To: T10 Technical Committee
From: Rob Elliott, HP (elliott@hp.com)
Date: 4 November 2003
Subject: T10/03-271r2 SAM-3 SPC-3 FCP-3 SAS-1.1 SRP-2 ADT Obsolete untagged tasks

Revision History

Revision 0 (7 August 2003) first revision
Revision 1 (14 September 2003) incorporated comments from September CAP WG. Removed the modified DQUE paragraph since it was made obsolete by 03-254r0 in spc3r15. Changed most I_T_L_x references to I_T_L_Q since only untagged tasks have only I_T_L nexuses. Added FCP-3 proposal to remove untagged support.
Revision 2 (4 November 2003) incorporated comments from November CAP WG.

Related Documents

sam3r09 – SCSI Architecture Model – 3 revision 9
spc3r15 – SCSI Primary Commands – 3 revision 15
fcp3r01 - Fibre Channel Protocol - 3 revision 1
sas1r01 – Serial Attached SCSI 1.1 revision 1
srp2-r00a – SCSI RDMA Protocol – 2 revision 00a
adt-r04 – Automation Drive Interface Transport Protocol revision 4

Overview

In SAM-1, transport protocols were required to specify a way to send untagged tasks.

In SAM-2, this was made optional. SRP, SPI-4 information unit mode, and SAS do not specify ways to send untagged tasks. FCP and iSCSI both overload their task attribute fields with an "untagged" value to implement this optional feature.

In SAM-3 and standards aligned with SAM-3, untagged tasks should be made obsolete.

Suggested Changes to SAM-3

1 Scope

1.1 Introduction

…

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

a) Contingent Allegiance; ~~and~~

b) The TARGET RESET task management function; and

c) untagged tasks.~~.~~

3.1.42 I_T_L_Q nexus: A nexus between a SCSI initiator port, a SCSI target port, a logical unit, and a ~~tagged~~ task (see 4.12).
[**Editor's note:** changing "tagged task" to just "task" everywhere]

3.1.xxx task tag: A field containing up to 64 bits that is a component of an I_T_L_Q nexus. See 4.11.
[**Editor's note:** changing "tag" to "task tag" everywhere.]

3.1.58 linked command: One in a series of SCSI commands processed by a single task that collectively make up a discrete I/O operation. In such a series, each command is represented by the same I_T_L_~~x~~ Q nexus, and all, except the last, have the LINK bit in the CDB CONTROL byte set to one.

4.4 The SCSI structural model

Figure 8 – Overall SCSI domain model
[Delete the “Untagged Task” box. Change “Tagged Task” box to “Task”.]

4.8 Logical units
A logical unit (see figure 14) contains:
a) A logical unit number;
  A) If access controls (see SPC-3) 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;
b) One or~~e~~ more logical unit names;
c) A device server;
d) A task manager; and
e) One or more task sets each of which may contain ~~zero or more untagged tasks or a combination of~~ zero or more ~~tagged~~ tasks ~~and zero or more untagged tasks~~.

Figure 14 — Logical unit model
[Delete the “Untagged Task” box. Change “Tagged Task” box to “Task”.]
…
A task set is composed of ~~zero or more untagged tasks or a combination of~~ zero or more ~~tagged~~ tasks (see 4.11) ~~and zero or more untagged tasks~~. ~~See 4.11 for additional restrictions on the untagged tasks and tagged tasks in a task set.~~

~~Task (see 4.11) refers to either a tagged task or an untagged task.~~ 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 5.9.1. 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-2).

4.11 Tasks
~~4.11.1 The task object~~ [removing both subsections and merging into 4.11]
~~The task object represents either a tagged task or an untagged task.~~ The composition of a task includes a definition of the work to be performed by the logical unit in the form of a command or a group of linked commands.

A ~~tagged~~ task is represented by an I_T_L_Q nexus (see 4.12) and is composed of a definition of the work to be performed by the logical unit~~,~~ and a task attribute (see 8.5). ~~An untagged task is represented by an I_T_L nexus (see 4.12) and is composed of a definition of the work to be performed by the logical unit, and implicitly a SIMPLE task attribute (see 8.5).~~

The I_T_L_Q nexus representing a ~~tagged~~ task includes a task tag ~~(see 4.11.2)~~ allowing many uniquely identified ~~tagged~~ tasks to be present concurrently in a single task set. A ~~tagged~~ task also includes one of the task attributes described in 8.5 that allows the application client to specify processing relationships between various ~~tagged~~ tasks.

~~An untagged task does not include a tag in its I_T_L nexus, thus restricting the number of concurrent untagged tasks in a single task set to one per SCSI initiator port. An untagged task is assumed to have a SIMPLE task attribute.~~

~~Every SCSI transport protocol shall support tagged tasks and may support untagged tasks. If the SCSI transport protocol upon which a SCSI device operates supports untagged tasks, the SCSI device is not required to support tagged tasks.~~

[Next paragraph is being moved from 4.11.2 to here. Change bars shown from original text in the paragraph.]

A task tag is a field containing up to 64 bits that is a component of an I_T_L_Q nexus. A SCSI initiator device assigns task tag values in each I_T_L_Q nexus in a way that ensures that the nexus uniqueness requirements ~~stated in 4.11.1~~ are met. Transport protocols may define restrictions on task tag assignment (e.g., restricting task tag length, requiring task tags to be unique per I_T nexus or per I_T_L nexus, or sharing values with other uses like a task management function association).

An I_T_L_~~x~~ 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 5.9.3). An I_T_L_~~x~~ Q nexus is unique if one or more of its components is unique within the specified time interval. ~~An untagged task shall be unique with respect to all tagged tasks in the task set.~~

~~A SCSI initiator device shall not cause the creation of more than one untagged task from a specific SCSI initiator port having identical values for the target port identifier and logical unit number.~~ A 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 tag.

~~4.11.2 Task tags~~ [merge into the previous section]

~~A tag is a field containing up to 64 bits that is a component of an I_T_L_Q nexus. A SCSI initiator device assigns tag values in each I_T_L_Q nexus in a way that ensures that the nexus uniqueness requirements stated in 4.11.1 are met.~~

4.12 The nexus object

The nexus object represents a relationship between a SCSI initiator port, a SCSI target port, optionally a logical unit~~.~~ , and optionally a task.

The nexus object may refer to any one or all of the following relationships:

a) One SCSI initiator port to one SCSI target port (an I_T nexus);
b) One SCSI initiator port to one SCSI target port to one logical unit (an I_T_L nexus);
c) One SCSI initiator port to one SCSI target port to one logical unit to one ~~tagged~~ task (an I_T_L_Q nexus); or
d) Either an I_T_L nexus or an I_T_L_Q nexus (denoted as an I_T_L_x nexus).

Table 19 — Mapping nexus to SAM-2 identifiers

I_T_L_Q	Task ~~T~~tag	4.11~~.2~~

4.14 Model for dependent logical units

…

Figure 23 — Dependent logical unit model

[Delete the "Untagged Task" box. Change "Tagged Task" box to "Task".]

5.1 The Execute Command procedure call

An application client requests the processing of a SCSI command by invoking the SCSI transport protocol services described in 5.4, the collective operation of which is conceptually modeled in the following procedure call:

Service Response =Execute Command (
IN (I_T_L_~~x~~ Q Nexus, CDB, ~~[~~Task Attribute~~]~~, [Data-In Buffer Size], [Data-Out Buffer], [Data-Out Buffer Size], [Command Reference Number]),
OUT ([Data-In Buffer], [Sense Data], [Sense Data Length], Status))

Input Arguments:
I_T_L_~~x~~ Q Nexus: ~~Either an I_T_L nexus or an~~The I_T_L_Q nexus identifying the task (see 4.12).
…
Task Attribute: A value specifying one of the task attributes defined in 8.5. ~~This argument shall not be specified for an untagged command or the second and subsequent commands in a sequence of linked commands. Untagged tasks shall implicitly have the SIMPLE attribute. The~~

~~attribute of a task that processes linked commands shall be set according to the Task Attribute argument specified for the first command in the sequence.~~ For a task that processes linked commands, the Task Attribute shall be that specified for the first command in the sequence of linked commands. The Task Attribute specified for the second and subsequent commands shall be ignored.

…

An application client requests processing of a linked command by setting the LINK bit to one in the CDB CONTROL byte as specified in 5.2. The task attribute is determined by the Task Attribute argument specified for the first command in the sequence. Upon receiving a response of LINKED COMMAND COMPLETE, an application client may issue the next command in the series through an **Execute Command** procedure call having the same I_T_L_~~x~~ Q nexus and omitting the Task Attribute argument. If the logical unit receives the next command in a series of linked
commands before completing the current command in that linked command series, the overlapped command condition described in 5.9.3 shall result.

5.3.1 Status codes

…

TASK SET FULL. This status shall be implemented ~~if the logical unit supports the creation of tagged tasks (see 4.11). This status shall not be implemented if the logical unit does not support the creation of tagged tasks~~by all logical units.

When the logical unit has at least one task in the task set for a SCSI initiator port and a lack of task set resources prevents accepting a received ~~tagged~~ task from that SCSI initiator port into the task set, TASK SET FULL shall be returned. When the logical unit has no task in the task set for a SCSI initiator port and a lack of task set resources prevents accepting a received ~~tagged~~ task from that SCSI initiator port into the task set, BUSY should be returned.

~~When the logical unit has at least one task in the task set and a lack of task set resources prevents accepting a received untagged task into the task set, BUSY should be returned.~~

The logical unit should allow at least one command in the task set for each supported SCSI initiator port that has identified itself to the SCSI target port by a SCSI transport protocol specific ~~procedure call~~manner or by the successful transmission of a command.

If the UA_INTLCK_CTRL field in the Control mode page contains 11b (see SPC-3), termination of a command with TASK SET FULL status shall cause an unit attention condition to be established for the SCSI initiator port that sent the command with an additional sense code of PREVIOUS TASK SET FULL STATUS unless a PREVIOUS TASK SET FULL STATUS unit attention condition already exists.

5.4.2 Execute Command request/confirmation SCSI transport protocol services

...

SCSI Transport Protocol Service Request:

Send SCSI Command (IN (I_T_L_~~x~~ Q Nexus, CDB, ~~[~~Task Attribute~~]~~, [Data-In Buffer Size], [Data-Out Buffer], [Data-Out Buffer Size], [Command Reference Number], [First Burst Enabled]))

Input Arguments:

I_T_L_~~x~~ Q Nexus: ~~Either an I_T_L nexus or a~~An I_T_L_Q nexus identifying the task (see 4.12). [Editor's note: it seems silly to define I_T_L_Q Nexus as just "An I_T_L_Q nexus", so added "identifying the task" to give it some meaning.]

...

SCSI Transport Protocol Service Indication:

SCSI Command Received (IN (I_T_L_~~x~~ Q Nexus, CDB, ~~[~~Task Attribute~~]~~, [Command Reference Number], [First Burst Enabled]))

Input Arguments:
I_T_L_~~x~~ Q Nexus: ~~Either an I_T_L nexus or a~~An I_T_Q nexus identifying the task (see 4.12).
...

SCSI Transport Protocol Service Response (from device server)**:**
Send Command Complete (IN (I_T_L_~~x~~ Q Nexus, [Sense Data], [Sense Data Length], Status, Service Response))

Input Arguments:
I_T_L_~~x~~ Q Nexus: ~~Either an I_T_L nexus or a~~An I_T_L_Q nexus identifying the task (see 4.12).
...

SCSI Transport Protocol Service Confirmation:
Command Complete Received (IN (I_T_L_~~x~~ Q Nexus, [Data-In Buffer], [Sense Data], [Sense Data Length], Status, Service Response))

Input Arguments:
I_T_L_~~x~~ Q Nexus: ~~Either an I_T_L nexus or a~~An I_T_L_Q nexus identifying the task (see 4.12).
...

5.4.3.2 Data-In delivery service
Request:
Send Data-In (IN (I_T_L_~~x~~ Q Nexus, Device Server Buffer, Application Client Buffer Offset, Request Byte Count))

Argument descriptions:
I_T_L_~~x~~ Q Nexus: ~~Either an I_T_L nexus or a~~An I_T_L_Q nexus identifying the task (see 4.12).
...

Confirmation:
Data-In Delivered (IN (I_T_L_~~x~~ 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.

Argument descriptions:
I_T_L_~~x~~ Q Nexus: ~~either an I_T_L nexus or a~~An I_T_L_Q nexus identifying the task (see 4.12).
...

5.4.3.3 Data-Out delivery service
Request:
Receive Data-Out (IN (I_T_L_~~x~~ Q Nexus, Application Client Buffer Offset, Request Byte Count, Device Server Buffer))

Argument descriptions:
I_T_L_~~x~~ Q Nexus: ~~either an I_T_L nexus or a~~An I_T_L_Q nexus identifying the task (see 4.12).
...

Confirmation:
Data-Out Received (IN (I_T_L_~~x~~ 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.

Argument descriptions:
I_T_L_~~x~~ Q Nexus: ~~either an I_T_L nexus or a~~An I_T_L_Q nexus identifying the task (see 4.12).

7 Task management functions
7.1 Introduction

…

Argument descriptions:
Nexus: An I_T Nexus, I_T_L Nexus, or I_T_L_Q Nexus (see 4.12) identifying the task(s) affected by the task management function.
I_T Nexus: ~~A SCSI initiator port and SCSI target port nexus (see 4.12)~~ An I_T nexus, identifying zero or more tasks.
I_T_L Nexus: ~~A SCSI initiator port, SCSI target port, and logical unit nexus (see 4.12)~~ An I_T_L Nexus, identifying zero or more tasks.
I_T_L_Q Nexus: ~~A SCSI initiator port, SCSI target port, logical unit, and tag nexus (see 4.12)~~ An I_T_L_Q nexus, identifying zero or one tasks.
[Editor's note: trying to make these argument descriptions more like those in the command section. The current definitions just restate the glossary definitions, not their reason for use as arguments here]

5.9.3 Overlapped commands

An overlapped command occurs when a task manager detects the use of a duplicate I_T_L_~~x~~Q nexus (see 4.11~~.1~~) in a command before a pending task holding that I_T_L_~~x~~Q nexus completes its task lifetime (see 5.5). Each SCSI transport protocol standard shall specify whether or not a task manager is required to detect overlapped commands.

7.2 ABORT TASK

This function shall be supported by all logical units ~~if it supports tagged tasks and may be supported by a logical unit if it does not support tagged tasks~~.

7.5 CLEAR TASK SET

~~This function shall be supported by all logical units, except in the following cases, when support for this function is optional:~~
~~a) The logical unit does not support tagged tasks (see 4.11); or~~
~~b) The logical unit supports the basic task management model (see 8.2).~~
This function may be supported by logical units supporting the basic task management model (see 8.2) and shall be supported by all other logical units.

7.8 Task management SCSI transport protocol services

…

Since the nexus used by all task management functions except ABORT TASK and QUERY TASK does not contain a task tag to uniquely identify the ~~transaction~~task, there may be no way for an application client to associate a confirmation with a request. A SCSI transport protocol that does not provide such an association should not allow a SCSI initiator port to have more than one pending task management request per I_T_L nexus.

8 Task Set Management

8.2 ~~Controlling task set management~~Task management models

8.2.1 Task management model overview [moving text into new subsection]

The Control mode page (see SPC-2) contains fields that specify particular task set management behaviors. The standard INQUIRY data ~~CmdQue~~CMDQUE bit and BQUE bit (see SPC-~~2~~3) indicate~~s~~ support for ~~tagged tasks (command queuing). One specific combination of task set management behaviors is identified as~~ the basic or full task management models.

~~Support for the basic task management model is indicated by values returned in the CMDQUE and BQUE bits in the standard INQUIRY data (see SPC-2).~~

8.2.2 Basic task management model [moving text into new subsection]

The basic task management model requires the following task set management behaviors:

a) The only task attribute supported shall be SIMPLE;

[note: 03-351 further changes item a) to allow either all SIMPLE or all ORDERED to be supported, which is how SBP-2 works]

b) The device server may reorder the actual processing sequence of tasks in any manner. Any data integrity exposures related to task sequence order shall be explicitly handled by the application client using the appropriate commands (i.e., they shall be handled as if the QUEUE ALGORITHM MODIFIER field is set to 1h in the Control mode page (see SPC-3));

c) All the tasks in the task set shall be aborted when a CHECK CONDITION status is returned ~~or when an ACA condition is established~~ for any task (i.e., they shall be handled as if the QERR field is set to 01b in the Control mode page (see SPC-3));

~~d) It shall not be possible to disable tagged queuing;~~

d) If the Control mode page is supported, the QUEUE ALGORITHM MODIFIER field shall be set to 1h and the QERR field shall be set to 01b; and

e) ~~Support for t~~The CLEAR TASK SET task management function ~~is optional~~may be supported.

8.2.3 Full task management model [new subsection with new text]

The full task management model requires the following task set management behaviors:

a) The SIMPLE task attribute 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-3) shall control the processing sequence of tasks having the SIMPLE task attribute;

d) The QERR field in the Control mode page (see SPC-3) shall control aborting of tasks when a CHECK CONDITION status is returned for any task; and

e) The CLEAR TASK SET task management function shall be supported.

Suggested changes to SPC-3

3.1.111 task set: A group of tasks within a logical unit, whose interaction is dependent on the task management (queuing) and ACA rules. See SAM-2 and the Control mode page (see 7.4.6).

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

4.5.6 Sense key and sense code definitions

...

Table 28 and Table C.1 - ASC and ASCQ assignments

4Dh NNh D T L PWR OMA E B K V F TAGGED OVERLAPPED COMMANDS (NN = ~~QUEUE~~ TASK TAG)

5.5.1 Reservations overview

…

Multiple persistent reserve service actions may be present in the task set at the same time. The order of execution of such service actions is defined by the ~~tagged queuing restrictions, if any~~task set management requirements (see SAM-3), but each is executed as a single indivisible command without any interleaving of actions that may be required by other reservation commands.

5.5.2.7.4.2 Failed persistent reservation preempt

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

6.4.2 Standard INQUIRY data

…

~~When the CMDQUE bit is set to zero, the BQUE bit shall have the following meaning. A BQUE bit set to zero indicates that the device does not support tagged tasks (command queuing) for this logical unit. A BQUE bit set to one indicates that the device supports, for this logical unit, the basic task management model defined by SAM-2.~~

~~…~~

~~A command queuing (CMDQUE) bit set to one indicates that the device supports tagged tasks (command queuing) for this logical unit (see SAM-2). A CMDQUE bit set to zero indicates the device server may support tagged tasks for this logical unit (see the BQUE bit, above). Table 79 summarizes the relationship of the BQUE and CMDQUE bits.~~

The command queuing (CMDQUE) and basic queuing (BQUE) bits indicate whether the logical unit supports the full task management model or the basic task management model as described in table 79.

Table 79 — Relationship of BQUE and CMDQUE bits

BQUE	CMDQUE	**Description**
0	0	~~No command queuing of any kind supported.~~Obsolete.
0	1	~~Command queuing with all types of task tags supported.~~Full task management model supported (see SAM-3).
1	0	Basic task management model supported (see SAM-~~2~~3).
1	1	Illegal combination of BQUE and CMDQUE bits.

7.4.6 Control mode page

The Control mode page (see table 223) provides controls over several SCSI features that are applicable to all device types such as ~~tagged queuing~~ task set management and error logging.

…

Table C.1 - ASC and ASCQ assignments

4Dh NNh D T L PWR OMA E B K V F TAGGED OVERLAPPED COMMANDS (NN = ~~QUEUE~~ TASK TAG)

Suggested changes to FCP-3

Introduction

...

Fibre Channel Protocol for SCSI, Second Version (FCP-2) is part of the SCSI family of standards developed by T10 to facilitate the use of the SCSI command sets for many different types of devices across many different types of physical interconnects. The architectural model for the family of standards is ~~NCITS Project 1157D, *Information Technology - SCSI Architecture Model - 2 (SAM-2)*~~*INCITS 1567-D, SCSI Architecture Model - 3 (SAM-3)*.

[Editor's note: Change all SAM-2 references to SAM-3 throughout the document]

3.1.57 tag: The initiator-specified component of a task identifier that uniquely identifies one task among the several tasks coming from an initiator to a logical unit. The fully qualified exchange identifier performs the function of the SCSI task tag in this standard. See 5.1 and SAM-~~2~~3.

9.1.2.3 TASK ATTRIBUTE

The TASK ATTRIBUTE field contains values that specify the task attribute (see SAM-2) associated with the CDB, as shown in table 20.

Table 20 - TASK ATTRIBUTE field values

Value, bits 2-0	task attribute
000b	SIMPLE
001b	HEAD OF QUEUE
010b	ORDERED
100b	ACA
101b	~~UNTAGGED~~Obsolete
others	Reserved

SIMPLE requests that the task be managed according to the rules for a SIMPLE task attribute.

HEAD OF QUEUE requests that the task be managed according to the rules for a HEAD OF QUEUE task attribute.

ORDERED requests that the task be managed according to the rules for an ORDERED task attribute. Mechanisms to assure delivery of commands to a device server in the correct order are described in 4.3.

ACA requests that the task be managed according to the rules for an automatic contingent allegiance (ACA) task attribute.

~~**UNTAGGED** requests that the task be managed according to the rules for an untagged task. Only one untagged task shall exist for each logical unit / initiator pair. Requesting a second untagged command for the same logical unit / initiator pair shall be treated as an overlapped command. See SAM-2.~~

12.4.1.5 FCP_RSP IU Recovery

This procedure shall be used only by FCP devices that have agreed to Sequence level recovery.

An error in transmitting an FCP_RSP IU is detected if:

a) the ACC for the REC Extended Link Service indicates that an FCP_RSP IU was sent by the target and no FCP_CONF IU was requested (i.e., E_STAT indicates that the Exchange is complete), but the initiator has not yet received the FCP_RSP IU; or

b) the ACC for the REC Extended Link Service indicates that an FCP_RSP IU Sequence was sent by the target and an FCP_CONF IU was requested (i.e., E_STAT indicates that the Exchange is not complete, that the initiator has initiative, and that, if the data transfer was from the initiator to the target, the data transfer indicates that all of the bytes expected to be transferred by the command have been transferred.)

When an error in transmitting an FCP_RSP IU is detected, the initiator shall issue an SRR FC-4 Link Service frame in a new Exchange to request retransmission of the FCP_RSP IU. The target shall first transmit the ACC for the SRR, then shall retransmit the FCP_RSP IU in a new Sequence.

An Exchange carrying a command that was terminated by a CHECK CONDITION requesting an FCP_CONF IU prior to transferring data may have the same REC values as an Exchange carrying a command having an FCP_XFER_RDY IU not received by the initiator. For a command transferring data from the initiator to the target with a non-zero FCP_DL, the parameters for the SRR shall indicate that an FCP_XFER_RDY IU is expected from the target. The target is aware of the actual present state of the transfer and response and shall either retry the FCP_XFER_RDY IU or, if the actual data transfer length for the command was zero, retry the FCP_RSP IU.

~~For non-tagged command queuing operations, the target shall retain the Exchange information until:~~

~~a) the next FCP_CMND IU has been received for that LUN from the same initiator;~~

~~b) an FCP_CONF IU is received for the Exchange; or~~

~~c) after RR_TOV times out.~~

~~For tagged command queuing operations, t~~The target shall retain Exchange information until:

a) an FCP_CONF IU is received for the Exchange; or

b) after RR_TOV times out.

A.1 Definition of procedure terms

...

Table A.2 - Procedure Terms

…

task tag SAM-2 this standard

…

A.3 Application client SCSI command services

The SCSI command services shall be requested by the application client using a procedure call defined as:

Service response =Execute Command (IN (I_T_L_~~x~~ Q Nexus, CDB, ~~[~~Task Attribute~~]~~, [Data-In Buffer Size], [Data-Out Buffer], [Data-Out Buffer Size], Autosense Request, [Command Reference Number]), OUT ([Data-In Buffer], [Sense Data], Status))

[Note: Autosense request needs to be deleted and [Sense Data Size] needs to be added in A.3 and A.4. Those changes are not related to this proposal so are not marked.]

A.4 Send SCSI command service

The send SCSI command service is a four step confirmed service that provides the means to transfer a command data block to a device server.

Processing the execute command procedure call for a send SCSI command service shall be composed of the 4 step confirmed service shown in table A.3.

Table A.3 - Processing of send SCSI command service procedure

Send SCSI Command (IN (I_T_L_~~x~~ Q Nexus, CDB, ~~[~~Task Attribute~~]~~, [Data-In Buffer Size], [Data-Out Buffer], [Data-Out Buffer Size], Autosense Request, [Command Reference Number]))

…

SCSI Command Received (IN (I_T_L_~~x~~ Q Nexus, ~~[~~Task Attribute~~]~~, CDB, Autosense Request, [Command Reference Number]))

…

Send Command Complete (IN (I_T_L_~~x~~ Q Nexus, [Sense Data], Status, Service Response))

…

Command Complete Received (IN (I_T_L_~~x~~ Q Nexus, [Data-In Buffer], [Sense Data], Status, Service Response))

A.5.2 Data-in delivery service

The data-in delivery service is a two step confirmed service (see table A.4) that provides the means to transfer a parameter list or data from a device server to an initiator~~.~~.

…

Table A.4 - Processing of data-in delivery service procedure

Send Data-In (IN (I_T_L_~~x~~ Q Nexus, Device Server Buffer, Application Client Buffer Offset, Request Byte Count))

…

Data-In Delivered (IN (I_T_L_~~x~~ Q Nexus))

A.5.3 Data-out delivery service

The data-out delivery service is a two step confirmed service (see table A.5) that provides the means to transfer a parameter list or data from an initiator to a device server.

Table A.5 - Processing of data-out delivery service procedure

Receive Data-Out (IN (I_T_L_~~x~~Q Nexus, Application Client Buffer Offset, Request Byte Count, Device Server Buffer))
Data-Out Received (IN (I_T_L_~~x~~Q Nexus))

A.6 Task management services
The task management function shall be requested from the application client using a procedure call defined as:
Service response = Function name (IN (~~I_T_x_x n~~Nexus))
[Note: Neither SAM-3 nor FCP-3 defines I_T_x_x as a term. SAM-3 just uses Nexus for this.]

Suggested changes to SAS-1.1

3.1.60 I_T_L_Q nexus: A nexus between a SCSI initiator port, a SCSI target port, a logical unit, and a ~~tagged~~ task. This relationship extends the prior I_T nexus or I_T_L nexus.
3.1.79 nexus: A relationship between a SCSI initiator port and a SCSI target port that may extend to a logical unit and a ~~tagged~~ task. See SAM-3.

9.2.1 SSP frame format
…
The TAG field contains a value that allows the SSP initiator port to establish a context for commands and task management functions.

For COMMAND and TASK frames, the SSP initiator port shall set the TAG field to a value that is unique for the I_T nexus established by the connection (see 7.12). An SSP initiator port shall not reuse the same tag when transmitting COMMAND or TASK frames to different LUNs in the same SSP target port; it may reuse a tag when transmitting frames to different SSP target ports. The TAG field in a COMMAND frame contains the task tag defined in SAM-3. The TAG field in a TASK frame does not correspond to a SAM-3 task tag, but corresponds to an SAM-3 association (see 10.2.1). The tag space used in the TAG fields is shared across COMMAND and TASK frames (e.g., if a tag is used for a COMMAND frame, it is not simultaneously used for a TASK frame).

For DATA, XFER_RDY, and RESPONSE frames, the SSP target port shall set the TAG field to the tag of the command or task management function to which the frame pertains.

10.2.1 SCSI transport protocol services
10.2.1.1 SCSI transport protocol services overview
An application client requests the processing of a SCSI command by invoking SCSI transport protocol services, the collective operation of which is conceptually modeled in the following remote procedure call (see SAM-3):
Service response = Execute Command (IN (I_T_L_~~x~~Q Nexus, CDB, ~~[~~Task Attribute~~]~~, [Data-In Buffer Size], [Data-Out Buffer], [Data-Out Buffer Size], [Autosense Request]), OUT ([Data-In Buffer], [Sense Data], Status))

10.2.1.3 SCSI Command Received transport protocol service
…
If an SSP target port calls SCSI Command Received () with a TAG already in use by another SCSI command (i.e., an overlapped command) or by a SCSI task management function, the device server responses are defined in SAM-3.

10.2.1.2 Send SCSI Command transport protocol service
An application client uses the Send SCSI Command transport protocol service request to request that an SSP initiator port transmit a COMMAND frame.
Send SCSI Command (IN (I_T_L_~~x~~Q Nexus, CDB, ~~[~~Task Attribute~~]~~, [Data-In Buffer Size], [Data-Out Buffer], [Data-Out Buffer Size], [Autosense Request], [Command Reference Number], [First Burst Enabled]))
…
I_T_L_~~x~~Q nexus
I_T_L_Q nexus, where:

a) I_T used to select a connection;
b) L used to set the LOGICAL UNIT NUMBER field in the COMMAND frame header; and
c) Q used to set the TAG field in the COMMAND frame header.
…
~~[~~Task Attribute~~]~~ Used to set the TASK ATTRIBUTE field in the COMMAND frame.
…

10.2.1.3 SCSI Command Received transport protocol service
An SSP target port uses the SCSI Command Received transport protocol service indication to notify a device server that it has received a COMMAND frame.
SCSI Command Received (IN (I_T_L_~~x~~ Q Nexus, CDB, ~~[~~Task Attribute~~]~~, [Autosense Request], [Command Reference Number]))
…
I_T_L_~~x~~ Q nexus
I_T_L_Q nexus, where:
a) I_T indicated by the connection;
b) L indicated by the LOGICAL UNIT NUMBER field in the COMMAND frame header; and
c) Q indicated by the TAG field in the COMMAND frame header.
…
~~[~~Task Attribute~~]~~ From the TASK ATTRIBUTE field in the COMMAND frame.
…
10.2.1.4 Send Command Complete transport protocol service
A device server uses the Send Command Complete transport protocol service response to request that an SSP target port transmit a RESPONSE frame.
Send Command Complete (IN (I_T_L_~~x~~ Q Nexus, [Sense Data], Status, Service Response))
…
I_T_L_~~x~~ Q nexus
I_T_L_Q nexus, where:
a) I_T used to select a connection;
b) L used to set the LOGICAL UNIT NUMBER field in the RESPONSE frame header; and
c) Q used to set the TAG field in the RESPONSE frame header.
…
10.2.1.5 Command Complete Received transport protocol service
An SSP initiator port uses the Command Complete Received transport protocol service confirmation to notify an application client that it has received a response for its COMMAND frame (e.g., a RESPONSE frame or a NAK).
Command Complete Received (IN (I_T_L_~~x~~ Q Nexus, [Data-In Buffer], [Sense Data], Status, Service Response))
…
I_T_L_~~x~~ Q nexus
I_T_L_Q nexus, where:
a) I_T indicated by the connection;
b) L indicated by the LOGICAL UNIT NUMBER field in the RESPONSE frame header or COMMAND frame header; and
c) Q indicated by the TAG field in the RESPONSE frame header or COMMAND frame header.

10.2.1.6 Send Data-In transport protocol service
A device server uses the Send Data-In transport protocol service request to request that an SSP target port transmit a DATA frame.
Send Data-In (IN (I_T_L_~~x~~ Q Nexus, Device Server Buffer, Application Client Buffer Offset, Request Byte Count))
…
I_T_L_~~x~~ Q nexus
I_T_L_Q nexus, where:
a) I_T used to select a connection;
b) L used to set the LOGICAL UNIT NUMBER field in the DATA frame header; and
c) Q used to set the TAG field in the DATA frame header.

10.2.1.7 Data-In Delivered transport protocol service
An SSP target port uses the Data-In Delivered transport protocol service indication to notify a device server of the results of transmitting a DATA frame.
Data-In Delivered (IN (I_T_L_~~x~~ Q Nexus))
…
I_T_L_~~x~~ Q nexus
I_T_L_Q nexus, where:
a) I_T indicated by the connection;
b) L indicated by the LOGICAL UNIT NUMBER field in the DATA frame header; and
c) Q indicated by the TAG field in the DATA frame header.

10.2.1.8 Receive Data-Out transport protocol service
A device server uses the Receive Data-Out transport protocol service request to request that an SSP target port transmit an XFER_RDY frame.
Receive Data-Out (IN (I_T_L_~~x~~ Q Nexus, Application Client Buffer Offset, Request Byte Count, Device Server Buffer))
…
I_T_L_~~x~~ Q nexus
I_T_L_Q nexus, where:
a) I_T used to select a connection;
b) L used to set the LOGICAL UNIT NUMBER field in the XFER_RDY frame header; and
c) Q used to set the TAG field in the XFER_RDY frame header.

10.2.1.9 Data-Out Received transport protocol service
An SSP target port uses the Data-Out Received transport protocol service indication to notify a device server of the result of transmitting an XFER_RDY frame (e.g., receiving DATA frames in response).
Data-Out Received (IN (I_T_L_~~x~~ Q Nexus))
…
I_T_L_~~x~~ Q nexus
I_T_L_Q nexus, where:
a) I_T indicated by the connection;
b) L indicated by the LOGICAL UNIT NUMBER field in the XFER_RDY frame header; and
c) Q indicated by the TAG field in the XFER_RDY frame header.

Suggested changes to SRP-2

A.4 Procedure objects
Table A.2 - Procedure objects
…
I_T_L_~~x~~ Q nexus this standard

A.5 Application client SCSI command services
A.5.1 Application client SCSI command services overview
The SCSI command services shall be requested by the application client using a procedure call defined as:
Execute Command (IN (I_T_L_~~x~~ Q nexus, command descriptor block, ~~[~~task attribute~~]~~, [data-in buffer size], [data-out buffer], [data-out buffer size], autosense request), OUT ([data-in buffer], [sense data], status, service response))
A.5.2 Send SCSI command service
Send SCSI command (IN (I_T_L_~~x~~ Q nexus, command descriptor block, ~~[~~task attribute~~]~~, [data-in buffer size], [data-out buffer], [data-out buffer size], autosense request))
SCSI command received (IN (I_T_L_~~x~~ Q nexus, command descriptor block, ~~[~~task attribute~~]~~, autosense request))
Send command complete (IN (I_T_L_~~x~~ Q nexus, [sense data], status, service response))
Command complete received (IN (I_T_L_~~x~~ Q nexus, [data-in buffer], [sense data], status, service response))

A.6 Device server SCSI command services

A.6.1 Device server SCSI command services overview

The SCSI data buffer movement services shall be requested from the device server using a procedure call defined as:

Move data buffer (IN (I_T_L_~~x~~ Q nexus, device server buffer, application client buffer offset, request byte count)).

Either data-in delivery, data-out delivery, both data-in and data-out delivery, or neither data delivery may be used while processing one command. If both are used, the device server shall combine the data-in and data-out service responses into one service response.

A.6.2 Data-out delivery service

Receive data-out (IN (I_T_L_~~x~~ Q nexus, application client buffer offset, request byte count, device server buffer))

See (see XX). [this line obviously needs work, but is not pertinent to this proposal so is ignored]

Data-out received (IN (I_T_L_x nexus))

A.6.3 Data-in delivery service

Send data-in (IN (I_T_L_~~x~~ Q nexus, device server buffer, application client buffer offset, request byte count))

See (see XX). [this line obviously needs work, but is not pertinent to this proposal so is ignored]

Data-In delivered (IN (I_T_L_~~x~~ Q nexus))

Suggested changes to ADT

[Editor's note: As of revision 4, ADT has no use of I_T_L_x nexus or [task attribute].]

~~**3.1.11 I_T nexus:** A nexus that exists between a SCSI initiator port and a SCSI target port.~~

~~**3.1.12 I_T_L nexus:** A nexus that exists between a SCSI initiator port, a SCSI target port, and a logical unit. This relationship replaces the prior I_T nexus.~~

~~**3.1.13 I_T_L_Q nexus:** A nexus between a SCSI initiator port, a SCSI target port, a logical unit, and a queue tag following the successful receipt of a queue tag. This relationship replaces the prior I_T nexus or I_T_L nexus.~~

3.1.11 I_T nexus: A nexus that exists between a SCSI initiator port and a SCSI target port.

3.1.12 I_T_L nexus: A nexus that exists between a SCSI initiator port, a SCSI target port, and a logical unit. This relationship extends the prior I_T nexus.

3.1.13 I_T_L_Q nexus: A nexus between a SCSI initiator port, a SCSI target port, a logical unit, and a task. This relationship extends the prior I_T nexus or I_T_L nexus.

7 Transport layer

7.1 SCSI Encapsulation

7.1.1 SCSI encapsulation overview

SCSI frames contain information required to implement the SCSI protocol. Each SCSI frame shall include a payload header at the start of the payload to carry other information that is required to perform the selected function. All SCSI commands use the ~~simple queue~~basic task management [if this is really intended] model described in SAM-2. The X_ORIGIN bit in the ADT Frame Header implies the SCSI initiator port and SCSI target port identities. The EXCHANGE ID value from the ADT Frame Header takes on the roll of the ~~Queue Tag~~task tag from SAM-2. The LUN is included in the SCSI Request IU payload header.