To: T10 Technical Committee
From: Rob Elliott, HP (elliott@hp.com)
Date: 16 April 2007
Subject: 07-177r0 SAS-2 Port layer wide port ordering

## Revision history

Revision 0 (16 April 2007) First revision

## Related documents

sas2r09 - Serial Attached SCSI - 2 (SAS-2) revision 9
06-341r1 SAM-4 Response Fence for protocol services (Mallikarjin Chadalapaka and Rob Elliott, HP)

## Overview

A wide port can have multiple connections open to another wide port at the same time. A port cannot prevent this from happening; for example, always rejecting the second incoming connection request could lead to a livelock if both ports implement that algorithm and their connection requests cross on different logical links. Although order is maintained within each connection, there is no ordering guarantee across multiple connections. If the wide port sends a frame on logical link 0 and then sends a frame on logical link 1, they might arrive at the destination in the original order, simultaneously, or in reverse order.

Provided it does not cause ordering problems, the wide port can and should send frames simultaneously on multiple logical links to take advantage of the extra bandwidth. This is particularly important for DATA frames, which are already non-interlocked within connections and should be allowed to be transferred concurrently with other frame types on other connections. It is not as critical for COMMAND, TASK, XFER_RDY, and RESPONSE frames, which are already interlocked within each connection.

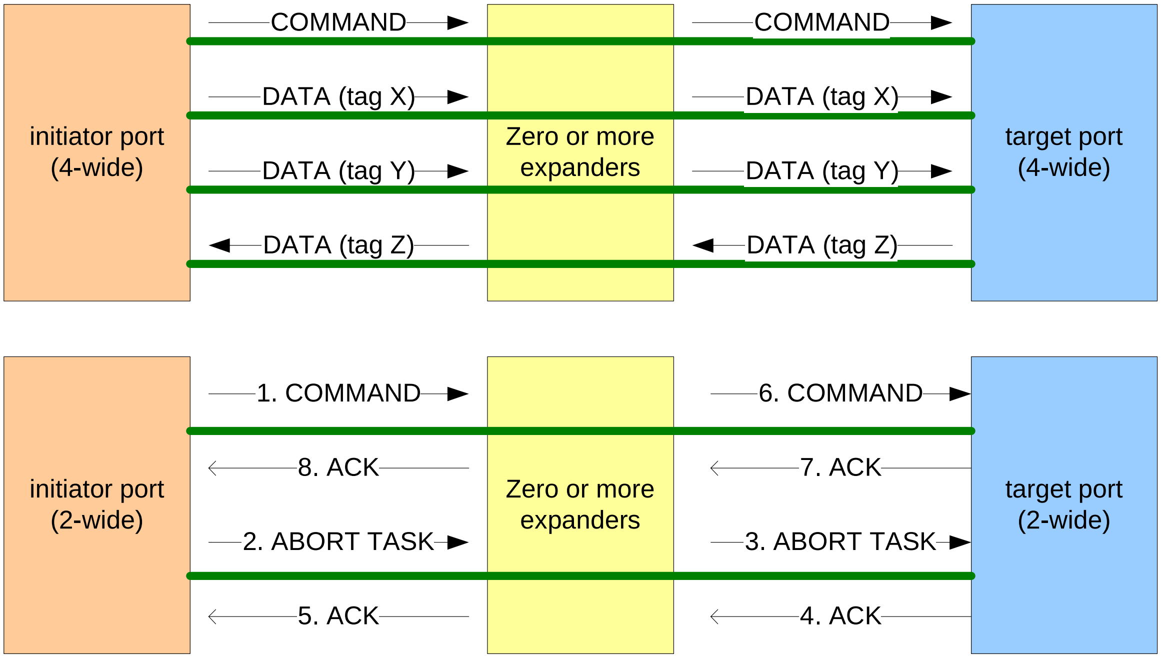

**Figure 1 — Wide connection examples**

The port layer (chapter 8) section 8.2.2.3.6 currently includes rules prohibiting sending a TASK frame until the wide port has received an ACK, a NAK, an ACK/NAK timeout, or a lost connection for each outgoing frame (e.g., COMMAND frames or write DATA frames) for each command that is affected by that task management function. For example, the port layer won't send an ABORT TASK while the command it is aborting is still in flight.

There are a few problems with this approach:

a) Architecturally, the port layer is not a good place to make this decision. The port layer should not be required to understand the semantic scope of each SCSI task management function - that belongs to the application layer;
b) The port layer also needs to prevent sending any new commands or TMFs *after* the TASK frame in question (e.g., ABORT TASK) until it receives an ACK or NAK for the TASK frame itself. Otherwise, those new commands or TMFs might arrive first and be affected by the TMF in question (e.g., the ABORT TASK could abort tasks that were sent by the application client after the ABORT TASK), or arrive later and be treated as unknown tags (not so bad) or tag conflicts (bad).
c) There are commands that affect the processing of other commands, like PERSISTENT RESERVE OUT with the PREEMPT AND ABORT service action, PERSISTENT RESERVE OUT with any service action changing the reservation state, ACCESS CONTROL OUT, etc. The COMMAND frames delivering these commands should be subject to the same rules as TASK frames.
d) Commands with ORDERED and HEAD OF QUEUE task attributes are not guaranteed to arrive at the destination in their original order if sent across different logical links. They are not useful unless they arrive in order.
e) Task management functions do not just affect commands; they can also affect task management functions (e.g., LOGICAL UNIT RESET wipes out all outstanding TMFs along with all commands). There is no rule restricting an I_T nexus to only one TMF at a time. A target device with multiple target ports and/or talking to multiple initiator ports cannot avoid deaing with multiple TMFs, at least one per I_T nexus.
f) The RESPONSE frame for a TMF (or for a command affecting other commands) also needs to be ordered in relation to the RESPONSE frames for the affected commands. Otherwise, the initiator will receive unknown tags (not so bad) or suffer tag conflicts (bad).

If a wide initiator port wants to send a TASK frame or COMMAND frame of that nature, it must:

1) wait for the ACK or NAK for all previously sent COMMAND and TASK frames on other logical links;
2) send the command; then
3) wait for the ACK or NAK before sending any subsequent COMMAND and TASK frames on any other logical link.

This can be modeled in SCSI architecture using a “Request Fence” argument to the Send SCSI Command () and Send Task Management Request () transport protocol services. If the application client is sending a command or TMF that is sensitive to order, it can specify the scope of fence required (I_T, I_T_L, or I_T_L_Q nexus). The port layer in the initiator port uses that argument to decide when to perform the fence.

If a wide target port wants to send a RESPONSE frame of that nature, it must:

1) wait for the ACK or NAK for all previously sent RESPONSE frames on other logical links;
2) send the command; then
3) wait for the ACK or NAK before sending any subsequent RESPONSE frame on any other logical link.

This can be modeled in SCSI architecture using a “Response Fence” argument to the Send Command Complete () and Task Management Function Executed () transport protocol services. If the device server is sending a RESPONSE frame that is sensitive to order, it can specify the scope of fence required (I_T, I_T_L, or I_T_L_Q nexus). The port layer in the target port uses that argument to decide when to perform the fence.

This leaves it up to the application client to decide when fencing is appropriate (and makes it a SAM-4 and command set issue).

Table 1 lists some examples where Request Fence would be asserted.

**Table 1 — Request Fence usage examples**

| When | Scope |
|---|---|
| ORDERED task attribute<br>HEAD OF QUEUE task attribute | I_T_L nexus |
| ABORT TASK<br>QUERY TASK | affected I_T_L_Q nexus |
| ABORT TASK SET<br>QUERY TASK SET<br>CLEAR TASK SET<br>CLEAR ACA | I_T_L nexus |
| LOGICAL UNIT RESET<br>I_T NEXUS RESET | I_T nexus |
| PERSISTENT RESERVE OUT command PREEMPT AND ABORT service action | I_T_L nexus |
| PERSISTENT RESERVE OUT command changing the reservation | I_T_L nexus |
| ACCESS CONTROL OUT command changing access rights | I_T_L nexus |

Table 2 lists some examples where Request Fence would be asserted.

**Table 2 — Response Fence usage examples**

| When | Scope |
|---|---|
| ABORT TASK SET<br>QUERY TASK SET<br>CLEAR TASK SET<br>CLEAR ACA | I_T_L nexus |
| LOGICAL UNIT RESET<br>I_T NEXUS RESET | I_T nexus |
| PERSISTENT RESERVE OUT command PREEMPT AND ABORT service action | I_T_L nexus |
| Any command returning status of CHECK CONDITION if ACA is enabled | I_T_L nexus |
| Any command returning a CHECK CONDITION/UNIT ATTENTION condition after commands were aborted on that I_T_L nexus | I_T_L nexus |

## Suggested changes to SAS-2

### 8.2.2.3.6 PL_OC2:Overall_Control state frame transmission

In order to prevent livelocks, If this port is a wide SSP port, has multiple connections established, and has a pending Tx Frame message, then this state shall send at least one Tx Frame message to a PL_PM state machine before sending a Close Connection message to the PL_PM state machine.

After this state receives a Connection Opened message from a PL_PM state machine, this state selects pending Tx Frame messages for the destination SAS address with the same initiator port bit and protocol arguments, and, as an option, the same connection rate argument, and sends the messages to the PL_PM state machine as Tx Frame messages.

This state may send a Tx Frame message to any PL_PM state machine that has established a connection with the destination SAS address when the initiator port bit and protocol arguments match those in the Tx Frame message.

After this state sends a Tx Frame message to a PL_PM state machine, it shall not send another Tx Frame message to that PL_PM state machine until it receives a Transmission Status (Frame Transmitted) message.

~~This state may send a Tx Frame message containing a COMMAND frame for a destination SAS address to a PL_PM state machine while waiting for one of the following messages for Tx Frame messages containing COMMAND frames for the same destination SAS address from different PL_PM state machines:~~

~~a) Transmission Status (ACK Received);~~
~~b) Transmission Status (NAK Received);~~
~~c) Transmission Status (ACK/NAK Timeout); or~~
~~d) Transmission Status (Connection Lost Without ACK/NAK).~~

~~This state shall not send a Tx Frame message containing a TASK frame for a task that only affects an I_T_L_Q nexus (e.g., an ABORT TASK or QUERY TASK task management function (see SAM-4)) to any PL_PM state machine until this state has received one of the following messages for each Tx Frame message with the same I_T_L_Q nexus:~~

~~a) Transmission Status (ACK Received);~~
~~b) Transmission Status (NAK Received);~~
~~c) Transmission Status (ACK/NAK Timeout); or~~
~~d) Transmission Status (Connection Lost Without ACK/NAK).~~

~~This state shall not send a Tx Frame message containing a TASK frame for a task that only affects an I_T_L nexus (e.g., an ABORT TASK SET, CLEAR TASK SET, QUERY TASK SET, QUERY UNIT ATTENTION, CLEAR ACA, or LOGICAL UNIT RESET task management function (see SAM-4)) to any PL_PM state machine until this state has received one of the following messages for each Tx Frame message with the same I_T_L nexus:~~

~~a) Transmission Status (ACK Received);~~
~~b) Transmission Status (NAK Received);~~
~~c) Transmission Status (ACK/NAK Timeout); or~~
~~d) Transmission Status (Connection Lost Without ACK/NAK).~~

~~This state shall not send a Tx Frame message containing a TASK frame for a task that only affects an I_T nexus (e.g., an I_T NEXUS RESET task management function (see SAM-4)) to any PL_PM state machine until this state has received one of the following messages for each Tx Frame message with the same I_T nexus:~~

~~a) Transmission Status (ACK Received);~~
~~b) Transmission Status (NAK Received);~~
~~c) Transmission Status (ACK/NAK Timeout); or~~
~~d) Transmission Status (Connection Lost Without ACK/NAK).~~

This state shall not send a Tx Frame message containing a Request Fence argument to any PL_PM state machine until this state has received one of the following messages for each Tx Frame message with the same I_T_L_Q nexus as specified by that Request Fence argument:

a) Transmission Status (ACK Received);
b) Transmission Status (NAK Received);
c) Transmission Status (ACK/NAK Timeout); or
d) Transmission Status (Connection Lost Without ACK/NAK).

After this state sends a Tx Frame message containing a Request Fence argument, it shall not send another Tx Frame message with the same I_T_L_Q nexus as specified by that Request Fence argument until it has received one of the following messages:

a) Transmission Status (ACK Received);
b) Transmission Status (NAK Received);
c) Transmission Status (ACK/NAK Timeout); or
d) Transmission Status (Connection Lost Without ACK/NAK).

Once this state has sent a Tx Frame message containing ~~a DATA frame~~Non-Interlocked argument to a PL_PM state machine, this state shall not send a Tx Frame message containing ~~a DATA~~

~~frame~~Non-Interlocked argument with the same I_T_L_Q nexus to another PL_PM state machine until this state has received one of the following messages for each Tx Frame message containing ~~a DATA frame~~Non-Interlocked argument for the same I_T_L_Q nexus:

a) Transmission Status (ACK Received);
b) Transmission Status (NAK Received);
c) Transmission Status (ACK/NAK Timeout); or
d) Transmission Status (Connection Lost Without ACK/NAK).

~~Read DATA frames and write DATA frames~~Frames with the Non-Interlocked argument for the same I_T_L_Q nexus may be transmitted and received simultaneously on the same or different phys.

If this port is an SMP initiator port, then this state shall send the Tx Frame message containing the SMP REQUEST frame to the PL_PM state machine on which the connection was established for the Tx Open message. If this port is an SMP target port, then this state shall send the Tx Frame message containing the SMP RESPONSE frame to the PL_PM state machine on which the connection was established for the Tx Open message. See 7.18 for additional information about SMP connections.

Characteristics of STP connections are defined by SATA (also see 7.17).

The following arguments shall be included with the Tx Frame message:

a) the frame to be transmitted; and
b) Balance Required or Balance Not Required.

A Balance Not Required argument shall only be included if:

a) the request was a Transmit Frame (Non-Interlocked) request (i.e., the request included a DATA frame); and
b) the last Tx Frame message sent to this PL_PM state machine while this connection has been established was for a DATA frame having the same logical unit number and tag value as the DATA frame in this Tx Frame message.

If a Balance Not Required argument is not included in the Tx Frame message, then a Balance Required argument shall be included.

If this state receives a Disable Tx Frames message from a PL_PM state machine, then this state should send no more Tx Frame messages to that state machine until a new connection is established.

### 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_Q Nexus, CDB, Task Attribute, [Data-In Buffer Size], [Data-Out Buffer], [Data-Out Buffer Size], [Task Priority], [Command Reference Number], [First Burst Enabled], [Request Fence]))

Table 181 shows how the arguments to the Send SCSI Command transport protocol service are used.

**Table 181 — Send SCSI Command transport protocol service arguments**

| Argument | SAS SSP implementation |
|---|---|
| I_T_L_Q nexus | I_T_L_Q nexus, where:<br>a) I specifies the initiator port to send the COMMAND frame;<br>b) T specifies the target port to which the COMMAND frame is to be sent;<br>c) L specifies the LOGICAL UNIT NUMBER field in the COMMAND frame header; and<br>d) Q specifies the TAG field in the COMMAND frame header. |
| CDB | Specifies the CDB field in the COMMAND frame. |
| Task Attribute | Specifies the TASK ATTRIBUTE field in the COMMAND frame. |
| [Data-In Buffer Size] | Maximum of $2^{32}$ |
| [Data-Out Buffer] | Internal to the SSP initiator port. |
| [Data-Out Buffer Size] | Maximum of $2^{32}$ |
| [Task Priority] | Specifies the TASK PRIORITY field in the COMMAND frame. |
| [First Burst Enabled] | Specifies the ENABLE FIRST BURST field in the COMMAND frame and to cause the SSP initiator port to transmit the number of bytes indicated by the FIRST BURST SIZE field in the Disconnect-Reconnect mode page (see 10.2.7.2.5) for the SCSI target port without waiting for an XFER_RDY frame. |
| [Request Fence] | If included, specifies an I_T nexus, I_T_L nexus, or I_T_L_Q nexus for which the COMMAND frame is fenced (see SAM-4). |

### 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_Q Nexus, [Sense Data], [Sense Data Length], Status, Service Response, [Response Fence]))

A device server shall only call Send Command Complete () after receiving SCSI Command Received ().

A device server shall not call Send Command Complete () for a given I_T_L_Q nexus until all its outstanding Receive Data-Out () calls have been responded to with Data-Out Received () and all its outstanding Send Data-In () calls have been responded to with Data-In Delivered ().

Table 183 shows how the arguments to the Send Command Complete transport protocol service are used.

**Table 183 — Send Command Complete transport protocol service arguments**

| Argument | SAS SSP implementation |
|---|---|
| I_T_L_Q nexus | I_T_L_Q nexus, where:<br>a) I specifies the initiator port to which the RESPONSE frame is to be sent;<br>b) T specifies the target port to send the RESPONSE frame;<br>c) L specifies the LOGICAL UNIT NUMBER field in the RESPONSE frame header; and<br>d) Q specifies the TAG field in the RESPONSE frame header. |
| [Sense Data] | Specifies the SENSE DATA field in the RESPONSE frame. |
| [Sense Data Length] | Specifies the SENSE DATA LENGTH field in the RESPONSE frame. |
| Status | Specifies the STATUS field in the RESPONSE frame. |
| Service Response | Specifies the DATAPRES field and STATUS field in the RESPONSE frame:<br>a) TASK COMPLETE: The DATAPRES field is set to NO_DATA or SENSE_DATA; or<br>b) SERVICE DELIVERY OR TARGET FAILURE: The DATAPRES field is set to RESPONSE_DATA and the RESPONSE CODE field is set to INVALID FRAME or OVERLAPPED TAG ATTEMPTED. |
| [Response Fence] | If included, specifies an I_T nexus, I_T_L nexus, or I_T_L_Q nexus for which the RESPONSE frame is fenced (see SAM-4). |

### 10.2.1.12 Send Task Management Request transport protocol service

An application client uses the Send Task Management Request transport protocol service request to request that an SSP initiator port transmit a TASK frame.

Send Task Management Request (IN (Nexus, Function Identifier, [Association], [Request Fence]))

Table 191 shows how the arguments to the Send Task Management Request transport protocol service are used.

**Table 191 — Send Task Management Request transport protocol service arguments**

| Argument | SAS SSP implementation |
|---|---|
| Nexus | I_T_L nexus or I_T_L_Q nexus (depending on the Function Identifier), where:<br>a) I specifies the initiator port to send the TASK frame;<br>b) T specifies the target port to which the TASK frame is sent;<br>c) L specifies the LOGICAL UNIT NUMBER field in the TASK frame header; and<br>d) Q (for an I_T_L_Q nexus) specifies the TAG OF TASK TO BE MANAGED field in the TASK frame header. |
| Function Identifier | Specifies the TASK MANAGEMENT FUNCTION field in the TASK frame. Only these task management functions are supported:<br>a) ABORT TASK (Nexus argument specifies an I_T_L_Q Nexus);<br>b) ABORT TASK SET (Nexus argument specifies an I_T_L Nexus);<br>c) CLEAR ACA (Nexus argument specifies an I_T_L Nexus);<br>d) CLEAR TASK SET (Nexus argument specifies an I_T_L Nexus);<br>e) I_T NEXUS RESET (Nexus argument specifies an I_T Nexus);<br>f) LOGICAL UNIT RESET (Nexus argument specifies an I_T_L Nexus);<br>g) QUERY TASK (Nexus argument specifies an I_T_L_Q Nexus);<br>h) QUERY TASK SET (Nexus argument specifies an I_T_L Nexus); and<br>i) QUERY UNIT ATTENTION (Nexus argument specifies an I_T_L Nexus). |
| [Association] | Specifies the TAG field in the TASK frame header. |
| [Request Fence] | If included, specifies an I_T nexus, I_T_L nexus, or I_T_L_Q nexus for which the TASK frame is fenced (see SAM-4). |

### 10.2.1.14 Task Management Function Executed transport protocol service

A task manager uses the Task Management Function Executed transport protocol service response to request that an SSP target port transmit a RESPONSE frame.

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

A task manager shall only call Task Management Function Executed () after receiving Task Management Request Received ().

Table 193 shows how the arguments to the Task Management Function Executed transport protocol service are used.

**Table 193 — Task Management Function Executed transport protocol service arguments**

| Argument | SAS SSP implementation |
|---|---|
| Nexus | I_T_L nexus or I_T_L_Q nexus (depending on the function), where:<br>a) I specifies the initiator port to which the RESPONSE frame is sent;<br>b) T specifies the target port to send the RESPONSE frame;<br>c) L specifies the LOGICAL UNIT NUMBER field in the RESPONSE frame header; and<br>d) Q (for an I_T_L_Q nexus) indirectly specifies the TAG field in the RESPONSE frame header. |
| Service Response | Specifies the DATAPRES field and RESPONSE CODE field in the RESPONSE frame:<br>a) FUNCTION COMPLETE: The RESPONSE frame DATAPRES field is set to RESPONSE_DATA and the RESPONSE CODE field is set to TASK MANAGEMENT FUNCTION COMPLETE;<br>b) FUNCTION SUCCEEDED: The RESPONSE frame DATAPRES field is set to RESPONSE_DATA and the RESPONSE CODE field is set to TASK MANAGEMENT FUNCTION SUCCEEDED;<br>c) FUNCTION REJECTED: The DATAPRES field is set to RESPONSE_DATA and the RESPONSE CODE field is set to TASK MANAGEMENT FUNCTION NOT SUPPORTED;<br>d) INCORRECT LOGICAL UNIT NUMBER: The DATAPRES field is set to RESPONSE_DATA and the RESPONSE CODE field is set to INCORRECT LOGICAL UNIT NUMBER; or<br>e) SERVICE DELIVERY OR TARGET FAILURE: The RESPONSE frame DATAPRES field is set to RESPONSE_DATA and the RESPONSE CODE field is set to:<br>A) INVALID FRAME;<br>B) TASK MANAGEMENT FUNCTION FAILED; or<br>C) OVERLAPPED TAG ATTEMPTED. |
| [Additional Response Information] | Specifies the ADDITIONAL RESPONSE INFORMATION field in the RESPONSE frame. |
| [Association] | Specifies the TAG field in the RESPONSE frame header. |
| [Response Fence] | If included, specifies an I_T nexus, I_T_L nexus, or I_T_L_Q nexus for which the RESPONSE frame is fenced (see SAM-4). |