To: T10 Technical Committee
From: Rob Elliott (elliott@hp.com) and Mallikarjun Chadalapaka, HP (mallikarjun.chadalapaka@hp.com)
Date: 15 January 2007
Subject: 06-341r1 SAM-4 Response Fence for protocol services

Revision history
Revision 0 (11 July 2006) First revision
Revision 1 (15 January 2007) Incorporated comments from July 2006 CAP WG

Related documents
T10/sam4r08 - SCSI Architecture Model - 4 (SAM-4) revision 8
T10/04-072r0 SAM-3 SPC-3 Unit attention for status delivery problems (Rob Elliott, HP)
IETF/RFC 3270 - iSCSI
IETF/RFC 3783 - SCSI Command Ordering Considerations With iSCSI
IETF/draft-ietf-ips-iscsi-impl-guide-03.txt - iSCSI Implementor's Guide revision 3 (by Mallikarjun Chadalapaka, HP). Available as http://www.ietf.org/internet-drafts/draft-ietf-ips-iscsi-impl-guide-03.txt through March 2007.

Overview
Some SCSI transport protocols support multiple concurrent communication paths for an I_T nexus at one time:

a) iSCSI supports multiple connections per session (i.e., multiple connections between an initiator port and a target port);
b) Serial Attached SCSI (SAS) supports multiple connections between a wide initiator port and a wide target port.

Many SCSI transport protocols allow multiple frames to be in flight over a single communications path (i.e., after sending one frame, send the next one without waiting for an ACK):

a) iSCSI, where frames are sent without waiting for any kind of ACK (they do arrive in-order since TCP/IP is ordered);
b) Serial Attached SCSI (SAS), for non-interlocked DATA frames, where frames are sent without waiting for ACKs for the previous frames; and
c) Fibre Channel Protocol (FCP) over Fibre Channel class 3, where frame delivery is not guaranteed and frames may arrive out-of-order.

Information sent over a communications path includes:

a) commands per the Send SCSI Command () transport protocol service (e.g., COMMAND frames)
a) task management functions per the Send Task Management Request () transport protocol service (e.g, COMMAND or TASK frames);
b) data per the Send Data-In () and Receive Data-Out () transport protocol services (including both DATA frames and XFER_RDY/R2T PDUs);
c) command status per the Send Command Complete () transport protocol service (e.g., RESPONSE frames); and
d) task management responses per the Task Management Function Executed () transport protocol service (e.g., REPSONSE frames).

SAM-4 does not define whether protocol service calls:

a) complete immediately;
b) complete after the information has been sent (e.g., after sending the frame or PDU); or
c) complete after the information has been confirmed to be delivered (e.g., after receiving an ACK in SAS, or a PDU with an ExpStatSN value higher than that of the RESPONSE PDU's StatSN value in iSCSI).

Nor is there any output value through which these calls report success or failure (e.g., there is no "Result = Send Command Complete ()").

With multiple connections for an I_T nexus, multiple types of information can be sent concurrently (with restrictuinos). For example, an initiator port can send two commands at the same time. iSCSI requires all the

PDUs for a command remain on the original connection; PDUs for different commands may occur on different connections.

SAM-4 provides no guarantees whatsoever about the ordering provided by SCSI transport protocols:

> "**4.3.3 Request/Response ordering**
> Request or response transactions are said to be in order if, relative to a given pair of sending and receiving SCSI ports, transactions are delivered in the order they were sent.
>
> A sender may require control over the order in which its requests or responses are presented to the receiver (e.g., the sequence in which requests are received is often important whenever a SCSI initiator device issues a series of commands with the ORDERED attribute to a logical unit as described in clause 8). In this case, the order in which these commands are completed, and hence the final state of the logical unit, may depend on the order in which these commands are received. The SCSI initiator device may develop knowledge about the state of pending commands and task management functions and may take action based on the nature and sequence of SCSI target device responses (e.g., a SCSI initiator device should be aware that further responses are possible from an aborted command because the command completion response may be delivered out of order with respect to the abort response).
>
> The manner in which ordering constraints are established is vendor specific. An implementation may delegate this responsibility to the application client (e.g., the device driver). In-order delivery may be an intrinsic property of a service delivery subsystem or a requirement established by the SCSI transport protocol standard.
>
> The order in which task management requests are processed is not specified by the SCSI architecture model. The SCSI architecture model does not require in-order delivery of such requests or processing by the task manager in the order received. To guarantee the processing order of task management requests referencing a specific logical unit, an application client should not have more than one such request pending to that logical unit.
>
> To simplify the description of behavior, the SCSI architecture model assumes in-order delivery of requests or responses to be a property of a service delivery subsystem. This assumption does not constitute a requirement. The SCSI architecture model makes no assumption about and places no requirement on the ordering of requests or responses for different I_T nexuses."

Rather than continuing to provide no guarantees, SAM-4 could start to allow application clients using transport protocols that do ensure ordering to take advantage of that knowledge and start using ORDERED task attributes, send more information concurrently, keep multiple frames in flight, etc.

Command/task management function ordering

Although it doesn't matter for commands with SIMPLE task attributes, an application client cannot send commands with ORDERED task attributes sensibly unless the initiator port sends the commands in the order in which Send SCSI Command () is called.

Since there are no task attributes for task management functions, the ordering of those is never technically necessary. SAM-4 already warns that an application client should only run one request at a time to be safe.

The ordering between commands and task management functions is unclear. However, it would not be very usable if the initiator port let a task management function jump past commands it has not yet sent; that would imply there is no protocol service mechanism to ensure commands are aborted.

This could be improved by:

a) mandate that initiator ports send all commands and task management functions in the order in which the application client sent them; or
b) add a Request Fence argument that instructs the initiator port to finish sending all pending commands and task management functions before sending this one

It may be appropriate to extend the Command Reference Number in FCP-4 to also apply to task management functions:

a) Change it from reserved to have meaning if the FCP_CMND frame is delivering a task management function
b) Expand the CRN argument definition in SAM-4 to cover task management functions
c) Expand the CRN argument definition in SAM-4 to represent asking the initiator port to guarantee ordered delivery to the target (not just ordered sending by the initiator, but ordered reception by the target)

Response ordering

SAM-4 warns that responses may still arrive from aborted commands. This is inconvenient for an initiator port that wants to reuse the tags - how long must it wait?

SAS includes some rules on tag reuse - the initiator port must not reuse a tag until it has evidence that the target port received the ACK for the RESPONSE frame for that command. This is harder to define for commands aborted that do not deliver RESPONSE frames.

iSCSI and Response Fence

Although it is not required, iSCSI ended up providing stronger ordering guarantees than most SCSI transport protocols:

a) commands and task management function requests are ordered session-wide (i.e., across all connections in a session)(I to T) using the Command Sequence Number (CmdSN);
b) data is ordered per command or part of a command (I to T and T to I) using a Data Sequence Number (DataSN);
c) R2T frames are ordered per command (T to I) using an R2T Sequence Number (R2TSN);
d) status and task management function responses are ordered connection-wide (T to I) using the Status Sequence Number (StatSN).

It does not, however, always provide feedback to a target port that status has arrived at the initiator. There are no ACKs delivered in the I to T direction in reply to the Response PDU. The next time the initiator chooses to send a frame to the target, it will include an updated ExpStatSN that serves as a NAK. However, there is no guarantee that the initiator will send another frame to the target.

To provide consistent ordering, the device server needs to ensure that the target port has sent certain previously requested RESPONSE frames before it sends a new RESPONSE frame. Examples:

a) Task management functions that abort commands for multiple I_T nexuses (e.g., CLEAR TASK SET and LOGICAL UNIT RESET). When the device server requests a RESPONSE frame be sent for one of these, it must not arrive at the initiator port before any RESPONSE frames for the tasks that were supposed to being cleared;
b) The PERSISTENT RESERVE OUT command with a PREEMPT AND ABORT service action. The RESPONSE frame for the command must not arrive until all the affected commands are aborted.
c) A command returning a CHECK CONDITION/UNIT ATTENTION condition after commands were aborted on that I_T nexus. The command completions should not show up at the initiator after the unit attention.
d) A command returning status of CHECK CONDITION if ACA is enabled. RESPONSE frames containing status other than ACA ACTIVE should not show up at the initiator after the RESPONSE frame indicating an ACA was established.
e) The CLEAR ACA task management function. RESPONSE frames containing ACA ACTIVE should not show up at the initiator after the RESPONSE frame for the CLEAR ACA.

The device server/task manager could guarantee this by single-threading calls to Send Command Complete () and Task Management Function Executed (). Several years ago, 04-072r0 proposed adding a Delivery Result output argument to Send Command Complete () [and Task Management Function Executed ()], indicating whether or not the RESPONSE frame was successfully delivered (e.g., ACKed). With that, the device server could simply wait for the previous protocol service(s) of interest to complete before issuing the critical one.

When originally proposed, the CAP WG didn't like 04-072 - its main goal was to allow the device server to create a unit attention condition and log the event if one of its RESPONSE frames could not be delivered, which was not viewed as compelling.

Even if Delivery Result were added, it doesn't solve the issue for iSCSI. An iSCSI target port doesn't receive an ACK for each RESPONSE PDU; it has to wait for the next PDU from the initiator with an ExpStatSN value higher than the StatSN of the RESPONSE PDU to serve as an ACK. This might not happen for a long time (or ever). The target can send a NOP-IN PDU to the initiator port and force a NOP-OUT PDU to be sent back with an updated ExpStatSN. However, it has to know when this is necessary - it's not appropriate for every RESPONSE PDU. The proposed Response Fence argument lets the device server/task manager tell the target port exactly when to do this.

SAS port layer's application layer knowledge

The SAS port layer requires than an initiator port not send a task management function that affects commands until it receives ACKs for all outstanding frame transmissions for all affected commands. This requires the port layer to understand what the task management function does, which is really something only known to the application layer. There should be more generic communication from the application layer to the port layer specifying when this type of behavior is appropriate (e.g., a Request Fence argument specifying the I_T_L_Q, I_T_L, or I_T nexus affected).

Suggested changes to SAM-4 (to add Response Fence)

[Additional changes that would accompany these changes:

a) Transport protocols: describe how they respond to the Response Fence argument
b) Command set standards and SAM-4: define the commands and task management function cases for which the device server/task manager sets the Response Fence argument. For example, CLEAR TASK SET would set it.]

5 SCSI command model

5.1 The Execute Command procedure call

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

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

Input ~~A~~arguments:

I_T_L_Q Nexus: The I_T_L_Q nexus identifying the task (see 4.12).
CDB: Command descriptor block (see 5.2).
Task Attribute: A value specifying one of the task attributes defined in 8.6. SCSI transport protocols may or may not provide the ability to specify a different task attribute for each task (see 8.6.1). For a task that processes linked commands, the Task Attribute shall be the value specified for the first command in a series of linked commands. The Task Attribute specified for the second and subsequent commands shall be ignored.
Data-In Buffer Size: The number of bytes available for data transfers to the Data-In Buffer (see 5.4.3). SCSI transport protocols may interpret this argument to include both the size and the location of the Data-In Buffer.
Data-Out Buffer: A buffer containing command specific information to be sent to the logical unit (e.g., data or parameter lists needed to process the command). The buffer size is indicated by the Data-Out Buffer Size argument. The content of the Data-Out Buffer shall not change during the lifetime of the command (see 5.5) as viewed by the application client.
Data-Out Buffer Size: The number of bytes available for data transfers from the Data-Out Buffer (see 5.4.3).
Command Reference Number (CRN): When this argument is used, all sequential commands of an I_T_L nexus shall include a CRN argument that is incremented by one. The CRN shall be set to one

for each I_T_L nexus involving the SCSI port after the SCSI port receives a hard reset or detects I_T nexus loss. The CRN shall be set to one after it reaches the maximum CRN value supported by the protocol. The CRN value zero shall be reserved for use as defined by the SCSI transport protocol. It is not an error for the application client to provide this argument when CRN is not supported by the SCSI transport protocol or logical unit.

Task Priority: The priority assigned to the task (see 8.7).

Output ~~A~~arguments:

Data-In Buffer: A buffer to contain command specific information returned by the logical unit by the time of command completion. The Execute Command procedure call shall not return a status of GOOD, CONDITION MET, INTERMEDIATE, or INTERMEDIATE-CONDITION MET unless the buffer contents are valid. The application client shall treat the buffer contents as invalid unless the command completes with a status of GOOD, CONDITION MET, INTERMEDIATE, or INTERMEDIATE-CONDITION MET. While some valid data may be present for other values of status, the application client should rely on additional information from the logical unit (e.g., sense data) to determine the state of the buffer contents. If the command ends with a service response of SERVICE DELIVERY OR TARGET FAILURE, the application client shall consider this argument to be undefined.

Sense Data: A buffer containing sense data returned in the same I_T_L_Q nexus transaction (see 3.1.47) as a CHECK CONDITION status (see 5.8.6). The buffer length is indicated by the Sense Data Length argument. If the command ends with a service response of SERVICE DELIVERY OR TARGET FAILURE, the application client shall consider this argument to be undefined.

Sense Data Length: The length in bytes of the Sense Data.

Status: A one-byte field containing command completion status (see 5.3). If the command ends with a service response of SERVICE DELIVERY OR TARGET FAILURE, the application client shall consider this argument to be undefined.

Retry Delay Timer: Additional information about the indicated status code (see 5.3.2).

Service Response assumes one of the following values:

TASK COMPLETE: A logical unit response indicating that the task has ended. The Status argument shall have one of the values specified in 5.3.

SERVICE DELIVERY OR TARGET FAILURE: The command has been ended due to a service delivery failure (see 3.1.120) or SCSI target device malfunction. All output parameters are invalid.

The SCSI transport protocol events corresponding to a response of TASK COMPLETE or SERVICE DELIVERY OR TARGET FAILURE shall be specified in each SCSI transport protocol standard.

...

5.4 SCSI transport protocol services in support of Execute Command

5.4.1 Overview

The SCSI transport protocol services that support the Execute Command procedure call are described in 5.4. Two groups of SCSI transport protocol services are described. The SCSI transport protocol services that support the ~~request and confirmation for the Execute Command procedure call~~delivery of the command and status are described in 5.4.2. The SCSI transport protocol services that support the data transfers associated with processing a command are described in 5.4.3.

5.4.2 ~~Execute Command request/confirmation SCSI transport protocol services~~Command and Status SCSI transport protocol services

5.4.2.1 Command and Status SCSI transport protocol services overview

All SCSI transport protocol standards shall define the SCSI transport protocol specific requirements for implementing the **Send SCSI Command** request (see 5.4.2.2), the **SCSI Command Received** indication (see 5.4.2.3), the **Send Command Complete** response (see 5.4.2.4), and the **Command Complete Received** confirmation (see 5.4.2.5) SCSI transport protocol services.

All SCSI initiator devices shall implement the **Send SCSI Command** request and the **Command Complete Received** confirmation SCSI transport protocol services as defined in the applicable SCSI transport protocol standards.

All SCSI target devices shall implement the **SCSI Command Received** indication and the **Send Command Complete** response SCSI transport protocol services as defined in the applicable SCSI transport protocol standards.

5.4.2.2 Send SCSI Command transport protocol service request

An application client uses the Send SCSI Command transport protocol service request to request that a SCSI initiator port send a SCSI command.

~~**SCSI Transport Protocol Service Request:**~~

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

Input ~~A~~arguments:

I_T_L_Q Nexus: The I_T_L_Q nexus identifying the task (see 4.12).
CDB: Command descriptor block (see 5.2).
Task Attribute: A value specifying one of the task attributes defined in 8.6. For specific requirements on the Task Attribute argument see 5.1.
Data-In Buffer Size: The number of bytes available for data transfers to the Data-In Buffer (see 5.4.3). SCSI transport protocols may interpret this argument to include both the size and the location of the Data-In Buffer.
Data-Out Buffer: A buffer containing command specific information to be sent to the logical unit (e.g., data or parameter lists needed to process the command (see 5.1)). The content of the Data-Out Buffer shall not change during the lifetime of the command (see 5.5) as viewed by the application client.
Data-Out Buffer Size: The number of bytes available for data transfers from the Data-Out Buffer (see 5.4.3).
Command Reference Number (CRN): When this argument is used, all sequential commands of an I_T_L nexus shall include a CRN argument that is incremented by one (see 5.1).
Task Priority: The priority assigned to the task (see 8.7).
First Burst Enabled: An argument specifying that a SCSI transport protocol specific number of bytes from the Data-Out Buffer shall be delivered to the logical unit without waiting for the device server to invoke the Receive Data-Out SCSI transport protocol service.

5.4.2.3 SCSI Command Received transport protocol service indication

A SCSI target port uses the SCSI Command Received transport protocol service indication to notify a device server that it has received a SCSI command.

~~**SCSI Transport Protocol Service Indication:**~~

SCSI Command Received (IN (I_T_L_Q Nexus, CDB, Task Attribute, [Command Reference Number], [Task Priority], [First Burst Enabled]))

Input ~~A~~arguments:

I_T_L_Q Nexus: The I_T_L_Q nexus identifying the task (see 4.12).
CDB: Command descriptor block (see 5.2).
Task Attribute: A value specifying one of the task attributes defined in 8.6. For specific requirements on the Task Attribute argument see 5.1.
Command Reference Number (CRN): When this argument is used, all sequential commands of an I_T_L nexus shall include a CRN argument that is incremented by one (see 5.1).
Task Priority: The priority assigned to the task (see 8.7).
First Burst Enabled: An argument specifying that a SCSI transport protocol specific number of bytes from the Data-Out Buffer are being delivered to the logical unit without waiting for the device server to invoke the Receive Data-Out SCSI transport protocol service.

5.4.2.4 Send Command Complete transport protocol service response

A device server uses the Send Command Complete transport protocol service response to request that a SCSI target port transmit command complete information.

~~**SCSI Transport Protocol Service Response (from device server):**~~

Send Command Complete (IN (I_T_L_Q Nexus, [Sense Data], [Sense Data Length], Status, Service Response, [Retry Delay Timer], [Response Fence]))

Input ~~A~~arguments:

I_T_L_Q Nexus: The I_T_L_Q nexus identifying the task (see 4.12).
Sense Data: If present, this argument instructs the SCSI target port to return sense data to the SCSI initiator port (see 5.8.6).
Sense Data Length: The length in bytes of the sense data to be returned to the SCSI initiator port.
Status: Command completion status (see 5.1).
Service Response: Possible service response information for the command (see 5.1).
Retry Delay Timer: The Retry Delay Timer code for the command (see 5.3.2).
Response Fence: The target port shall:

a) ensure that it has delivered to the SCSI initiator port the information for all previous Send Command Complete responses and all previous Task Management Function Executed responses (see 7.10.4) for the I_T_L nexus in the specified I_T_L_Q nexus before sending this response; and
b) ensure that the information for this response has been delivered to the SCSI initiator port before processing any subsequent Send Command Complete responses and Task Management Function Executed responses for the I_T_L nexus in the specified I_T_L_Q nexus.

5.4.2.5 Command Complete Received transport protocol service confirmation

A SCSI initiator port uses the Command Complete Received transport protocol service confirmation to notify an application client that it has received command complete information.

~~**SCSI Transport Protocol Service Confirmation:**~~

Command Complete Received (IN (I_T_L_Q Nexus, [Data-In Buffer], [Sense Data], [Sense Data Length], Status, Service Response, [Retry Delay Timer]))

Input ~~A~~arguments:

I_T_L_Q Nexus: The I_T_L_Q nexus identifying the task (see 4.12).
Data-In Buffer: A buffer containing command specific information returned by the logical unit on command completion (see 5.1).
Sense Data: Sense data returned in the same I_T_L_Q nexus transaction (see 3.1.47) as a CHECK CONDITION status (see 5.8.6).
Sense Data Length: The length in bytes of the received sense data.
Status: Command completion status (see 5.1).
Service Response: Service response for the command (see 5.1).
Retry Delay Timer: The Retry Delay Timer code for the command (see 5.3.2).

5.4.3 Data transfer SCSI transport protocol services

5.4.3.1 Introduction

The data transfer services described in 5.4.3 provide mechanisms for moving data to and from the SCSI initiator port in response to commands transmitted using the **Execute Command** procedure call. All SCSI transport protocol standards shall define the protocols required to implement these services.

...

The STPL confirmed services specified in 5.4.3.2 and 5.4.3.3 are used by the device server to request the transfer of data to or from the application client Data-In Buffer or Data-Out Buffer, respectively. The SCSI initiator device SCSI transport protocol service interactions are unspecified.

This standard provides only for the transfer phases to be sequential. Provision for overlapping transfer phases is outside the scope of this standard.

5.4.3.2 Data-In delivery service

5.4.3.2.1 Send Data-In transport protocol service request

A device server uses the Send Data-In transport protocol service request to request that a SCSI target port send data.

~~**Request:**~~

Send Data-In (IN (I_T_L_Q Nexus, Device Server Buffer, Application Client Buffer Offset, Request Byte Count))

~~Argument descriptions~~Input arguments:

I_T_L_Q Nexus: The I_T_L_Q nexus identifying the task (see 4.12).
Device Server Buffer: The buffer in the device server from which data is to be transferred.
Application Client Buffer Offset: Offset in bytes from the beginning of the application client's buffer (i.e., the Data-In Buffer) to the first byte of transferred data.
Request Byte Count: Number of bytes to be moved by this request.

5.4.3.2.2 Data-In Delivered transport protocol service confirmation

A SCSI target port uses the Data-In Delivered transport protocol service confirmation to notify a device server that it has sent data.

~~**Confirmation:**~~

Data-In Delivered (IN (I_T_L_Q Nexus, Delivery Result))

This confirmation notifies the device server that the specified data was successfully delivered to the application client buffer, or that a service delivery subsystem error occurred while attempting to deliver the data.

~~Argument descriptions~~Input arguments:

I_T_L_Q Nexus: The I_T_L_Q nexus identifying the task (see 4.12).
Delivery Result: an encoded value representing one of the following:
DELIVERY SUCCESSFUL:The data was delivered successfully.
DELIVERY FAILURE:A service delivery subsystem error occurred while attempting to deliver the data.

5.4.3.3 Data-Out delivery service

5.4.3.3.1 Receive Data-Out transport protocol service request

A device server uses the Receive Data-Out transport protocol service request to request that a SCSI target port receive data.

~~**Request:**~~

Receive Data-Out (IN (I_T_L_Q Nexus, Application Client Buffer Offset, Request Byte Count, Device Server Buffer))

~~Argument descriptions~~Input arguments:

I_T_L_Q Nexus: The I_T_L_Q nexus identifying the task (see 4.12).
Device Server Buffer: The buffer in the device server to which data is to be transferred.
Application Client Buffer Offset: Offset in bytes from the beginning of the application client's buffer (i.e., the Data-Out Buffer) to the first byte of transferred data.
Request Byte Count: Number of bytes to be moved by this request.

If the SCSI Command Received SCSI transport protocol service included a First Burst Enabled argument and random buffer access is not supported, first burst data shall be transferred to the Device Server Buffer until all first burst data has been transferred. If the SCSI Command Received SCSI transport protocol service included a First Burst Enabled argument and random buffer access is supported, first burst data should be transferred to the Device Server Buffer but first burst data may be re-transferred across the service delivery subsystem.

5.4.3.3.2 Data-Out Received transport protocol service confirmation

A SCSI target port uses the Data-Out Received transport protocol service confirmation to notify a device server that it has received data.

~~**Confirmation:**~~

Data-Out Received (IN (I_T_L_Q Nexus, Delivery Result))

This confirmation notifies the device server that the requested data has been successfully delivered to its buffer, or that a service delivery subsystem error occurred while attempting to receive the data.

~~Argument descriptions~~Input arguments:

I_T_L_Q Nexus: The I_T_L_Q nexus identifying the task (see 4.12).
Delivery Result: an encoded value representing one of the following:
DELIVERY SUCCESSFUL:The data was delivered successfully.
DELIVERY FAILURE:A service delivery subsystem error occurred while attempting to receive the data.

5.4.3.4 Terminate Data Transfer service

5.4.3.4.1 Terminate Data Transfer service overview

The terminate data transfer request and confirmation may be used by a task manager to terminate partially completed transfers to the Data-In Buffer or from the Data-Out Buffer.

The Terminate Data Transfer SCSI transport protocol service allows a device server to specify that one or more Send Data-In or Receive Data-Out SCSI transport protocol service requests be terminated by a SCSI target port.

5.4.3.4.2 Terminate Data Transfer transport protocol service request

A device server uses the Terminate Data Transfer transport protocol service request to request that a SCSI target port terminate data transfers.

~~**Request:**~~

Terminate Data Transfer (IN (Nexus))

~~Argument descriptions~~Input arguments:

Nexus: An I_T ~~N~~nexus, I_T_L ~~N~~nexus, or I_T_L_Q ~~N~~nexus (see 4.12).

The SCSI target port terminates all transfer service requests for the specified nexus (e.g., if an I_T_L nexus is specified, then the SCSI target port terminates all transfer service requests from the logical unit for the specified SCSI initiator port).

5.4.3.4.3 Data Transfer Terminated transport protocol service confirmation

A SCSI target port uses the Data Transfer Terminated transport protocol service confirmation to notify a device server that it has terminated all outstanding data transfers for a specified nexus.

~~**Confirmation:**~~

Data Transfer Terminated (IN (Nexus))

~~Argument descriptions~~Input arguments:

Nexus: An I_T ~~N~~nexus, I_T_L ~~N~~nexus, or I_T_L_Q ~~N~~nexus (see 4.12).

This confirmation is returned in response to a Terminate Data Transfer request whether or not the specified nexus existed in the SCSI target port when the request was received. After a Data Transfer Terminated SCSI transport protocol service confirmation has been sent in response to a Terminate Data Transfer SCSI transport protocol service request, Data-In Delivered or Data-Out Received SCSI transport protocol service confirmations shall not be sent for the tasks specified by the nexus.

7 Task management functions

7.1 Introduction

An application client requests the processing of a task management function by invoking the SCSI transport protocol services described in 7.10, the collective operation of which is modeled in the following procedure call: [make bold and indent:]

Service Response = Function name (IN (nexus))

The task management function names are summarized in table 34.

~~Argument descriptions~~Input arguments:

Nexus: An I_T ~~N~~nexus, I_T_L ~~N~~nexus, or I_T_L_Q ~~N~~nexus (see 4.7) identifying the task or tasks affected by the task management function.
I_T Nexus: A SCSI initiator port and SCSI target port nexus (see 4.7).
I_T_L Nexus: A SCSI initiator port, SCSI target port, and logical unit nexus (see 4.7).
I_T_L_Q Nexus: A SCSI initiator port, SCSI target port, logical unit, and task tag nexus (see 4.7).

~~Service Response:~~

~~One of the following SCSI transport protocol specific responses shall be returned:~~

Service Response assumes one of the following values:

FUNCTION COMPLETE: A task manager response indicating that the requested function is complete. Unless another response is required, the task manager shall return this response upon completion of a task management request supported by the logical unit or SCSI target device to which the request was directed.
FUNCTION SUCCEEDED: An optional task manager response indicating that the requested function is supported and completed successfully. This task manager response shall only be used by functions that require notification of success (e.g., QUERY TASK).
FUNCTION REJECTED: An task manager response indicating that the requested function is not supported by the logical unit or SCSI target device to which the function was directed.
INCORRECT LOGICAL UNIT NUMBER: An optional task router response indicating that the function requested processing for an incorrect logical unit number.
SERVICE DELIVERY OR TARGET FAILURE: The request was terminated due to a service delivery failure (see 3.1.120) or SCSI target device malfunction. The task manager may or may not have successfully performed the specified function.

Each SCSI transport protocol standard shall define the events comprising each of these service responses.

The task manager response to task management requests is subject to the presence of access restrictions, as managed by ACCESS CONTROL OUT and ACCESS CONTROL IN commands (see SPC-3), as follows:

a) A task management request of ABORT TASK, ABORT TASK SET, CLEAR ACA, I_T NEXUS RESET, or QUERY TASK shall not be affected by the presence of access restrictions;
b) A task management request of CLEAR TASK SET or LOGICAL UNIT RESET received from a SCSI initiator port that is denied access to the logical unit, either because it has no access rights or because it is in the pending-enrolled state, shall not cause any changes to the logical unit; and c) The task management function service response shall not be affected by the presence of access restrictions.

...

7.10 Task management SCSI transport protocol services

7.10.1 Task management SCSI transport protocol services overview

The SCSI transport protocol services described in this subclause are used by a SCSI initiator device and SCSI target device to process a task management procedure call. The following arguments are passed:

Nexus: An I_T ~~N~~nexus, I_T_L ~~N~~nexus, or I_T_L_Q ~~N~~nexus (see 4.12).
Function Identifier: Argument encoding the task management function to be performed.

All SCSI transport protocol standards shall define the SCSI transport protocol specific requirements for implementing the **Send Task Management Request** request (see 7.10.2), the **Task Management Request Received** indication (see 7.10.3), the **Task Management Function Executed** response (see 7.10.4), and the

Received Task Management Function Executed confirmation (see 7.10.5) SCSI transport protocol services ~~described in this subclause~~.

A SCSI transport protocol standard may specify different implementation requirements for the **Send Task Management Request** request SCSI transport protocol service for different values of the Function Identifier argument.

All SCSI initiator devices shall implement the **Send Task Management Request** request and the **Received Task Management Function Executed** confirmation SCSI transport protocol services as defined in the applicable SCSI transport protocol standards.

All SCSI target devices shall implement the **Task Management Request Received** indication and the **Task Management Function Executed** response SCSI transport protocol services as defined in the applicable SCSI transport protocol standards.

7.10.2 Send Task Management Request transport protocol service request

An application client uses the Send Task Management Request transport protocol service request to request that a SCSI initiator port transmit a task mangement request.

~~Request sent by an application client:~~

Send Task Management Request (IN (Nexus, Function Identifier))

~~Argument descriptions~~Input arguments:

Nexus: An I_T ~~N~~nexus, I_T_L ~~N~~nexus, or I_T_L_Q ~~N~~nexus (see 4.12).
Function Identifier: Argument encoding the task management function to be performed.

7.10.3 Task Management Request Received transport protocol service indication

A SCSI target port uses the Task Management Request Received transport protocol service indication to notify a task manager that it has received a task management request.

~~Indication received by the task manager:~~

Task Management Request Received (IN (Nexus, Function Identifier))

~~Argument descriptions~~Input arguments:

Nexus: An I_T ~~N~~nexus, I_T_L ~~N~~nexus, or I_T_L_Q ~~N~~nexus (see 4.12).
Function Identifier: Argument encoding the task management function to be performed.

7.10.4 Task Management Function Executed transport protocol service response

A task manger uses the Send Task Management Request transport protocol service response to request that a SCSI target port send task management function executed information.

~~Response from task manager:~~

Task Management Function Executed (IN (Nexus, Service Response, [Response Fence]))

~~Argument descriptions~~Input arguments:

Nexus: An I_T ~~N~~nexus, I_T_L ~~N~~nexus, or I_T_L_Q ~~N~~nexus (see 4.12).
Service Response: An encoded value representing one of the following:
FUNCTION COMPLETE: The requested function has been completed.
FUNCTION SUCCEEDED: The requested function is supported and completed successfully.
FUNCTION REJECTED: The task manager does not implement the requested function.
INCORRECT LOGICAL UNIT NUMBER: An optional task router response indicating that the function requested processing for an incorrect logical unit number.
SERVICE DELIVERY OR TARGET FAILURE: The request was terminated due to a service delivery failure (see 3.1.112) or SCSI target device malfunction. The task manager may or may not have successfully performed the specified function.
Response Fence: If the specified nexus is an I_T nexus or an I_T_L nexus, the target port shall:
a) ensure that it has delivered to the SCSI initiator port the information for all previous Send Command Complete responses (see 5.4.2.4) and all previous Task Management Function Executed responses

for the specified nexus before sending this response; and

b) ensure that the information for this response has been delivered to the SCSI initiator port before processing any subsequent Send Command Complete responses and Task Management Function Executed responses for the specified nexus.

7.10.5 Received Task Management Function Executed transport protocol service confirmation

A SCSI initiator port uses the Received Task Management Function Executed transport protocol service confirmation to notify an application client that it has received task management function executed information.

~~Confirmation received by application client:~~

Received Task Management Function Executed (IN (Nexus, Service Response))

~~Argument descriptions~~Input arguments:

Nexus: An I_T ~~N~~nexus, I_T_L ~~N~~nexus, or I_T_L_Q ~~N~~nexus (see 4.12).
Service Response: An encoded value representing one of the following:

FUNCTION COMPLETE: The requested function has been completed.
FUNCTION SUCCEEDED: The requested function is supported and completed successfully.
FUNCTION REJECTED: The task manager does not implement the requested function.
INCORRECT LOGICAL UNIT NUMBER: An optional task router response indicating that the function requested processing for an incorrect logical unit number.
SERVICE DELIVERY OR TARGET FAILURE: The request was terminated due to a service delivery failure (see 3.1.112) or SCSI target device malfunction. The task manager may or may not have successfully performed the specified function.

Each SCSI transport protocol shall allow a **Received Task Management Function Executed** confirming completion of the requested task to be associated with the corresponding **Send Task Management Request**.