Working Draft

T11.1 / Project XXXX-X / Rev 1.3

September 29, 1999

Information Technology - SCSI on Scheduled Transfer Protocol (SST)

Secretariat : National Committee for Information Technology Standardization (NCITS)

This is an internal working draft of T11.1, a Task Group of Technical Committee T11 of Accredited Standards Committee NCITS. As such, this is not a completed standard. The contents are actively being modified by T11.1.

ABSTRACT

This document specifies a mapping for SCSI commands, data transfers, and responses using the Scheduled Transfer (ST) protocol (NCITS: T11.1, Project 1245-D: Scheduled Transfer) as the lower level protocol. The ST protocol is defined for a variety of network media including Ethernet, ATM, and HIPPI-6400. The mapping of SCSI storage data onto the ST protocol layer enables storage area networking (SAN) implementations on a wide variety of common network infrastructures.

Contacts:

T11.1 Chairman
Roger Ronald
Power Micro Research
Suite 1100
1411 East Campbell Road
Richardson, TX 75081
Voice : 972-437-9461
FAX : 972-994-0888
E-mail : rronald@pmr.com

T11.1 Vice Chairman and Technical Editor
Don Woelz
Genroco Inc.
255 Info Highway
Slinger, WI 53086
Voice : 414-644-2505
Fax : 414-644-6667
E-mail : don@genroco.com

Reference number
ISO/IEC 11518-xx : 199x
NCITS.*** - 199x
Printed 9/29/99

Other Points of Contact:

	T11 Chairman	T11 Vice-Chairman	NCITS Secretariat
	Kumar Malavalli	Edward L. Grivna	NCITS Secretariat, ITI
	Brocade Communications	Cypress Semiconductor	1250 Eye Street, NW Suite 200
	1901 Guadalupe Parkway	2401 East 86th Street	Washington, DC 20005
	San Jose, CA 95131	Bloomington, MN 55425	
Voice :	408-487-8156	612-851-5046	202-737-8888
FAX :	408-524-8601	612-851-5087	202-638-4922
E-mail :	kumar@brocade.com	elg@cypress.com	ncitssec@itic.nw.dc.us

T11.1 E-mail Reflector (for HIPPI and ST technical discussions and notification of things on the web site)

Internet address for subscribing to the reflector:	Majordomo@nsco.network.com
Message should contain a line stating…	subscribe hippi <*your E-mail address*>
Internet address for distribution via the HIPPI reflector	hippi@nsco.network.com

T11 E-mail Reflector (for T11 meeting notices, agendas etc.)

Internet address for subscription to the T11 reflector:	Majordomo@nsco.network.com
Message should contain a line stating…	subscribe T11 <*your E-mail address*>
Internet address for distribution via T11 reflector:	t11@nsco.network.com

Web sites:

HIPPI and ST Standards Activities	http://www.hippi.org/
T11 Activities	http://www.t11.org/
NCITS	http://www.ncits.org/

T11 Document Distribution :

Global Engineering
15 Inverness Way East
Englewood, CO 80112-5704
Voice : 303-792-2181 or 800-854-7179
FAX : 303-792-2192

PATENT STATEMENT

CAUTION: The developers of this standard have requested that holder's of patents that may be required for the implementation of the standard, disclose such patents to the publisher. However, neither the developers nor the publisher have undertaken a patent search in order to identify which, if any, patents may apply to this standard.

As of the date of publication of this standard and following calls for the identification of patents that may be required for the implementation of the standard, some such claims have been made. No further patent search is conducted by the developer or the publisher in respect to any standard it processes. No representation is made or implied that licenses are not required to avoid infringement in the use of this standard.

Comments on Rev 3.1

This is a preliminary document undergoing lots of changes. Many of the additions are just place holders, or are put there to stimulate discussion. Hence, do not assume that the items herein are correct, or final – everything is subject to change. This page tries to outline where we are; what has been discussed and semi-approved, and what has been added or changed recently and deserves your special attention. This summary relates to changes since the previous revision. Also, previous open issues are outlined with a single box, new open issues ones are marked with a double bar on the left edge of the box.

Changes are marked with margin bars so that changed paragraphs are easily found, and then highlights mark the specific changes. The list below just describes the major changes, for detail changes please compare this revision to the previous revision. **The major technical changes are printed in bold.**

Please help us in this development process by sending comments, corrections, and suggestions to the Technical Editor, Don Woelz, of GENROCO, Inc., at don@genroco.com. If you would like to address the whole group working on this document, then send the comment(s) to hippi@nsco.network.com.

1. Changed many items in this document that did not conform to ANSI formatting guidelines. Due to the volume of these changes, many are not referenced by highlights or changebars.
2. Combined the T11.1 Vice-Chairman and Technical Editor into one list on the T11.1 cover page.
3. Changed the Abstract on the Title page, the Forward page, and the Scope in clause 1.
4. Globally changed the following words from upper case to lower case as necessary: Protocol, Interconnect, Figure, Table, Sequence
5. Globally removed the word "Section" in clause references and generally cleaned up clause references to conform to ANSI document standards.
6. Globally changed the words "send", "sent", or "sending" when referencing the issuance of an ST command to "issue", issued, or "issuing" respectively.
7. Globally changed "optional payload" to "option payload"
8. Globally changed "If…., …" to "If…, then …" to clarify the text.
9. Changed the title of figure 1 and added the block titled "Various ST-enabled networks".
10. Changed format of references in 2.1 and 2.2.
11. Added clause numbers to individual definitions in 3.1.
12. Added definitions for exposed, STTVC, Initiator Transaction Identifier, and SPMR to the definitions in 3.1.
13. In 3.1.14, changed "ST" references to "SST".
14. Clause 5.3 was misnumbered and was changed to 5.2. All subsequent clause 5 numbering was changed to reflect the new numbering. All references in these comments to these clauses uses the corrected numbering sequence.
15. The last two paragraphs of 5.2.1 were deleted. The two paragraphs before that were moved to clause 7. A reference to clause 7 was inserted in their place.
16. The remaining paragraphs of 5.2.1 (except for what is now the last paragraph) were re-written.
17. A new figure 2 was added at the end of 5.2.1. All subsequent figure numbers were incremented. Further references in these comments relates to the renumbered figures as they appear in this version of the document.
18. The word "standard" was deleted in 5.2.2.
19. Clause 5.2.3 was deleted in its entirety.
20. In 5.3, paragraph 3, changed "including some command" to "including command".
21. In 5.3, paragraph 5, changed two instances of "is" to "shall be" and deleted the words "Note that".
22. In 5.3, paragraph 7, deleted the last sentence.
23. In 5.3, paragraph 10, added a reference to clause 6.3.
24. In 5.3, paragraph 11, changed "is" to "shall be" and changed "software" to "entity".

25. In 5.3, paragraph 12, changed "proper status" to "proper SCSI status".
26. In 5.3, paragraph 14, changed "exchanges" to "transactions".
27. In 5.3.1, paragraph 3, changed "which have been sent" to "issued".
28. In 5.4, last paragraph, deleted the words "as always".
29. 6.2 was moved before 6.1 and was renumbered to 6.1. All previous 6.1 clauses were demoted (i.e., 6.1 became 6.1.1, 6.1.1 became 6.1.1.1, etc.) to become subclauses of this new 6.1. All further clause references in these Comments shall refer to the newly numbered paragraphs.
30. In 6.1.1.4, in the paragraph describing the DATA_CHANNEL field of figure 4, several wording changes were made to clarify the item.
31. **In 6.1.1.4, the paragraph describing the AUTHENTICATE field in figure 4 was deleted.**
32. **In 6.1.1.4, figure 4, the ONE_CTS_READ bit was moved from byte 2 to byte 3 of the option payload (in place of the ATHENTICATE field). The order in which these bits are discussed was changed in the text to be consistent with the figure.**
33. **In 6.1.1.5, paragraph 1, two instances of the word "should" were changed to "shall".**
34. In 6.1.1.6 the words "rejected or" were deleted.
35. In table 4, SSTVC Parameters were added.
36. In 6.2.1, "ST_OPTION_CODE" was changed to "ST_OC".
37. In 6.2.4, the word "ANSI" was added to the standards reference and the 2nd and 3rd paragraphs were deleted.
38. In 6.2.5, the words "a number of" were deleted.
39. In 6.2.5.1, the 1st paragraph, the wording regarding the reference was changed to clarify the sentence.
40. In 6.2.5.1, all paragraphs between table 5 and the end of the clause were deleted.
41. In 6.2.5.2, all paragraphs after the 1st paragraph were deleted.
42. In 6.2.6, the text describing the location of the CDB command byte and subsequent bytes of the CDB was changed to clarify meaning.
43. In 6.3, the equations describing the computation of the Bufx/Offset were changed to clarify the calculation.
44. In 6.3, paragraph 2, all sentences after the first sentence were deleted.
45. In 6.4, paragraph 3, the second sentence was deleted.
46. In 6.4.1, paragraph 1 was deleted.
47. In 6.4.1, the last paragraph had the word "ANSI" added to the standards reference.
48. In 6.4.2, paragraph 1, deleted all sentences after the first sentence.
49. In 6.4.3, paragraph 3, the word "always" was deleted.
50. In 6.3.4, paragraph 4, the word "always" was deleted.
51. In 6.3.5, deleted "the" two times, changed "proscribed" to "specified", changed an errant reference to 6.4.6, and added "in table 6".
52. In 6.3.6, the ANSI standard reference was clarified in two places, and "described" was changed to "specified".
53. In 6.4, the words "or End_Ack" were deleted in two places.
54. In 6.4.1, "ST_OPTION_CODE" was changed to "ST_OC" in two places.
55. In 6.4.2, paragraph 1, the word "sink" was changed to "destination".
56. In 7, the text from 5.2.1 was added.
57. The title of 7.1 was changed.
58. In 7.1, the words "endpoint Target" were changed to "Target endpoint" in two places.

ANSI ®
NCITS xxx-199x

American National Standard
for Information Technology –

SCSI on Scheduled Transfer Protocol (SST)

Secretariat

Information Technology Industry Council (ITI)

Approved , 199x

American National Standards Institute, Inc.

Abstract

This document specifies a mapping for SCSI commands, data transfers, and responses using the Scheduled Transfer (ST) protocol (NCITS: T11.1, Project 1245-D: Scheduled Transfer) as the lower level protocol. The ST protocol is defined for a variety of network media including Ethernet, ATM, and HIPPI-6400. The mapping of SCSI storage data onto the ST protocol layer enables storage area networking (SAN) implementations on a wide variety of common network infrastructures.

American National Standard

Approval of an American National Standard requires verification by ANSI that the requirements for due process, consensus, and other criteria for approval have been met by the standards developer.

Consensus is established when, in the judgment of the ANSI Board of Standards Review, substantial agreement has been reached by directly and materially affected interests. Substantial agreement means much more than a simple majority, but not necessarily unanimity. Consensus requires that all views and objections be considered, and that a concerted effort be made towards their resolution.

The use of American National Standards is completely voluntary; their existence does not in any respect preclude anyone, whether he has approved the standards or not, from manufacturing, marketing, purchasing, or using products, processes, or procedures not conforming to the standards.

The American National Standards Institute does not develop standards and will in no circumstances give interpretation on any American National Standard. Moreover, no person shall have the right or authority to issue an interpretation of an American National Standard in the name of the American National Standards Institute. Requests for interpretations should be addressed to the secretariat or sponsor whose name appears on the title page of this standard.

CAUTION NOTICE: This American National Standard may be revised or withdrawn at any time. The procedures of the American National Standards Institute require that action be taken periodically to reaffirm, revise, or withdraw this standard. Purchasers of American National Standards may receive current information on all standards by calling or writing the American National Standards Institute.

CAUTION: The developers of this standard have requested that holder's of patents that may be required for the implementation of the standard disclose such patents to the publisher. However, neither the developers nor the publisher have undertaken a patent search in order to identify which, if any, patents may apply to this standard.

As of the date of publication of this standard, following calls for the identification of patents that may be required for the implementation of the standard, notice of one or more such claims has been received.

By publication of this standard, no position is taken with respect to the validity of this claim or of any rights in connection therewith. The known patent holder(s) has (have), however, filed a statement of willingness to grant a license under these rights on reasonable and nondiscriminatory terms and conditions to applicants desiring to obtain such a license. Details may be obtained from the publisher.

No further patent search is conducted by the developer or the publisher in respect to any standard in process. No representation is made or implied that this is the only license that may be required to avoid infringement in the use of this standard.

Contents

Tables

Figures

Foreword (This foreword is not part of American National Standard NCITS xxx-199x.)

This document specifies a mapping for SCSI commands, data transfers, and responses using the Scheduled Transfer (ST) protocol (NCITS: T11.1, Project 1245-D: Scheduled Transfer) as the lower level protocol. The ST protocol is defined for a variety of network media including Ethernet, ATM, and HIPPI-6400. The mapping of SCSI storage data onto the ST protocol layer enables storage area networking (SAN) implementations on a wide variety of common network infrastructures.

Requests for interpretation, suggestions for improvement or addenda, or defect reports are welcome. They should be sent to the National Committee for Information Technology Standards, 1250 Eye Street, NW, Suite 200, Washington, DC 20005.

This standard was processed and approved for submittal to ANSI by NCITS. Committee approval of the standard does not necessarily imply that all committee members voted for approval. At the time it approved this standard, the NCITS had the following members:

(List of NCITS Committee members, and other active participants, at the time the document is forwarded for public review, will be included by the Technical Editor.)

Technical Committee T11 on Device Level Interfaces, which reviewed this standard, had the following participants:

(List of T11 Committee members, and other active participants, at the time the document is forwarded for public review, will be included by the Technical Editor.)

Task Group T11.1 on the High-Performance Parallel Interface, which developed this standard, had the following participants:

(List of T11.1 Task Group members, and active participants, at the time of document is forwarded for public review will be included by the Technical Editor.)

Introduction

The SCSI-3 family of standards is developed by NCITS T10 to facilitate the use of the SCSI command sets for different types of devices over a variety of physical interconnects. The master architectural document of the family of standards is ANSI X3.270-1996. Information Technology - SCSI-3 Architecture Model (SAM). The SAM document contains a guide to other SCSI-3 documents.

The SCSI on ST (SST) specification defines a protocol within the SCSI-3 SAM as shown in figure 1. The SAM interconnects to which the SST protocol may attach are not defined within this specification, but rather, are any interconnects or other protocols on which the basic ST protocol may operate.

Clause 4 describes how SCSI operations are structurally mapped onto ST operations in the SST protocol.

Clause 5 defines specific message formats for the SST protocol.

Clause 6 defines error recovery procedures for the SST protocol.

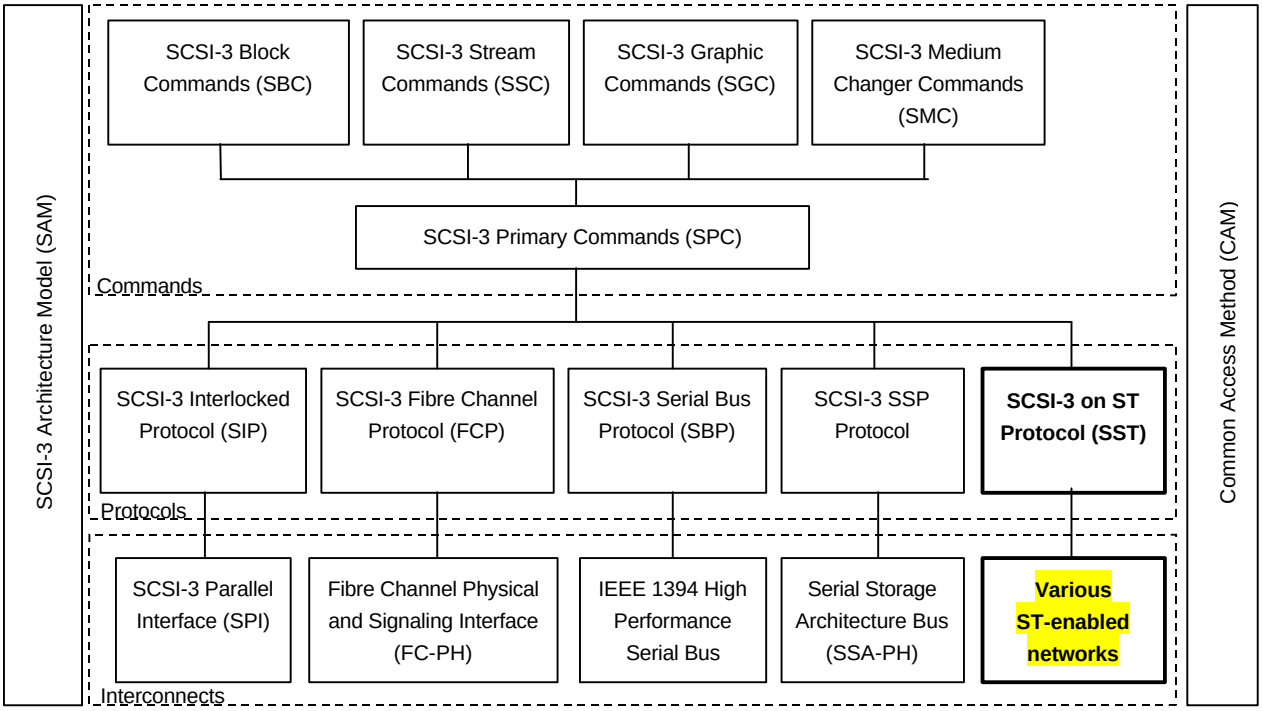

Figure 1 –Relationship of SST to SAM

American National Standard for Information Technology –

Scheduled Transfer Protocol (ST)

1 Scope

This document specifies a mapping for SCSI commands, data transfers, and responses using the Scheduled Transfer (ST) protocol (NCITS: T11.1, Project 1245-D: Scheduled Transfer) as the lower level protocol. The ST protocol is defined for a variety of network media including Ethernet, ATM, and HIPPI-6400. The mapping of SCSI storage data onto the ST protocol layer enables storage area networking (SAN) implementations on a wide variety of common network infrastructures.

Specifications are included for

– connection management,

– device management,

– task management,

– SCSI command service requests using ST Request_To_Send, Request_To_Receive, and Nop operations,

– SCSI data delivery requests using the ST Clear_To_Send operation,

– SCSI data delivery actions using ST Data operations,

– SCSI command service responses using the ST Put operation to a Persistent Memory Region,

– SCSI I/O operations using ST Read, Write, Nop, and Put sequences, and

– aborting connections and operations.

2 Normative references

The following standards contain provisions that, through 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 can be obtained from ANSI: Approved ANSI standards, approved and draft international and regional standards (ISO, IEC, CEN/CENELEC, ITUT) and approved 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. Additional availability contact information is provided below as needed.

2.1 Approved references

ANSI NCITS 323-1998, *Information Systems – High-Performance Parallel Interface – 6400 Mbit/s Physical Layer (HIPPI-6400-PH)*

ANSI X3.183-1991 (R1996), *Information Systems – High-Performance Parallel Interface – Mechanical, Electrical, and Signaling Protocol Specification (HIPPI-PH)*

ANSI X3.210-1998, *Information Systems – High-Performance Parallel Interface – Framing Protocol (HIPPI-FP)*

ANSI X3.218-1997, *Information Systems – High-Performance Parallel Interface – Physical Switch Control (HIPPI-SC)*

ANSI X3.222-1993, *Information Systems – High-Performance Parallel Interface – Encapsulation of ISO 8802-2 (IEEE Std 802.2) Logical Link Control Protocol Data Units (HIPPI-LE)*

ANSI X3.131-1994, *Information Systems – Small Computer System Interface – 2 (SCSI-2)*

ANSI X3.270-1996, *Information Systems – SCSI-3 Architecture Model (SAM)*

2.2 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. For information about obtaining copies of this document or for more information on the current status of the document, contact National Committee for Information Technology Standards, 1250 Eye Street, NW, Suite 200, Washington, DC 20005, 202-626-5746.

ANSI NCITS 324-1999, *High-Performance Parallel Interface – 6400 Mbit/s Physical Switch Control (HIPPI-6400-SC)*

NCITS T11.1, Project 1245-D, *Scheduled Transfer (ST)*

3 Definitions and conventions

3.1 Definitions

For the purposes of this standard, the follow-ing definitions apply.

3.1.1 Block: An ordered set of one or more STUs within a Read, Write, Put, Get or FetchOp sequence.

3.1.2 Buffer Index (Bufx): A 32-bit parameter identifying the starting address of a data buffer.

3.1.3 Data Channel: The logical channel that carries the data payload.

3.1.4 Data operation: A data transmission consisting of a Schedule Header and up to 4 gigabytes of user payload.

3.1.5 Destination: The end device that receives an operation or data.

3.1.6 exposed: A memory region that is enabled for data operations.

3.1.7 Initiator: The end device that starts a sequence of operations. This is typically a host computer system, but may also be a non-transparent translator, bridge, or router.

3.1.8 Initiator Transaction Identifier: The ST Initiator Sequence Identifier used on the ST Request_To_Send, ST Request_To_Receive, or SST Request_Zero_Length operation which begins an SST Transaction (see 5.3).

3.1.9 Key: A local identifier used to select and validate operations.

3.1.10 lower-layer protocol (LLP): A protocol below the Scheduled Transfer Protocol, e.g., a physical layer.

3.1.11 Offset: A parameter specifying the data's starting point relative to the start of a Bufx.

3.1.12 Opaque data: Four bytes of Source ULP to Destination ULP peer-to-peer information carried in a Data operation's Schedule Header separately from the data payload.

3.1.13 operation: The procedure defined by the parameters in a Schedule Header, and any payload associated with that Schedule Header. The code in the Schedule Header's "Op" field identifies the operation's name/function.

3.1.14 optional: Characteristics that are not required by SST. However, if any optional characteristic is implemented, then it shall be implemented as defined in SST.

3.1.15 persistent: Memory that is maintained for multiple Put, Get, and FetchOp operations.

3.1.16 Put: An operation to write data into a persistent memory region on a remote end device.

3.1.17 Scheduled Transfer: An information transfer, normally used for bulk data movement, where the end devices prearrange the transfer using the protocol defined in the ST standard.

3.1.18 Scheduled Transfer Unit (STU): The data payload portion of a Data operation. STUs

are the basic components of Blocks and are the smallest units transferred.

3.1.19 sequence: An ordered group of ST operations providing a particular function, e.g., Read, Write, Get, etc., between an Initiator and a Responder. The roles of Initiator and Responder are constant for all operations in the sequence.

3.1.20 SST Virtual Connection (SSTVC): An ST Virtual Connection is opened on the SST reserved ULP port number, and managed according to the rules defined in 5.2.

3.1.21 ST Buffer Size: The unit of memory addressed by ST for Bufx and Offset calculations and expressed as 2Bufsize bytes.

3.1.22 Status Persistent Memory Region (SPMR): An ST Persistent Memory Region established by the SSTVC Initiator to contain SCSI status information (see 5.2.1)

3.1.23 Transaction: One or more distinct ST sequences (e.g. Read sequence, Write sequence, Persistent Memory Region (PMR) Put sequence), which are logically associated to implement a complete SCSI function (see 4.1).

3.1.24 Transfer: An ordered set of one or more Blocks within a Scheduled Transfer.

3.1.25 upper-layer protocol (ULP): The protocol above ST. A ULP could be implemented in hardware or software, or could be distributed between the two.

3.1.26 Virtual Connection: A bi-directional logical connection used for Scheduled Transfers between two end devices. A Virtual Connection contains a logical Control Channel and one or more logical Data Channels in each direction.

3.2 Editorial conventions

A number of conditions, sequence parameters, events, states, or similar terms are printed with the first letter of each word in uppercase and the rest lowercase (e.g., Block, Transfer). Any lowercase uses of these words have the normal technical English meaning.

The word shall, when used in this American National standard, states a mandatory rule or requirement. The word should, when used in this standard, states a recommendation.

Multiword parameters and field names are joined with an underscore, e.g., D_Port. A parameter associated with a particular end device uses a single letter prefix and a hyphen as a joiner, e.g., I-Key denoting the Initiator's Key.

All numbers are represented as unsigned integers.

Operations contained within <…> are conditional, and may not occur.

3.2.1 Binary notation

Binary notation is used to represent relatively short fields. For example a two-bit field containing the binary value of 10 is shown in binary format as b'10'. An "x" in a bit position indicates a "don't care" value.

3.2.2 Hexadecimal notation

Hexadecimal notation is used to represent some fields. For example a two-byte field containing a binary value of b'1100010000000011' is shown in hexadecimal format as x'C403'.

3.3 Acronyms and other abbreviations

FC	Fibre Channel
HIPPI	High-Performance Parallel Interface
id	identifier
IEEE	Institute of Electrical and Electronic Engineers
IP	Internet Protocol
KB	kilobyte (i.e., 1024 bytes)
LAN	local area network
LLP	lower-layer protocol
MAC	Media Access Control
MB	megabyte (i.e., 1,048,576 bytes)
num	number, as in B_num
PMR	Persistent Memory Region
RFC	Request For Comment
SCSI	Small Computer System Interconnect
SNAP	SubNetwork Access Protocol
SPMR	Status Persistent Memory Region
SST	SCSI on Scheduled Transfer Protocol
SSTVC	SST Virtual Connection
ST	Scheduled Transfer Protocol
STU	Scheduled Transfer Unit
TM	Task Management
ULA	Universal LAN address
ULP	upper-layer protocol

4 Overview

4.1 Structure and concepts

Scheduled Transfer (ST) is a data transfer protocol which may be implemented on a wide variety of Lower Layer Protocols (LLP), which supports a connection-oriented data transfer model with multiple outstanding data transfers per connection.

Three kinds of functional management are defined by the SST specification:

– Connection management;

– Device management;

– Task management.

For the purposes of describing this SST protocol, we define the term Transaction which consists of one or more distinct ST sequences (e.g. Read sequence, Write sequence, Persistent Memory Region (PMR) Put sequence), which are logically associated to implement a complete SCSI function from the list above.

The correspondence between SCSI functions, elements of the SST protocol, and elements of the ST protocol on which SST is built is shown in table 1.

Table 1 – Functional correspondence between SCSI, SST, and ST operations

SCSI	SST	ST
Command Service Request	Request_To_Send or Request_To_Receive or Request_Zero_Length	Request_To_Send operation Request_To_Receive operation Nop operation
Data delivery request	Clear_To_Send operation	Clear_To_Send operation
Data delivery action	Data operation (STU)	Data operation
Command service response	Status Put	Put operation
I/O operation	Transaction	Read sequence and Put sequence or Write sequence and Put sequence or Nop operation and Put sequence
Task Management operation	Transaction or Request_Zero_Length or End sequence	Nop operation and Put operation or Nop operation or End sequence

5 SST characteristics

5.1 Common ST operation characteristics

All ST Control operations used in the SST protocol shall have the ST Interrupt flag bit set.

Non-final ST Data operations of Read sequences and Write sequences shall have the ST Flag bits Interrupt = b'0', Last = b'0', and Silent = b'1'.

The final ST Data operation of Read sequences and Write sequences shall have the ST Flag bits Interrupt = b'1', Last = b'1', and Silent = b'0'.

All ST Data operations for an SST connection shall be sent on the ST Data Channel specified in the SST connection setup protocol (see 5.2). ST Data Channel flag bits shall be appropriately set to reflect this.

The SST protocol does not use the operation pairs for reliable data movement (see 10.2 of NCITS T11.1, Project 1245-D, Scheduled Transfer). Instead, entire SST Transactions are retried as described in 6.1, Transaction Timeouts.

5.2 Connection management

An SST Virtual Connection (SSTVC) is an extension of the basic ST Virtual Connection.

5.2.1 Connection setup

An SSTVC shall be requested by issuing an ST Request_Connect operation to the well known port for SST with the Out_Of_Order (O) flag bit = b'0'. SST requires that out-of-order operation not be requested in order to support the Clear_To_Send operation accounting protocol described in 5.4.1.

Note: A well-known port for SST must be established.

If the SSTVC Responder is willing to accept the SSTVC, it shall respond with an ST Connection_Answer operation with the Out_Of_Order (O) flag bit = b'0'. Otherwise the SSTVC Responder shall reject the SSTVC with an ST Connection_Answer operation with the flag bit R = b'1'.

If the SSTVC is accepted by the SSTVC Responder, the SSTVC Initiator shall issue an SSTVC_Parameters operation with its SSTVC parameters in the option payload (see 6.1).

If the SSTVC parameters are unacceptable to the SSTVC Responder, then the Responder shall disconnect the SSTVC using the standard ST Virtual Connection Disconnect sequence. If the SSTVC parameters are acceptable to the SSTVC Responder then the SSTVC Responder shall issue an SSTVC_Parameters operation with its SSTVC parameters in the option payload (see 6.1). If the SSTVC parameters specified by the SSTVC Responder are unacceptable to the SSTVC Initiator then the SSTVC Initiator shall disconnect the SSTVC using the standard ST Virtual Connection Disconnect sequence.

If the SSTVC parameters in the SSTVC Initiator's SSTVC_Parameters operation indicate that the SSTVC Initiator can function as a SCSI Initiator and the SSTVC Responder intends to function as a SCSI Target, then the SSTVC Responder shall use an ST Request_Memory_Region operation to request that the SSTVC Initiator expose a Persistent Memory Region, called the Status Persistent Memory Region (SPMR), for the ST Data Channel specified in the SSTVC_Parameters Option Payload. The SPMR size shall be specified in SSTVC_Parameters operation issued by the SSTVC Initiator. The SSTVC Initiator shall expose the SPMR and respond with an appropriate Memory_Region_Available operation. The initial Offset of the SPMR shall be a multiple of 512 bytes, the SPMR segment size.

When the SSTVC_Parameters operation issued by the SSTVC Responder is received by the SSTVC Initiator, and the connection parameters specify that the SSTVC Responder can function as a SCSI Initiator, and the SSTVC Initiator intends to function as a SCSI Target, then the SSTVC Initiator shall request that the SSTVC Responder expose an SPMR as described above.

An SSTVC endpoint shall not initiate any SCSI operations until the SPMR has been requested and exposed, and the ST Memory_Region_Available operation sent. An SST Initiator shall keep a timer awaiting the ST Request_Memory_Region operation, and if none

arrives within the duration of the timer, then disconnect the SSTVC using the ST Virtual Connection Disconnect sequence and retry the SSTVC sequence. Connection error recovery shall be done according to 7.

Note: The timer should be specified.

Figure 2 shows a typical SSTVC connection setup sequence.

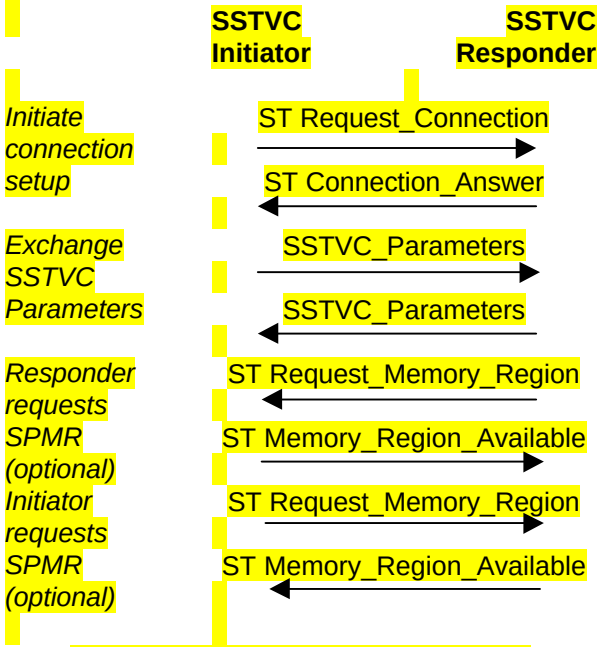

Figure 2 – SSTVC Connection setup sequence

5.2.2 Connection disconnect

An SSTVC shall be disconnected using the ST Virtual Connection Disconnect sequence.

5.3 Device management

An application client begins an SST I/O Transaction when it provides to the SST layer a request for an Execute command service. A single request or a list of linked requests may be presented to the software interface. The SST layer then performs the following actions using ST services to perform the SCSI command.

The ST endpoint that is the SCSI Initiator for the command starts a Transaction by issuing an ST Request_To_Send, ST Request_To_Receive, or SST Request_Zero_Length operation (see 6.1) to begin an SST device management sequence. A Request_To_Send operation is issued if the SCSI command involves data transfer from the SCSI Initiator to the SCSI Target. A Request_To_Receive is issued if the SCSI command involves data transfer from the SCSI Target to the SCSI Initiator. An SST Request_Zero_Length is issued if the command involves no data transfer.

The ST Request_To_Send, ST Request_To_Receive or SST Request_Zero_Length operation has an option payload including command control flags, addressing information, and the SCSI Command Descriptor Block (CDB).

This ST Request_To_Send, ST Request_To_Receive or SST Request_Zero_Length operation is the Execute Command service request and starts the I/O operation. The Transaction that is started is identified by the ST Initiator Sequence Identifier qualified by the ST Virtual Connection. The ST Initiator Sequence Identifier used on the ST Request_To_Send, ST Request_To_Receive, or SST Request_Zero_Length operation which begins an SST Transaction is called the Initiator Transaction Identifier and is used appropriately on all ST operations associated with the Transaction.

When the Target for the command has completed the interpretation of the command, has determined that a data transfer is required, and is prepared to request the data delivery service, it shall indicate this to the Initiator, and data shall be transferred as described below. The amount of data transferred shall be the minimum of:

– the transfer length specified in the ST Request_To_Send or Request_To_Receive operation which initiated the Transaction; or,

– the amount of data the Target requires to be transferred as specified in the SCSI CDB.

If the Transaction was initiated with an ST Request_To_Send operation, then the Target shall issue one or more ST Clear_To_Send operations for the amount of data that it is prepared to receive.

The Initiator shall then respond with one or more ST Data operations to satisfy the outstanding Clear_To_Send operations received from the Target.

When the Target is prepared to receive additional data, it may issue one or more additional ST Clear_To_Send operations to which the Initiator will respond with additional Data operations until the required amount of data has been transferred.

If the Transaction was initiated with an ST Request_To_Receive operation, then the Target shall respond to the ST Request_To_Receive operation with an ST Request_To_Send operation. Data is then transferred from the Target to the Initiator as above, except that the Initiator issues ST Clear_To_Send operations and the Target issues ST Data operations.

After all of the data has been transferred, the Target shall transmit the Execute Command service response by issuing an ST Put sequence to the SPMR defined by the connection setup protocol at an address (Bufx/Offset) computed from the Initiator Transaction Identifier for the Transaction (see 6.3). This ST Put sequence shall contain the SCSI status. If an unusual condition has been detected, then it shall also contain the SCSI REQUEST SENSE information and the SST Response information that describes the condition. The SST Status Put shall terminate the command. The SCSI logical unit determines whether additional commands will be performed in the SST I/O operation. If this is the last or only command executed in the SST I/O operation, then the SST I/O operation and the Transaction shall be terminated.

When the command is completed, returned information shall be used to prepare and return the Execute Command service confirmation information to the entity that requested the operation. The returned status shall indicate whether or not the command was successful. The successful completion of the command indicates that the SCSI device performed the desired operations with the transferred data and that the information was successfully transferred to or from the SCSI Initiator.

If the command is linked to another command, then the ST Put payload shall contain the proper SCSI status indicating that another command will be executed. The Initiator shall continue the same Transaction with an ST Request_To_Send, ST Request_To_Receive, or SST Request_Zero_Length operation beginning the next SCSI command. All SCSI commands linked in the SST I/O operation except the last shall be executed in the manner described above.

Note that when an SST Transaction executes more than one linked SCSI command, the same Initiator Transaction Identifier shall be used for all ST sequences in the SST Transaction.

The number of SST I/O operations that may be active at one time depends on the queuing capabilities of the particular SCSI devices and the number of concurrent transactions supported by the ST endpoints.

A brief way of summarizing this is that an SST Transaction consists of:

– an ST Read, Write or Nop sequence; followed by;

– an ST Put sequence to the SPMR containing status;

for each command in a single SCSI Execute service request. Note that commands may be linked.

5.3.1 Underrun Residual handling

If the amount of data the Target requires to be transferred as specified in the SCSI CDB is less than the length of the ST data transfer specified in the ST Request_To_Send or Request_To_Receive operation, either as a result of a successful or unsuccessful SCSI command completion, then this is called an Underrun Residual.

If an ST Write sequence in an SST Transaction results in an Underrun Residual, then the SST Target shall receive all data it has requested with outstanding ST Clear_To_Send operations before performing the SST Status Put operation which signifies the completion of the SCSI command.

If an ST Read sequence in an SST Transaction results in an Underrun Residual and the SST Initiator has sent ST Clear_To_Send operations for this ST Read sequence for data beyond the

final block in which the SST Target has transferred data, then the SST Initiator shall ensure that all Clear_To_Send operations have been received by the SST Target by issuing an ST End operation with an option payload that indicates the highest block number of Clear_To_Send operations issued (see 6.4). After all outstanding Clear_To_Send operations have been received, the SST Target shall respond to the ST End operation with an ST End_Ack operation as described by the ST protocol. In order to support this Clear_To_Send accounting protocol, an SST Initiator shall only issue Clear_To_Send operations for a contiguous sequence of Blocks for SST Read Transactions.

5.3.2 Third party SCSI commands

Certain third-party SCSI commands and parameters specify a 64-bit field that is defined to access other SCSI devices addressable from that port. These commands include COPY, RESERVE, and several others.

The ST protocol does not specify any form of addressing. However, it is usually run on an LLP that does specify an address format. Therefore, the addressing formats for third party SCSI commands in the SST protocol are a function of the LLP addressing format. Addressing formats are beyond the scope of this specification.

An application client requests a Task Management (TM) function when a task or some group of tasks must be aborted or terminated.

SCSI TM functions are mapped onto SST and the underlying ST protocol as shown in Table 2.

The SST Request_Zero_Length operation sent to initiate a TARGET RESET, ABORT TASK SET, CLEAR TASK SET or CLEAR ACA TM function shall be sent as the first operation in a new Transaction. As with all other Transactions, the target shall respond to TARGET RESET, ABORT TASK SET, CLEAR TASK SET and CLEAR ACA TM functions with an SST Status Put using the Initiator Transaction Identifier specified in the Request_Zero_Length operation that requested the TM function.

The SST Request_Zero_Length operation sent to initiate a TERMINATE TASK TM function shall be sent using the Initiator Transaction Identifier of the task to be terminated. Unlike other TM functions initiated with an SST Request_Zero_Length operation, the target will not respond to a TERMINATE TASK TM request with an SST Status Put operation specifically for the TERMINATE TASK TM function. However, when the requested task is terminated, the target shall perform an SST Status Put operation with the status of the terminated task.

5.4 Task management

Table 2 – SCSI Task Management function mapping

SCSI	SST	ST	Required
ABORT TASK	End sequence	End sequence	Y
TERMINATE TASK	Request_Zero_Length with Terminate Task bit set	Nop operation	N
TARGET RESET	Transaction with Target Reset bit set	Nop operation and Put operation	Y
ABORT TASK SET	Transaction with Abort Task Set bit set	Nop operation and Put operation	Y
CLEAR TASK SET	Transaction with Clear Task Set bit set	Nop operation and Put operation	Y
CLEAR ACA	Transaction with Clear Auto Contingent Allegiance bit set	Nop operation and Put operation	Y (if ACA is supported)

5.4.1 Abort Task

The SCSI Abort Task TM function may be used to terminate, prematurely, an outstanding SCSI function. The SST Abort Task protocol used to implement the SCSI Abort Task TM function ensures that in addition to aborting the SCSI function, all outstanding ST operations for that SST Transaction have either reached their destination or been discarded before the SST Abort Task protocol is complete.

Once the SST Abort Task protocol is complete, both the SST Initiator and SST Target are free to reuse the ST Sequence Identifiers, including the SST Transaction Identifier, associated with the SST Transaction.

For the purpose of the Abort Task TM function, the Abort Task Timeout shall be selected as the sum of the maximum time required for a target to perform internal functions associated with aborting the Transaction, and the maximum lifetime of an ST operation on the network. Generally, the Abort Task Timeout can be the same as the Transaction Timeout for a Transaction, since it should not take longer to abort a Transaction than it would take to complete the Transaction normally.

To abort an SST Write Transaction, the Initiator shall:

a) stop sending data;

b) issue an ST End for the Transaction to abort;

c) for every block for which a Clear_To_Send has been received, ensure that an in-order STU with ST Flags Last=b'1', Interrupt=b'1' and Silent=b'0' has been sent;

d) wait for an End_Ack for the Transaction being aborted;

e) if an End_Ack is not received within the Abort Task Timeout, determine the viability of the Virtual Connection (see 7.1) and:

1) if the Virtual Connection is not viable, then tear down the Virtual Connection; otherwise,

2) retry steps b through d an appropriate number of times after which the Write Transaction shall be declared aborted.

The Target shall issue an End_Ack if a Transaction with the Initiator Transaction Identifier specified in the End is not in progress, otherwise:

a) for each Clear_To_Send operation issued, wait for an STU with the ST flag bit Last = b'1';

b) issue an End_Ack.

To abort an SST Read Transaction, the Initiator shall:

a) stop issuing Clear_To_Send operations;

b) issue an End operation indicating the Block number of the last Clear_To_Send operation in the option payload (see 6.4);

c) wait for a STU with the ST Flag Last = b'1' for every Clear_To_Send operation issued, and an End_Ack operation for the Transaction being aborted;

d) if the operations awaited in c are not received within the Abort Task Timeout, then determine the viability of the Virtual Connection (see 7.1) and:

1) if the Virtual Connection is not viable, then tear down the Virtual Connection, otherwise;

2) retry steps b and c an appropriate number of times, after which the Read Transaction shall be declared aborted.

The Target shall issue an End_Ack if a Transaction with the Initiator Transaction Identifier specified in the End is not in progress, otherwise:

a) stop sending data;

b) for each Clear_To_Send operation reported sent by the Option Payload of the End operation, ensure that an in-order STU with ST Flags Last = b'1', Interrupt = b'1' and Silent = b'0' has been sent;

c) issue an End_Ack.

To abort an SST Request_Zero_Length Transaction, the Initiator shall:

a) issue an End;

b) wait for an End_Ack operation for the Transaction being aborted.

c) if the awaited End_Ack is not received within the Abort Task Timeout, then determine the viability of the Virtual Connection and:

1) if the Virtual Connection is not viable, then tear down the Virtual Connection, otherwise;

2) retry steps b and c an appropriate number of times, after which the Read Transaction shall be declared aborted.

The Target shall issue an End_Ack.

6 SST protocol operation formats

All multibyte quantities are formed using big-endian ordering.

6.1 ST Nop operations in SST

SST distinguishes different uses of the ST Nop operation using the 16-bit Param field as an operation code as shown in table 4. ST considers this field to be opaque.

Table 4 – SST Nop operations

Definition	Value
Request_Zero_Length	x‘0’
SSTVC_Parameters	x'1'

The 32-bit S_id field of an SST Request_Zero_Length operation contains the Initiator Transaction Identifier. ST considers this field to be opaque.

The remainder of the Nop fields (which ST considers opaque) in the SST Request_Zero_Length operation are unused. The Option Payload of the Request_Zero_Length operation contains the Option Payload for SST command operations as described in 6.2.

6.1.1 Option Payload for SSTVC_Parameters Operation

Figure 3 shows the format of the Option Payload data for the Connection Request and Connection Answer operations.

			Byte
ST_OC	LENGTH	MAX_TARGET	00-03
FLAGS			04-07
SPMR_SIZE			08-11

Figure 3 – Option Payload for Connection_Request and Connection_Answer

6.1.1.1 ST_OC

ST_OC = x’03’, indicating that the ST option payload field is valid and contains a ULP parameter.

6.1.1.2 LENGTH

LENGTH = x’0C’, indicating that this ST option payload field is 12 bytes long.

6.1.1.3 MAX_TARGET

MAX_TARGET is the maximum supported value of the TARGET field (see 6.2.3) in the Option Payload for an ST Request_To_Send, an ST Request_To_Receive, or an SST Return_Zero_Length operation. Note that the maximum number of targets is MAX_TARGET+1.

6.1.1.4 FLAGS

The FLAGS field contains a number of supported and required feature flags as shown in figure 4 and described in the following text.

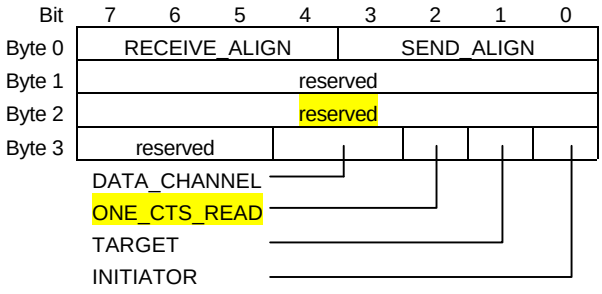

Figure 4 – Option Payload Flags for Connection operations

RECEIVE_ALIGN is the log, base 2, of the minimum required alignment of the offset parameter of any ST Data operations sent to the SST Virtual Connection endpoint specifying the

RECEIVE_ALIGN value. This necessarily implies that a SST endpoint will never issue a Clear_To_Send operation with an initial offset that does not conform to the RECEIVE_ALIGN value it specified during the SST Connection setup.

SEND_ALIGN is the log, base 2, of the minimum required alignment of the offset parameter of any ST Clear_To_Send operations sent to the SST Virtual Connection endpoint specifying the SEND_ALIGN value.

DATA_CHANNEL indicates the ST Data Channel to request for the SST SPMR.

ONE_CTS_READ = b'1' indicates that an SST Target requires an Initiator to issue only a single ST Clear_To_Send operation to cover the entire data transfer of an SST Read Transaction.

TARGET = b'1' indicates that the SST endpoint is capable of performing SCSI Target functions.

INITIATOR = b'1' indicates that the SST endpoint is capable of performing SCSI Initiator functions.

6.1.1.5 SPMR_SIZE

This field indicates the number of 512-byte Status Persistent Memory Region segments an SST Target shall request that an SST Initiator expose.

In other words, the size of the Status Persistent Memory Region requested shall be:

SPMR_SIZE * 512 bytes

6.1.1.6 SST Virtual Connection Reject reason codes

If the parameters of an SST Virtual Connection are unacceptable, then the Virtual Connection request shall be closed with an appropriate reason code as shown in table 3.

Table 3 – SST Connection Reject reason codes

Definition	Value
Single CTS Read unsupported	TBD
Target function unsupported	TBD
Busy (no resources)	TBD

Open issue: Reason codes required.

6.2 Option Payload for SST command operations

The Option Payload for ST Request_To_Send, ST Request_To_Receive, and SST Request_Zero_Length operations is organized as shown in figure 5.

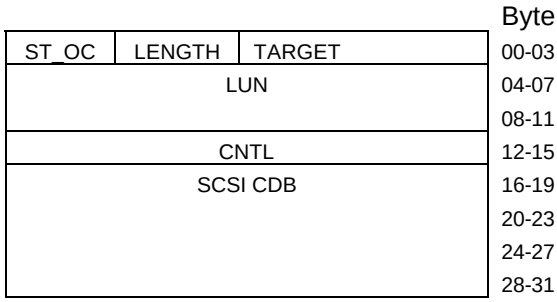

Figure 5 – Option Payload for SST command operations

6.2.1 ST_OC

ST_OC = x'03' indicates that the ST Option Payload field is valid and contains a ULP parameter

6.2.2 LENGTH

LENGTH = x'20' indicates that the ST option payload field is 32 bytes long.

6.2.3 TARGET

Selects one of a set of individual SCSI Targets addressable through a single SST Virtual Connection. The maximum allowable value of the TARGET field is established in the SST connection setup protocol.

6.2.4 LUN

The SST Logical Unit Number (LUN) is the address of the desired logical unit in the attached subsystem. The LUN field is specified by ANSI X3.270.

Open issue: Confirm the above reference.

6.2.5 CNTL

The CNTL field contains control flags and control bits as shown in figure 6.

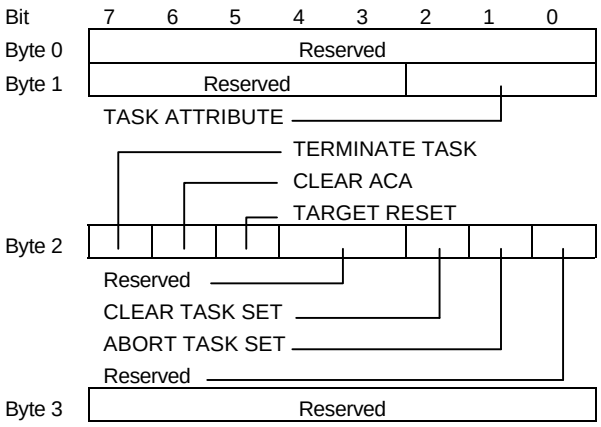

Figure 6 – CNTL field definitions for SST command operations

6.2.5.1 Task Codes, Byte 1

TASK ATTRIBUTE shall be selected as defined in table 5 (see ANSI X3.270), described here as an aid to the reader.

Table 5 – TASK ATTRIBUTE definitions for SST Command operations

Value, bits 2-0	TASK ATTRIBUTE
b'000'	SIMPLE_Q
b'001'	HEAD_OF_Q
b'010'	ORDERED_Q
b'100'	ACA_Q
b'101'	UNTAGGED
others	reserved

6.2.5.2 Task Management Flags, Byte 2

If any TM flag is set to b'1', then the CDB and CDB related CNTL flags (task codes and execution management codes) are not valid and shall be ignored. No more than one TM flag shall be set to b'1' in any SST Request_Zero_Length Transaction. No TM flags shall be set for an ST Request_To_Send or ST Request_To_Receive operation.

6.2.6 CDB

The CDB field shall contain the actual CDB to be interpreted by the addressed logical unit. The maximum CDB length shall be 16 bytes. The CDB is not valid and shall be ignored if any TM flag is set to 1. The CDB command byte (CDB byte 0) shall be the first byte following the CNTL parameter. Subsequent CDB bytes shall be stored in subsequent bytes of the option payload. Bytes beyond the last byte of the CDB are not defined by SST, shall be ignored by the Target, and may have any value.

6.3 SST Status Put sequence

The SST Status Put sequence shall be directed to an address (Bufx/Offset) which is computed as:

Status Put Bufx = address ÷ I-Bufxsize

Status Put Offset =
address modolo I-Bufxsize

where:
I-Bufxsize = $2^{\text{Initiator Bufsize}}$, and

address = (SPMR initial Bufx × I-Bufxsize + SPMR initial offset) + 512 × Transaction Initiator Identifier.

The Transaction Initiator Identifier shall be sent in the ST S_id field (opaque in the ST specification) of the ST Data operations of the SST Status Put.

The SST Status Put sequence shall consist of a single ST Data STU with the entire SST Status Put payload.

The payload of the SST Status Put sequence is shown in figure 7.

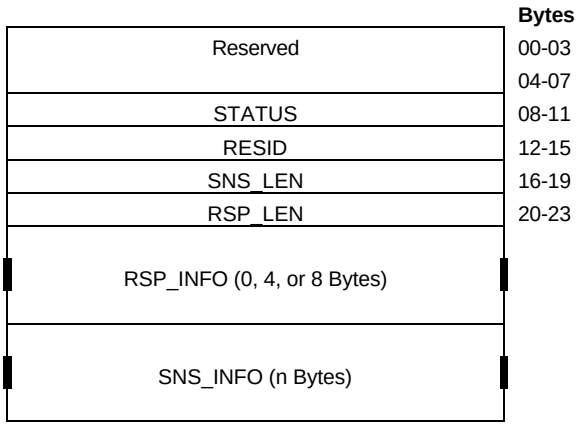

Figure 7 - Payload Parameters for an SST Status Put

6.3.1 STATUS

Figure 8 shows the format of the STATUS field.

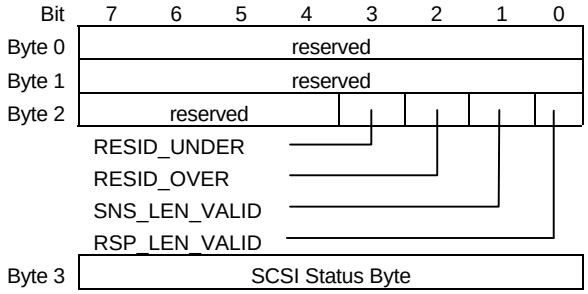

Figure 8 – STATUS field definitions for SST Status PUT

RESID_UNDER = b'1' indicates that the RESID field is valid and contains the count of bytes that were expected to be transferred, but were not transferred.

RESID_OVER = b'1' indicates that the RESID field is valid and contains the count of bytes that could not be transferred because the ST transfer length was not sufficient.

SNS_LEN_VALID = b'1' indicates that the SNS_LEN field is valid and contains the count of bytes in the SNS_INFO field.

RSP_LEN_VALID = 'b'1' indicates that the RSP_LEN field is valid and contains the count of bytes in the RSP_INFO field.

Byte 3 contains the status byte from the SCSI logical unit. The status byte codes are defined by ANSI X3.270.

6.3.2 RESID

If RESID_UNDER = b'1' or RESID_OVER = b'1', then the RESID field contains a count of the number of residual data bytes that were not transferred for this SCSI command.

If RESID_UNDER = b'1', then a transfer that did not fill the buffer to the expected displacement specified by the ST transfer length was performed and the value of RESID is a number equal to:

ST transfer length – highest offset of any byte transmitted

A condition of RESID_UNDER may not be an error for some devices and some commands.

If RESID_OVER = b'1', then the transfer was truncated because the data transfer required by the SCSI command extended beyond the transfer length specified in the ST Request_to_Send or Request_to_Receive operations. Those bytes that could be transferred without violating the ST transfer length value may be transferred. RESID is a number equal to:

(Transfer length required by command) – ST transfer length

If a condition of RESID_OVER is detected, then the termination state of the SST I/O operation is not certain. Data may or may not have been transferred and the SCSI status byte may or may not provide correct command completion information.

If the RESID_UNDER and the RESID_OVER bits are 0, then the RESID field is not meaningful and may contain any value.

6.3.3 SNS_LEN

If SNS_LEN_VALID = b'1', then the SNS_LEN field specifies the number of valid bytes of SNS_INFO.

If SNS_LEN_VALID = b'0', then the SNS_LEN field is not valid and no SNS_INFO is provided.

The SNS_LEN field shall be included in the SST Status Put.

6.3.4 RSP_LEN

If RSP_LEN_VALID = b'1', then the RSP_LEN field specifies the number of valid bytes of RSP_INFO. The number of valid bytes shall be x'00 00 00 00', x'00 00 00 04', or x'00 00 00 08'. Other values of length are reserved for future standardization.

RSP_LEN = b'00 00 00 00' specifies that no bytes of response information are being provided.

If RSP_LEN_VALID = b'0', then the RSP_LEN field is not valid and no RSP_INFO is provided.

The RSP_LEN field shall be included in the SST Status Put.

6.3.5 RSP_INFO

The RSP_INFO field contains information describing only protocol failures detected during the execution of an SST I/O operation. RSP_INFO does not contain SCSI logical unit error information since that is contained in the SNS_INFO field as specified in 6.3.6. The RSP_INFO field shall contain valid information if the Target detects any of the conditions indicated by a RSP_CODE in table 6. The format of the RSP_INFO field is shown in figure 9.

Bit	7	6	5	4	3	2	1	0
Byte 0	reserved							
Byte 1	reserved							
Byte 2	reserved							
Byte 3	RSP_CODE							

Figure 9 – RSP_INFO field definitions for SST Status PUT

The valid RSP_CODE values are shown in table 6.

Table 6 – RSP_CODE definitions for SST Status PUT

RSP_CODE definition	Value
No failure or Task Management function complete	x'00'
reserved	x'01'
SST CTS or RTS payload fields invalid	x'02'
reserved	x'03
Task Management Function Not Supported	x'04
Task Management Function Failed	x'05'
Nonexistent Target	x'06'
Busy (no resources)	x'07'
reserved	X'08' – x'FF'

6.3.6 SNS_INFO

The SNS_INFO field contains the information specified by ANSI X3.270 for presentation by the REQUEST SENSE command. The proper SNS_INFO shall be presented when the SCSI status byte of CHECK CONDITION or COMMAND TERMINATED is presented as specified by ANSI X3.270. SST implements the autosense mechanism as specified in ANSI X3.270.

6.4 Option Payload for ST End operations

When the ST End operation is required by the SST protocol to specify the block number of the last ST Clear_To_Send operation issued, the ST End operation option payload shall be as shown in figure 10.

			Byte
ST_OC	LENGTH	reserved	00-03
B_NUM			04-07

Figure 10 – Option Payload for ST End operations

6.4.1 ST_OC

ST_OC = x'03' indicates that the ST option payload field is valid and contains a ULP parameter.

6.4.2 LENGTH

LENGTH = x'08' indicates that the ST option payload field is 8 bytes long.

6.4.2 B_NUM

Indicates the ST Block number of the last ST Clear_To_Send operation sent by the data destination.

Since the SST protocol requires in-order request of ST data Blocks, the ST Block number of the last ST Clear_To_Send operation permits the data source to ensure that all ST Clear_To_Send operations for a Transaction have been received before reusing the ST Sequence Identifiers associated with the Transaction. This permits the endpoints to avoid aliasing of ST Clear_To_Send operations across Transactions when Sequence Identifiers are reused.

7 Error recovery procedures

An SST Target shall respond to any Transaction initiating operation (Request_To_Send, Request_To_Receive or Request_Zero_Length) using an ST Request_Answer with a reason code of SPMR_Not_Established until the ST Memory_Region_Available operation is received.

Note: An SPMR Not Estalibshed reason code must be assigned.

If the initial offset returned in the ST Memory_Region_Available operation for the SST SPMR is not a multiple of the Status Segment Size, 512 bytes, then an SST Target shall respond to any data transfer request using an ST Request_Answer with a reason code of SPMR Unaligned.

Note: An SPMR Unaligned reason code must be assigned.

7.1 Determining the viability of a VC

An SST Target endpoint shall implement ST Virtual Connection keep-alive sequences as described in the ST specification (see 10.4). An SST Initiator endpoint may implement ST Virtual Connection keep-alive sequences as described in the ST specification.

If an ST Virtual Connection is discovered to be dead by a failure of the keep-alive test, then all SST Transactions on the Virtual Connection shall be closed and their resources reclaimed, as well as the resources for the Virtual Connection itself.

Determining the viability of a Virtual Connection consists of performing a Request_State/Request_State_Response sequence with an appropriate number of retries.

For a very low loss medium, a single retry would be an appropriate default for any step in the SST Abort Task protocol that requires retries.

7.2 Transaction timeouts

An SST Initiator shall maintain timeouts on all outstanding device and task management operations from the time a Request_To_Send, Request_To_Receive or Request_Zero_Length operation is sent until the final Data and SST Status Put operations are received.

The Transaction Timeout shall be at least the sum of:

– the maximum time required to perform the SCSI command and transfer the associated data;

– the maximum time an ST operation can remain in the network.

The actual determination of the Transaction Timeout value is beyond the scope of this document.

This timeout ensures that no ST operations remain in the network or will be subsequently generated by an SST Target for an SST Transaction which guarantees that an SST Initiator may reuse the SST Transaction Identifier (ST I-id) associated with the SST Transaction immediately.

If a Transaction Timeout expires, then the Initiator shall first determine the viability of the SST Virtual Connection using an ST Request_State / Request_State_Response sequence. If a Request_State_Response operation is not received within an appropriate period of time, then the Request_State – Request_State_Response sequence shall be retried an appropriate number of times after which the Virtual Connection shall be declared closed and its resources reclaimed (see 6.1).

If a Request_State_Response is received, then the operation that timed out shall be aborted with the SST Abort Task protocol as described in 4.5.1, Abort Task.

An SST Target shall not maintain timeouts on outstanding Transactions. An SST Target's resources for a Transaction shall only be reclaimed when:

– the Target completes the Transaction,

– the Initiator ends the Transaction with the SST Abort Task protocol, or

– the SST Virtual Connection is ended

The requirement that the SST Initiator perform the SST Abort Task protocol for timed out operations ensures that an SST Target can reclaim resources associated with a failed Transaction.