To: T10 Technical Committee
From: Rob Elliott, HP (elliott@hp.com)
Date: 25 January 2005
Subject: 05-024r2 SAS-1.1 Credit-based starvation

Revision history
Revision 0 (24 December 2004) First revision
Revision 1 (4 January 2005) Deleted item f) in list (about credit) which was a remnant of an earlier draft.
Revision 2 (25 January 2005) Incorporated comments from January SAS protocol WG - changed the key rule from “shall not” to “should not.”

Related documents
04-341r1 SAS-1.1 Do not reset Arbitration Wait Timer on incoming OPEN
sas1r07 - Serial Attached SCSI 1.1 revision 7

Overview
1. 04-341 discusses problems with AWT being reset too often, leading to livelocks. It corrected the major problem by requiring ports to keep the AWT running if they send OPEN_REJECT (but still stopping it on OPEN_ACCEPT). It left open one other problem, however: credit is not guaranteed to be granted by the winner in any particular connection, so resetting the AWT on transmitting OPEN_ACCEPT leaves open the possibilty that the port’s request will starve.

This can happen to both initiators or targets.

The SAS credit rules are as follows. An initiator port is required to *eventually* grant credit to a target port; it is not allowed to make this eventual acceptance contingent on being able to send something to a target port (any reason for not providing credit has to be temporary and self-healing, e.g. PCI bus congestion in an HBA). A target port, on the other hand, may refuse to grant credit until it is able to send something to an initiator port.

The initiator port is not required to grant this credit in any particular connection; whenever it wins a connection, it is not obligated to provide credit in that connection. It only must do so (eventually) in a connection that it loses (i.e. one where it sends OPEN_ACCEPT rather than receives OPEN_ACCEPT).

For a target sending OPEN_ACCEPT: The target port may not be able to send its frame until the initiator stops requesting new connections and sending it new frames.

If the initiator is sending COMMAND or TASK frames, eventually the task set will fill up and the target port will have to send OPEN_REJECT (RETRY) to avoid dropping an incoming frame. Although it can return BUSY or TASK SET FULL status for commands, it has to generate and return a RESPONSE frame to do so; it can only do that for a limited number of commands. There is no BUSY or TASK SET FULL kind of response for task management functions; they must be processed if received. When it sends the OPEN_REJECT, per 04-341 the target will lets its AWT continue to run and eventually rise above the initiator’s AWT, which makes the target’s request win (initiator sends OPEN_ACCEPT) and unstarves the target.

If the initiator is sending DATA frames, eventually it will run out of reason to do so and stop making connection requests. Since XFER_RDY defines a maximum write transfer length of 4 GiB, this could take 4 GiB * (task set size) worth of DATA transfers if the maximum number of XFER_RDYs were pending (for comparison, AWT maxes out at 32 sec):

a) 4 GiB (maximum XFER_RDY transfer size) / (150 MB/sec) (slowest link rate) = 28.6 sec
b) 128 KiB (reasonable XFER_RDY transfer size) / (300 MB/sec) = 4.369 usec
c) 28.6 sec * 1024 (task set size) = 29,320 sec (488 minutes) to drain all the data
d) 4.369 usec * 1024 (task set size) = 4.473 sec to drain all the data
e) 4.369 usec * 32 (task set size) = 139.8 msec to drain all the data

Thus, it could be a long time before the target can send a RESPONSE frame and clear room to accept another task management function. The target needs higher priority on its connection requests so this doesn’t happen.

For an SSP initiator port sending OPEN_ACCEPT: The initiator port may not be able to send its frame until the target stops requesting new connections and sending it new frames. If the target is sending DATA or

RESPONSE frames for many queued commands, eventually those commands will complete (may take 488 minutes using the extreme example above for a lot of huge read commands) and the target will stop making connection requests. This could block the initiator from sending a task management function to clear the commands.

Proposed solution: rather than asking the port to reset its AWT if credit is granted but keep it running if it is not, just let the port keep the AWT running on all inbound connections (accepted or rejected).

This is unfair if fairness is defined as “equal opportunity to send frames”, since the port is able to make forward progress yet still has arbitration priority the next time it makes a request. It is fair, however, if fairness is defined as “equal opportunity to win connection requests.”

2. A phy is supposed to reset its AWT if an OPEN_REJECT comes from a destination phy. However, OPEN_REJECT (CONNECTION RATE NOT SUPPORTED) and the reserved OPEN_REJECT reasons cannot be conclusively determined to be from the destination or from an expander. A specific list needs to be provided. Based on the aforementioned problems with resetting AWT too aggressively, the list is conservative and only lists OPEN_REJECTs that clearly come from a destination phy.

Suggested changes

7.12.3 Arbitration fairness

SAS supports least-recently used arbitration fairness for connection requests.

Each SAS port and expander port shall include an Arbitration Wait Time timer which counts the time from the moment when the port makes a connection request until the request is accepted or rejected. The Arbitration Wait Time timer shall count in microseconds from 0 μs to 32 767 μs and in milliseconds from 32 768 μs to 32 767 ms + 32 768 μs. The Arbitration Wait Time timer shall stop incrementing when its value reaches 32 767 ms + 32 768 μs.

SAS ports (i.e., SAS initiator ports and SAS target ports) shall start the Arbitration Wait Time timer (see 8.2.2) when they transmit the first OPEN address frame (see 7.8.3) for the connection request. When the SAS port retransmits the OPEN address frame (e.g., after losing arbitration and handling an inbound OPEN address frame), it shall set the ARBITRATION WAIT TIME field to the current value of the Arbitration Wait Time timer.

SAS ports should set the Arbitration Wait Time timer to zero when they transmit the first OPEN address frame for the connection request. A SAS initiator port or SAS target port may be unfair by setting the ARBITRATION WAIT TIME field in the OPEN address frame (see 7.8.3) to a higher value than its Arbitration Wait Time timer indicates. However, unfair SAS ports shall not set the ARBITRATION WAIT TIME field to a value greater than or equal to 8000h; this limits the amount of unfairness and helps prevent livelocks.

~~A port shall stop the Arbitration Wait Time timer and set it to zero when it wins arbitration (i.e., it receives either OPEN_ACCEPT or OPEN_REJECT from the destination SAS port rather than from an intermediate expander device).~~

~~If a port receives a connection request that satisfies its arbitration request (i.e., it receives an OPEN address frame from the destination SAS port with the INITIATOR PORT bit set to the opposite value and a matching PROTOCOL field), it shall not stop the Arbitration Wait Time timer unless it accepts the request (i.e., transmits an OPEN_ACCEPT rather than an OPEN_REJECT).~~

A SAS port shall stop the Arbitration Wait Time timer and set it to zero when it receives one of the following connection responses:

a) OPEN_ACCEPT;
b) OPEN REJECT (PROTOCOL NOT SUPPORTED);
c) OPEN_REJECT (STP RESOURCES BUSY);
d) OPEN_REJECT (WRONG DESTINATION); or
e) OPEN_REJECT (RETRY).

NOTE 1 Connection responses that are conclusively from the destination phy (see table 66 in 7.2.5.11 and table 67 in 7.2.5.12) are included in the list. Connection responses that are only from or may be from expander phys are not included.

A SAS port *should not* stop the Arbitration Wait Time timer and set it to zero when it receives an incoming OPEN address frame that has priority over the outgoing OPEN address frame according to table 1, regardless of whether it replies with an OPEN_ACCEPT or an OPEN_REJECT.

The expander port that receives an OPEN address frame shall set the Arbitration Wait Time timer to the value of the incoming ARBITRATION WAIT TIME field and start the Arbitration Wait Time timer as it arbitrates for internal access to the outgoing expander port. When the expander device transmits the OPEN address frame out another expander port, it shall set the outgoing ARBITRATION WAIT TIME field to the current value of the Arbitration Wait Time timer maintained by the incoming expander port.

When arbitrating for access to an outgoing expander port, the expander device shall select the connection request based on the rules described in 7.12.4.

If two connection requests pass on a physical link, the phy shall determine the winner by comparing OPEN address frame field contents using the arbitration priority described in table 1.

Table 1 — Arbitration priority for OPEN address frames passing on a physical link

Bits 79-64 (79 is MSB)	Bits 63-0 (0 is LSB)
ARBITRATION WAIT TIME field value	SOURCE SAS ADDRESS field value

See 7.8.3 for details on the OPEN address frame and the ARBITRATION WAIT TIME field.