliott Subject: Note Date: 10/27/2007 1:22:10 PM
add the following after the Service Response = line:
where:
Function Name is one of the task management function names listed in table 34
Nexus is either:
a) an I_T Nexus argument;
b) an I_T_L Nexus Argument; or
c) an I_T_L_Q Nexus argument
Additional Response Information is the Additional Response Information output argument described below

Status
George Penokie Rejected 11/2/2007 9:02:17 AM
Author: George Penokie Subject: Note Date: 2/13/2008 12:12:30 PM -06'00'
Added an <<(i.e., function name >>, and << nexus argument >> into the headings. Added << nexus >> into each nexus column. Changed the nexus input argument to << Contains an i_T nexus argument, ... >>

Author: relliott Subject: Highlight Date: 10/27/2007 1:22:04 PM
Nexus: An I_T nexus, I_T_L nexus, or I_T_L_Q nexus (see 4.7) identifying the task or
tasks affected by the task management function.
I_T Nexus: A SCSI initiator port and SCSI target port nexus (see 4.7).
I_T_L Nexus: A SCSI initiator port, SCSI target port, and logical unit nexus (see 4.7).
I_T_L_Q Nexus: A SCSI initiator port, SCSI target port, logical unit, and task identifier nexus (see 4.7).

s/b
I_T Nexus: The I_T nexus (see 4.7) affected by the task management function.
I_T_L Nexus: The I_T_L nexus (see 4.7) affected by the task management function.
I_T_L_Q Nexus: The I_T_L_Q nexus (see 4.7) affected by the task management function.

Status
George Penokie Accepted 11/2/2007 9:05:36 AM

One of the following ~~SCSI transport protocol specific service responses~~ shall be returned:

FUNCTION COMPLETE:	A task manager response indicating that the requested function is complete. Unless another response is required, the task manager shall return this response upon completion of a task management request supported by the logical unit or SCSI target device to which the request was directed.
FUNCTION SUCCEEDED:	A task manager response indicating that the requested function is supported and completed successfully. This task manager response shall only be used by functions that require notification of success (e.g., QUERY TASK, QUERY TASK SET, or QUERY UNIT ATTENTION).
FUNCTION REJECTED:	A task manager response indicating that the requested function is not supported by the logical unit or SCSI target device to which the function was directed.
INCORRECT LOGICAL UNIT NUMBER:	A task router response indicating that the function requested processing for an incorrect logical unit number.
SERVICE DELIVERY OR TARGET FAILURE:	The request was terminated due to a service delivery failure (see 3.1.113) or SCSI target device malfunction. The task manager may or may not have successfully performed the specified function.

Each SCSI transport protocol standard shall define the events for each of these service responses.

The task manager response to task management requests is subject to the presence of access restrictions, as managed by ACCESS CONTROL OUT and ACCESS CONTROL IN commands (see SPC-4), as follows:

a) A task management request of ABORT TASK, ABORT TASK SET, CLEAR ACA, I_T NEXUS RESET, QUERY TASK, QUERY TASK SET, or QUERY UNIT ATTENTION shall not be affected by the presence of access restrictions;
b) A task management request of CLEAR TASK SET or LOGICAL UNIT RESET received from a SCSI initiator port that is denied access to the logical unit, either because it has no access rights or because it is in the pending-enrolled state, shall not cause any changes to the logical unit; and
c) The task management function service response shall not be affected by the presence of access restrictions.

7.2 ABORT TASK

Request:

Service Response = ABORT TASK (IN (I_T_L_Q Nexus))

Description:

This function shall be supported by all logical units.

The task manager shall abort the specified task, if any, as described in 5.6. Previously established conditions, including mode parameters, reservations, and ACA shall not be changed by the ABORT TASK function.

A response of FUNCTION COMPLETE shall indicate that the task was aborted or was not in the task set. In either case, the SCSI target device shall guarantee that no further requests or responses are sent from the task.

All SCSI transport protocol standards shall support the ABORT TASK task management function.

Page: 103

Author: relliott Subject: Highlight Date: 10/27/2007 1:21:52 PM
One of the following SCSI transport protocol specific service responses shall be returned
s/b
Service Response assumes one of the following values

to match wording in section 5.1 for Execute Command. Wording could be changed in both places if "assumes" is not agreeable. There is no need for the Service Response values to be determined by the transport protocol here. When included in RESPONSE frames over the wire, they are; when returned from the initiator port to the application client, the values are probably remapped to protocol-independent values (so generic SCSI software isn't affected by the transport protocol choice).

Status
George Penokie Completed 11/2/2007 9:17:56 AM
gpenokie Rejected 2/13/2008 11:57:44 AM -06'00'
Author: George Penokie Subject: Note Date: 11/2/2007 9:17:50 AM
Changed the wording in section 5.1 to "One of the following SCSI transport protocol specific service responses shall be returned:" so it will match what is stated here and gets rid of the "assumes" term.

Author: relliott Subject: Note Date: 10/10/2007 2:08:40 PM
At end of SERVICE DELIVERY OR TARGET FAILURE description, add "All output arguments are invalid."

That means Additional Response Information is not usable.

Status
gpenokie Accepted 2/13/2008 11:53:30 AM -06'00'
Author: relliott Subject: Highlight Date: 10/10/2007 1:57:48 PM
Request
s/b
Procedure call:

(this is at the same level as Execute Command, not the same level as Send SCSI Command)

Status
George Penokie Accepted 11/1/2007 5:22:41 PM

7.3 ABORT TASK SET

Request:

Service Response = ABORT TASK SET (IN (I_T_L Nexus))

Description:

This function shall be supported by all logical units.

The task manager shall abort all tasks in the task set that were received on the specified I_T nexus as described in 5.6. Tasks received on other I_T nexuses or in other task sets shall not be aborted. This task management function performed is equivalent to a series of ABORT TASK requests.

All pending status and sense data for the tasks that were aborted shall be cleared. Other previously established conditions, including mode parameters, reservations, and ACA shall not be changed by the ABORT TASK SET function.

All SCSI transport protocol standards shall support the ABORT TASK SET task management function.

7.4 CLEAR ACA

Request:

Service Response = CLEAR ACA (IN (I_T_L Nexus))

Description:

This function shall be supported by a logical unit if it supports ACA (see 5.2).

For the CLEAR ACA task management function, the task set shall be the one defined by the TST field in the Control mode page (see SPC-4).

An application client requests a CLEAR ACA using the faulted I_T nexus (see 3.1.38) to clear an ACA condition from the task set serviced by the logical unit. The state of all tasks in the task set shall be modified as described in 8.8. For a task with the ACA task attribute (see 8.6.5) receipt of a CLEAR ACA function shall have the same effect as receipt of an ABORT TASK function (see 7.2) specifying that task. If successful, this function shall be terminated with a service response of FUNCTION COMPLETE.

If the task manager clears the ACA condition, any task within that task set may be completed subject to the requirements for task set management specified in clause 8.

The service response for a CLEAR ACA request received from an I_T nexus other than the faulted I_T nexus shall be FUNCTION REJECTED.

All SCSI transport protocol standards shall support the CLEAR ACA task management function.

7.5 CLEAR TASK SET

Request:

Service Response = CLEAR TASK SET (IN (I_T_L Nexus))

Description:

This function shall be supported by all logical units.

Page: 104

Author: relliott Subject: Highlight Date: 10/10/2007 1:57:58 PM
Request
s/b
Procedure call:

(this is at the same level as Execute Command, not the same level as Send SCSI Command)

Status
George Penokie Accepted 11/1/2007 5:22:49 PM

Author: relliott Subject: Highlight Date: 10/10/2007 1:58:05 PM
Request
s/b
Procedure call:

(this is at the same level as Execute Command, not the same level as Send SCSI Command)

Status
George Penokie Accepted 11/1/2007 5:22:57 PM

Author: relliott Subject: Highlight Date: 10/8/2007 6:26:29 PM
ACA task attribute
s/b ACA (smallcaps lowercase) task attribute

Status
George Penokie Accepted 11/1/2007 5:24:28 PM

Author: relliott Subject: Highlight Date: 10/10/2007 1:58:12 PM
Request
s/b
Procedure call:

(this is at the same level as Execute Command, not the same level as Send SCSI Command)

Status
George Penokie Accepted 11/1/2007 5:23:05 PM

The task manager shall abort all tasks in the task set as described in 5.6.

If the TST field is set to 001b (i.e., per I_T nexus) in the Control mode page (see SPC-4), there is one task set per I_T nexus. As a result, no other I_T nexuses are affected and CLEAR TASK SET is equivalent to ABORT TASK SET (see 7.2).

All pending status and sense data for the task set shall be cleared. Other previously established conditions, including mode parameters, reservations, and ACA shall not be changed by the CLEAR TASK SET function.

All SCSI transport protocol standards shall support the CLEAR TASK SET task management function.

7.6 I_T NEXUS RESET

Request:

Service Response = I_T NEXUS RESET (IN (I_T Nexus))

Description:

SCSI transport protocols may or may not support I_T NEXUS RESET and may or may not require logical units accessible through SCSI target ports using such transport protocols to support I_T NEXUS RESET.

Each logical unit accessible through the SCSI target port shall perform the I_T nexus loss functions specified in 6.3.4 for the I_T nexus on which the function request was received, then the SCSI target device shall return a FUNCTION COMPLETE response. After returning a FUNCTION COMPLETE response, the logical unit(s) and the SCSI target port shall perform any additional functions specified by the SCSI transport protocol.

7.7 LOGICAL UNIT RESET

Request:

Service Response = LOGICAL UNIT RESET (IN (I_T_L Nexus))

Description:

This function shall be supported by all logical units.

Before returning a FUNCTION COMPLETE response, the logical unit shall perform the logical unit reset functions specified in 6.3.3.

NOTE 13 - Previous versions of this standard only required LOGICAL UNIT RESET support in logical units that supported hierarchical logical units.

All SCSI transport protocol standards shall support the LOGICAL UNIT RESET task management function.

7.8 QUERY TASK

Request:

Service Response = QUERY TASK (IN (I_T_L_Q Nexus))

Description:

Page: 105

Author: relliott Subject: Highlight Date: 10/10/2007 1:58:19 PM
Request
s/b
Procedure call:

(this is at the same level as Execute Command, not the same level as Send SCSI Command)

Status
George Penokie Accepted 11/1/2007 5:23:13 PM

Author: relliott Subject: Highlight Date: 10/10/2007 1:58:26 PM
Request
s/b
Procedure call:

(this is at the same level as Execute Command, not the same level as Send SCSI Command)

Status
George Penokie Accepted 11/1/2007 5:23:20 PM

Author: relliott Subject: Highlight Date: 10/10/2007 1:58:35 PM
Request
s/b
Procedure call:

(this is at the same level as Execute Command, not the same level as Send SCSI Command)

Status
George Penokie Accepted 11/1/2007 5:23:30 PM

SCSI transport protocols may or may not support QUERY TASK and may or may not require logical units accessible through SCSI target ports using such transport protocols to support QUERY TASK.

The task manager in the specified logical unit shall:

a) if the specified task is present in the task set, then return a service response set to FUNCTION SUCCEEDED; and
b) if the specified task is not present in the task set, then return a service response set to FUNCTION COMPLETE.

7.9 QUERY TASK SET

Request:

Service Response = QUERY TASK SET (IN (I_T_L Nexus))

Description:

SCSI transport protocols may or may not support QUERY TASK SET and may or may not require logical units accessible through SCSI target ports using such transport protocols to support QUERY TASK SET.

The task manager in the specified logical unit shall:

a) if there is any task present in the task set from the specified I_T nexus, then return a service response set to FUNCTION SUCCEEDED; and
b) if there is no task present in the task set from the specified I_T nexus, then return a service response set to FUNCTION COMPLETE.

7.10 QUERY UNIT ATTENTION

Request:

Service Response = QUERY UNIT ATTENTION (IN (I_T_L Nexus), OUT ([Additional Response Information]))

Description:

A SCSI transport protocol may or may not support QUERY UNIT ATTENTION. A SCSI transport protocol supporting QUERY UNIT ATTENTION may or may not require logical units accessible through SCSI target ports using that transport protocol to support QUERY UNIT ATTENTION.

The task manager in the specified logical unit shall:

a) if there is a unit attention condition (see 5.8.7) or a deferred error (see SPC-4) pending for the specified I_T nexus, then return a service response set to FUNCTION SUCCEEDED; and
b) if there is no unit attention condition or deferred error pending for the specified I_T nexus, then return a service response set to FUNCTION COMPLETE.

If the service response is not FUNCTION SUCCEEDED, then the task manager shall set the Additional Response Information argument to 000000h.

If the service response is FUNCTION SUCCEEDED, then the task manager shall set the Additional Response Information argument as defined in table 38.

Page: 106

Author: relliott Subject: Highlight Date: 10/10/2007 2:01:19 PM
Request
s/b
Procedure call:

(this is at the same level as Execute Command, not the same level as Send SCSI Command)

Status
George Penokie Accepted 11/1/2007 5:23:37 PM

Author: relliott Subject: Highlight Date: 10/10/2007 2:01:25 PM
Request
s/b
Procedure call:

(this is at the same level as Execute Command, not the same level as Send SCSI Command)

Status
George Penokie Accepted 11/1/2007 5:23:41 PM

Table 38 — Additional Response Information argument for QUERY UNIT ATTENTION

Bit Byte	7	6	5	4	3	2	1	0
0	Reserved		UADE DEPTH		SENSE KEY			
1	ADDITIONAL SENSE CODE							
2	ADDITIONAL SENSE CODE QUALIFIER							

The UADE DEPTH field indicates the number of pending unit attention conditions or deferred errors and is defined in table 39.

Table 39 — UADE DEPTH field

Code	Description
00b	The combined number of unit attention conditions and deferred errors is unknown
01b	The combined number of unit attention conditions and deferred errors is one
10b	The combined number of unit attention conditions and deferred errors is greater than one
11b	Reserved

The SENSE KEY field indicates the value of the SENSE KEY field that would be returned in the sense data for the highest-priority pending unit attention condition or deferred error (see SPC-4).

The ADDITIONAL SENSE CODE field indicates the value of the ADDITIONAL SENSE CODE field in the highest-priority pending unit attention condition or deferred error (see SPC-4).

The ADDITIONAL SENSE CODE QUALIFIER field indicates the value of the ADDITIONAL SENSE CODE QUALIFIER field in the highest-priority pending unit attention condition or deferred error (see SPC-4).

7.11 Task management function lifetime

The task manager shall create a task management function upon receiving a Task Management Request Received indication (see 7.12). The task management function shall exist until:

a) the task manager sends a SCSI transport protocol service response for the task management function;
b) an I_T nexus loss (see 6.3.4);
c) a logical unit reset (see 6.3.3);
d) a hard reset (see 6.3.2); or
e) a power on condition (see 6.3.1).

The application client maintains an application client task to interact with the task management function from the time the **Send Task Management Request** SCSI transport protocol service request is invoked until it receives one of the following SCSI target device responses:

a) A service response of FUNCTION COMPLETE, FUNCTION SUCCEEDED, FUNCTION REJECTED, or SERVICE DELIVERY OR TARGET FAILURE is received for that task management function;
b) Notification of a unit attention condition with any additional sense code whose ADDITIONAL SENSE CODE field contains 29h (e.g., POWER ON, RESET, OR BUS DEVICE RESET OCCURRED; POWER ON

Page: 107

Author: relliott Subject: Note Date: 10/27/2007 1:58:54 PM
Add "power loss expected (see 6.3.5)" to the list of things that cause a task management function to no longer exist. Make it item b) ahead of I_T nexus loss.

Status
gpenokie Accepted 2/13/2008 11:32:00 AM -06'00'

Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:21:51 PM
"The application client maintains an application client task to interact with the task management function from the time the Send Task Management Request SCSI transport protocol service request is invoked until it receives one of the following SCSI target device responses:"
s/b
"An application client maintains an application client task to interact with the task management function from the time the Send Task Management Request SCSI transport protocol service request is invoked until the application client receives one of the following SCSI target device responses:"

Status
George Penokie Accepted 11/2/2007 9:36:22 AM

Author: relliott Subject: Highlight Date: 10/16/2007 7:44:25 PM
application client task to interact with the task management function
s/b
application client task management function to represent the task management function

Status
George Penokie Accepted 11/2/2007 9:38:48 AM

Author: relliott Subject: Highlight Date: 10/16/2007 7:54:18 PM
FUNCTION COMPLETE, FUNCTION SUCCEEDED, FUNCTION REJECTED, or SERVICE DELIVERY OR TARGET FAILURE

1. Add INCORRECT LOGICAL UNIT NUMBER

2. If SERVICE DELIVERY OR TARGET FAILURE remains in this list, then there is no reason to list all of them - any service response suffices, so delete the list and just leave "A service response is received". See other comment about excluding SERVICE DELIVERY OR TARGET FAILURE, though.

Status
George Penokie Accepted 12/26/2007 1:51:32 PM -06'00'

Author: George Penokie Subject: Note Date: 11/20/2007 9:01:58 AM -06'00'
Added "incorrect logical unit number' to the list

Author: relliott Subject: Note Date: 11/2/2007 9:44:25 AM
For commands (5.5), a service response of SERVICE DELIVERY OR TARGET FAILURE leaves the application client task in existence until the initiator receives something else from the target that assures it is gone (a response to a TMF aborting that task).

Task management functions should be handled the same way. It is not safe to reuse the task identifier (task tag) if a SERVICE DELIVERY OR TARGET FAILURE is returned. The task management function should be assumed to exist until an I_T NEXUS RESET or LOGICAL UNIT RESET is successfully run (or a unit attention occurs reporting a reset).

Status
gpenokie Accepted 2/13/2008 11:47:23 AM -06'00'

Author: gpenokie Subject: Sticky Note Date: 2/13/2008 2:23:57 PM -06'00'
Added << If a service response of service delivery or target failure is received for a task management function (e.g., when an I_T nexus loss is detected by the SCSI initiator port), the application client shall maintain an application client task management function to represent the task management function until the application client has determined that the task management function is no longer known to the device server. >> to this section.

OCCURRED; SCSI BUS RESET OCCURRED; BUS DEVICE RESET FUNCTION OCCURRED; DEVICE INTERNAL RESET; or I_T NEXUS LOSS OCCURRED);

c) Notification of a unit attention condition with an additional sense code of MICROCODE HAS BEEN CHANGED; or

d) Notification of a unit attention condition with an additional sense code of COMMANDS CLEARED BY POWER LOSS NOTIFICATION.

NOTE 14 - The names of the unit attention conditions listed in the subclause (e.g., SCSI BUS RESET OCCURRED) are based on usage in previous versions of this standard. The use of these unit attention condition names is not to be interpreted as a description of how the unit attention conditions are represented by any given SCSI transport protocol.

7.12 Task management SCSI transport protocol services

7.12.1 Task management SCSI transport protocol services overview

The SCSI transport protocol services described in this subclause are used by a SCSI initiator device and SCSI target device to process a task management procedure call. The following arguments are passed:

Nexus: An I_T nexus, I_T_L nexus, or I_T_L_Q nexus (see 4.7).

Function Identifier: Argument encoding the task management function to be performed.

All SCSI transport protocol standards shall define the SCSI transport protocol specific requirements for implementing the **Send Task Management Request** request (see 7.12.2), the **Task Management Request Received** indication (see 7.12.3), the **Task Management Function Executed** response (see 7.12.4), and the **Received Task Management Function Executed** (see 7.12.5) confirmation SCSI transport protocol services.

A SCSI transport protocol standard may specify different implementation requirements for the **Send Task Management Request** request SCSI transport protocol service for different values of the Function Identifier argument.

All SCSI initiator devices shall implement the **Send Task Management Request** and the **Received Task Management Function Executed** confirmation SCSI transport protocol services as defined in the applicable SCSI transport protocol standards.

All SCSI target devices shall implement the **Task Management Request Received** indication and the **Task Management Function Executed** response SCSI transport protocol services as defined in the applicable SCSI transport protocol standards.

7.12.2 Send Task Management Request transport protocol service request

An application client uses the Send Task Management Request transport protocol service request to request that a SCSI initiator port send a task management function.

Send Task Management Request transport protocol service request:

Send Task Management Request (IN (Nexus, Function Identifier))

Input arguments:

Nexus: An I_T nexus, I_T_L nexus, or I_T_L_Q nexus (see 4.7).

Function Identifier: Argument encoding the task management function to be performed.

Page: 108

Author: relliott Subject: Note Date: 10/16/2007 7:56:51 PM
Items b) c) and d) should be qualified with knowledge that the unit attention condition was reported after the task management request arrived at the target port. Otherwise, it might still be in flight. This is the subtle ordering assumption in 4.3.3.

Status
gpenokie Accepted 2/13/2008 11:51:54 AM -06'00'
Author: gpenokie Subject: Sticky Note Date: 2/13/2008 2:29:44 PM -06'00'
This << Items other than a) assume in-order delivery (see 4.3.3). >> was added to this section.

Author: relliott Subject: Highlight Date: 10/10/2007 1:53:20 PM
(see 7.12.5) confirmation
s/b
confirmation (see 7.12.5)

Status
George Penokie Accepted 11/1/2007 5:26:20 PM

Author: relliott Subject: Highlight Date: 10/10/2007 1:53:42 PM
Send Task Management Request
s/b
Send Task Management Request request

Status
George Penokie Accepted 11/1/2007 5:26:59 PM

7.12.3 Task Management Request Received transport protocol service indication

A task router (see 4.5.8) uses the Task Management Request Received transport protocol service indication to notify a task manager that it has received a task management function.

Task Management Request Received transport protocol service indication:

Task Management Request Received (IN (Nexus, Function Identifier))

Input arguments:

Nexus: An I_T nexus, I_T_L nexus, or I_T_L_Q nexus (see 4.7).

Function Identifier: Argument encoding the task management function to be performed.

7.12.4 Task Management Function Executed transport protocol service response

A task manger uses the Task Management Function Executed transport protocol service response to request that a SCSI target port transmit task management function executed information.

Task Management Function Executed transport protocol service response:

Task Management Function Executed (IN (Nexus, Service Response))

Input arguments:

Nexus: An I_T nexus, I_T_L nexus, or I_T_L_Q nexus (see 4.7).

Service Response: An encoded value representing one of the following:

FUNCTION COMPLETE:	The requested function has been completed.
FUNCTION SUCCEEDED:	The requested function is supported and completed successfully.
FUNCTION REJECTED:	The task manager does not implement the requested function.
INCORRECT LOGICAL UNIT NUMBER:	An optional task router response indicating that the function requested processing for an incorrect logical unit number.
SERVICE DELIVERY OR TARGET FAILURE:	The request was terminated due to a service delivery failure (see 3.1.113) or SCSI target device malfunction. The task manager may or may not have successfully performed the specified function.

7.12.5 Received Task Management Function Executed transport protocol service confirmation

A SCSI initiator port uses the Received Task Management Function Executed transport protocol service confirmation to notify an application client that it has received task management function executed information.

Received Task Management Function Executed transport protocol service confirmation:

Received Task Management Function Executed (IN (Nexus, Service Response))

Page: 109

Author: relliott Subject: Highlight Date: 10/10/2007 1:54:12 PM
task manger
s/b
task manager

Status
George Penokie Accepted 11/1/2007 5:27:34 PM

Author: relliott Subject: Highlight Date: 10/27/2007 1:21:12 PM
(IN (Nexus, Service Response))
s/b
(IN (Nexus, Service Response, [Additional Response Information]))

with this added to Input arguments:
Additional Response Information: The Additional Response Information output argument for the task management procedure call (see 7.1):

Status
George Penokie Completed 11/2/2007 10:07:34 AM
gpenokie Accepted 2/13/2008 11:27:51 AM -06'00'

Author: relliott Subject: Highlight Date: 10/27/2007 1:21:10 PM
(IN (Nexus, Service Response))
s/b
(IN (Nexus, Service Response, [Additional Response Information]))

with this added to Input arguments:
Additional Response Information: The Additional Response Information output argument for the task management procedure call (see 7.1):

Status
George Penokie Completed 11/2/2007 10:10:49 AM
gpenokie Accepted 2/13/2008 11:23:55 AM -06'00'

Input arguments:

Nexus: An I_T nexus, I_T_L nexus, or I_T_L_Q nexus (see 4.7).

Service Response: An encoded value representing one of the following:

FUNCTION COMPLETE:	The requested function has been completed.
FUNCTION SUCCEEDED:	The requested function is supported and completed successfully.
FUNCTION REJECTED:	The task manager does not implement the requested function.
INCORRECT LOGICAL UNIT NUMBER:	An optional task router response indicating that the function requested processing for an incorrect logical unit number.
SERVICE DELIVERY OR TARGET FAILURE:	The request was terminated due to a service delivery failure (see 3.1.113) or SCSI target device malfunction. The task manager may or may not have successfully performed the specified function.

Each SCSI transport protocol shall allow a **Received Task Management Function Executed** confirming completion of the requested task to be associated with the corresponding **Send Task Management Request**.

7.13 Task management function example

Figure 42 shows the sequence of events associated with a task management function.

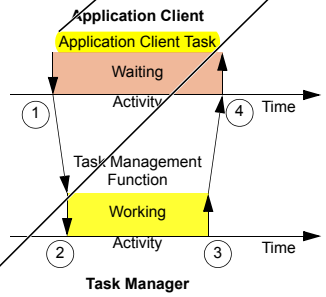


Figure 42 — Task management processing events

The numbers in figure 42 identify the events described as follows:

1) The application client task issues a task management request by invoking the **Send Task Management Request** SCSI transport protocol service.
2) The task manager is notified through a **Task Management Request Received** transport protocol service indication and begins processing the function.

Page: 110

Author: relliott Subject: Highlight Date: 10/16/2007 7:40:10 PM
Application Client Task
s/b
Application Client Task Management Function

Status
George Penokie Accepted 11/1/2007 5:31:21 PM

Author: relliott Subject: Highlight Date: 10/16/2007 7:40:30 PM
application client task
s/b
application client

Status
George Penokie Accepted 11/2/2007 10:12:14 AM

3) The task manager performs the requested function and responds by invoking the **Task Management Function Executed** SCSI transport protocol service to notify the application client. The service response argument is set to a value of FUNCTION COMPLETE.
4) A **Received Task Management Function Executed** confirmation is received by the application client task.

Page: 111

Author: relliott Subject: Highlight Date: 10/27/2007 2:13:40 PM
service response argument
s/b
Service Response argument

or just replace "service response argument is set to a value of" with "service response of" (this is how most of the standard is worded)

Status
George Penokie Rejected 11/2/2007 10:17:11 AM
Author: George Penokie Subject: Note Date: 2/13/2008 11:10:51 AM -06'00'
Changed to "...client of a service response of FUNCTION COMPLETE."

Author: relliott Subject: Highlight Date: 10/16/2007 7:40:52 PM
application client task
s/b
application client

Status
George Penokie Accepted 11/2/2007 10:17:39 AM

8 Task set management

8.1 Introduction to task set management

This clause describes some of the controls that application clients have over task set management behaviors (see 8.3). This clause also specifies task set management requirements in terms of:

a) Task states (see 8.5);
b) Task attributes (see 8.6);
c) Task priority (see 8.7);
d) The events that cause transitions between task states (see 8.4 and 8.5); and
e) A map of task state transitions (see 8.8).

This clause concludes with several task set management examples (see 8.9).

Task behavior, as specified in this clause, refers to the functioning of a task as observed by an application client, including the results of command processing and interactions with other tasks.

The requirements for task set management only apply to a task after it has been entered into a task set. A task shall be entered into a task set unless:

a) A condition exists that causes that task to be completed with a status of BUSY, RESERVATION CONFLICT, TASK SET FULL, or ACA ACTIVE;
b) Detection of an overlapped command (see 5.8.3) causes that task to be completed with a CHECK CONDITION status; or
c) SCSI transport protocol specific errors cause that task to be completed with CHECK CONDITION status.

8.2 Implicit head of queue

A command standard (see 3.1.20) may define commands each of which may be processed by the task manager as if the task's task attribute is HEAD OF QUEUE even if the task is received with a SIMPLE task attribute, an ORDERED task attribute~~, or no task attribute.~~

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 is vendor specific.

8.3 Task management model

The task management model requires the following task set management behaviors:

a) The SIMPLE task attribute (see 8.6.1) shall be supported;
b) Task attributes other than SIMPLE may be supported;
c) The QUEUE ALGORITHM MODIFIER field in the Control mode page (see SPC-4) shall control the processing sequence of tasks having the SIMPLE task attribute;
d) The QERR field in the Control mode page (see SPC-4) shall control aborting of tasks when a CHECK CONDITION status is returned for any task; and
e) The CLEAR TASK SET task management function (see 7.5) shall be supported.

Page: 112

Author: relliott Subject: Cross-Out Date: 10/27/2007 1:32:08 PM
Delete ", or no task attribute." and move the "or" to earlier in the sentence.

SAM-4 requires each task have a task attribute.

Status
George Penokie Accepted 11/2/2007 10:19:34 AM

Page: 115

Author: relliott Subject: Note Date: 10/27/2007 1:20:52 PM
When referring to "(simple, ordered, head of queue, and ACA) task", Mixed Case should be used, not lowercase.

Status
George Penokie Rejected 11/2/2007 10:26:25 AM
Author: George Penokie Subject: Note Date: 11/2/2007 10:26:20 AM
I see no instance where it is not correct. The table is a list of task attributes which are allways small caps.

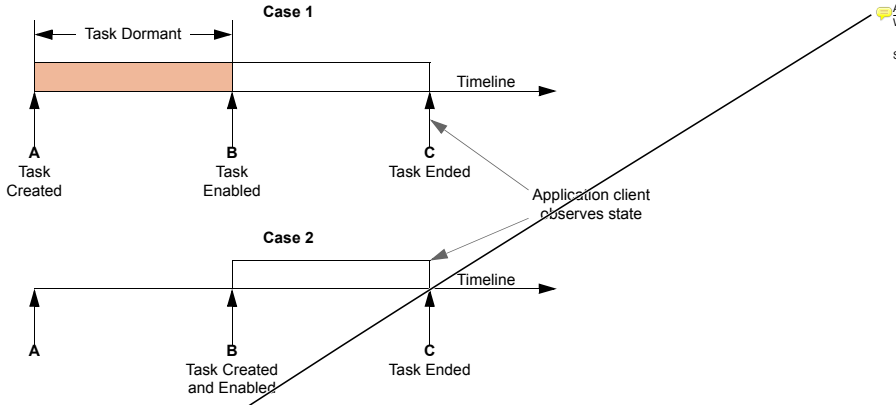


Figure 43 — Example of Dormant state task behavior

8.6 Task attributes

8.6.1 Overview

The application client shall assign a task attribute (see table 41) to each task.

Table 41 — Task attributes

Task Attribute	Reference
SIMPLE	8.6.2
ORDERED	8.6.3
HEAD OF QUEUE	8.6.4
ACA	8.6.5

SCSI transport protocols shall provide the capability to specify a unique task attribute for each task.

8.6.2 Simple task

If accepted, a task having the SIMPLE task attribute shall be entered into the task set in the dormant task state. The task shall not enter the enabled task state until all head of queue tasks and older ordered tasks in the task set have ended (see 8.4).

The QUEUE ALGORITHM MODIFIER field in the Control mode page (see SPC-4) provides additional constraints on task completion order for tasks having the SIMPLE task attribute.

8.6.3 Ordered task

If accepted, a task having the ORDERED task attribute shall be entered into the task set in the dormant task state. The task shall not enter the enabled task state until all head of queue tasks and all older tasks in the task set have ended (see 8.4).

8.6.4 Head of queue task

If accepted, a task having the HEAD OF QUEUE task attribute shall be entered into the task set in the enabled task state.

8.6.5 ACA task

If accepted, a task having the ACA task attribute shall be entered into the task set in the enabled task state. There shall be no more than one ACA task per task set (see 5.8.2.2).

8.7 Task priority

Task priority specifies the relative scheduling importance of a task having a SIMPLE task attribute in relation to other tasks having SIMPLE task attributes already in the task set. If the task has a task attribute other than SIMPLE, the task priority is not used. Task priority is a value in the range of 0h through Fh. A task with either no task priority or a task priority set to 0h has a vendor-specific level of scheduling importance. A task with a task priority set to 1h has the highest scheduling importance, with increasing task priority values indicating decreasing scheduling importance. A task with a task priority set to Fh has the lowest scheduling importance.

If the Task Priority argument is set to zero or is not contained within the Send SCSI Received SCSI transport protocol service indication (see 5.4.2) and a priority has been assigned to the I_T_L nexus, the device server shall use that priority as the task priority. A priority may be assigned to an I_T_L nexus by a SET PRIORITY command (see SPC-4) or by the INITIAL PRIORITY field in the Control Extension mode page (see SPC-4). If no priority has been assigned to the I_T_L nexus using the SET PRIORITY command and the logical unit does not support the INITIAL PRIORITY field in the Control Extension mode page the device server shall set the task priority to 0h (i.e., vendor specific) or the task shall have no task priority.

A task manager may use task priority to determine an ordering to process tasks with the SIMPLE task attribute within the task set. A difference in task priority between tasks may not override other scheduling considerations (e.g., different times to access different logical block addresses) or vendor specific scheduling considerations. However, processing of a collection of tasks with different task priorities should cause the subset of tasks with the higher task priorities to return status sooner in aggregate than the same subset would if the same collection of tasks were submitted under the same conditions but with all task priorities being equal.

8.8 Task state transitions

This subclause describes task state transitions, actions and associated triggering events as they appear to an application client. The logical unit response to events affecting multiple tasks (e.g., a CLEAR TASK SET) may be different from the response to an event affecting a single task. To the application client, the collective behavior appears as a series of state changes occurring to individual tasks.

The task state diagram of figure 44 shows the behavior of a single task in response to an external event.

Page: 116

Author: relliott Subject: Highlight Date: 12/11/2007 4:33:10 PM -06'00'
"task priority" might also be renameable at this point. How much is it used in SPC-4?

Or, we could strive to have all these extra attributes of a command outside the CDB to have a "task " prefix. Task attribute is unlikely to change to command attribute.

Status
gpenokie Accepted 2/13/2008 2:10:23 PM -06'00'
Author: gpenokie Subject: Sticky Note Date: 2/13/2008 11:10:19 AM -06'00'
Change all << task priority >> to << command priority >> send note to Ralph

Author: Mark Evans, WDC Subject: Highlight Date: 10/29/2007 1:20:57 PM
"If the task has a task attribute other than SIMPLE, the task priority is not used."
s/b
"If a task has a task attribute other than SIMPLE, then task priority is not used."

Status
George Penokie Accepted 11/2/2007 10:27:31 AM
Author: Mark Evans, WDC Subject: Highlight Date: 10/25/2007 12:52:03 PM
"If the Task Priority argument is set to zero or is not contained within the Send SCSI Received SCSI transport protocol service indication (see 5.4.2) and a priority has been assigned to the I_T_L nexus, the device server shall use that priority as the task priority."
s/b
"If the Task Priority argument is set to zero or is not contained within the Send SCSI Received SCSI transport protocol service indication (see 5.4.2), and a priority has been assigned to the I_T_L nexus, then the device server shall use the priority specified for the I_T_L nexus as the task priority."

Status
George Penokie Rejected 11/2/2007 10:31:15 AM
Author: George Penokie Subject: Note Date: 11/8/2007 3:26:50 PM -06'00'
Change to: If the Task Priority argument is set to zero or is not contained within the Send SCSI Received SCSI transport protocol service indication (see 5.4.2), and a priority has been assigned to the I_T_L nexus, then the device server shall use the specified priority for the I_T_L nexus as the task priority.

Author: relliott Subject: Highlight Date: 10/8/2007 7:33:21 PM
A priority may be assigned
s/b
A priority is assigned

Status
George Penokie Accepted 2/13/2008 4:51:19 PM -06'00'
Author: Mark Evans, WDC Subject: Highlight Date: 10/25/2007 12:53:40 PM
"If no priority has been assigned to the I_T_L nexus using the SET PRIORITY command and the logical unit does not support the INITIAL PRIORITY field in the Control Extension mode page the device server shall set the task priority to 0h (i.e., vendor specific) or the task shall have no task priority."
s/b
"If no priority has been assigned to the I_T_L nexus using the SET PRIORITY command, and the logical unit does not support the INITIAL PRIORITY field in the Control Extension mode page, then the device server shall set the task priority to 0h (i.e., vendor specific), or the task shall have no task priority."

Status
George Penokie Accepted 11/2/2007 10:33:02 AM

Annex A
(informative)

Identifiers and names for objects

A.1 Identifiers and names overview

This annex summarizes identifiers and names.

The following SCSI architecture model objects have identifiers and names summarized in this annex:

a) SCSI initiator port (see 3.1.97);
b) SCSI target port (see 3.1.101);
c) Logical unit (see 3.1.62); and
d) SCSI device (see 3.1.92)

A.2 Identifiers and names

This standard defines the identifier attributes and name attributes for the attribute listed in A.1. The size requirements placed on identifier attributes by this standard are as shown in table A.1. This standard places no requirements on the sizes of name attributes as shown in table A.2. Table A.1 also lists whether this standard or SPC-4 requires SCSI transport protocols and logical units to support identifier attributes. Table A.2 also lists whether this standard or SPC-4 requires SCSI transport protocols and logical units to support name attributes.

Table A.1 — Identifier attribute size and support requirements

Attribute	Identifier	
	Size	Support Requirements
Initiator port identifier	not specified	mandatory
Target port identifier	not specified	mandatory
Logical unit number	8 bytes (maximum)	mandatory

Page: 122

Author: Mark Evans, WDC Subject: Highlight Date: 10/25/2007 12:57:00 PM
"This standard defines the identifier attributes and name attributes for the attribute listed in A.1."
s/b
"This standard defines the identifier attributes and name attributes for the SCSI architecture model objects listed in A.1."

Status
George Penokie Rejected 11/2/2007 10:38:04 AM
Author: George Penokie Subject: Note Date: 11/2/2007 10:37:31 AM
Changed to "This standard defines the identifier attributes and name attributes listed in A.1."

Author: Emulex Subject: Highlight Date: 10/30/2007 2:02:51 PM
Emulex-014
Page: 122 A.2 first sentence "attribute" s/b "attributes"

Status
George Penokie Rejected 11/2/2007 10:37:09 AM
Author: George Penokie Subject: Note Date: 11/2/2007 10:37:06 AM
Changed to "This standard defines the identifier attributes and name attributes listed in A.1.

Author: Emulex Subject: Highlight Date: 10/30/2007 2:13:27 PM
Emulex-015
Page: 122 A.2 last two sentences of first paragraph: To clarify that the tables do not specify which standard makes requirements, change to: "Table A.1 also lists whether SCSI transport protocols and logical units are required to support identifier attributes by either this standard or SPC-4. Table A.2 also lists whether SCSI transport protocols and logical units are required to support name attributes by either this standard or SPC-4."

Status
George Penokie Rejected 11/8/2007 6:37:38 PM -06'00'
Author: George Penokie Subject: Note Date: 11/8/2007 6:37:30 PM -06'00'
Made into footnotes. One for table A.1 and one for table A.2 and deleted the two senetences referenced in the comment.

A.3.2 ISO/IEC 13213:1994: See ISO/IEC 13213:1994, Information technology - Microprocessor systems - Control and Status Registers Architecture for microcomputer buses [ANSI/IEEE 1212, 1994 Edition]. See http://www.iso.org/.

A.3.3 NAA: Name Address Authority (see SPC-4)

A.3.4 SAS-2 SSP: SAS-2 (see A.3.4) Serial SCSI Protocol.

A.3.5 UTF-8: See ISO/IEC 10646-1:2000, Information technology - Universal Multiple-Octet Coded Character Set (UCS) - Part 1: Architecture and Basic Multilingual Plane. See http://www.iso.org/.

Page: 127

Author: Emulex Subject: Highlight Date: 10/30/2007 2:14:48 PM
Emulex-016
Page: 127 A.3.4: This seems to be a self reference. Should this be "SAS-2 Serial SCSI Protocol (see SAS-2)."

Status
George Penokie Accepted 11/8/2007 6:26:23 PM -06'00'

Page: 129

Author: relliott Subject: Highlight Date: 9/28/2007 6:44:34 PM
Information
s/b
Additional Response Information

Status
George Penokie Accepted 11/2/2007 10:47:09 AM

Table B.1 — SCSI Initiator Port attributes and SCSI Target Port attributes supported by SCSI transport protocols (part 2 of 2)

Attribute	ADT-2	FCP-4	iSCSI	SAS-2	~~SRP~~
Retry Delay Timer supported	no	yes	no	yes	no
Information supported	no	yes	no	yes	no
Bidirectional Commands supported	yes				
Task Management Functions supported [e]	ABORT TASK, ABORT TASK SET, CLEAR TASK SET, LOGICAL UNIT RESET, CLEAR ACA, QUERY TASK	All	ABORT TASK, ABORT TASK SET, CLEAR TASK SET, LOGICAL UNIT RESET, CLEAR ACA	All	ABORT TASK, ABORT TASK SET, CLEAR TASK SET, LOGICAL UNIT RESET, CLEAR ACA

[a] SPC-4 defines the maximum length of a CDB as being 260 bytes.
[b] Maximum CRN of zero indicates that CRN is not supported.
[c] SPC-4 defines the maximum length of sense data as being 252 bytes.
[d] 3E8h represents 1 000 bytes.
[e] The task management function name is the name of the procedure call, not an argument to a procedure call.

Annex C
(informative)

Terminology mapping

The introduction of a UML model into this standard resulted in changes in terminology between SAM-4 and SAM-3 (see table C.1).

Table C.1 — SAM-4 to SAM-3 terminology mapping

SAM-4 equivalent term	SAM-3 term
task identifier	task tag

Page: 130

Author: relliott Subject: Highlight Date: 10/27/2007 1:15:54 PM
Terminology mapping
s/b
Terminology mapping to previous versions of this standard

Status
George Penokie Accepted 11/1/2007 5:35:43 PM
Author: relliott Subject: Highlight Date: 10/27/2007 1:13:37 PM
Change "SAM-3" to "previous versions of this standard"

or add SAM-3 as a normative reference in 2.1.

Status
George Penokie Accepted 11/1/2007 5:37:13 PM
Author: relliott Subject: Highlight Date: 10/27/2007 1:14:14 PM
Change "SAM-4 to SAM-3 terminology mapping"
to
"Terminology mapping to previous versions of this standard"

or add SAM-3 as a normative reference in 2.1.

Status
George Penokie Accepted 11/1/2007 5:37:21 PM
Author: relliott Subject: Note Date: 10/27/2007 1:15:14 PM
in table C.1, center the left column and left-justify the right column (including the headers)

Status
George Penokie Completed 11/1/2007 5:40:30 PM
Author: relliott Subject: Highlight Date: 10/27/2007 1:14:47 PM
SAM-4 equivalent term
s/b
Term used in this standard

Status
George Penokie Accepted 11/1/2007 5:39:09 PM
Author: relliott Subject: Highlight Date: 10/27/2007 1:14:39 PM
Change "SAM-3 term"
to
"Term used in previous versions of this standard"

or add SAM-3 as a normative reference in 2.1.

Status
George Penokie Accepted 11/1/2007 5:38:35 PM

3.1.14 class diagram: Shows a set of classes and their relationships. Class diagrams are used to illustrate the static design view of a system. See 3.6.2.

3.1.15 client-server: A relationship established between a pair of distributed entities where one (the client) requests the other (the server) to perform some operation or unit of work on the client's behalf. See 4.3.

3.1.16 client: An entity that requests a service from a server. This standard defines one client, the application client.

3.1.17 ~~code value:~~ ~~A defined numeric value, possibly a member of a series of defined numeric values, representing an identified and described instance or condition. Code values are defined to be used in a specific field (see 3.1.44), in a procedure call input argument (see 3.6.4), in a procedure call output argument, or in a procedure call result.~~

3.1.18 command: A request describing a unit of work to be performed by a device server.

3.1.19 command descriptor block (CDB): A structure used to communicate a command from an application client to a device server. A CDB may have a fixed length of 6 bytes, 10 bytes, 12 bytes, or 16 bytes, or a variable length of between 12 and 260 bytes. See 5.2 and SPC-4.

3.1.20 command identifier: The portion of an I_T_L_Q nexus (i.e., the Q) that is the numerical identifier of the command (see 3.1.18) within an I_T_L nexus. See 4.7.2.

3.1.21 command priority: The relative scheduling importance of a command having the SIMPLE task attribute among the set of commands having the SIMPLE task attribute already in the task set. See 8.7.

3.1.22 command standard: A SCSI standard that defines the model, commands, and parameter data for a device type (e.g., SPC-4, SBC-3). See clause 1.

3.1.23 completed command: A command that has ~~ended by returning~~ completed with a ~~status and~~ service response of TASK COMPLETE.

3.1.24 ~~completed task~~confirmation: 3.1.25 ~~confirmation:~~ A response returned to an application client or device server that signals the completion of a service request.

3.1.26 confirmed SCSI transport protocol service: A service available at the SCSI transport protocol service interface that includes a confirmation of completion. See 4.9.

3.1.27 constraint: When referring to classes (see 3.1.13) and objects (see 3.1.78), a mechanism for specifying semantics or conditions that are maintained as true between entities (e.g., a required condition between associations).

3.1.28 current ~~task~~command: A ~~task~~ command that has a data transfer SCSI transport protocol service request in progress (see 5.4.3) or is in the process of sending command status. Each SCSI transport protocol standard may define the SCSI transport protocol specific conditions under which a ~~task~~ command is considered a current ~~task~~command.

3.1.29 deferred error: An error generated by a background operation (see SPC-4).

3.1.30 dependency: A relationship between two elements in which a change to one element (e.g., the ~~supplier~~server) may affect or supply information needed by the other element (e.g., the client).

3.1.31 dependent logical unit: A logical unit that is addressed via some other logical unit(s) in a hierarchical logical unit structure (see 3.1.47)~~, also a logical unit~~ and that is at a higher numbered level in the hierarchy than the referenced logical unit (see 4.5.19.4).

Summary of Comments on SCSI Architecture Model - 4

Page: 20

Author: relliott Subject: Highlight Date: 4/25/2008 12:46:56 PM
Change all cases for << TASK COMPLETE >> to << COMMAND COMPLETE >>.

Status
George Penokie Accepted 4/25/2008 12:47:10 PM

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 consisting 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 consisting of only the Western-Arabic numerals 0 through 9 not immediately followed by a lower-case b or lower-case h (e.g., 25).

~~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), shows some examples of decimal numbers represented using the ISO and American conventions.~~

This standard uses the following conventions for representing decimal numbers:

a) the decimal separator (i.e., separating the integer and fractional portions of the number) is a period;
b) the thousands separator (i.e., separating groups of three digits in a portion of the number) is a space;
c) the thousands separator is used in both the integer portion and the fraction portion of a number; and
d) the decimal representation for a year is 1999 not 1 999.

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

Table 1 — Numbering conventions

French	English	This standard
0,6	0.6	0.6
3,141 592 65	3.14159265	3.141 592 65
1 000	1,000	1 000
1 323 462,95	1,323,462.95	1 323 462.95

3.6 Notation conventions

3.6.1 Notation conventions overview

This standard uses class diagrams and object diagrams with notation that is based on the Unified Modeling Language (UML).

See 3.6.2 for the conventions used for class diagrams.

See 3.6.3 for the conventions used for object diagrams.

~~Within class~~ Class diagrams and object diagrams ~~there~~ may ~~be constraints~~ include constraints, which specify ~~requirements~~ requirements, and ~~notes~~ notes, which are informative~~.~~ .

A constraint is specified as text encapsulated with a { } notation within a box. See figure 5 for an example of a constraint.

A note is specified as text within a box (i.e., no { }). See figure 6 for an example of a note.

Page: 29

Author: relliott Subject: Note Date: 3/12/2008 9:38:56 AM
As proposed in 08-113r1 for SPC-4, add a figure right here showing constraints and notes.

Status
George Penokie Accepted 3/26/2008 8:42:24 AM

Solid lines with arrowheads (see figure 4) ~~are~~ is the notation used to describe the association relationship between classes in class diagrams. Multiplicity notation occurs at each end of the solid line.

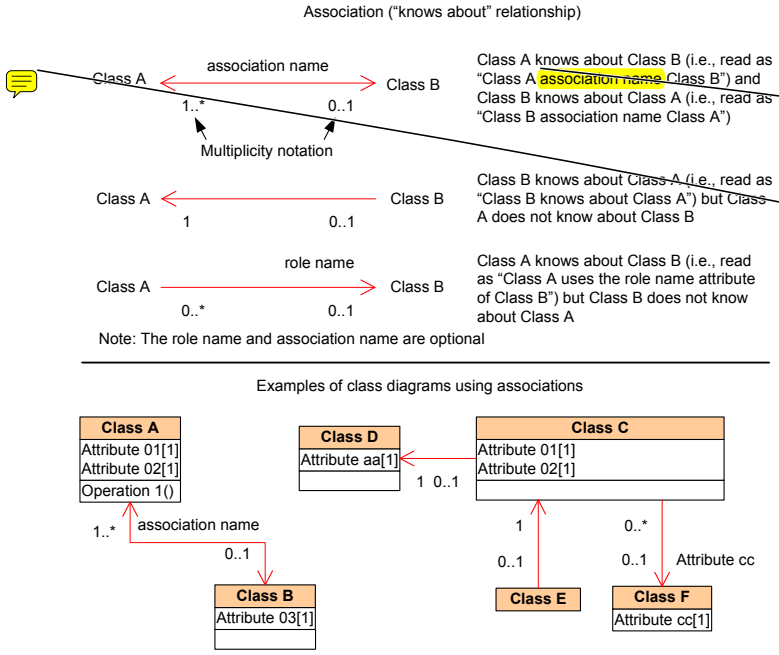


Figure 4 — Notation for association relationships for class diagrams

Page: 32

Author: relliott Subject: Note Date: 3/12/2008 9:36:31 AM
"is" is very awkward given that all the nouns on the left are plural.
Restructure as
"The notation used to describe the association relationship between classes in class diagrams is solid lines with arrowheads (see figure 4).

Apply to all the figure introductions in this section.

Status
George Penokie Rejected 3/26/2008 8:51:22 AM
Author: George Penokie Subject: Sticky Note Date: 3/26/2008 8:51:12 AM
Changed to << Table 5 shows the notation used to denote association (i.e., “knows about”) relationships between classes. >>

Author: relliott Subject: Highlight Date: 3/12/2008 9:25:35 AM
association name

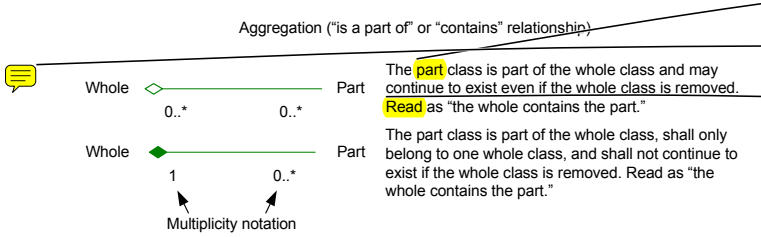
in <> or something

Status
George Penokie Accepted 3/26/2008 8:52:02 AM
Author: George Penokie Subject: Sticky Note Date: 3/26/2008 8:51:56 AM
Changed to << association_name >>

Author: relliott Subject: Note Date: 3/12/2008 9:24:42 AM
Class should be in shaded box

Status
George Penokie Accepted 3/26/2008 8:52:18 AM

Solid lines with diamonds (see figure 5) ~~are~~ is the notation used to describe the aggregation relationship between classes in class diagrams. Multiplicity notation occurs at each end of the solid line.



Examples of class diagrams using aggregation

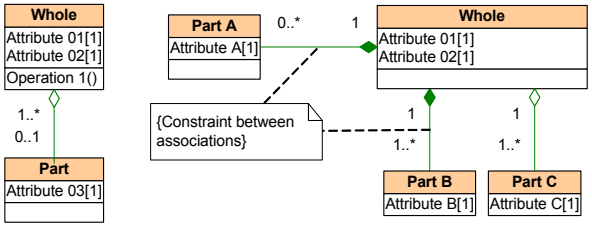


Figure 5 — Notation for aggregation relationships for class diagrams

Page: 33

Author: relliott Subject: Highlight Date: 3/12/2008 9:28:46 AM
part and whole should be Mixed case

Status
George Penokie Accepted 3/26/2008 8:52:26 AM
Author: relliott Subject: Note Date: 3/12/2008 9:28:19 AM
Whole and Part should be shaded boxes

Status
George Penokie Accepted 3/26/2008 8:52:34 AM
Author: relliott Subject: Highlight Date: 3/12/2008 9:29:38 AM
"Read as" clauses should be in (i.e.,)s like in earlier figure

Status
George Penokie Accepted 3/26/2008 8:52:42 AM

Solid lines with triangles (see figure 6) ~~are~~ is the notation used to describe the generalization relationship between classes in class diagrams.

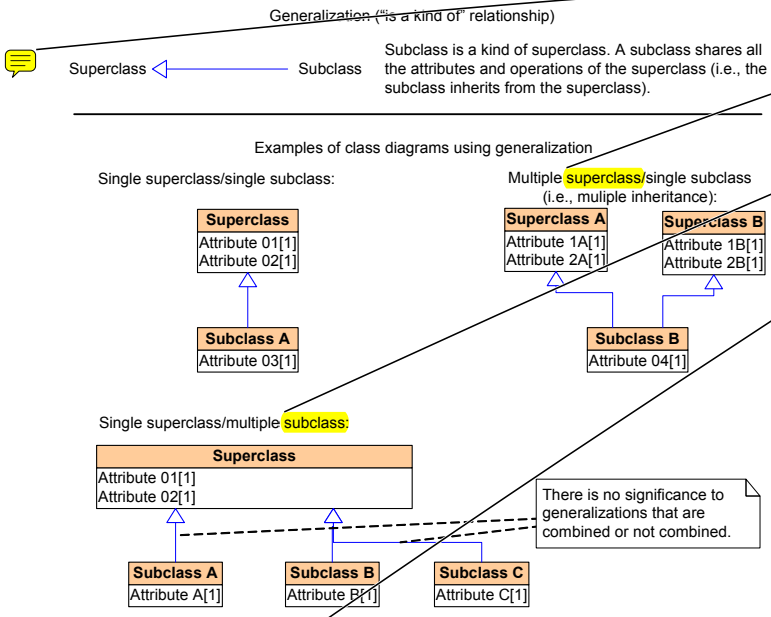


Figure 6 — Notation for generalization relationships for class diagrams

Dashed lines with arrowheads (see figure 7) ~~are~~ is the notation used to describe the dependency relationship between classes in class diagrams.

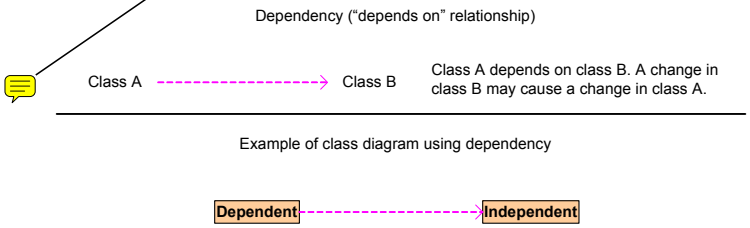


Figure 7 — Notation for dependency relationships for class diagrams

Page: 34

Author: relliott Subject: Note Date: 3/12/2008 9:30:59 AM
Superclass and Subclass should be in shaded boxes

Status
George Penokie Accepted 3/26/2008 8:52:51 AM

Author: relliott Subject: Highlight Date: 3/12/2008 9:33:08 AM
superclass
s/b
superclasses

Status
George Penokie Accepted 3/26/2008 8:54:32 AM

Author: relliott Subject: Highlight Date: 3/12/2008 9:33:29 AM
subclass
s/b
subclasses

Status
George Penokie Accepted 3/26/2008 8:54:41 AM

Author: relliott Subject: Note Date: 3/12/2008 9:34:14 AM
Class A and B should be in shaded boxes

Status
George Penokie Accepted 3/26/2008 8:54:49 AM

Output arguments:

Data-In Buffer: A buffer (see 5.4.3) to contain command specific information returned by the logical unit by the time of command completion. The **Execute Command** procedure call shall not return complete with a status of GOOD or CONDITION MET unless the buffer contents are valid. The application client shall treat the buffer contents as invalid unless the command completes with a status of GOOD or CONDITION MET. While some valid data may be present for other values of status, the application client should rely on additional information from the logical unit (e.g., sense data) to determine the state of the buffer contents. If the command ends with a service response of SERVICE DELIVERY OR TARGET FAILURE, the application client shall consider the buffer to be undefined.

Sense Data: A buffer containing sense data returned in the same I_T_L_Q nexus transaction (see 3.1.50) as a CHECK CONDITION status (see 5.14). The buffer length is indicated by the Sense Data Length argument. If the command ends with a service response of SERVICE DELIVERY OR TARGET FAILURE, the application client shall consider the sense data to be undefined.

Sense Data Length: The length in bytes of the Sense DataData (see 5.14).

Status: A one-byte field containing command completion status (see 5.3). If the command ends completes with a service response of SERVICE DELIVERY OR TARGET FAILURE, the application client shall consider status to be undefined.

Retry Delay TimerStatus Qualifier: Additional information about the indicated status code (see 5.3.3).

Service Response assumes one of the following value

One of the following SCSI transport protocol specific service responses shall be returned:

TASK COMPLETE: A logical unit response indicating that the command has completed. The Status argument shall have one of the values specified in 5.3.

SERVICE DELIVERY OR TARGET FAILURE: The command has been ended due to a service delivery failure (see 3.1.116) or SCSI target device malfunction. All output arguments are invalid.

The SCSI transport protocol events corresponding to a response of TASK COMPLETE or SERVICE DELIVERY OR TARGET FAILURE shall be specified in each SCSI transport protocol standard.

5.2 Command descriptor block (CDB)

The CDB defines the operation to be performed by the device server. See SPC-4 for the CDB formats.

For all commands, if the logical unit detects an invalid parameter field in the CDB, then the logical unit shall not process the command.

All CDBs shall have an OPERATION CODE field as the first byte.

Some operation codes provide for modification of their operation based on a service action. In such cases, the combination of operation code value and service action code value may be modeled as a single, unique command determinatecommand. The location of the SERVICE ACTION field in the CDB varies depending on the operation code value.

All CDBs shall contain a CONTROL byte (see table 24). The location of the CONTROL byte within a CDB depends on the CDB format (see SPC-4).

Page: 90

Author: relliott Subject: Highlight Date: 3/14/2008 4:30:08 AM
undo

I think the procedure calls still "return" their outputs; avoid using "complete" there

Status
George Penokie Accepted 3/26/2008 9:34:30 AM
Author: relliott Subject: Highlight Date: 3/14/2008 4:29:18 AM
the command completes with
s/b
Execute Command procedure call returns

Status
George Penokie Accepted 3/26/2008 9:36:48 AM
Author: relliott Subject: Note Date: 5/8/2008 11:19:08 AM
Problem: completed command is defined as one that returns TASK COMPLETE service response. A service response of SERVICE DELIVERY OR TARGET FAILURE doesn't fit...

Status
George Penokie Rejected 5/8/2008 11:20:05 AM
Author: George Penokie Subject: Sticky Note Date: 3/26/2008 9:40:02 AM
Changed << completes >> to << terminates >>

Author: George Penokie Subject: Sticky Note Date: 5/8/2008 11:19:45 AM
No change made as it was determined that the wording is correct.

Author: relliott Subject: Highlight Date: 3/14/2008 4:31:26 AM
status
s/b
Status

or
command completion status

Status
George Penokie Accepted 3/26/2008 9:40:34 AM
Author: George Penokie Subject: Sticky Note Date: 3/26/2008 9:43:43 AM
Changed to << command completion status >>

Author: relliott Subject: Highlight Date: 3/14/2008 4:31:50 AM
status code
s/b
Status

or
command completion status

inconsistent terminology

Status
George Penokie Accepted 5/8/2008 12:02:54 PM
Author: George Penokie Subject: Sticky Note Date: 3/26/2008 9:43:48 AM
Changed to << command completion status >>

Author: relliott Subject: Highlight Date: 3/14/2008 4:32:20 AM
note use of "ended" here

Status
George Penokie Accepted 5/8/2008 11:34:21 AM
Author: George Penokie Subject: Sticky Note Date: 3/26/2008 10:09:13 AM
Changed << ended >> to << terminated >> and << end >> to << terminate >> in all cases relating to task management functions. Where <<ended>> refers to a command ending changed to << completed >>

Author: George Penokie Subject: Sticky Note Date: 5/8/2008 11:33:41 AM
Changed << ended >> to << terminated >> and << end >> to << terminate >> in all cases relating to task management functions. Where <<ended>> refers to a command ending without an error changed to << completed >>. Where <<ends>> refers to a command ending in error changed to << terminated >>.

Page: 93

Author: relliott Subject: Note Date: 3/13/2008 2:06:23 PM
there seems to be a _ after SCOPE field above

Status
George Penokie Rejected 3/26/2008 10:35:26 AM
Author: George Penokie Subject: Sticky Note Date: 3/26/2008 10:35:19 AM
I see none in the source file.

Table 26 — Status qualifier format

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

The SCOPE field (see table 27) indicates the logical unit(s) to which the qualifier applies.

Table 27 — SCOPE field

Code	Affected logical unit(s)	Affected nexus(es)
00b	The logical unit addressed by the command associated with the status.	All I_T nexus(es).
01b	All logical units accessible by the target port through which the command associated with the status was routed.	I_T nexus through with the status was returned.
10b	All logical unit(s) contained within the SCSI device that contains the logical unit addressed by the command associated with the status.	All I_T nexus(es).
11b	Reserved	

The ~~retry delay timer~~ QUALIFIER field codes are ~~specified~~ defined in table 28 ~~and provide additional information about the reason for the status code~~.

Page: 94

Author: relliott Subject: Highlight Date: 4/25/2008 1:36:17 PM
Add in a new row that states:
All : 0000h : No additional information (i.e., the same as returning no status qualifier)
Delete the two rows that have a 0000h QUALIFIER field.

In SAS-2 or FCP-4, the "status qualifier" structure is always returned. If you return GOOD status, for example, this means SCOPE=00b and QUALIFIER=0000h are being returned. 0000h being lumped into the "Reserved" description means the meaning of that combination might change in the future. Really, it cannot; it will always have to mean "No additional information" for backwards compatibility.

Status
George Penokie Accepted 5/8/2008 11:50:44 AM

Table 28 — QUALIFIER field

Status code	QUALIFIER field	Description
BUSY	0000h	No additional information (i.e., the same as BUSY with no status qualifier).
	0001h - 3FEFh	The number of 100 milliseconds increments the application client should wait before sending another command to the logical unit(s) indicated by the SCOPE field using the nexus(es) indicated by the SCOPE field.
	3FF0h - 3FFDh	Reserved
	3FFEh	The application client should stop sending commands to the logical unit(s) indicated by the SCOPE field using the nexus(es) indicated by the SCOPE field.
	3FFFh	The logical unit(s) indicated by the SCOPE field are not able to accept the command because they are servicing too many other I_T nexuses.
TASK SET FULL [a]	0000h	No additional information (i.e., the same as TASK SET FULL with no status qualifier).
	0001h - 3FEFh	The application client should wait before sending another command to the logical unit on any I_T nexus until: a) at least the number of 100 milliseconds increments indicated in the QUALIFIER field have elapsed; or b) a command addressed to the logical unit on any I_T nexus completes or terminates.
	3FF0h - 3FFFh	Reserved
GOOD	0000h - 3FFFh	Reserved
CHECK CONDITION	0000h - 3FFFh	Reserved
CONDITION MET	0000h - 3FFFh	Reserved
RESERVATION CONFLICT	0000h - 3FFFh	Reserved
ACA ACTIVE	0000h - 3FFFh	Reserved
TASK ABORTED	0000h - 3FFFh	Reserved
All others	0000h - 3FFFh	Reserved

[a] The SCOPE field shall be set to zero.

Page: 95

5.3.4 Status precedence

If a device server detects that more than one of the following conditions applies to a completed ~~task~~command, it shall select the condition to report based on the following precedence:

1) ~~An~~ an ACA ACTIVE status;
2) ~~A~~ a CHECK CONDITION status for any of the following unit attention conditions (i.e., with a sense key set to UNIT ATTENTION and one of the following additional sense codes):
   A) POWER ON, RESET, OR BUS DEVICE RESET OCCURRED;
   B) POWER ON OCCURRED;
   C) SCSI BUS RESET OCCURRED;
   D) MICROCODE HAS BEEN CHANGED;
   E) BUS DEVICE RESET FUNCTION OCCURRED;
   F) DEVICE INTERNAL RESET; ~~or~~
   G) COMMANDS CLEARED BY POWER LOSS NOTIFICATION; or
   H) I_T NEXUS LOSS OCCURRED;
3) ~~A~~ a RESERVATION CONFLICT status;

   and

4) ~~A~~ a status of:
   A) CHECK ~~CONDITION~~CONDITION status, for any reason not listed in 2);
   B) ~~GOOD~~;
   C) GOOD status;
   D) CONDITION ~~MET~~MET status; or
   E) TASK ~~ABORTED~~ABORTED status.

NOTE 7 - The names of the unit attention conditions listed in this subclause (e.g., SCSI BUS RESET OCCURRED) are based on usage in previous versions of this standard. The use of these unit attention condition names is not to be interpreted as a description of how the unit attention conditions are represented by any given SCSI transport protocol.

A device server may report the following status codes with any level of precedence:

a) BUSY status;
b) TASK SET FULL status; or
c) CHECK CONDITION status with a sense key set to ILLEGAL REQUEST.

Any unit attention condition that was established for all logical units should be reported with a higher precedence than a CHECK CONDITION status with a sense key set to ILLEGAL REQUEST and an additional sense code set to LOGICAL UNIT NOT SUPPORTED.

5.4 SCSI transport protocol services in support of Execute Command

5.4.1 Overview

The SCSI transport protocol services that support the **Execute Command** procedure call are described in 5.4. ~~Two~~ The following two groups of SCSI transport protocol services are described. ~~The SCSI transport protocol services that support the delivery of the command and status are described in 5.4.2. The SCSI transport protocol services that support the data transfers associated with processing a command are described in 5.4.3.~~:

a) the SCSI transport protocol services that support the delivery of the command and status (see 5.4.2); and
b) the SCSI transport protocol services that support the data transfers associated with processing a command (see 5.4.3).

Author: relliott Subject: Cross-Out Date: 3/13/2008 2:10:05 PM
status

Status
George Penokie Accepted 3/26/2008 10:36:32 AM
Author: relliott Subject: Highlight Date: 3/13/2008 2:12:34 PM
any reason not listed in...

also need to exclude "with a sense key set to ILLEGAL REQUEST" from item c) 8 lines below.

Status
George Penokie Accepted 5/8/2008 11:36:26 AM
Author: George Penokie Subject: Sticky Note Date: 3/26/2008 10:40:54 AM
Changed to << CHECK CONDITION, other than with a sense key set to ILLEGAL REQUEST or for any reason not listed in 2)>>

Author: relliott Subject: Cross-Out Date: 3/13/2008 2:10:07 PM
status

Status
George Penokie Accepted 3/26/2008 10:41:06 AM
Author: relliott Subject: Cross-Out Date: 3/13/2008 2:10:09 PM
status

Status
George Penokie Accepted 3/26/2008 10:41:12 AM
Author: relliott Subject: Cross-Out Date: 3/13/2008 2:10:11 PM
status

Status
George Penokie Accepted 3/26/2008 10:41:22 AM
Author: relliott Subject: Highlight Date: 3/13/2008 2:17:07 PM
after "for all logical units"

add the example that prompted this change:
"(e.g., REPORTED LUNS DATA HAS CHANGED)"

Status
George Penokie Accepted 3/26/2008 10:42:17 AM
Author: relliott Subject: Cross-Out Date: 3/13/2008 2:17:49 PM
two

Status
George Penokie Accepted 3/26/2008 10:43:29 AM

longer known to the device server~~. An application client may determine that a task is no longer known to the device server by detecting:~~.

a) ~~Completion of an ABORT TASK task management function specifying that task;~~
b) ~~Completion of an ABORT TASK SET or an I_T NEXUS RESET task management function on the I_T nexus used to deliver that task; or~~
c) ~~Completion of a CLEAR TASK SET or LOGICAL UNIT RESET task management function.~~

NOTE 9 - The names of the unit attention conditions listed in the subclause (e.g., SCSI BUS RESET OCCURRED) are based on usage in previous versions of this standard. The use of these unit attention condition names is not to be interpreted as a description of how the unit attention conditions are represented by any given SCSI transport protocol.

~~To the application client, the command is pending from the time it calls the **Send SCSI Command** SCSI transport protocol service until one of the responses described in this subclause.~~

The device server shall create a command upon receiving a **SCSI Command Received** indication.

The command shall exist until:

a) the device server sends a SCSI transport protocol service response for the command of TASK COMPLETE; or
b) the command is aborted as described in 5.6.

When a SCSI transport protocol does not require state synchronization (see 4.3.2), there may be a time skew between the completion of a device server request-response transaction as seen by the application client and device server. As a result, the lifetime of a ~~task or~~ command as it appears to the application client is different from the lifetime observed by the device server.

~~Some commands (e.g., commands with immediate bits like SEND DIAGNOSTIC, or write commands when a write cache is enabled) start background operations that operate after the task containing the command is no longer in the task set. Background operations may be aborted by power on, hard resets, or logical unit resets. Background operations shall not be aborted by I_T nexus loss.~~

Some commands initiate background operations that are processed after the ~~task containing the~~ command is no longer in the task set (i.e., status has been returned for the command) (e.g., a SEND DIAGNOSTIC command when used to initiate a background self-test (see SPC-4) or a write command when write cache is enabled (see SBC-3)). Background operations may be aborted by power on, hard reset, or logical unit reset. Background operations shall not be aborted by I_T nexus loss or power loss expected.

Background operations may generate deferred errors that are reported in the sense data for a subsequent ~~completed~~ command (see SPC-4). Information that a deferred error occurred may be cleared before it is reported (e.g., by power on, hard reset, or logical unit reset). Deferred errors should not be cleared by I_T nexus ~~loss~~loss or power loss expected.

Unless a command completes with a GOOD status or CONDITION MET ~~status~~ status, the degree to which the required command processing has been completed is vendor specific.

5.6 Aborting ~~tasks~~commands

A ~~task~~ command is aborted when a SCSI device condition (see 6.3), command, or task management function causes termination of the ~~task~~ command prior to its completion by the device server.

See table 29 for a list of the SCSI device conditions that cause ~~tasks~~ commands to be aborted in a SCSI initiator device.

Page: 104

Author: relliott Subject: Cross-Out Date: 3/20/2008 4:47:38 PM
task containing the

Status
George Penokie Accepted 3/26/2008 10:44:59 AM

See table 32 for a list of the command related conditions that cause ~~tasks~~ commands to be aborted.

Table 32 — Command related conditions that abort ~~tasks~~commands

Command related conditions	Scope	Unit attention condition (see 5.15) additional sense code, if any [a]	TASK ABORTED status [b]	Reference
CHECK CONDITION status if: a) the QERR field is set to 01b; and b) the TST field is set to 000b (i.e., shared) in the Control mode page (see SPC-4)	All ~~tasks~~ commands in the task set	COMMANDS CLEARED BY ANOTHER INITIATOR	Yes	5.9.3 and 5.10.2
CHECK CONDITION status if: a) the QERR field is set to 01b; and b) the TST field is set to 001b (i.e., per I_T nexus) in the Control mode page (see SPC-4)	All ~~tasks~~ commands in the task set [c]	None	No	5.9.3 and 5.10.2
Completion of a command with a CHECK CONDITION status if the QERR field is set to 11b in the Control mode page (see SPC-4)	All ~~tasks~~ commands in the task set with the same I_T nexus as the command that was terminated	None	No	5.9.3 and 5.10.2
Processing of a PERSISTENT RESERVE OUT command with a PREEMPT AND ABORT service action with a reservation key that is associated with the I_T nexus on which the ~~task~~ command was received (see SPC-4)	All ~~tasks~~ commands from all I_T nexuses with the specified reservation key	COMMANDS CLEARED BY ANOTHER INITIATOR	Yes	SPC-4
The ~~return~~ completion of an Execute Command service response of SERVICE DELIVERY OR TARGET FAILURE	The indicated ~~task~~command	None	No	5.1
Termination of an overlapped command	All ~~tasks~~ commands with the same I_T nexus as the command that was terminated	None	No	5.11

[a] If the TAS bit is set to zero in the Control mode page (see SPC-4), the device server creates this unit attention condition for each I_T nexus that had ~~task~~command(s) aborted other than the I_T nexus that delivered the task management function. If the TAS bit is set to one in the Control mode page (see SPC-4), the device server does not create this unit attention condition.

[b] "Yes" indicates that each ~~task~~ command that is aborted on an I_T nexus other than the one that delivered the command is ~~terminated~~ completed with TASK ABORTED status, if the TAS bit is set to one in the Control mode page (see SPC-4). "No" indicates that no status is returned for aborted ~~tasks~~commands.

[c] As a result of the TST field being set to 001b, there is one task set per I_T nexus, so no other I_T nexuses are affected.

Page: 108

Author: relliott Subject: Highlight Date: 3/13/2008 2:30:58 PM
undo this

the service response is a value that is returned

Status
George Penokie Accepted 3/26/2008 10:45:59 AM

If one or more ~~tasks~~ commands are cleared or aborted, the affected ~~tasks~~ commands are also cleared from the SCSI initiator ports in a manner that is outside the scope of this standard.

When a device server receives a command or task management function on an I_T nexus that causes ~~tasks~~ commands on the same I_T nexus to be aborted, the device server shall not ~~return~~ complete those commands with any notification that ~~those tasks~~ they have been aborted other than:

a) ~~the~~ a completion response for the command or task management function that caused the ~~task~~command(s) to be aborted; and
b) notification(s) associated with related effects of the command or task management function (e.g., a reset unit attention condition).

When a device server receives a command or task management function on an I_T nexus that causes ~~tasks~~ commands on other I_T nexuses to be aborted, the device server shall ~~return~~ complete with notifications for those ~~tasks~~ commands based on the setting of the TAS bit in the Control mode page (see SPC-4):

a) ~~If~~ if the TAS bit is set to ~~zero,~~ the device server:
   A) ~~shall not return status for the tasks that were aborted; and~~
   B) shall not complete commands that were aborted with any status; and
   C) shall establish a unit attention condition for the SCSI initiator port associated with each I_T nexus containing ~~tasks~~ commands that were aborted with an additional sense code set as defined in table 31 and table 32;

   or

b) ~~If~~ if the TAS bit is set to one, the device server:
   A) ~~shall return TASK ABORTED status for each aborted task; and~~
   B) shall complete each aborted command with a TASK ABORTED status; and
   C) shall not establish a unit attention condition for this reason.

When a logical unit ~~is aborting~~ completes one or more ~~tasks~~ commands received on an I_T nexus ~~using~~ with a status of TASK ABORTED, the ~~TASK ABORTED status it~~ logical unit should complete all of ~~those tasks~~ the affected commands before entering ~~additional tasks~~ any other commands received on that I_T nexus into the task set~~.~~.

5.7 Command processing example

A command is used to show the events associated with the processing of a single device service request (see figure 41). This example does not include error or exception conditions.

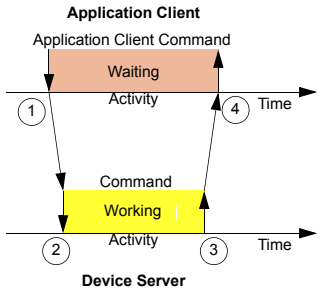


Figure 41 — Command processing events

Page: 109

Author: relliott Subject: Highlight Date: 3/20/2008 4:49:45 PM
undo, "return" was better

Status
George Penokie Accepted 3/26/2008 10:47:53 AM
Author: relliott Subject: Highlight Date: 3/20/2008 4:49:51 PM
undo, this doesn't make sense now

Status
George Penokie Accepted 3/26/2008 10:51:36 AM
Author: relliott Subject: Highlight Date: 3/13/2008 2:35:25 PM
undo this change; the old wording reads better. This sounds like commands are "aborted with status"

Status
George Penokie Accepted 3/26/2008 10:53:10 AM

The numbers in figure 41 identify the events described as follows:

1) ~~The~~ the application client task performs an **Execute Command** procedure call by invoking the **Send SCSI Command** SCSI transport protocol service to send the CDB and other input parameters to the logical unit.
2) ~~The~~ the device server is notified through a **SCSI Command Received** indication containing the CDB and command parameters. ~~.~~ A ~~task~~ command is created and entered into the task set. ~~.~~ The device server may invoke the appropriate data delivery service one or more times to complete command processing.
3) ~~The task~~ the command ~~ends upon completion of the command~~. On command completion, the **Send Command Complete** SCSI transport protocol service is invoked to ~~return~~ complete with a status of GOOD and a service response of TASK COMPLETE.
4) ~~A~~ a confirmation of **Command Complete Received** is passed to the application client ~~task~~ by the SCSI initiator port.

5.8 ~~Command processing considerations and exception conditions~~

5.9 Commands that complete with CHECK CONDITION status

5.9.1 Overview

~~When a command completes with a CHECK CONDITION status, the application client may request that the device server alter command processing by establishing an ACA condition, using the NACA bit in the CONTROL byte of the CDB as follows:~~

An application client uses the NACA bit in the CONTROL byte of the CDB (see 5.2) to specify whether or not the device server establishes an ACA condition when a command completes with CHECK CONDITION status. The meaning of the value in the NACA bit is as follows:

a) If the NACA bit is set to zero, an ACA condition shall not be established; or
b) If the NACA bit is set to one, an ACA condition shall be established (see 5.10).

The requirements that apply when the ACA condition is not in effect are described in 5.9.2.

When a command completes with a CHECK CONDITION status and an ACA condition is not established, ~~tasks~~ commands other than the ~~task for the~~ command ~~returning~~ completing with a the CHECK CONDITION status may be aborted as described in 5.9.3.

5.9.2 Handling ~~tasks~~ commands when ACA is not in effect

Table 33 describes the handling of ~~tasks~~ commands when an ACA condition is not in effect for the task set. Which I_T nexuses are associated with ~~the~~ a task set is ~~influenced~~ specified by the TST field in the Control mode page (see SPC-4).

Page: 110

Author: George Penokie Subject: Cross-Out Date: 5/8/2008 11:38:50 AM
Delete << the command ends upon completion of the command >> as it makes no sense now that tasks are gone.

Status
George Penokie Accepted 5/8/2008 11:38:39 AM

Author: relliott Subject: Highlight Date: 3/20/2008 4:50:49 PM
undo

Status
George Penokie Accepted 3/26/2008 10:53:50 AM

Author: relliott Subject: Highlight Date: 3/20/2008 4:51:25 PM
a command completes
s/b
it completes a command

Author: George Penokie Subject: Sticky Note Date: 3/26/2008 11:02:20 AM
In addition changed << completes >> to << teminates >>

Author: relliott Subject: Highlight Date: 3/20/2008 4:52:25 PM
Which I_T nexuses are
s/b
The I_T nexuses that are

Status
George Penokie Accepted 3/26/2008 11:04:25 AM

Page: 111

Author: relliott Subject: Note Date: 3/20/2008 4:53:49 PM
Make everything in the table header lowercase except first letter in each cell. "attribute" is the only one that is correct

Status
George Penokie Accepted 3/26/2008 11:05:05 AM
Author: George Penokie Subject: Sticky Note Date: 3/26/2008 11:05:01 AM
Changed globally

Author: relliott Subject: Highlight Date: 3/20/2008 4:52:54 PM
except
s/b
except the

Status
George Penokie Accepted 3/26/2008 11:05:43 AM
Author: relliott Subject: Note Date: 3/20/2008 4:54:04 PM
Make column wider to avoid wrap

Status
George Penokie Accepted 3/26/2008 11:06:03 AM

Table 33 — ~~Task~~ Command handling when ACA is not in effect

New ~~Task~~ Command Properties		Device Server Action	ACA Established if New ~~Task~~ Command Terminates with a CHECK CONDITION status
Task attribute [a]	NACA Value [b]		
Any task attribute except ~~ACA~~ACA task attribute	0	Process the ~~task~~command. [c]	No
	1		Yes
~~ACA~~ ACA task attribute	0	Process an invalid task attribute condition as described in 5.13.	No
	1		Yes

[a] Task attributes are described in 8.6.
[b] The NACA bit is in the CONTROL byte in the CDB (see 5.2).
[c] All the conditions that affect the processing of commands (e.g., reservations) apply.

5.9.3 Aborting ~~other tasks when~~ commands completed with a CHECK CONDITION status ~~is returned~~ without establishing an ACA

When a command completes with a CHECK CONDITION status ~~is returned for a command~~ where the NACA bit is set to zero in the command's CDB CONTROL byte (i.e., when an ACA condition is not established), ~~tasks~~ commands in the dormant command state or enabled ~~task~~ command state (see 8.5) may be aborted based on the contents of the TST field and QERR field in the Control mode page (see SPC-4) as shown in table 34. The TST field specifies the type of task set in the logical unit. The QERR field specifies how the device server handles commands in the blocked command state and dormant ~~tasks~~ command state when another ~~task receives~~ command completes with a CHECK CONDITION status.

Table 34 — Aborting ~~tasks~~ commands when an ACA is not established

QERR	TST	Action
00b	000b	~~Tasks~~ Commands other than the ~~task returning~~ command completed with a CHECK CONDITION status shall not be aborted.
	001b	
01b	000b	All enabled and dormant ~~tasks~~ commands received on all I_T nexuses shall be aborted (see 5.6).
	001b	All enabled and dormant ~~tasks~~ commands received on the I_T nexus on which the CHECK CONDITION status was returned shall be aborted (see 5.6). All ~~tasks~~ commands received on other I_T nexuses shall not be aborted.
11b	000b	All enabled and dormant ~~tasks~~ commands received on the I_T nexus on which the CHECK CONDITION status was returned shall be aborted (see 5.6). All ~~tasks~~ commands received on other I_T nexuses shall not be aborted.
	001b	

5.10 Auto contingent allegiance (ACA)

5.10.1 ACA Overview

~~When a command completes with a CHECK CONDITION status, the~~ The application client may request that the device server alter command processing when a command completes with a CHECK CONDITION status by establishing an ACA ~~condition,~~ condition using the NACA bit in the CONTROL byte ~~of the CDB as follows:~~ (see 5.9.1).

a) ~~If the NACA bit is set to zero, an ACA condition shall not be established (see 5.9.1); or~~
b) ~~If the NACA bit is set to one, an ACA condition shall be established.~~

The steps taken by the device server to establish an ACA condition are described in 5.10.2. Upon establishment of the ACA condition, some ~~tasks~~ commands other than the ~~task returning~~ command completing with the CHECK CONDITION status may be aborted and continued processing of other ~~tasks~~ commands may be blocked as described in 5.10.2.

While the ACA condition is in effect and the TMF_ONLY bit is set to zero in the Control mode page (see SPC-4), new ~~tasks~~ commands received by the logical unit from the faulted I_T nexus are not allowed to enter the task set unless they have the ~~ACA~~ ACA task attribute (see 8.6.5). One of the results of the ~~ACA~~ ACA task attribute requirement is that commands in-flight when the CHECK CONDITION status occurs are ~~returned~~ completed unprocessed with an ACA ACTIVE status. Multiple commands may be sent one at a time using the ~~ACA~~ ACA task attribute to recover from the event that resulted in the ACA condition without clearing the ACA.

While the ACA condition is in effect and the TMF_ONLY bit is set to one, no new ~~tasks~~ commands received by the logical unit from the faulted I_T nexus are allowed to enter the task set.

While the ACA condition is in effect:

a) ~~New tasks~~ new commands received on the faulted I_T nexus shall be handled as described in 5.10.3, and
b) ~~New tasks~~ new commands received on I_T nexuses other than the faulted I_T nexus shall be handled as described in 5.10.4.

The methods for clearing an ACA condition are described in 5.10.5.

5.10.2 Establishing an ACA

When a device server ~~terminates~~ completes a command with a CHECK CONDITION status and the NACA bit was set to one in the CONTROL byte of the faulting command, the device server shall create an ACA condition.

When an ACA condition is established, ~~tasks~~ commands in the dormant command state or enabled ~~task~~ command state (see 8.5) shall either be aborted or blocked based on the contents of the TST field and QERR field in the Control mode page (see SPC-4) as shown in table 35. The TST field specifies the type of task set in the logical unit. The QERR field specifies how the device server handles commands in the blocked command state and dormant ~~tasks~~ command state when another ~~task receives~~ command completes with a CHECK CONDITION status.

Page: 112

Author: relliott Subject: Highlight Date: 3/20/2008 4:55:34 PM
ACA
s/b
ACA condition

Status
George Penokie Accepted 3/26/2008 11:06:47 AM

Author: relliott Subject: Highlight Date: 3/20/2008 4:57:57 PM
the preceding paragraphs and this one's item a) seem to overlap for "new commands received on the faulted I_T nexus". Can they be merged?

Status
George Penokie Rejected 3/26/2008 11:11:39 AM

Author: George Penokie Subject: Sticky Note Date: 3/26/2008 11:11:33 AM
I do not thnk merging them to save a few words would help any in the understanding.

Author: relliott Subject: Highlight Date: 3/20/2008 4:58:46 PM
and
s/b
and the

Status
George Penokie Accepted 3/26/2008 11:12:29 AM

Page: 114

Author: relliott Subject: Note Date: 3/20/2008 4:59:45 PM
Make everything in the table header lowercase except the first letter of each cell

Status
George Penokie Accepted 3/26/2008 11:12:38 AM

Author: relliott Subject: Highlight Date: 3/20/2008 4:59:17 PM
except
s/b
except the

Status
George Penokie Accepted 3/26/2008 11:13:06 AM

Table 36 — Handling for new ~~tasks~~ commands received on a faulted I_T nexus during ACA

New ~~Task~~ Command Properties		ACA ~~Task~~ Command Present in the Task Set	TMF_ONLY value [c]	Device ~~Server~~ Action	ACA Established If New ~~Task Terminates~~ Command Completes with a CHECK CONDITION status
Task attribute [a]	NACA Value [b]				
~~ACA~~ ACA task attribute	0	No	0	Process the ~~task~~command. [e]	No [d]
	1	No	0		Yes [d]
	n/a	n/a	1	~~Terminate~~ Complete the ~~task~~ command with ACA ACTIVE status.	n/a
	0 or 1	Yes	n/a		n/a
Any task attribute except ~~ACA~~ACA task attribute	0 or 1	n/a	n/a	~~Terminate~~ Complete the ~~task~~ command with ACA ACTIVE status.	n/a

[a] Task attributes are described in 8.6.
[b] The NACA bit is in the CONTROL byte in the CDB (see 5.2).
[c] The TMF_ONLY bit is in the Control mode page (see SPC-4).
[d] If a ~~task~~ command with the ~~ACA~~ ACA task attribute ~~terminates~~ completes with a CHECK CONDITION status, the existing ACA condition shall be cleared and the value of the NACA bit shall control the establishment of a new ACA condition.
[e] All the conditions that affect the processing of commands (e.g., reservations) apply.

5.10.4 Handling new ~~tasks~~ commands received on non-faulted I_T nexuses when ACA is in effect

5.10.4.1 Command processing permitted for ~~tasks~~ commands received on non-faulted I_T nexuses during ACA

The device server shall process a PERSISTENT RESERVE OUT command with a PREEMPT AND ABORT service action (see SPC-4) while an ACA condition is established when the command is received on a non-faulted I_T nexus.

NOTE 10 - The processing of specific commands (e.g., PERSISTENT RESERVE OUT command with a PREEMPT AND ABORT service action) received on a non-faulted I_T nexus while an ACA condition is in effect provides SCSI initiator ports not associated with the faulted I_T nexus the opportunity to recover from error conditions that the initiator port associated with the faulted I_T nexus is unable to recover from itself.

5.10.4.2 Handling new ~~tasks~~ commands received on non-faulted I_T nexuses when ACA is in effect

The handling of ~~tasks~~ commands received on I_T nexuses other than the faulted I_T nexus depends on the value in the TST field in the Control mode page (see SPC-4).

Table 37 describes the handling of new ~~tasks~~ commands received on I_T nexuses other than the faulted I_T nexus when ACA is in effect.

Page: 115

Author: relliott Subject: Note Date: 3/20/2008 5:00:12 PM
Make everything in the table header lowercase except the first letter of each cell

Status
George Penokie Accepted 3/26/2008 11:13:16 AM
Author: relliott Subject: Highlight Date: 3/20/2008 5:00:24 PM
except
s/b
except the

Status
George Penokie Accepted 3/26/2008 11:13:48 AM
Author: relliott Subject: Highlight Date: 3/20/2008 5:00:34 PM
except
s/b
except the

Status
George Penokie Accepted 3/26/2008 11:13:54 AM

Table 37 — Handling for new ~~tasks~~ commands received on non-faulted I_T nexuses during ACA

TST Field Value in Control mode page	New ~~Task~~ Command Properties		New Command Permitted During ACA [c]	Device Server Action	ACA Established if New ~~task Terminates~~ Command Complete with a CHECK CONDITION status
	Task attribute [a]	NACA Value [b]			
000b	~~ACA~~ ACA task attribute	n/a	n/a	~~Terminate~~ Complete the ~~task~~ command with ACA ACTIVE status.	n/a
	Any task attribute except ~~ACA~~ACA task attribute	0	No	~~Terminate~~ Complete the ~~task~~ command with BUSY status.	n/a
		1	No	~~Terminate~~ Complete the ~~task~~ command with ACA ACTIVE status.	n/a
		0	Yes	Process the ~~task~~command.	No [d]
		1	Yes		Yes [d]
001b	~~ACA~~ ACA task attribute	0	n/a	Process an invalid task attribute condition as described in 5.13.	No
		1			Yes
	Any task attribute except ~~ACA~~ACA task attribute	0 or 1	n/a	Process the ~~task~~command. [e]	See 5.9.2.

[a] Task attributes are described in 8.6.
[b] The NACA bit is in the CONTROL byte in the CDB (see 5.2).
[c] See 5.10.4.1.
[d] If a permitted command ~~terminates~~ completes with a CHECK CONDITION status, the existing ACA condition shall be cleared and the value of the NACA bit shall control the establishment of a new ACA condition.
[e] When the TST field in the Control mode page contains 001b, commands received on a non-faulted I_T nexus shall be processed as if the ACA condition does not exist (see 5.9.2). In this case, the logical unit shall be capable of handling concurrent ACA conditions and sense data associated with each I_T nexus.

5.10.5 Clearing an ACA condition

An ACA condition shall only be cleared:

a) ~~As~~ as a result of a hard reset (see 6.3.2), logical unit reset (see 6.3.3), or I_T nexus loss (see 6.3.4);
b) ~~By~~ by a CLEAR ACA task management function (see 7.4) received on the faulted I_T nexus;
c) ~~By~~ by a PERSISTENT RESERVE OUT command with a PREEMPT AND ABORT service action with the ~~ACA~~ ACA task attribute received on the faulted I_T nexus that clears the ~~tasks~~ commands received on the faulted I_T nexus (see SPC-4);

d) By by a PERSISTENT RESERVE OUT command with a PREEMPT AND ABORT service action with a task attribute other than ACA ACA task attribute received on a non-faulted I_T nexus that clears the tasks commands received on the faulted I_T nexus;
e) When when a command with the ACA ACA task attribute received on the faulted I_T nexus terminates completes with a CHECK CONDITION status; or
f) When when a PERSISTENT RESERVE OUT command with a PREEMPT AND ABORT service action terminates completes in a CHECK CONDITION status.

Cases e) and f) may result in the establishment of a new ACA based on the value of the NACA bit.

When an ACA condition is cleared and no new ACA condition is established, the state of all tasks commands in the task set shall be modified as described in 8.8.

5.11 Overlapped commands

An overlapped command occurs when a task manager or a task router detects the use of a duplicate I_T_L_Q nexus (see 4.5.6) in a command before a task holding that I_T_L_Q nexus completes its task command lifetime (see 5.5). Each SCSI transport protocol standard shall specify whether or not a task manager or a task router is required to detect overlapped commands.

A task manager or a task router that detects an overlapped command shall abort all tasks commands received on the I_T nexus on which the overlapped command was received and the device server shall return complete with a CHECK CONDITION status for the overlapped command. The sense key shall be set to ABORTED COMMAND and the additional sense code shall be set to OVERLAPPED COMMANDS ATTEMPTED.

NOTE 11 - An overlapped command may be indicative of a serious error and, if not detected, may result in corrupted data. This is considered a catastrophic failure on the part of the SCSI initiator device. Therefore, vendor specific error recovery procedures may be required to guarantee the data integrity on the medium. The SCSI target device logical unit may return complete the overlapped command with additional sense data to aid in this error recovery procedure (e.g., sequential-access devices may return complete the overlapped command with the residue of blocks remaining to be written or read at the time the second command was received).

5.12 Incorrect logical unit selection

The SCSI target device's response to a command addressed to an incorrect logical unit number is described in this subclause.

In response to a REQUEST SENSE command, a REPORT LUNS command, or an INQUIRY command the SCSI target device shall respond as defined in SPC-4.

Any command except REQUEST SENSE, REPORT LUNS, or INQUIRY:

a) Shall shall be terminated completed with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and with the additional sense code set to LOGICAL UNIT NOT SUPPORTED, if:
   A) The the SCSI target device is not capable of supporting the logical unit (e.g., some SCSI target devices support only one peripheral device); or
   B) The the SCSI target device supports the logical unit, but the peripheral device is not currently connected to the SCSI target device;

   or

b) Is is responded to in a vendor specific manner, if:
   A) The the SCSI target device supports the logical unit and the peripheral device is connected, but the peripheral device is not operational; or
   B) The the SCSI target device supports the logical unit but is incapable of determining if the peripheral device is connected or is not operational because the peripheral device is not ready.

Page: 116

Author: relliott Subject: Highlight Date: 3/20/2008 5:01:19 PM
than
s/b
than the

Status
George Penokie Accepted 3/26/2008 11:15:15 AM
Author: relliott Subject: Cross-Out Date: 3/20/2008 5:01:51 PM
a

Status
George Penokie Rejected 3/26/2008 11:18:23 AM
Author: George Penokie Subject: Sticky Note Date: 3/26/2008 11:18:19 AM
Old wording put back.

Author: relliott Subject: Cross-Out Date: 3/20/2008 5:01:53 PM
a

Status
George Penokie Rejected 3/26/2008 11:18:42 AM
Author: George Penokie Subject: Sticky Note Date: 3/26/2008 11:18:38 AM
Old wording put back.

Author: relliott Subject: Highlight Date: 3/20/2008 5:01:37 PM
in a
s/b
with

Status
George Penokie Accepted 3/26/2008 11:19:01 AM
Author: relliott Subject: Highlight Date: 3/20/2008 5:02:41 PM
undo

Status
George Penokie Accepted 3/26/2008 11:20:31 AM
Author: relliott Subject: Highlight Date: 3/20/2008 5:10:38 PM
undo

Status
George Penokie Accepted 3/26/2008 11:20:40 AM

5.13 Task attribute exception conditions

If a command is received with a task attribute that is not supported or is not valid (e.g., an ACA task attribute when an ACA condition does not exist), the command shall be ~~terminated~~ completed with CHECK CONDITION ~~status,~~ status with the sense key set to ILLEGAL ~~REQUEST.~~ REQUEST and the additional sense code set to INVALID MESSAGE ERROR.

NOTE 12 - The use of the INVALID MESSAGE ERROR additional sense code is based on its similar usage in previous versions of this standard. The use of the INVALID MESSAGE ERROR additional sense code is not to be interpreted as a description of how the task attributes are represented by any given SCSI transport protocol.

Task attribute support should be reported with the Extended INQUIRY Data VPD page (see SPC-4).

5.14 Sense data

Sense data shall be made available by the logical unit in the event that a command completes with ~~a~~ CHECK CONDITION status or other conditions (e.g., the processing of a REQUEST SENSE command). The format, content, and conditions under which sense data shall be prepared by the logical unit are specified in this standard, SPC-4, the applicable command standard, and the applicable SCSI transport protocol standard.

Sense data associated with an I_T nexus shall be preserved by the logical unit until:

a) ~~The~~ the sense data is transferred;
b) ~~A~~ a logical unit reset (see 6.3.3) occurs;
c) ~~Power~~ power loss expected (see 6.3.5) occurs; or
d) ~~An~~ an I_T nexus loss (see 6.3.4) occurs for the I_T nexus associated with the preserved sense data.

When a command completes with a CHECK CONDITION status, sense data shall be returned in the same I_T_L_Q nexus transaction (see 3.1.50) as the CHECK CONDITION status. After the sense data is returned, it shall be cleared except when it is associated with a unit attention condition and the UA_INTLCK_CTRL field in the Control mode page (see SPC-4) contains 10b or 11b.

~~The return of~~ Completion with sense data in the same I_T_L_Q nexus transaction as a CHECK CONDITION status shall not affect ACA (see 5.10) or the sense data associated with a unit attention condition when the UA_INTLCK_CTRL field contains 10b or 11b.

5.15 Unit ~~Attention~~ attention condition

Each logical unit shall ~~generate~~ establish a unit attention condition whenever one of the following events occurs:

a) ~~A~~ a power on (see 6.3.1), hard reset (see 6.3.2), logical unit reset (see 6.3.3), I_T nexus loss (see 6.3.4), or power loss expected (see 6.3.5) occurs;
b) ~~A removable medium may have been changed;~~
c) ~~The mode parameters associated with this I_T nexus have been changed by a task received on another I_T nexus (i.e., SCSI initiator ports share mode parameters, see SPC-4);~~
d) ~~The log parameters associated with this I_T nexus have been changed by a task received on another I_T nexus (i.e., SCSI initiator ports share log parameters, see SPC-4);~~
e) ~~The version or level of microcode has been changed (see SPC-4);~~
f) ~~Tasks~~ Commands received on this I_T nexus have been cleared by a ~~task~~ command or a task management function associated with another I_T nexus and the TAS bit was set to zero in the Control mode page associated with this I_T nexus (see 5.6);
g) ~~INQUIRY data has been changed (see SPC-4);~~
h) ~~The~~ the logical unit inventory has been changed (see 4.5.19.1)~~;~~ or
i) ~~The mode parameters in effect for the associated I_T nexus have been restored from non-volatile memory (see SPC-4); or~~
j) ~~Any~~ any other event requiring the attention of the SCSI initiator device.

Page: 117

Author: relliott Subject: Cross-Out Date: 3/20/2008 5:11:30 PM
a

Status
George Penokie Rejected 3/26/2008 11:22:04 AM
Author: George Penokie Subject: Sticky Note Date: 3/26/2008 11:22:00 AM
Replaced with terminate wording.

Author: relliott Subject: Note Date: 3/20/2008 5:12:00 PM
rearrange c) and d)

Status
George Penokie Accepted 3/26/2008 11:23:26 AM

~~Logical units may queue unit attention conditions. After the first unit attention condition is cleared, another unit attention condition may exist (e.g., a unit attention condition with an additional sense code set to POWER ON OCCURRED may be followed by one with an additional sense code set to MICROCODE HAS BEEN CHANGED).~~

Unit attention conditions are classified by precedence levels. Table 38 defines the unit attention condition precedence levels.

Table 38 — Unit attention condition precedence level

Unit attention condition additional Sense Code	Unit attention condition precedence
POWER ON, RESET, OR BUS DEVICE RESET OCCURRED	highest
POWER ON OCCURRED or DEVICE INTERNAL RESET	
SCSI BUS RESET OCCURRED or MICROCODE HAS BEEN CHANGED or protocol specific	
BUS DEVICE RESET FUNCTION OCCURRED	
I_T NEXUS LOSS OCCURRED	
COMMANDS CLEARED BY POWER LOSS NOTIFICATION	
all others	Lowest

For unit attention conditions with the lowest precedence level with a given ADDITIONAL SENSE CODE field value, the unit attention condition with the ADDITIONAL SENSE CODE QUALIFIER field set to 00h has higher precedence level than the unit attention conditions with the ADDITIONAL SENSE CODE QUALIFIER field set to value other than 00h (e.g., PARAMETERS CHANGED has precedence over MODE PARAMETERS CHANGED and LOG PARAMETERS CHANGED). A unit attention condition with the lowest precedence level has equal priority with all unit attention conditions with the lowest precedence level with different ADDITIONAL SENSE CODE field values.

NOTE 13 - The unit attention additional sense code specificity order defined in 6.2 determines which unit attention condition is allowed to be established when certain conditions occur. The unit attention condition precedence defined in this subclause determines which unit attention conditions are allowed to clear other unit attention conditions if they have not yet been reported.

The device server shall maintain a queue of unit attention conditions of unspecified order for each I_T nexus. The queue should be large enough to hold every unit attention condition that the device server is capable of reporting.

When a device server establishes a unit attention condition:

1) the device server may clear unit attention conditions from the queue that are no longer needed as follows:
   A) the device server may clear any pending unit attention conditions in the queue that have lower precedence levels (e.g., BUS DEVICE RESET FUNCTION OCCURRED may clear I_T NEXUS LOSS OCCURRED and all unit attention conditions with a lower precedence); and
   B) the device server should clear pending unit attention conditions that have the same additional sense code (i.e., the device server should not add the same unit attention condition twice);
2) if a queue slot is available, then:
   A) if a higher precedence unit attention condition is not in the queue, the device server shall add the unit attention condition to the queue; or
   B) if a higher precedence unit attention condition is in the queue, the device server should add the unit attention condition to the queue.

Page: 118

Author: relliott Subject: Highlight Date: 3/20/2008 5:19:11 PM
Sense Code

lowercase

Status
George Penokie Accepted 3/26/2008 11:23:47 AM
Author: relliott Subject: Highlight Date: 3/20/2008 5:12:39 PM
ADDITIONAL SENSE CODE

smcaps

Status
George Penokie Accepted 3/26/2008 11:24:18 AM
Author: relliott Subject: Highlight Date: 3/20/2008 5:12:57 PM
value
s/b
values

Status
George Penokie Accepted 3/26/2008 11:24:56 AM
Author: relliott Subject: Highlight Date: 3/20/2008 5:20:18 PM
this is confusing

In the proposal 07-459, the levels are given numbers 1 to 6 in the table, and this sentence is saying that within the "all others" camp, all the non-00h values have the same priority.

Maybe the table should get two rows:
all others with additional sense code qualifier fields set to 00h (second lowest)
all others with additional sense code qualifier fields not set to 00h (lowest)

However, that would imply that xxh/00h always has higher precedence than yyh/non-00h, which is untrue; it is only higher than xxh/non-zero.

Perhaps another table is needed. "all others" would point to the other table, and it would explain the precedence within each additional sense code value.

Status
George Penokie Rejected 3/26/2008 11:26:05 AM
Author: George Penokie Subject: Sticky Note Date: 3/26/2008 11:26:27 AM
I think its ok the way it is.

In the sense data for the unit attention condition, the device shall either:

A) not include sense-key specific sense data; or
B) include sense-key specific sense data and set the OVERFLOW bit to zero (see SPC-4);

or

3) if a queue slot is not available, the device server shall either:
   A) replace any unit attention condition in the queue; or
   B) not add the unit attention condition to the queue.

The device server shall include sense-key specific sense data and set the OVERFLOW bit to one (see SPC-4) for at least one unit attention condition in the queue. If the device server establishes multiple unit attention conditions as a result of the same event or a series of events, the device server may establish the unit attention conditions in any order (e.g., in direct-access block devices, if a MODE SELECT command changes the initial command priority value, the device server may report PRIORITY CHANGED before MODE PARAMETERS CHANGED or may report MODE PARAMETERS CHANGED before PRIORITY CHANGED).

When the device server reports and clears a unit attention condition, it:

a) may select any unit attention condition in the queue to report; and
b) shall clear the unit attention condition from the queue after reporting it.

A unit attention condition shall persist ~~on the logical unit for the SCSI initiator port associated with each I_T nexus~~ until the ~~SCSI initiator port associated with the I_T nexus~~ device server clears the unit attention condition. Unit attention conditions are affected by the processing of commands as follows:

a) ~~If~~ if an INQUIRY command enters the enabled ~~task~~ command state, the device server shall ~~perform~~ process the INQUIRY command and shall neither report nor clear any unit attention condition;
b) ~~If~~ if a REPORT LUNS command enters the enabled ~~task~~ command state, the device server shall ~~perform~~ process the REPORT LUNS command and shall not report any unit attention condition~~.~~.
c) ~~If~~ if the UA_INTLCK_CTRL field in the Control mode page is set to 00b (see SPC-4), the SCSI target device shall clear any pending unit attention condition with an additional sense code of REPORTED LUNS DATA HAS CHANGED established for the initiator port associated with that I_T nexus in each logical unit accessible by the I_T nexus on which the REPORT LUNS command was received. Other pending unit attention conditions shall not be cleared~~.~~;
d) ~~If~~ if the UA_INTLCK_CTRL field in the Control mode page contains 10b or 11b, the SCSI target device shall not clear any unit attention condition(s);
e) ~~If~~ if a REQUEST SENSE command enters the enabled ~~task~~ command state while a unit attention condition exists for the SCSI initiator port associated with the I_T nexus on which the REQUEST SENSE command was received, then the device server shall ~~return~~ complete with GOOD status and either:
   A) ~~Report~~ report any pending sense data as parameter data and preserve all unit attention conditions on the logical unit; or
   B) ~~Report~~ report a unit attention condition as parameter data for the REQUEST SENSE command to the SCSI initiator port associated with the I_T nexus on which the REQUEST SENSE command was received. The logical unit may discard any pending sense data and shall clear the reported unit attention condition for the SCSI initiator port associated with that I_T nexus. If the unit attention condition has an additional sense code of REPORTED LUNS DATA HAS CHANGED, the SCSI target device shall clear any pending unit attention conditions with an additional sense code of REPORTED LUNS DATA HAS CHANGED established for the I_T nexus on which the command was received in each logical unit accessible by that I_T nexus;
f) ~~If~~ if the device server has already generated the ACA condition (see 5.10) for a unit attention condition, the device server shall report the unit attention condition (i.e., option c)B) above);
g) if the device server supports the NOTIFY DATA TRANSFER DEVICE command (see ADC-2) and a NOTIFY DATA TRANSFER DEVICE command enters the enabled ~~task~~ command state, then the device server shall ~~perform~~ process the NOTIFY DATA TRANSFER DEVICE command and shall neither report nor clear any unit attention condition; ~~and~~
h) ~~If~~ if a command other than INQUIRY, REPORT LUNS, REQUEST SENSE, or NOTIFY DATA TRANSFER DEVICE enters the enabled ~~task~~ command state while a unit attention condition exists for the SCSI initiator port associated with the I_T nexus on which the command was received, the device server shall ~~terminate~~ complete the command with ~~a~~ CHECK CONDITION status. The device server

Page: 119

Author: relliott Subject: Highlight Date: 3/20/2008 5:22:57 PM
OVERFLOW

smcaps

Status
George Penokie Accepted 3/26/2008 11:27:09 AM
Author: relliott Subject: Highlight Date: 3/20/2008 5:23:50 PM
The device server shall include sense-key specific sense data and set the OVERFLOW bit to one (see SPC-4) for at least one unit attention condition in the queue

This sentence is supposed to be unorderedtext0 under 3), not part of this paragraph

Status
George Penokie Accepted 3/26/2008 11:29:47 AM
Author: relliott Subject: Highlight Date: 3/20/2008 5:24:17 PM
the device server
s/b
it

Status
George Penokie Accepted 3/26/2008 11:31:12 AM
Author: relliott Subject: Note Date: 3/20/2008 5:26:48 PM
spc4r14 calls it "initial priority" not "initial command priority"

Maybe spc4 should change?

Status
George Penokie Accepted 5/8/2008 11:42:52 AM
Author: George Penokie Subject: Sticky Note Date: 3/26/2008 11:35:43 AM
Changed to << initial priority >>

Author: George Penokie Subject: Sticky Note Date: 5/8/2008 11:42:46 AM
Change was accepted and changed.

Author: relliott Subject: Highlight Date: 3/20/2008 5:28:58 PM
return complete with
s/b
complete the command with

or should it just say "process the command"? If there is a CDB field error, it still has the right to return CHECK CONDITION status.

Status
George Penokie Accepted 5/8/2008 11:45:44 AM
Author: George Penokie Subject: Sticky Note Date: 3/26/2008 11:37:55 AM
Changed to << process the command >>

Author: relliott Subject: Cross-Out Date: 3/20/2008 5:28:04 PM
a

Status
George Penokie Rejected 3/26/2008 11:38:15 AM
Author: George Penokie Subject: Sticky Note Date: 3/26/2008 11:38:32 AM
Changed to old wording.

Page: 124

Author: relliott Subject: Highlight Date: 4/28/2008 9:39:06 AM
This << and >> should be an << or >>

Status
George Penokie Accepted 4/28/2008 9:39:26 AM

Table 39 — Unit attention additional sense codes for events detected by SCSI target devices

Condition	Additional Sense Code	Specificity
Logical unit is unable to distinguish between the conditions	POWER ON, RESET, OR BUS DEVICE RESET OCCURRED	Lowest
Power on	POWER ON OCCURRED or DEVICE INTERNAL RESET [a]	
Hard reset	SCSI BUS RESET OCCURRED or MICROCODE HAS BEEN CHANGED [b] or protocol specific [c]	
Logical unit reset	BUS DEVICE RESET FUNCTION OCCURRED	
I_T nexus loss	I_T NEXUS LOSS OCCURRED	
Power loss expected	COMMANDS CLEARED BY POWER LOSS NOTIFICATION	Highest

[a] Used after a vendor-specific power on event has occurred (e.g., a firmware reboot).
[b] Only used if microcode has been changed (see SPC-4).
[c] Only used if a protocol-specific reset event has occurred.

NOTE 14 - The names of the unit attention conditions listed in the subclause (e.g., SCSI BUS RESET OCCURRED) are based on usage in previous versions of this standard. The use of these unit attention condition names is not to be interpreted as a description of how the unit attention conditions are represented by any given SCSI transport protocol.

A logical unit ~~may~~ should use the I_T NEXUS LOSS OCCURRED additional sense code when establishing a unit attention condition for an I_T nexus loss if:

a) ~~The~~ the SCSI initiator port to which the sense data is being delivered is the SCSI initiator port that was associated with the I_T nexus loss, and the logical unit has maintained all state information specific to that SCSI initiator port since the I_T nexus loss; and
b) ~~The~~ the I_T nexus being used to deliver the sense data is the same I_T nexus that was lost, and the logical unit has maintained all state information specific to that I_T nexus since the I_T nexus loss.

Otherwise, the logical unit shall use one of the less specific additional sense codes (e.g., POWER ON OCCURRED) when establishing a unit attention ~~condition~~condition for an I_T nexus loss.

6.3 Conditions resulting from SCSI events

6.3.1 Power on

Power on is a SCSI device condition resulting from a power on event. When a SCSI device is powered on, it shall cause a hard reset.

The power on condition applies to both SCSI initiator devices and SCSI target devices.

Power on events include:

a) power being applied to the SCSI device; and
b) vendor-specific events that cause the SCSI device to behave as if power has been applied (e.g., firmware reboot).

6.3.2 Hard reset

Hard reset is a SCSI device condition resulting from:

a) ~~A~~ a power on condition (see 6.3.1);
b) ~~Microcode~~ microcode change (see SPC-4); or
c) ~~A~~ a reset event indicated by a **Transport Reset** event notification (see 6.4).

The definition of reset events and the notification of their detection is SCSI transport protocol specific.

Each SCSI transport protocol standard that defines reset events shall specify a SCSI target port's protocol specific actions in response to reset events. Each SCSI transport protocol standard that defines reset events should specify when those events result in the delivery of a **Transport Reset** event notification to the SCSI applications layer.

SCSI transport protocols may include reset events that have no SCSI effects (e.g., a Fibre Channel non-initializing loop initialization primitive).

The hard reset condition applies to both SCSI initiator devices and SCSI target devices.

A SCSI target port's response to a hard reset condition shall include a logical unit reset condition (see 6.3.3) for all logical units to which the SCSI target port has access. A hard reset condition shall not affect any other SCSI target ports in the SCSI target device, however, the logical unit reset condition established by a hard reset may affect ~~tasks~~ commands and task management functions that are communicating via other SCSI target ports.

Although the task manager response to task management requests is subject to the presence of access restrictions, as managed by ACCESS CONTROL OUT commands (see SPC-4), a hard reset condition (see 6.3.2) shall not be prevented by access controls.

When a SCSI initiator port detects a hard reset condition, it should ~~terminate~~ end all its outstanding **Execute Command** procedure calls and all its outstanding task management procedure calls with a service response of SERVICE DELIVERY OR TARGET FAILURE. A hard reset condition shall not affect any other SCSI initiator ports in the SCSI initiator device, however, the logical unit reset condition established in a SCSI target device by a hard reset may affect ~~tasks~~ commands and task management functions that are communicating via other SCSI initiator ports.

A SCSI port's response to a hard reset condition shall include establishing an I_T nexus loss condition (see 6.3.4) for every I_T nexus associated with that SCSI port.

6.3.3 Logical unit reset

Logical unit reset is a logical unit condition resulting from:

a) ~~A~~ a hard reset condition (see 6.3.2); or
b) ~~A~~ a logical unit reset event indicating that a LOGICAL UNIT RESET task management request (see 7.7) has been processed.

The logical unit reset condition applies only to SCSI target devices.

When responding to a logical unit reset condition, the logical unit shall:

a) ~~Abort~~ abort all ~~tasks~~ commands as described in 5.6;
b) end all task management functions;
c) ~~Clear~~ clear all ACA conditions (see 5.9.5) in all task sets in the logical unit;
d) ~~Establish~~ establish a unit attention condition (see 5.14 and 6.2);
e) ~~Initiate~~ initiate a logical unit reset for all dependent logical units (see 4.5.19.4); and
f) ~~Perform~~ perform any additional functions required by the applicable command standards.

6.3.4 I_T nexus loss

I_T nexus loss is a SCSI device condition resulting from:

a) ~~A~~ a hard reset condition (see 6.3.2);
b) ~~An~~ an I_T nexus loss event (e.g., logout) indicated by a **Nexus Loss** event notification (see 6.4); or

Page: 125

Author: relliott Subject: Highlight Date: 3/20/2008 5:32:09 PM
terminate end
s/b
complete??

this is a global comment; "end" suddenly starts to appear as the chosen word here, when I thought "complete" was the choice earlier

Status
George Penokie Accepted 5/8/2008 11:46:46 AM
Author: George Penokie Subject: Sticky Note Date: 3/26/2008 11:40:59 AM
End was not searched for in the wording switch. It has now been removed and replaced with complete or terminate as needed.

7.5 CLEAR TASK SET

~~Request~~Procedure call:

Service Response = CLEAR TASK SET (IN (I_T_L Nexus))

Description:

This function shall be supported by all logical units.

The task manager shall abort all ~~tasks~~ commands in the task set as described in 5.6.

If the TST field is set to 001b (i.e., per I_T nexus) in the Control mode page (see SPC-4), there is one task set per I_T nexus. As a result, no other I_T nexuses are affected and CLEAR TASK SET is equivalent to ABORT TASK SET (see 7.2).

All pending status and sense data for the task set shall be cleared. Other previously established conditions, including mode parameters, reservations, and ACA shall not be changed by the CLEAR TASK SET function.

All SCSI transport protocol standards shall support the CLEAR TASK SET task management function.

7.6 I_T NEXUS RESET

~~Request~~Procedure call:

Service Response = I_T NEXUS RESET (IN (I_T Nexus))

Description:

SCSI transport protocols may or may not support I_T NEXUS RESET and may or may not require logical units accessible through SCSI target ports using such transport protocols to support I_T NEXUS RESET.

Each logical unit accessible through the SCSI target port shall perform the I_T nexus loss functions specified in 6.3.4 for the I_T nexus on which the function request was received, then the SCSI target device shall ~~return~~ complete with a FUNCTION COMPLETE response. After returning a FUNCTION COMPLETE response, the logical unit(s) and the SCSI target port shall perform any additional functions specified by the SCSI transport protocol.

7.7 LOGICAL UNIT RESET

~~Request~~Procedure call:

Service Response = LOGICAL UNIT RESET (IN (I_T_L Nexus))

Description:

This function shall be supported by all logical units.

Before returning a FUNCTION COMPLETE response, the logical unit shall perform the logical unit reset functions specified in 6.3.3.

NOTE 15 - Previous versions of this standard only required LOGICAL UNIT RESET support in logical units that supported hierarchical logical units.

All SCSI transport protocol standards shall support the LOGICAL UNIT RESET task management function.

Page: 131

Author: relliott Subject: Highlight Date: 3/20/2008 5:33:28 PM
undo

or "complete the task management funcion with"

Status
George Penokie Accepted 3/26/2008 11:43:09 AM
Author: relliott Subject: Highlight Date: 3/20/2008 5:32:58 PM
with

Status
George Penokie Accepted 3/26/2008 11:46:21 AM
Author: relliott Subject: Highlight Date: 3/20/2008 5:34:11 PM
returning a FUNCTION COMPLETE response
change to
completing the task management function

Maybe merge this with the preceding paragraph into a 1) 2) list?

Status
George Penokie Accepted 5/8/2008 11:46:59 AM
Author: George Penokie Subject: Sticky Note Date: 3/26/2008 11:45:26 AM
Made into 1,2 list.

7.8 QUERY TASK

~~Request~~Procedure call:

Service Response = QUERY TASK (IN (I_T_L_Q Nexus))

Description:

SCSI transport protocols may or may not support QUERY TASK and may or may not require logical units accessible through SCSI target ports using such transport protocols to support QUERY TASK.

The task manager in the specified logical unit shall:

a) if the specified ~~task~~ command is present in the task set, then ~~return~~ complete with a service response set to FUNCTION SUCCEEDED; and
b) if the specified ~~task~~ command is not present in the task set, then ~~return~~ complete with a service response set to FUNCTION COMPLETE.

7.9 QUERY TASK SET

~~Request~~Procedure call:

Service Response = QUERY TASK SET (IN (I_T_L Nexus))

Description:

SCSI transport protocols may or may not support QUERY TASK SET and may or may not require logical units accessible through SCSI target ports using such transport protocols to support QUERY TASK SET.

The task manager in the specified logical unit shall:

a) if there is any ~~task~~ command present in the task set from the specified I_T nexus, then ~~return~~ complete with a service response set to FUNCTION SUCCEEDED; and
b) if there is no ~~task~~ command present in the task set from the specified I_T nexus, then ~~return~~ complete with a service response set to FUNCTION COMPLETE.

7.10 QUERY UNIT ATTENTION

~~Request~~Procedure call:

Service Response = QUERY UNIT ATTENTION (IN (I_T_L Nexus), OUT ([Additional Response Information]))

Description:

A SCSI transport protocol may or may not support QUERY UNIT ATTENTION. A SCSI transport protocol supporting QUERY UNIT ATTENTION may or may not require logical units accessible through SCSI target ports using that transport protocol to support QUERY UNIT ATTENTION.

The task manager in the specified logical unit shall:

a) if there is a unit attention condition (see 5.14) or a deferred error (see SPC-4) pending for the specified I_T nexus, then ~~return~~ complete with a service response set to FUNCTION SUCCEEDED; and
b) if there is no unit attention condition or deferred error pending for the specified I_T nexus, then ~~return~~ complete with a service response set to FUNCTION COMPLETE.

Page: 132

Author: relliott Subject: Highlight Date: 3/20/2008 5:34:33 PM
undo or fix

Status
George Penokie Accepted 3/26/2008 11:47:13 AM
Author: relliott Subject: Highlight Date: 4/28/2008 9:42:45 AM
This << and >> should be an << or >>

Status
George Penokie Accepted 4/28/2008 9:42:57 AM
Author: relliott Subject: Highlight Date: 3/20/2008 5:34:39 PM
undo or fix

Status
George Penokie Accepted 3/26/2008 11:47:20 AM
Author: relliott Subject: Highlight Date: 3/20/2008 5:34:47 PM
undo or fix

Status
George Penokie Accepted 3/26/2008 11:47:58 AM
Author: relliott Subject: Highlight Date: 4/28/2008 9:42:30 AM
This << and >> should be an << or >>

Status
George Penokie Accepted 4/28/2008 9:43:08 AM
Author: relliott Subject: Highlight Date: 3/20/2008 5:34:53 PM
undo or fix

Status
George Penokie Accepted 3/26/2008 11:48:05 AM
Author: relliott Subject: Highlight Date: 4/28/2008 9:57:03 AM
QUERY UNIT ATTENTION should be called QUERY ASYNCHRONOUS EVENT as it now includes deferred errors.

Status
George Penokie Accepted 5/8/2008 11:48:34 AM
Author: relliott Subject: Highlight Date: 3/20/2008 5:35:00 PM
undo or fix

Status
George Penokie Accepted 3/26/2008 11:48:42 AM
Author: relliott Subject: Highlight Date: 4/28/2008 9:43:46 AM
This << and >> should be an << or >>

Status
George Penokie Accepted 4/28/2008 9:43:58 AM
Author: relliott Subject: Highlight Date: 3/20/2008 5:35:05 PM
undo or fix

Status
George Penokie Accepted 3/26/2008 11:48:48 AM

If the service response is not FUNCTION SUCCEEDED, then the task manager shall set the Additional Response Information argument to 000000h.

If the service response is FUNCTION SUCCEEDED, then the task manager shall set the Additional Response Information argument as defined in table 41.

Table 41 — Additional Response Information argument for QUERY UNIT ATTENTION

Bit Byte	7	6	5	4	3	2	1	0
0	Reserved		UADE DEPTH		SENSE KEY			
1	ADDITIONAL SENSE CODE							
2	ADDITIONAL SENSE CODE QUALIFIER							

The UADE DEPTH field indicates the number of pending unit attention conditions or deferred errors and is defined in table 42.

Table 42 — UADE DEPTH field

Code	Description
00b	The combined number of unit attention conditions and deferred errors is unknown
01b	The combined number of unit attention conditions and deferred errors is one
10b	The combined number of unit attention conditions and deferred errors is greater than one
11b	Reserved

The SENSE KEY field indicates the value of the SENSE KEY field that would be returned in the sense data for the ~~highest priority pending~~ next unit attention condition or deferred error that is going to be reported (see SPC-4).

The ADDITIONAL SENSE CODE field indicates the value of the ADDITIONAL SENSE CODE field in the ~~highest priority pending~~ next unit attention condition or deferred error going to be reported (see SPC-4).

The ADDITIONAL SENSE CODE QUALIFIER field indicates the value of the ADDITIONAL SENSE CODE QUALIFIER field in the ~~highest priority pending~~ next unit attention condition or deferred error going to be reported (see SPC-4).

7.11 Task management function lifetime

The task manager shall create a task management function upon receiving a Task Management Request Received indication (see 7.12). The task management function shall exist until:

a) the task manager sends a SCSI transport protocol service response for the task management function;
b) an I_T nexus loss (see 6.3.4);
c) a logical unit reset (see 6.3.3);
d) a hard reset (see 6.3.2); ~~or~~
e) power loss expected (see 6.3.5); or
f) a power on condition (see 6.3.1).

Page: 133

Author: relliott Subject: Highlight Date: 3/20/2008 5:35:47 PM
going
s/b
that is going

Status
George Penokie Accepted 3/26/2008 11:49:34 AM

Author: relliott Subject: Highlight Date: 3/20/2008 5:35:52 PM
going
s/b
that is going

Status
George Penokie Accepted 3/26/2008 11:49:47 AM

Page: 138

Author: relliott Subject: Cross-Out Date: 3/20/2008 5:37:14 PM
a

Status
George Penokie Rejected 3/26/2008 11:50:44 AM
Author: George Penokie Subject: Sticky Note Date: 3/26/2008 11:50:40 AM
Terminate wording used here.

8 Task set management

8.1 Introduction to task set management

This clause describes some of the controls that application clients have over task set management behaviors (see 8.3). This clause also specifies task set management requirements in terms of:

a) ~~Task~~ task states (see 8.5);
b) ~~Task~~ task attributes (see 8.6);
c) ~~Task~~ command priority (see 8.7);
d) ~~The~~ the events that cause transitions between ~~task~~ command states (see 8.4 and 8.5); and
e) ~~A~~ a map of ~~task~~ command state transitions (see 8.8).

This clause concludes with several task set management examples (see 8.9).

~~Task~~ Command behavior, as specified in this clause, refers to the functioning of a ~~task~~ command as observed by an application client, including the results of command processing and interactions with other ~~tasks~~commands.

The requirements for task set management only apply to a ~~task~~ command after it has been entered into a task set. A ~~task~~ command shall be entered into a task set unless:

a) ~~A~~ a condition exists that causes that ~~task~~ command to be completed with a status of BUSY, RESERVATION CONFLICT, TASK SET FULL, or ACA ACTIVE;
b) ~~Detection~~ detection of an overlapped command (see 5.10) causes that ~~task~~ command to be completed with a CHECK CONDITION status; or
c) SCSI transport protocol specific errors cause that ~~task~~ command to be completed with CHECK CONDITION status.

8.2 Implicit head of queue

A command standard (see 3.1.21) may define commands each of which may be processed by the task manager as if the ~~task~~command's task attribute is HEAD OF QUEUE task attribute even if the ~~task~~ command is received with a SIMPLE task ~~attribute,~~ attribute or an ORDERED ~~task attribute, or no~~ task attribute.

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 task attribute because whether the ORDERED task attribute is honored is vendor specific.

8.3 ~~Task~~ Command management model

The ~~task~~ command management model requires the following task set management behaviors:

a) ~~The~~ the SIMPLE task attribute (see 8.6.1) shall be supported;
b) ~~Task~~ task attributes other than SIMPLE may be supported;
c) ~~The~~ the QUEUE ALGORITHM MODIFIER field in the Control mode page (see SPC-4) shall control the processing sequence of ~~tasks~~ commands having the SIMPLE task attribute;
d) ~~The~~ the QERR field in the Control mode page (see SPC-4) shall control aborting of ~~tasks~~ commands when any command completes with ~~a~~ a CHECK CONDITION ~~status is returned for any task~~status; and
e) ~~The~~ the CLEAR TASK SET task management function (see 7.5) shall be supported.

Page: 139

Author: relliott Subject: Note Date: 3/20/2008 5:38:18 PM
lowercase table 43 title except for first letter

Status
George Penokie Accepted 3/26/2008 11:51:03 AM

Author: relliott Subject: Note Date: 3/20/2008 5:38:00 PM
make sure columns are wide enough (change marks may be causing extra wrapping)

Status
George Penokie Accepted 3/26/2008 11:51:16 AM

8.4 ~~Task~~ Command management events

The following describe the events that cause changes in ~~task~~ command state.

All older ~~tasks~~ commands ended: If the TST field in the Control mode page (see SPC-4) equals 000b, all ~~tasks~~ commands received on all I_T nexuses and accepted earlier in time than the referenced ~~task~~ command have ended. If the TST field equals 001b, all ~~tasks~~ commands received on the referenced I_T nexus and accepted earlier in time than the referenced ~~task~~ command have ended.

All head of queue and older ordered ~~tasks~~ commands ended: If the TST field equals 000b, all the following ~~tasks~~ commands received on all I_T nexuses have ended:

a) ~~All~~ all head of queue ~~tasks~~commands; and
b) ~~All~~ all ordered ~~tasks~~ commands accepted earlier in time than the referenced ~~task~~command.

If the TST field equals 001b, the following ~~tasks~~ commands received on the referenced I_T nexus have ended:

a) ~~All~~ all head of queue ~~tasks~~commands; and
b) ~~All~~ all ordered ~~tasks~~ commands accepted earlier in time than the referenced ~~task~~command.

ACA establishment: An ACA condition has been established (see 5.8).

~~task~~ command abort: A ~~task~~ command has been aborted as described in 5.6.

~~task~~ command completion: The device server has sent a service response of TASK COMPLETE for the ~~task~~ command (see 5.1 and 5.5).

~~task~~ command ended: A ~~task~~ command has ~~completed~~ terminated or aborted.

ACA cleared: An ACA condition has been cleared (see 5.9.5).

8.5 ~~Task~~ Command states

8.5.1 Overview

8.5.1.1 ~~Task~~ Command state nomenclature

This standard defines four ~~tasks~~ command states, summarized in table 43.

Table 43 — ~~Task~~ Command State Nomenclature

~~Task~~ Command State Name	Reference	~~Tasks~~ Commands in This State May Be Called
Enabled ~~task~~ command state	8.5.2	Enabled ~~tasks~~commands
Blocked ~~task~~ command state	8.5.3	Blocked ~~tasks~~commands
Dormant ~~task~~ command state	8.5.4	Dormant ~~tasks~~commands
Ended ~~task~~ command state	8.5.5	Ended ~~tasks~~commands

Page: 140

Author: relliott Subject: Highlight Date: 3/20/2008 5:38:39 PM
a status of GOOD
s/b
GOOD status

Status
George Penokie Accepted 3/26/2008 11:56:23 AM

8.5.1.2 Suspended information

Any information the logical unit has or accepts for a ~~task~~ command in the blocked ~~task~~ command state (see 8.5.3) or dormant ~~task~~ command state (see 8.5.4) is required to be held in a condition where it is not available to the ~~task~~command. Such information is called suspended information.

8.5.2 Enabled ~~task~~ command state

A ~~task~~ command in the enabled ~~task~~ command state may become a current ~~task~~ command and may complete at any time, subject to the ~~task~~ command completion constraints specified in the Control mode page (see SPC-4). A ~~task~~ command that has been accepted into the task set shall not complete or become a current ~~task~~ command unless it is in the enabled ~~task~~ command state.

Except for the use of resources required to preserve ~~task~~ command state, a ~~task~~ command shall produce no effects detectable by the application client before the ~~task~~command's first transition to the enabled ~~task~~ command state. Before entering this state for the first time, the ~~task~~ command may perform other activities visible at the STPL (e.g., pre-fetching data to be written to the media), however this activity shall not result in a detectable change in state as perceived by an application client. In addition, the behavior of a completed ~~task~~command, as defined by the commands it has processed, shall not be affected by the ~~task~~command's states before it enters the enabled ~~task~~ command state.

8.5.3 Blocked ~~task~~ command state

A ~~task~~ command in the blocked ~~task~~ command state is prevented from completing due to an ACA condition. A ~~task~~ command in this state shall not become a current ~~task~~command. While a ~~task~~ command is in the blocked ~~task~~ command state, any information the logical unit has or accepts for the ~~task~~ command shall be suspended. If the TST field in the Control mode page (see SPC-4) equals 000b the blocked ~~task~~ command state is independent of I_T nexus. If the TST field equals 001b the blocked ~~task~~ command state applies only to the faulted I_T nexus.

8.5.4 Dormant ~~task~~ command state

A ~~task~~ command in the dormant ~~task~~ command state is prevented from completing due to the presence of certain other ~~tasks~~ commands in the task set. -A ~~task~~ command in this state shall not become a current ~~task~~command. - While a ~~task~~ command is in the dormant ~~task~~ command state, any information the logical unit has or accepts for the ~~task~~ command shall be suspended.

8.5.5 Ended ~~task~~ command state

A ~~task~~ command in the ended ~~task~~ command state is removed from the task set.

8.5.6 ~~Task~~ Command states and ~~task~~ command lifetimes

Figure 45 shows the events corresponding to two ~~task~~ command processing sequences. Except for the dormant ~~task~~ command state between times A and B in case 1, logical unit conditions and the commands processed by the ~~task~~ command are identical. Assuming in each case the ~~task~~ command completes with a status of GOOD at time C, the state observed by the application client for case 1 shall be indistinguishable from the state observed for case 2.

Page: 142

Author: relliott Subject: Note Date: 3/20/2008 5:40:40 PM
it might be helpful to include a table here showing the priority. It would stand out better than a dense paragraph:

Command priority Description
0h vendor-specific scheduling importance
1h highest scheduling importance
...
Fh lowest scheduling importance

Status
George Penokie Accepted 3/26/2008 1:23:02 PM

8.6.4 ~~Ordered task~~

8.6.5 Commands having the ORDERED task attribute

If accepted, a ~~task~~ command having the ORDERED task attribute shall be entered into the task set in the dormant ~~task~~ command state. The ~~task~~ command shall not enter the enabled ~~task~~ command state until all commands having a HEAD OF QUEUE ~~tasks~~ task attribute and all older ~~tasks~~ commands in the task set have ended (see 8.4).

8.6.6 ~~Head of queue task~~

8.6.7 Commands having the HEAD OF QUEUE task attribute

If accepted, a ~~task~~ command having the HEAD OF QUEUE task attribute shall be entered into the task set in the enabled ~~task~~ command state.

8.6.8 ~~ACA task~~

8.6.9 Commands having the ACA task attribute

If accepted, a ~~task~~ command having the ACA task attribute shall be entered into the task set in the enabled ~~task~~ command state. There shall be no more than one ~~ACA~~ command having the ACA task attribute per task set (see 5.9.2).

8.7 ~~Task~~ Command priority

~~Task~~ Command priority specifies the relative scheduling importance of a ~~task~~ command having a SIMPLE task attribute in relation to other ~~tasks~~ commands having SIMPLE task attributes already in the task set. If the ~~task~~ command has a task attribute other than SIMPLE, ~~the task~~ then command priority is not used. ~~Task~~ Command priority is a value in the range of 0h through Fh. A ~~task~~ command with either no ~~task~~ command priority or a ~~task~~ command priority set to 0h has a vendor-specific level of scheduling importance. A ~~task~~ command with a ~~task~~ command priority set to 1h has the highest scheduling importance, with increasing ~~task~~ command priority values indicating decreasing scheduling importance. A ~~task~~ command with a ~~task~~ command priority set to Fh has the lowest scheduling importance.

If the ~~Task~~ Command Priority argument is set to zero or is not contained within the Send SCSI Received SCSI transport protocol service indication (see 5.4.2)~~,~~ , and a priority has been assigned to the I_T_L nexus, then the device server shall use ~~that~~ the specified priority for the I_T_L nexus as the ~~task~~ command priority. A priority ~~may be~~ is assigned to an I_T_L nexus by a SET PRIORITY command (see SPC-4) or by the INITIAL PRIORITY field in the Control Extension mode page (see SPC-4). If no priority has been assigned to the I_T_L nexus using the SET PRIORITY command and the logical unit does not support the INITIAL PRIORITY field in the Control Extension mode ~~page~~ page, then the device server shall set the ~~task~~ command priority to 0h (i.e., vendor specific~~)~~), or the ~~task~~ command shall have no ~~task~~ command priority.

A task manager may use ~~task~~ command priority to determine an ordering to process ~~tasks~~ commands with the SIMPLE task attribute within the task set. A difference in ~~task~~ command priority between ~~tasks~~ commands may not override other scheduling considerations (e.g., different times to access different logical block addresses) or vendor specific scheduling considerations. However, processing of a collection of ~~tasks~~ commands with different task priorities should cause the subset of ~~tasks~~ commands with the higher task priorities to ~~return~~ complete with status sooner in aggregate than the same subset would if the same collection of ~~tasks~~ commands were submitted under the same conditions but with all task priorities being equal.

If the TST field in the Control mode page (see SPC-4) contains 000b, then the transition from dormant ~~task~~ command to enabled ~~task~~ command shall not occur while an ACA is in effect for any I_T nexus (see 5.9.3 and 5.9.4). If the TST field contains 001b, then dormant ~~tasks~~ commands from the faulted I_T nexus shall not transition to the enabled ~~task~~ command state while an ACA is in effect for that I_T nexus (see 5.9.3).

Transition S2:S3: The establishment of an ACA condition (see 8.4) shall cause zero or more enabled ~~tasks~~ commands to enter the blocked ~~task~~ command state as described in 5.9.2.

Transition S3:S2: When an ACA condition is cleared (see 8.4), ~~tasks~~ commands that entered the blocked command state when the ACA condition was established (see 5.9.2) shall re-enter the enabled ~~task~~ command state.

Transition S2:S4: A ~~task~~ command that has completed (see 8.4) or aborted (see 8.4 and 5.6) shall enter the ended ~~task~~ command state. This is the only state transition out of S2:Enabled that applies to commands having an ~~ACA task~~ACA task attribute.

Transitions S1:S4, S3:S4: A ~~task~~ command abort event (see 8.4 and 5.6) shall cause the ~~task~~ command to unconditionally enter the ended ~~task~~ command state.

8.9 Task set management examples

8.9.1 Introduction

Several task set management scenarios are shown in 8.9.3, 8.9.5, and 8.9.7. The examples are valid for configurations with one or multiple SCSI initiator ports when the TST field contains 000b (i.e., the interaction among ~~tasks~~ commands in a task set is independent of the I_T nexus on which a ~~task~~ command is received). The examples are also valid for a single I_T nexus when the TST field contains 001b (i.e., task set management proceeds independently for each I_T nexus and the events and transitions for the task set associated with one I_T nexus do not affect the task set management for task sets associated with other I_T nexuses). Throughout these examples, the scope of the task set box drawn in each snapshot depends on the setting of the TST field in the Control mode page (see SPC-4).

The figure accompanying each example shows successive snapshots of a task set after various events (e.g., ~~task~~ command creation or completion). In all cases, the constraints on ~~task~~ command completion order established using Control mode page (see SPC-4) fields other that the TST field (e.g., the QUEUE ALGORITHM MODIFIER field) are not in effect.

A task set is shown as an ordered list or queue of ~~tasks~~ commands with ~~the~~ commands having a HEAD OF ~~the~~ QUEUE task attribute towards the top of the figure. A new command having a HEAD OF QUEUE task attribute always enters the task set at the head, displacing older commands having a HEAD OF QUEUE ~~tasks~~task attribute. ~~Simple~~A command having a SIMPLE task attribute, ORDERED ~~and ACA tasks~~ task attribute, or ACA task attribute always enter the task set at the end of the queue.

~~Tasks~~Command, denoted by rectangles, are numbered in ascending order from oldest to most recent. Fill, shape and line weight are used to distinguish ~~task~~ command states and task attributes are shown in table 45.

Table 45 — Task attribute and state indications in examples

Task Attribute	Box Shape	Line Weight	~~Task~~ Command State
SIMPLE	Rounded Corners	Thin	Enabled
ORDERED	Square Corners	Thin	Dormant
HEAD OF QUEUE	Square Corners	Thick	Blocked
ACA	Square Corners	Thin Dashed	

Page: 144

Author: relliott Subject: Highlight Date: 3/20/2008 5:41:30 PM
enter
s/b
enters

Status
George Penokie Accepted 3/26/2008 1:00:35 PM

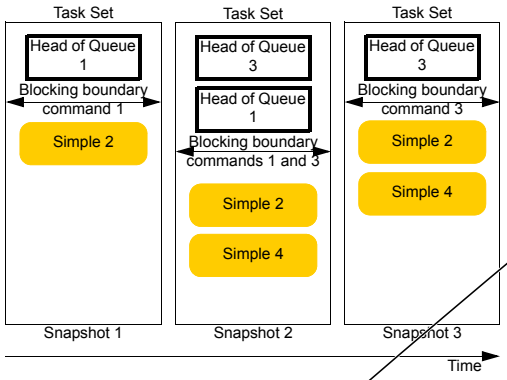


Figure 48 — ~~Head~~ Commands having the HEAD OF QUEUE ~~tasks~~ task attribute and blocking boundaries (example 2)

Because the blocking boundary remains in place for a command having a HEAD OF QUEUE ~~task~~task attribute, both the commands having a SIMPLE ~~tasks~~ task attribute remain in the dormant ~~task~~ command state in snapshot 3. The blocking boundary is not removed until all commands having a HEAD OF QUEUE ~~tasks~~ task attribute complete.

8.9.4 ~~Ordered tasks~~

8.9.5 Commands having the ORDERED task attribute

An example of commands having an ORDERED task attribute and commands having a SIMPLE task attribute interaction is shown in figure 50.

Page: 146

Author: relliott Subject: Highlight Date: 3/13/2008 1:18:02 PM
an ORDERED task attribute
s/b
ORDERED task attributes

Status
George Penokie Accepted 3/26/2008 1:01:50 PM

Author: relliott Subject: Highlight Date: 3/13/2008 1:18:11 PM
a SIMPLE task attribute
s/b
SIMPLE task attributes

Status
George Penokie Accepted 3/26/2008 1:01:43 PM

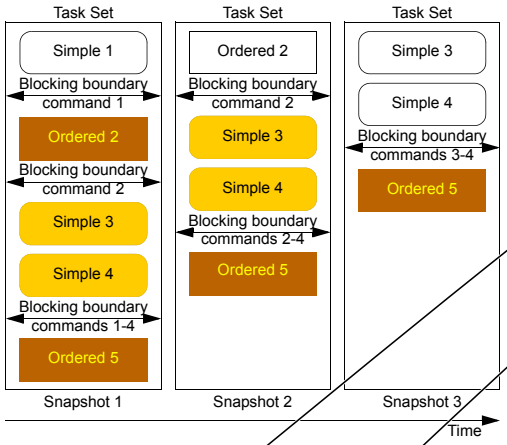


Figure 49 — ~~Ordered tasks and blocking boundaries~~

Figure 50 — Commands having an ORDERED task attribute and blocking boundaries

The state of dormant ~~task~~ command 2 through ~~task~~ command 5 is determined by the requirements shown in table 46.

Table 46 — Dormant ~~task~~ command blocking boundary requirements

~~Task~~Command	Reason for blocking boundary
2	An command having an ORDERED task attribute is not allowed to enter the enabled ~~task~~ command state until all commands having a HEAD OF QUEUE ~~tasks~~ task attribute and all older ~~tasks~~ commands have ended.
5	
3	A command having a SIMPLE task attribute is not allowed to enter the enabled ~~task~~ command state until all commands having a HEAD OF QUEUE task attribute and all older ordered ~~tasks~~ commands have ended.
4	

The table 46 constraints are shown by the blocking boundaries in snapshot 1.

In snapshot 2, the completion of ~~task~~ command 1 allows the command having an ORDERED task attribute 2 to enter the enabled ~~task~~ command state. Since the initial constraints on ~~task~~ command 3, ~~task~~ command 4 and ~~task~~ command 5 are still in effect, these ~~tasks~~ commands are required to remain in the dormant ~~task~~ command state. As shown in snapshot 3, the completion of ~~task~~ command 2 triggers two state changes, with ~~task~~ command 3 and ~~task~~ command 4 transitioning to the enabled ~~task~~ command state. Task 5 is required to remain in the dormant ~~task~~ command state until ~~task~~ command 3 and ~~task~~ command 4 complete.

Page: 147

Author: relliott Subject: Highlight Date: 3/13/2008 1:19:16 PM
an ORDERED task attribute
s/b
ORDERED task attributes

Status
George Penokie Accepted 3/26/2008 1:02:32 PM
Author: relliott Subject: Note Date: 3/20/2008 5:42:50 PM
Make right column wider to reduce number of rows

Assume that wrapping will also be reduced when changes are not shown

Status
George Penokie Accepted 3/26/2008 1:03:04 PM

Page: 148

Author: relliott Subject: Highlight Date: 3/13/2008 1:19:40 PM
ACA task attribute
s/b
ACA task attributes

Status
George Penokie Accepted 3/26/2008 1:03:32 PM

8.9.6 ~~ACA task~~

8.9.7 Commands having the ACA task attribute

Figure 52 shows the effects of an ACA condition on the task set. This example assumes the QERR field contains 00b in the Control mode page (see SPC-4). Consequently, clearing an ACA condition does not cause ~~tasks~~ commands to be aborted

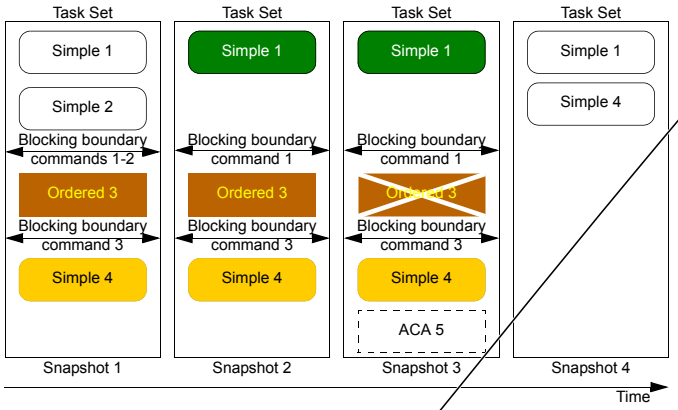


Figure 51 — ~~ACA task example~~

Figure 52 — Commands having ACA task attribute example

The completion of ~~task~~ command 2 with CHECK CONDITION status causes ~~task~~ command 1 to enter the blocked ~~task~~ command state shown in snapshot 2. In snapshot 3, command having an ORDERED task attribute 3 is aborted using the ABORT TASK task management function and ~~ACA~~ command having an ACA task attribute 5 is created to perform additional handling for the exception. Once the ACA condition is cleared (i.e., snapshot 4), command having a SIMPLE task attribute 1 is allowed to reenter the enabled ~~task~~ command state. Since there are no commands having a HEAD OF QUEUE ~~tasks~~ task attribute or older commands having an ORDERED ~~tasks~~task attribute, ~~task~~ command 4 also transitions to the enabled ~~task~~ command state.

Page: 150

Author: relliott Subject: Note Date: 3/20/2008 5:43:43 PM
check b) and c), maybe a Framemaker comparison anomoly?

Status
George Penokie Accepted 3/26/2008 1:04:42 PM
Author: George Penokie Subject: Sticky Note Date: 3/26/2008 1:04:36 PM
Comparison anomoly

Table A.2 — Name attribute size and support requirements

Attribute	Name	
	Size	Support Requirements [c]
SCSI device name	not specified [a]	optional
Initiator port name	not specified	optional
Target port name	not specified [a]	optional
Logical unit name	not specified [a]	mandatory

[a] Reported in the Device Identification VPD page (see SPC-4) from a logical unit within a SCSI target device.

[b]

[c] As defined in this standard or SPC-4.

Each SCSI transport protocol defines the size and format of identifier attributes and name attributes.

See table A.3 for a list of the sizes for identifier attributes for each SCSI transport protocol.

Table A.3 — Identifier attribute size for each SCSI transport protocol

Attribute	Size					
	FCP-4	SRP	iSCSI	SBP-3	SAS-2 SSP	ADT-2
Initiator port identifier	3 bytes	16 bytes	241 bytes [a]	2 bytes	8 bytes	none
Target port identifier	3 bytes	16 bytes	233 bytes [a]	11 bytes	8 bytes	none
LUN	8 bytes	8 bytes	8 bytes	2 bytes	8 bytes	2 bytes

[a] Maximum size, including the terminating null character byte.

See table A.6 for a list of the format of the name attributes for each SCSI transport protocol.

Table A.6 — Name attribute format for each SCSI transport protocol

Attribute	Format [a]					
	FCP-4	SRP	iSCSI [b]	SBP-3	SAS-2 SSP	ADT-2
SCSI device name	not specified	not specified	SCSI name string format	not specified	NAA IEEE Registered format	not specified
Initiator port name	Fibre Channel name_ identifier	EUI-64 || 8 byte extension [d]	iSCSI name [c] || ",i,0x" || Initiator Session Identifier [e]	EUI-64	not specified	not specified
Target port name	Fibre Channel name_ identifier	EUI-64 || 8 byte extension [d]	iSCSI name [c] || ",t,0x" || Target Portal Group Tag [f]	EUI-64 || Discovery ID [g]	not specified	not specified
Logical unit name	Device Identification VPD page name (see SPC-4)					

Key:
- \|\| means "concatenated with"
- ",i,0x" means a UTF-8 string composed of the following five characters: comma, lowercase i, comma, zero, and lowercase x.
- ",t,0x" means a UTF-8 string composed of the following five characters: comma, lowercase t, comma, zero, and lowercase x.

[a] In addition to the name formats shown in this table, any SCSI transport protocol may support the SCSI name string format (see SPC-4).
[b] iSCSI identifiers are concatenated strings containing no null characters except after the last string in the concatenation.
[c] The iSCSI name portion of the string is a worldwide unique UTF-8 string no more than 223 bytes long, not including null character termination.
[d] Required to be worldwide unique and recommend to be EUI-64 concatenated with an 8 byte extension.
[e] The Initiator Session Identifier (ISID) portion of the string is a UTF-8 encoded hexadecimal representation of a six byte binary value. This portion of the string contains no more than 12 bytes, not including null character termination if any.
[f] The Target Portal Group Tag (TPGT) portion of the string is a UTF-8 encoded hexadecimal representation of a two byte binary value. This portion of the string contains no more than 4 bytes, not including null character termination if any.
[g] See ISO/IEC 13213:1994 for more information on the Discovery ID.

A.3 SCSI transport protocol acronyms and bibliography

A.3.1 EUI-64 (Extended Unique Identifier, a 64-bit globally unique identifier): The IEEE maintains a tutorial describing EUI-64 at http://standards.ieee.org/regauth/oui/tutorials/EUI64.html.

Page: 153

Author: relliott Subject: Highlight Date: 3/13/2008 1:20:54 PM
name_
identifier

I think FC standards capitalize that as Name_Identifier

Status
George Penokie Accepted 3/26/2008 1:06:51 PM

Author: relliott Subject: Highlight Date: 3/13/2008 1:21:00 PM
name_
identifier

I think FC standards capitalize that as Name_Identifier

Status
George Penokie Accepted 3/26/2008 1:06:57 PM

Author: relliott Subject: Highlight Date: 3/20/2008 5:44:44 PM
recommend
s/b
recommended

Status
George Penokie Accepted 3/26/2008 1:07:25 PM

Annex C
(informative)

~~Terminology mapping~~

Terminology mapping to previous versions of this standard

The introduction of a UML model into this standard resulted in changes in terminology between ~~SAM-4~~ this standard and ~~SAM-3~~ previous versions of this standard (see table C.1).

Table C.1 — Terminology mapping to previous versions of this standard

Term used in this standard	Term used in previous versions of this standard
command identifier	task tag

Page: 157

Author: relliott Subject: Note Date: 3/20/2008 5:45:52 PM
how about listing

task -> command?

Status
George Penokie Accepted 3/26/2008 1:08:27 PM