To: T10 Technical Committee
From: Rob Elliott, HP (elliott@hp.com)
Date: 25 April 2004
Subject: 04-115r1 SAS-1.1 Miscellaneous changes

Revision history
Revision 0 (21 April 2004) First revision
Revision 1 (25 April 2004) Move physical layer topics to 04-131. Added SL issue as #27. Kept numbering the same as in revision 0.

Related documents
sas1r04 - Serial Attached SCSI 1.1 revision 4

Overview
This collects a variety of minor technical (or major editorial) changes, most of which have been highlighted as editor's notes in the last few revisions of SAS-1.1.

Suggested changes

A wide variety of topics follow.

1 Clause 3 - moved to 04-131
2 Clause 4 - add SMP initiator port to figure 12
3 Clause 4 - correct when partial pathway is blocked
4 Clause 4 - mention the bound on phy identifiers in REPORT GENERAL
5 Clause 4 - base the Phy Status on the last AIP sent
6 Clause 5 - moved to 04-131
7 Clause 5 - moved to 04-131
8 Clause 5 - moved to 04-131
9 Clause 7.2 - distinguish between repeated and continued primitive sequences
10 Clause 7.2 - allow sending two SATA_CONTs
11 Clause 7.2 - expanders need not enforce SATA_CONT rules
12 Clause 7.2 - allow expanders to delete invalid dwords rather than replace them
13 Clause 7.2 - add invalid dword and ERROR to reasons for a NAK (CRC ERROR)
14 Clause 7.2 - Assorted invalid dword changes
15 Clause 7.12 - Expander arbitration rules
16 Clause 7.12 - change "at least one" to "all" in Partial Pathway Timer wording
17 Clause 7.16 - handling RRDY after CREDIT_BLOCKED
18 Clause 7.17 - clarify LINK RESET does not clear affiliation
19 Clause 7.17 - send SATA_SYNC immediately after OPEN_ACCEPT
20 Clause 7.18 - SMP receipt of consecutive SOFs
21 Clause 8 - handling phy loss during frame transmission
22 Clause 9.2 - combine Sense Data and Response Data fields
23 Clause 9.3 - handling multiple initial SATA FISes
24 Clause 10 - SATA spinup hold begins at COMSAS Timeout Detected
25 Clause 10 - honor programmed link rate changes for LINK RESET/HARD RESET
26 Clause 10 - handling errors in programmed link rate changes
27 Clause 7 - add confirmation when SL rejects an incoming OPEN

Note: these proposals are in order. If a proposal modifies text modified by a previous proposal in the list, the changes from the previous proposal may have been incorporated.

1 Clause 3 - moved to 04-131

Moved to 04-131.

2 Clause 4 - add SMP initiator port to figure 12

Add an optional *SMP initiator port* to figure 12, since if present it shares the expander's own SAS address with the SMP target port. Also add some more phys to the ports in the picture and show the SAS addresses.

4.1.5 Expander devices (edge expander devices and fanout expander devices)

...

The updated figure follows:

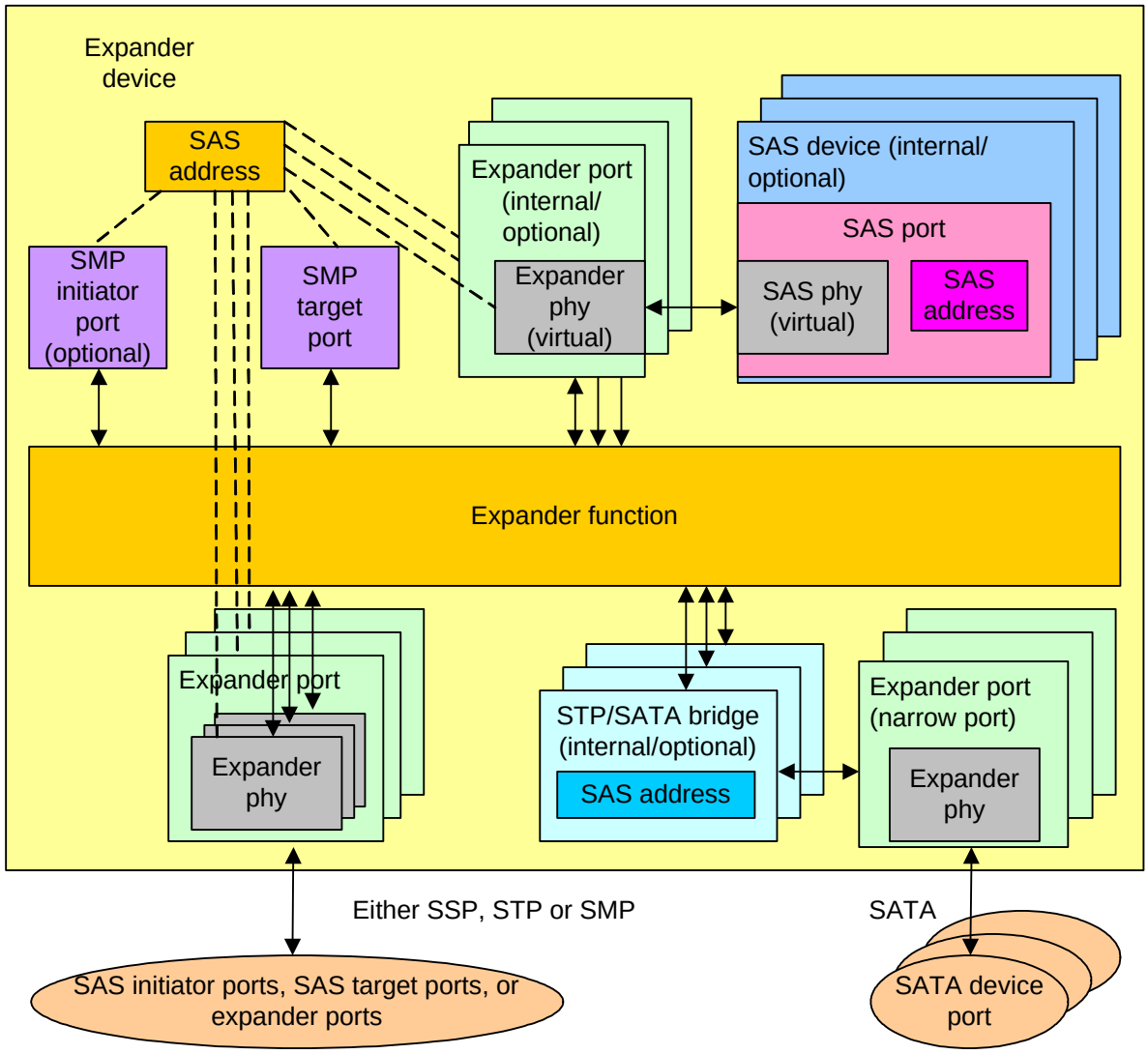

Figure 1 — Expander device [updated figure 12]

3 Clause 4 - correct when partial pathway is blocked

The term “blocked” has a very specific meaning to the ECM rules and XL state machine, so the introduction needs to be consistent.

4.1.9 Pathways

...

A partial pathway is the set of physical links participating in a connection request that has not reached the destination phy (e.g., the OPEN address frame has been transmitted by the source phy but the OPEN address frame has not yet reached the destination phy)(see 7.12).

A partial pathway is blocked when path resources it requires are held by ~~either another connection or~~ another partial pathway (see 7.12).

4 Clause 4 - mention the bound on phy identifiers in REPORT GENERAL

Mention that phy identifiers have to be less than the NUMBER OF PHYS field reported in REPORT GENERAL.

4.2.7 Phy identifiers

Each SAS phy and expander phy shall be assigned an identifier that is unique within the SAS device and/or expander device. The phy identifier is used for management functions (see 10.4).

Phy identifiers shall be greater than or equal to 00h and less than 80h, and should be numbered starting with 00h. In an expander device or in a SAS device containing an SMP target port, phy identifiers shall be less than the value of the NUMBER OF PHYS field in the SMP REPORT GENERAL function (see 10.4.3.3).

5 Clause 4 - base the Phy Status on the last AIP sent

The Phy Status definitions are based in part on “has transmitted or received an AIP (WHATEVER)”. They should be “the last AIP transmitted or received is an AIP (WHATEVER)”. A phy can change among AIP (WAITING ON PARTIAL), AIP (WAITING ON CONNECTION), and AIP (WAITING ON DEVICE) as conditions change. Those changes need to propagate too.

4.6.6.3 ECM interface

...

Table 1 describes the responses from an expander phy to the ECM.

Table 1 — Expander phy to ECM responses

Message	Description
Phy Status (Partial Pathway)	Response meaning that an expander phy: a) is being used for an unblocked partial pathway (i.e., the expander phy is in the XL3:Open_Confirm_Wait state or XL6:Open_Response_Wait state and ~~has not transmitted or received an AIP (WAITING ON PARTIAL)~~ the last AIP received or transmitted is not AIP (WAITING ON PARTIAL)); or b) has sent a Request Path request to the ECM and is receiving Arbitrating (Waiting On Partial) from the ECM.
Phy Status (Blocked Partial Pathway)	Response meaning that an expander phy: a) is being used for a blocked partial pathway (i.e., the expander phy is in the XL3:Open_Confirm_Wait state or XL6:Open_Response_Wait state and ~~has transmitted or received an AIP (WAITING ON PARTIAL)~~ the last AIP received or transmitted is AIP (WAITING ON PARTIAL)); or b) has sent a Request Path request to the ECM and is receiving Arbitrating (Blocked On Partial) from the ECM.
Phy Status (Connection)	Response meaning that an expander phy: a) is being used for a connection (i.e., the expander phy is in the XL7:Connected or XL8:Close_Wait state); or b) has sent a Request Path request to the ECM and is receiving Arbitrating (Waiting On Connection) from the ECM.

6 Clause 5 - moved to 04-131

Moved to 04-131.

7 Clause 5 - moved to 04-131

Moved to 04-131.

8 Clause 5 - moved to 04-131

Moved to 04-131.

9 Clause 7.2 - distinguish between repeated and continued primitive sequences

ATA/ATAPI-7 Volume 3 revision 4a table 21 “Description of primitives” uses the term “repeated” for primitives that might not be used with CONT. Change “repeated primitive” to “continued primitive” to differentiate between the two classes.

a) Repeated = ATA/ATAPI-7 “repeated”
b) Continued = ATA/ATAPI-7 “Repeated Note 1”

SATA_PMACK and SATA_PMNAK are now called repeated rather than single.

Also, the term “STP primitive” is used but is lacking a definition.

3.1 Defintiions

3.1.xx: STP primitive: A primitive used only inside STP connections and on SATA physical links. See 7.2.2.

7.2.2 Primitive summary

...

Table 2 lists the primitives used only inside STP connections and on SATA physical links. These are called STP primitives.

Table 2 — Primitives used only inside STP connections and on SATA physical links

Primitive	Use [a]	From [b]			To [b]			Primitive sequence type [c]
		I	E	T	I	E	T	
SATA_CONT	STP, SATA	I		T	I		T	Single
SATA_DMAT	STP, SATA	I		T	I		T	Single
SATA_EOF	STP, SATA	I		T	I		T	Single
SATA_ERROR	SATA		E				T	Single
SATA_HOLD	STP, SATA	I		T	I		T	~~Repeated~~ Continued
SATA_HOLDA	STP, SATA	I		T	I		T	~~Repeated~~ Continued
SATA_PMACK	STP, SATA							~~Single~~ Repeated
SATA_PMNAK	STP, SATA	I	E				T	~~Single~~ Repeated
SATA_PMREQ_P	STP, SATA							~~Repeated~~ Continued
SATA_PMREQ_S	STP, SATA							~~Repeated~~ Continued
SATA_R_ERR	STP, SATA	I		T	I		T	~~Repeated~~ Continued
SATA_R_IP	STP, SATA	I		T	I		T	~~Repeated~~ Continued
SATA_R_OK	STP, SATA	I		T	I		T	~~Repeated~~ Continued
SATA_R_RDY	STP, SATA	I		T	I		T	~~Repeated~~ Continued
SATA_SOF	STP, SATA	I		T	I		T	Single
SATA_SYNC	STP, SATA	I		T	I		T	~~Repeated~~ Continued
SATA_WTRM	STP, SATA	I		T	I		T	~~Repeated~~ Continued
SATA_X_RDY	STP, SATA	I		T	I		T	~~Repeated~~ Continued

[a] The Use column indicates when the primitive is used:
a) STP: SAS physical links, inside STP connections; or
b) SATA: SATA physical links.

[b] The From and To columns indicate the type of ports that originate each primitive or are the intended destinations of each primitive:
a) I for STP initiator ports and SATA host ports;
b) E for expander ports; and
c) T for STP target ports and SATA device ports.
Expander ports are not considered originators of primitives that are passing through from expander port to expander port.

[c] The Primitive sequence type columns indicate whether the primitive is sent as a single primitive sequence, a repeated primitive sequence, a continued primitive sequence, a triple primitive sequence, or a redundant primitive sequence (see 7.2.4) [change in table 50, 51, and 52]

7.2.4.3 Primitive sequences

7.2.4.3.1 Primitive sequences overview

Table 3 summarizes the types of primitive sequences.

Table 3 — Primitive sequences

Primitive sequence type	~~Number of times to transmit~~ Number of times the transmitter transmits the primitive to transmit the primitive sequence	~~Number of times received to detect~~ Number of times the receiver receives the primitive to detect the primitive sequence
Single	1	1
Repeated	~~2~~ 1 or more	1
Continued	2 followed by SATA_CONT	1
Triple	3	3
Redundant	6	3

Any number of ALIGNs and NOTIFYs may be sent inside primitive sequences without affecting the count or breaking the consecutiveness requirements. Rate matching ALIGNs and NOTIFYs shall be sent inside primitive sequences inside of connections if rate matching is enabled (see 7.13).

7.2.4.2 Single primitive sequence

Primitives labeled as single primitive sequences (e.g., RRDY, SATA_SOF) are sent one time.

Receivers count each single primitive sequence received as distinct.

7.2.4.3 Repeated primitive sequence

Primitives that form repeated primitive sequences (e.g., SATA_PMACK) shall be transmitted one or more times. Only STP primitives form repeated primitive sequences. ALIGNs and NOTIFYs may be sent inside repeated primitive sequences as described in 7.2.4.1.

Figure 2 shows an example of transmitting a repeated primitive sequence.

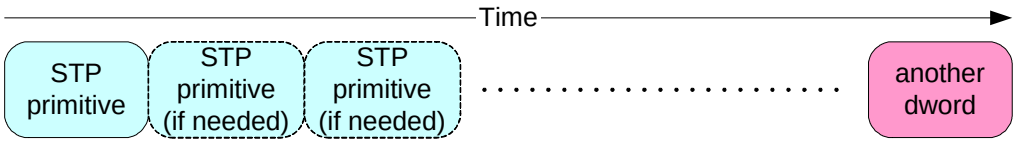

Figure 2 — Transmitting a repeated primitive sequence

Receivers do not count the number of times a repeated primitive is received; they are simply in the state of receiving the primitive.

Figure 5 shows an example of receiving a repeated primitive sequence.

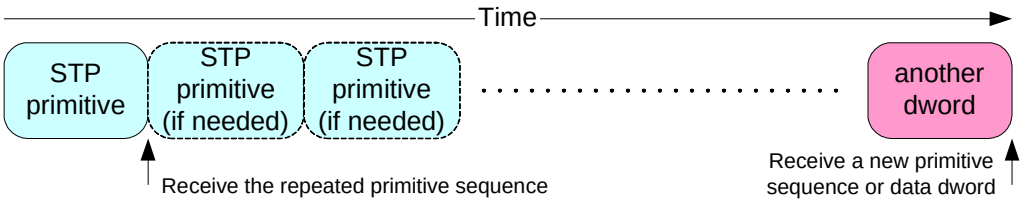

Figure 3 — Receiving a repeated primitive sequence

7.2.4.3 ~~Repeated~~ Continued primitive sequence

Primitives that form ~~repeated~~ continued primitive sequences (e.g., SATA_HOLD) shall be ~~sent~~ transmitted two times, then be followed by SATA_CONT (if needed), then be followed by vendor-specific scrambled data dwords (if needed). ALIGNs and NOTIFYs may be sent inside continued primitive sequences as described in 7.2.4.1. After the SATA_CONT, during the vendor-specific scrambled data dwords:

a) a SATA_CONT continues the continued primitive sequence; and
b) any other STP primitive, including the primitive that is being continued, ends the continued primitive sequence.

Figure 4 shows an example of transmitting a ~~repeated~~ continued primitive sequence.

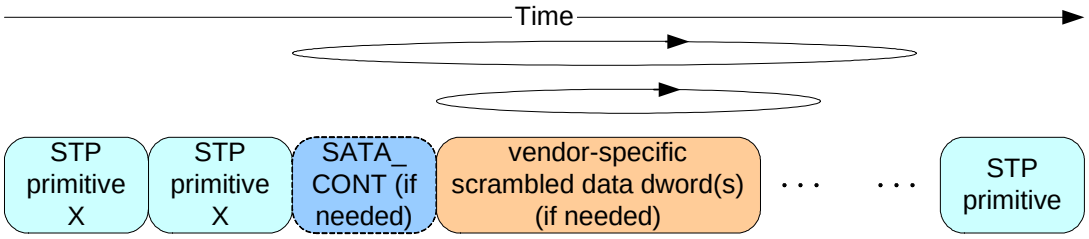

Figure 4 — Transmitting a continued primitive sequence [figure updated]

Receivers shall detect a continued primitive sequence after at least one primitive is received. The primitive may be followed by one or more SATA_CONTs, each of which may be followed by vendor-specific data dwords. Receivers shall ignore invalid dwords before, during, or after the SATA_CONT(s). Receivers do not count the number of times the continued primitive, the SATA_CONTs, or the vendor-specific data dwords are received; they are simply in the state of receiving the primitive.

Figure 5 shows an example of receiving a continued primitive sequence.

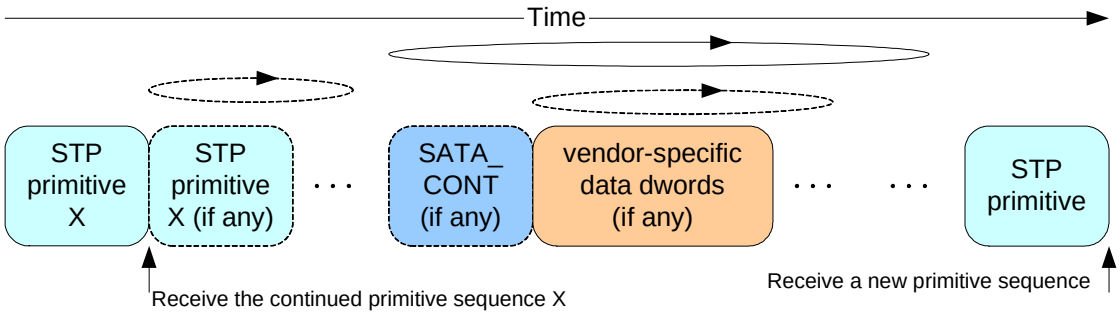

Figure 5 — Receiving a continued primitive sequence

7.6 Scrambling

Scrambling is used to reduce the probability of long strings of repeated patterns appearing on the physical link.

All data dwords are scrambled. Table 4 lists the scrambling for different types of data dwords.

Table 4 — Scrambling for different data dword types

Connection state	Data dword type	Description of scrambling
Outside connections	SAS idle dword	When a connection is not open and there are no other dwords to transmit, vendor-specific scrambled data dwords shall be transmitted.
	Address frame	After an SOAF, all data dwords shall be scrambled until the EOAF.
Inside SSP connection	SSP frame	After an SOF, all data dwords shall be scrambled until the EOF.
	SSP idle dword	When there are no other dwords to transmit, vendor-specific scrambled data dwords shall be transmitted.
Inside SMP connection	SMP frame	After an SOF, all data dwords shall be scrambled until the EOF.
	SMP idle dword	When there are no other dwords to transmit, vendor-specific scrambled data dwords shall be transmitted.
Inside STP connection	STP frame	After a SATA_SOF, all data dwords shall be scrambled until the SATA_EOF.
	~~Repeated SATA~~ Continued primitive	After a SATA_CONT, vendor-specific scrambled data dwords shall be sent until a primitive other than ALIGN or NOTIFY is transmitted.

...

For detailed requirements about scrambling of data dwords following SATA_SOF and SOF_CONT, see ATA/ATAPI-7 V3.

NOTE 1 - STP scrambling uses two linear feedback shift registers, since ~~repeated SATA~~ continued primitive~~s~~ sequences may occur inside STP frames and the~~y~~ STP frame and the continued primitive sequence have independent scrambling patterns.

10 Clause 7.2 - allow sending two SATA_CONTs

The existing SAS and SATA requirement for sending the continued primitive at least twice tolerates single-bit errors in the primitive itself. Likewise, single-bit errors in the scrambled data don't matter. A single-bit error in the SATA_CONT itself, however, confuses the receiver, which sees a primitive followed by scrambled data.

In SAS, mention that STP ports may send two SATA_CONTs rather than one to survive a single bit error in SATA_CONT.

7.2.4.3 Continued primitive sequence

Primitives that form continued primitive sequences (e.g., SATA_HOLD) shall be sent two times, then be followed by one or two SATA_CONTs (if needed), then be followed by vendor-specific scrambled data dwords (if needed). Two SATA_CONTs provides more tolerance for bit errors. ALIGNs and NOTIFYs may be sent inside continued primitive sequences as described in 7.2.4.1. After the SATA_CONT, during the vendor-specific scrambled data dwords:

a) a SATA_CONT continues the continued primitive sequence; and
b) any other STP primitive, including the primitive that is being continued, ends the continued primitive sequence.

Figure 6 shows an example of transmitting a continued primitive sequence with two SATA_CONTs.

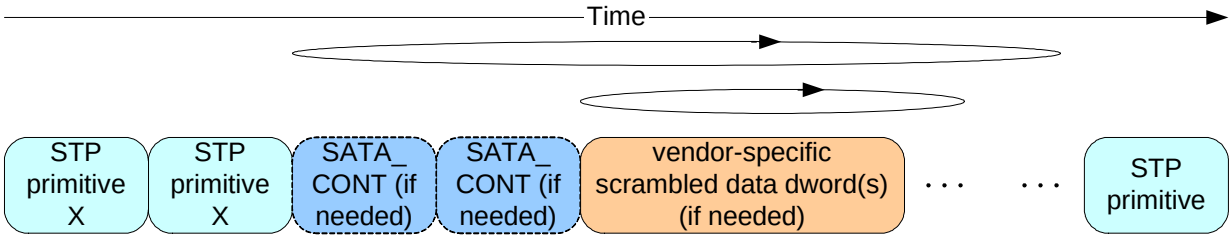

Figure 6 — Transmitting a continued primitive sequence [figure updated]

Editor's Note 1: as as, the transmit picture from the previous proposal allows for this (assume it loops on (SATA_CONT followed by data dwords) where no data dwords are included) so updating the picture might not be necessary.

---

11 Clause 7.2 - expanders need not enforce SATA_CONT rules

SAS requires STP ports use SATA_CONT (SATA did not require this, making it useless). This mandate was intended for native STP initiator ports and STP target ports. It is unclear whether it should also apply to expanders. Are they required to alter the dword stream from a SATA drive to ensure that the STP target port transmitting on its behalf uses SATA_CONT according to the SAS rules?

Proposed answer: no.

7.2.4.3 Continued primitive sequence

Primitives that form continued primitive sequences (e.g., SATA_HOLD) shall be sent two times, then be followed by one or two SATA_CONTs (if needed), then be followed by vendor-specific scrambled data dwords (if needed). Two SATA_CONTs provides more tolerance for bit errors. ALIGNs and NOTIFYs may be sent inside continued primitive sequences as described in 7.2.4.1. After the SATA_CONT, during the vendor-specific scrambled data dwords:

a) a SATA_CONT continues the continued primitive sequence; and
b) any other STP primitive, including the primitive that is being continued, ends the continued primitive sequence.

Figure 6 shows an example of transmitting a continued primitive sequence with two SATA_CONTs.

...

Receivers shall detect a continued primitive sequence after at least one primitive is received. The primitive may be followed by one or more SATA_CONTs, each of which may be followed by vendor-specific data dwords. Receivers shall ignore invalid dwords before, during, or after the SATA_CONT(s). Receivers do not count the number of times the continued primitive, the SATA_CONTs, or the vendor-specific data dwords are received; they are simply in the state of receiving the primitive.

Figure 5 shows an example of receiving a continued primitive sequence.

...

Expanders forwarding dwords are not required to detect an incoming sequence of the same primitive and convert it into a continued primitive sequence.

7.4 Idle physical links

Idle dwords are vendor-specific data dwords.

Phys shall transmit idle dwords if there are no other dwords to transmit and:

a) no connection is open; or
b) an SSP or SMP connection is open.

While an STP connection is open, STP phys transmit SATA_SYNC between frames (see ATA/ATAPI-7 V3). After transmitting two SATA_SYNCs, STP phys shall transmit SATA_CONT and start transmitting idle dwords.

> NOTE 2 SATA devices are allowed but not required to transmit SATA_CONT for primitives that form continued primitive sequences such as SATA_SYNC. Expanders forwarding dwords are not required to detect an incoming sequence of the same primitive and convert it into a continued primitive sequence.

Idle dwords are scrambled (see 7.6).

12 Clause 7.2 - allow expanders to delete invalid dwords rather than replace them

If ALIGNs or NOTIFYs are corrupted, but they are replaced in the data stream by expander(s) with valid ERROR primitives, the elasticity buffer of an expander or the frame recipient could overflow at a later time. If the expander determines that its elasticity buffer is getting full and it is due an ALIGN, it should be allowed to choose to delete the invalid dword (assuming it might have been an ALIGN) rather than replace it with an ERROR.

This could also apply to SATA_ERROR. However, that is only sent on the last link. If the dword that was corrupted was a SAS clock skew management ALIGN, deleting it would avoid unnecessarily corrupting the SATA dword stream. If the dword was an STP initiator phy throttling ALIGN or another dword, forwarding it as an error is appropriate.

7.2.5.7 ERROR

ERROR ~~is~~ may be sent by an expander device when it is forwarding dwords from a SAS physical link or SATA physical link to a SAS physical link and it receives an invalid dword or an ERROR.

See 7.15 for details on error handling by expander devices.

SAS phys may ignore ERROR or treat it as an invalid dword.

7.2.7 Primitives used only inside STP connections and on SATA physical links

7.2.7.1 SATA_ERROR

SATA_ERROR ~~is~~ may be sent by an expander device when it is forwarding dwords from a SAS physical link to a SATA physical link and it receives an invalid dword or an ERROR. SATA_ERROR is an invalid dword.

7.15.10 XL7:Connected state

7.15.10.1 State description

If an invalid dword is received with the Dword Received message and the expander phy is forwarding to an expander phy attached to a SAS physical link, the expander phy shall:

a) send an ERROR primitive with the Transmit Dword request instead of the invalid dword or
b) delete the invalid dword.

If an invalid dword or an ERROR primitive is received with Dword Received message and the expander phy is forwarding to an expander phy attached to a SATA physical link, the expander phy shall:

a) send a SATA_ERROR primitive with the Transmit Dword request instead of the invalid dword or ERROR primitive; or
b) delete the invalid dword or ERROR primitive.

7.15.11 XL8:Close_Wait state

7.15.11.1 State description

If an invalid dword is received with the Dword Received message and the expander phy is forwarding to an expander phy attached to a SAS physical link, the expander phy shall:

a) send an ERROR primitive with the Transmit Dword request instead of the invalid dword; or
b) delete the invalid dword

If an invalid dword or an ERROR primitive is received with Dword Received message and the expander phy is forwarding to an expander phy attached to a SATA physical link, the expander phy shall:

a) send a SATA_ERROR primitive with the Transmit Dword request instead of the invalid dword or ERROR primitive; or
b) delete the invalid dword or ERROR primitive.

13 Clause 7.2 - add invalid dword and ERROR to reasons for a NAK (CRC ERROR)

Frames can be NAKed if they contain an invalid dword, not just if they have CRC errors. A disparity error, for example, can result in a good CRC. NAKing based on the invalid dword may be the only way to report the error on the frame in which it happens.

Also, clarify whether a frame containing an invalid dword shall/should/may generate a NAK. The state machine says shall while the text says may. Proposed as only a should.

7.2.6.5 NAK (Negative acknowledgement)

NAK indicates the negative acknowledgement of an SSP frame and the reason for doing so.

The versions of NAK representing different reasons are defined in table 65.

Table 5 — NAK primitives

Primitive	Description
NAK (CRC ERROR)	The frame had a bad CRC, an invalid dword, or an ERROR primitive.
NAK (RESERVED 0)	Reserved. Processed the same as NAK (CRC ERROR).
NAK (RESERVED 1)	Reserved. Processed the same as NAK (CRC ERROR).
NAK (RESERVED 2)	Reserved. Processed the same as NAK (CRC ERROR).

7.16.3 SSP frame transmission and reception

...

Receiving SSP phys shall acknowledge SSP frames within 1 ms if not discarded as described in 7.16.7.7 with either a positive acknowledgement (ACK) or a negative acknowledgement (NAK). ACK means the SSP frame was received into a frame buffer without errors. NAK (CRC ERROR) means the SSP frame was received with a CRC error, an invalid dword, or an ERROR primitive.

14 Clause 7.2 - Assorted invalid dword changes

SAS requires that frames received with invalid dwords generate a NAK, but allows incoming ERRORs to be either ignored and not generate a NAK, or generate a NAK.

Invalid dwords and ERRORs should be treated the same. There should be no functional difference if the error happened on the immediate physical link, resulting in an invalid dword being received, rather than on a remote physical link, resulting in an ERROR being received (from an expander).

Discussion of “invalid dwords and unexpected primitives” in the SL_CC state machine is inappropriate, since it only acts on messages. The SL receiver should send an Invalid Dword Received message to to be handled. Unexpected messages are ignored by the convention of the state machines.

The “default” crutch in the SL state machine overview about sending idle dwords when there is nothing else to send belongs in the SL transmitter section.

Similar changes are proposed for the XL and SSP state machines. XL shouldn’t handwave on how ERRORs get forwarded through the ECR.

The SMP state machines forgot to mention invalid dwords at all. The same handling as in the other state machines is proposed.

Discussion item: Current defined terms are:

a) **dword** - four characters
b) **data dword** - Dxx.y, Dxx.y, Dxx.y, Dxx.y - no 8b10b problems
c) **primitive** - K28.5 or K28.3, Dxx.y Dxx.y, Dxx.y - no 8b10b problems, but does not necessarily decode to a defined primitive. ERROR itself is just a primitive.
d) **invalid dword** - any dword that is not a data dword or primitive

A suggestion has been received to define the following terms:

a) **dword** - four characters
b) **data dword** - Dxx.y, Dxx.y, Dxx.y, Dxx.y - no 8b10b problems
c) **primitive** - K28.5 or K28.3, Dxx.y Dxx.y, Dxx.y - no 8b10b problems, but does not necessarily decode to a defined primitive. ERROR itself is just a primitive.
d) **invalid data dword** - Dxx.y followed 3 characters where they are not each a perfect Dxx.y, or where Dxx.y itself has a disparity problem
e) **invalid primitive** - K28.5 or K28.3 followed by 3 characters where they are not each a perfect Dxx.y, or where K28.5/K28.3 itself has a disparity problem
f) **invalid dword** - problem in the first character other than disparity, followed by 3 characters with or without problems (i.e. a dword that is not a data dword, primitive, invalid data dword, or invalid primitive).

SP_DWS would have to trigger on invalid dword, invalid primitve, and invalid data dword rather than just invalid dword. Other state machines could make distinctions.

I prefer to use the “shall generate a NAK” policy as aggressively as possible - these intermediate levels might lead to “shall generate a NAK” on an invalid data dword but not on an invalid primitive or invalid dword, which is better than the current “may generate a NAK” for the generic “invalid dword” rule.

This revision of the proposal just uses the currently defined “invalid dword” term, and deletes a few “invalid data dwords” that exist without definition.

7.2.5.7 ERROR

ERROR may be sent by an expander device when it is forwarding dwords from a SAS physical link or SATA physical link to a SAS physical link and it receives an *invalid dword or an ERROR*.

See 7.15 for details on error handling by expander devices.

SAS phys may ignore ERROR.

7.2.6.5 NAK (Negative acknowledgement)

NAK indicates the negative acknowledgement of an SSP frame and the reason for doing so.

The versions of NAK representing different reasons are defined in table 65.

Table 6 — NAK primitives

Primitive	Description
NAK (CRC ERROR)	The frame had a bad CRC, *an invalid dword, or an ERROR primitive*.
...	...

7.2.7.1 SATA_ERROR

SATA_ERROR may be sent by an expander device when it is forwarding dwords from a SAS physical link to a SATA physical link and it receives an *invalid dword or an ERROR*. SATA_ERROR is an invalid dword.

See 6.8 for details on error handling by expander devices.

7.14.2 SL transmitter and receiver

The SL transmitter receives the following messages from the SL state machines specifying primitive sequences, frames, and dwords to transmit:

a) Transmit Idle Dword;
b) Transmit SOAF/Data Dwords/EOAF;
c) Transmit OPEN_ACCEPT;
d) Transmit OPEN_REJECT with an argument indicating the specific type (e.g., Transmit OPEN_REJECT (Retry));
e) Transmit BREAK;
f) Transmit BROADCAST; and
g) Transmit CLOSE with an argument indicating the specific type (e.g., Transmit CLOSE (Normal)).

When there is no outstanding message specifying a dword to transmit, the SL transmitter shall transmit idle dwords.

The SL transmitter sends the following messages ~~from~~ to the SL state machines based on dwords that have been transmitted:

a) SOAF/Data Dwords/EOAF Transmitted.

The SL receiver sends the following messages to the SL state machines indicating primitive sequences and dwords received:

a) SOAF Received;
b) Data Dword Received;
c) EOAF Received;
d) Invalid Dword Received;
e) BROADCAST Received with an argument indicating the specific type (e.g., BROADCAST Received (Change));
f) BREAK Received;
g) OPEN_ACCEPT Received;
h) OPEN_REJECT Received with an argument indicating the specific type (e.g., OPEN_REJECT Received (No Destination));
i) AIP Received; and
j) CLOSE Received with an argument indicating the specific type (e.g., CLOSE Received (Normal)).

The SL receiver shall ignore all other dwords.

...

7.14.4 SL_CC (connection control) state machine

7.14.4.1 SL_CC state machine overview

...

~~Unless otherwise stated within the state description, all invalid dwords and unexpected primitives (i.e., any primitive not described in the description of the SL_CC state) received within any SL state shall be ignored and idle dwords shall be transmitted.~~

7.15 XL (link layer for expander phys) state machine

7.15.1 XL state machine overview

~~...~~

~~Unless otherwise stated within a state description, all invalid dwords and unexpected primitives received within any XL state shall be ignored.~~

7.15.2 XL transmitter and receiver

The XL transmitter receives the following messages from the XL state machine ~~indicating~~ specifying primitive sequences, frames, and dwords to transmit:

a) Transmit Idle Dword;
b) Transmit AIP with an argument indicating the specific type (e.g., Transmit AIP (Normal));
c) Transmit BREAK;
d) Transmit BROADCAST with an argument indicating the specific type (e.g., Transmit BROADCAST (Change));
e) Transmit CLOSE with an argument indicating the specific type (e.g., Transmit CLOSE (Normal));
f) Transmit OPEN_ACCEPT;
g) Transmit OPEN_REJECT, with an argument indicating the specific type (e.g., Transmit OPEN_REJECT (No Destination));
h) Transmit OPEN Address Frame; and
i) Transmit Dword.

The XL transmitter sends the following messages to the XL state machine based on dwords that have been transmitted:

a) a) OPEN Address Frame Transmitted.

...

When there is no outstanding message specifying a dword to transmit, the XL transmitter shall transmit idle dwords.

The XL receiver sends the following messages to the XL state machine indicating primitive sequences, frames, and dwords received:

a) AIP Received with an argument indicating the specific type (e.g., AIP Received (Normal));
b) BREAK Received;
c) BROADCAST Received;
d) CLOSE Received;
e) OPEN_ACCEPT Received;
f) OPEN_REJECT Received;
g) OPEN Address Frame Received;
h) Dword Received; and
i) Invalid Dword Received.

The XL receiver shall ignore all other dwords.

7.15.10 XL7:Connected state

7.15.10.1 State description

...

If:

a) an ~~invalid dword is received with the~~ Invalid Dword Received message is received; and
b) the expander phy is forwarding to an expander phy attached to a SAS physical link,

the expander phy shall:

a) send an ERROR primitive with the Transmit Dword request instead of the invalid dword; or
b) delete the invalid dword.

If:

a) an ~~invalid dword or an~~ ERROR primitive is received with the Dword Received message or an Invalid Dword Received message is received; and
b) the expander phy is forwarding to an expander phy attached to a SATA physical link,

the expander phy shall

a) send a SATA_ERROR primitive with the Transmit Dword request instead of the invalid dword or ERROR primitive; or
b) delete the ERROR primitive or invalid dword.

7.15.11 XL8:Close_Wait state

7.15.11.1 State description

If:

a) an ~~invalid dword is received with the~~ Invalid Dword Received message is received;
b) and the expander phy is forwarding to an expander phy attached to a SAS physical link,

the expander phy shall:

a) send an ERROR primitive with the Transmit Dword request instead of the invalid dword; or
b) delete the invalid dword.

If:

a) an ~~invalid dword or an~~ ERROR primitive is received with the Dword Received message or an Invalid Dword Received message is received; and
b) the expander phy is forwarding to an expander phy attached to a SATA physical link,

the expander phy shall:

a) send a SATA_ERROR primitive with the Transmit Dword request instead of the invalid dword or ERROR primitive; or
b) delete the invalid dword or ERROR primitive.

...

7.16.3 SSP frame transmission and reception

...

Receiving SSP phys shall acknowledge SSP frames within 1 ms if not discarded as described in 7.16.7.7 with either a positive acknowledgement (ACK) or a negative acknowledgement (NAK). ACK means the SSP frame was received into a frame buffer without errors. NAK (CRC ERROR) means the SSP frame was received with a CRC error, an invalid dword, or an ERROR primitive.

7.16.7 SSP (link layer for SSP phys) state machines

7.16.7.1 SSP state machines overview

In Figure 99, add Invalid Dword message into SSP_RF.

7.16.7.2 SSP transmitter and receiver

The SSP transmitter receives the following messages from the SSP state machines ~~indicating~~ specifying primitive sequences and frames to transmit:

a) Transmit RRDY with an argument indicating the specific type (e.g., Transmit RRDY (Normal));
b) Transmit CREDIT_BLOCKED;
c) Transmit ACK;
d) Transmit NAK with an argument indicating the specific type (e.g., Transmit NAK (CRC Error));
e) Transmit Frame (i.e., SOF/data dwords/EOF); and
f) Transmit DONE with an argument indicating the specific type (e.g., Transmit DONE (Normal)).

The SSP transmitter sends the following messages to the SSP state machines based on dwords that have been transmitted:

a) DONE Transmitted;
b) RRDY Transmitted;
c) CREDIT_BLOCKED Transmitted;
d) ACK Transmitted;
e) NAK Transmitted; and
f) Frame Transmitted.

~~When the SSP transmitter is not processing a message to transmit, it shall transmit idle dwords.~~

When there is no outstanding message specifying a dword to transmit, the SSP transmitter shall transmit idle dwords.

The SSP receiver sends the following messages to the SSP state machines indicating primitive sequences and dwords received:

a) ACK Received;
b) NAK Received;
c) RRDY Received;
d) CREDIT_BLOCKED Received;
e) EOF Received;
f) DONE Received with an argument indicating the specific type (e.g., DONE Received (Normal));
g) SOF Received;
h) Data Dword Received;
i) EOF Received; and
j) Invalid Dword Received.

The SSP receiver shall ignore all other dwords.

7.16.7.7 SSP_RF (receive frame control) state machine

...

[this section is already vague, not using the existing message names like Data Dword Received, so no upgrade is proposed for the “invalid dword” references]

If the frame CRC is good and the frame contained no invalid ~~data~~ dwords, this state machine shall send a Frame Received (Successful) message to the SSP_TAN1:Idle state and:

a) if the last Rx Balance Status message received had an argument of Balanced, send a Frame Received (ACK/NAK Balanced) confirmation to the port layer; or
b) if the last Rx Balance Status message received had an argument of Not Balanced, send a Frame Received (ACK/NAK Not Balanced) confirmation to the port layer.

If the frame CRC is bad or the frame contained invalid ~~data~~ dwords, this state machine shall send a Frame Received (Unsuccessful) message to the SSP_TAN1:Idle state.

7.16.7.11 SSP_TAN (transmit ACK/NAK control) state machine

...

Any time this state machine receives a Frame Received (Unsuccessful) message it shall send a Transmit NAK (CRC Error) message to the SSP transmitter.

7.18.4.2 SMP transmitter and receiver

The SMP transmitter receives the following messages from the SMP state machines ~~indicating~~ specifying dwords and frames to transmit:

a) Transmit Idle Dword; and
b) Transmit Frame.

The SMP transmitter sends the following messages to the SMP state machines:

a) Frame Transmitted.

When there is no outstanding message specifying a dword to transmit, the SMP transmitter shall transmit idle dwords.

The SMP receiver sends the following messages to the SMP state machines indicating primitive sequences and dwords received:

a) SOF Received;
b) Dword Received;
c) EOF Received; and
d) Invalid Dword Received.

The SMP receiver shall ignore all other dwords.

7.18.4.3.4 SMP_IP3:Receive_Frame state

This state checks the SMP response frame and determines if the SMP response frame was successfully received (e.g., no CRC error).

If the SMP response frame is received with a CRC error or with an invalid dword, this state shall send a Frame Received (SMP Failure) confirmation to the port layer.

If the number of dwords between the SOF and EOF of the SMP response frame is less than 2, or the number of dwords after an SOF is greater than 258, this state shall send a Frame Received (SMP Failure) confirmation to the port layer. If the SMP response frame is received with no CRC error and the SMP response frame is valid, this state shall:

a) send a Frame Received confirmation to the port layer; and
b) send a Request Close message to the SL state machines (see 7.14).

If an SMP Transmit Break request is received, this state shall send a Request Break message to the SL state machines and terminate.

This state shall request idle dwords be transmitted by repeatedly sending Transmit Idle Dword messages to the SMP transmitter.

7.18.4.4.2 SMP_TP1:Receive_Frame state

7.18.4.4.2.1 State description

This state waits for an SMP frame and determines if the SMP frame was successfully received (e.g., no CRC error).

If an SMP frame is received, this state shall send a Request Break message to the SL state machines (see 7.14) and terminate if:

a) the SMP frame has a CRC error or an invalid dword;
b) the number of data dwords between the SOF and EOF is less than 2; or
c) the number of data dwords after the SOF is greater than 258.

Otherwise, this state shall send a Frame Received confirmation to the port layer.

This state shall request idle dwords be transmitted by repeatedly sending Transmit Idle Dword messages to the SMP transmitter.

15 Clause 7.12 - Expander arbitration rules

The word “available” in unclear, leading to some confusion in the arbitration rules. It needs to be replaced by specific Phy Status message names.

A reference to “partial pathway” should be “blocked partial pathway” in the Arb Reject reason list.

The phrase “that contains at least one” should be “that all contain” in the Arb Reject reason list (see next proposal).

(for Gil Romo, QLogic) The rules for generating Arb Won only specifies what happens when an expander phy has a request and there is a higher priority request for that phy as the destination. It doesn’t mention that when two or more peer requests contend for the same destination phy only one must be selected.

7.12.4 Arbitration and resource management in an expander device

7.12.4.1 Arbitration overview

The ECM shall arbitrate and assign or deny path resources for ~~connection attempts requested by~~ Request Path requests from each expander phy ~~in response to receiving valid OPEN address frames~~.

...

The ECM shall generate the Arb Reject confirmation when any of the following conditions are met and all the Arb Won conditions are not met:

a) Arb Reject (No Destination) or Arb Reject (Bad Destination) if the connection request does not map to ~~a valid~~ an expander phy that is not part of the same expander port as the requesting expander phy (i.e., there is no direct routing or table routing match and there is no subtractive phy);
b) Arb Reject (Bad Connection Rate) if the connection request ~~specifies an unsupported~~ does not map to any expander phy that supports the connection rate; or
c) Arb Reject (Pathway Blocked) if the connection request ~~specifies a destination port that contains at least one partial pathway~~ maps to expander phys that all contain blocked partial pathways (i.e., are returning Phy Status (Blocked Partial Pathway)) and pathway recovery rules require this connection request to release path resources.

Editor’s Note 2: note the “partial pathway” to “blocked partial pathway” change in c)

Editor’s Note 3: note the “at least one” to “all” change in c)

The ECM shall generate the Arb Lost confirmation when all of the following conditions are met:

a) the connection request maps to an ~~available~~ expander phy ~~at a supported~~ that:
   A) supports the connection rate; and
   B) is not reporting a Phy Status (Partial Pathway), Phy Status (Blocked Partial Pathway), or Phy Status (Connection) response unless that expander phy is arbitrating for the expander phy making this connection request;
b) there are sufficient routing resources to complete the connection request; and
c) the destination expander phy of this connection request has received a higher priority OPEN address frame with this expander phy as its destination (i.e., when two expander phys both receive an OPEN address frame destined for each other, the ECM shall provide the Arb Lost confirmation to the expander phy that received the lowest priority OPEN address frame).

The ECM shall generate the Arb Won confirmation when all of the following conditions are met:

a) the connection request maps to an ~~available~~ expander phy ~~at a supported~~ that:
   A) supports the connection rate; and

B) is not reporting a Phy Status (Partial Pathway), Phy Status (Blocked Partial Pathway), or Phy Status (Connection) response, unless that expander phy is arbitrating for the expander phy making this connection request;

b) there are sufficient routing resources to complete the connection request~~.~~; ~~and~~

c) no higher priority connection requests are present with this expander phy as the destination; and

d) the connection request is chosen as the highest priority connection request in the expander device mapping to the specified destination expander phy.

16 Clause 7.12 - change “at least one” to “all” in Partial Pathway Timer wording

One sentence was not changed by “03-346r2 Pathway Recover Priority corrections” that needed to be changed to match the other changes. The timer does not need to start until all phys are blocked. Some of them might be connected, which should cause everything to clear when a connection is closed. (also see previous proposal regarding Arb Reject).

7.12.4.3 Partial Pathway Timeout timer

Each expander phy shall maintain a Partial Pathway Timeout timer. This timer is used to identify potential deadlock conditions and to request resolution by the ECM. An expander phy shall initialize the Partial Pathway Timeout timer to the partial pathway timeout value it reports in the SMP DISCOVER function (see 10.4.3.5) and run the Partial Pathway Timeout timer whenever the ECM provides confirmation to the expander phy that all expander phys within the requested destination port are blocked waiting on partial pathways.

NOTE 3 The partial pathway timeout value allows flexibility in specifying how long an expander device waits before attempting pathway recovery. The recommended default value (see 10.4.3.5) was chosen to cover a wide range of topologies. Selecting small partial pathway timeout value values within a large topology may compromise performance because of the time a device waits after receiving OPEN_REJECT (PATHWAY BLOCKED) before it retries the connection request. Similarly, selecting large partial pathway timeout value values within a small topology may compromise performance due to waiting longer than necessary to detect pathway blockage.

When the Partial Pathway Timeout timer is not running, an expander phy shall initialize and start the Partial Pathway Timeout timer when ~~all of the following conditions are met:~~

~~a) there are no unallocated expander phys within a requested destination port available to complete the connection; and~~

b) ~~at least one~~ all expander phy~~s~~ within the requested destination port contain~~s~~ a blocked partial pathway (i.e., are returning Phy Status (Blocked Partial Pathway)).

NOTE 4 The Partial Pathway Timeout timer is not initialized and started if one or more of the expander phys within a requested destination port is being used for a connection.

Editor’s Note 4: with b) changed to all, a) is kind of meaningless

---

When one of the conditions above is not met, the expander phy shall stop the Partial Pathway Timeout timer. If the timer expires, pathway recovery shall occur (see 7.12.4.4).

17 Clause 7.16 - handling RRDY after CREDIT_BLOCKED

Clarify handling of CREDIT_BLOCKED under some error conditions.

1. CREDIT_BLOCKED shall not be sent while previous RRDY credits are still available for use.
2. The sender shall not send any additional RRDYs after CREDIT_BLOCKED.
3. If a receiver sees CREDIT_BLOCKED at unexpected times, it shall honor it.
4. If a receiver sees RRDY after CREDIT_BLOCKED, it should ignore it.

Also, delete “Credit” from the argument names of Tx/Rx Credit Status (Credit Whatever) to make the usage consistent (usually they were sent with “Credit” in the name but received without it in the name).

7.2.6.2 CREDIT_BLOCKED

CREDIT_BLOCKED is transmitted when no credit is currently extended and indicates that no more credit is going to be sent during this connection.

See 7.16.4 for details on SSP flow control.

7.2.6.6 RRDY (Receiver ready)

RRDY is used to increase SSP frame credit.

The versions of RRDY representing different reasons are defined in table 66.

...

A phy shall not transmit RRDY after transmitting CREDIT_BLOCKED in a connection. See 7.16.4 for details on SSP flow control.

7.16.4 SSP flow control

SSP phys grant credit for permission to transmit frames with RRDY. Each RRDY increments credit by one frame. Frame transmission decrements credit by one frame. Credit of zero frames is established at the beginning of each connection.

SSP phys shall not increment credit past 255 frames.

To prevent deadlocks where an SSP initiator port and SSP target port are both waiting on each other to provide credit, an SSP initiator port shall never refuse to provide credit by withholding RRDY because it needs to transmit a frame itself. It may refuse to provide credit for other reasons (e.g., temporary buffer full conditions).

An SSP target port may refuse to provide credit for any reason, including because it needs to transmit a frame itself.

When credit is zero, SSP phys that are going to be unable to provide credit for 1 ms shall send CREDIT_BLOCKED. After sending CREDIT_BLOCKED, they shall not send any additional RRDYs.

7.16.5 Interlocked frames

...

An SSP phy may transmit primitives responding to traffic it is receiving (e.g. an ACK or NAK to acknowledge an SSP frame, an RRDY to grant more receive credit, or a CREDIT_BLOCKED to indicate credit is not forthcoming) while waiting for an interlocked frame it transmitted to be acknowledged. These primitives may also be interspersed within an SSP frame.

...

7.16.6 Closing an SSP connection

DONE shall be exchanged prior to closing an SSP connection (see 8.2.2.3.5). There are several versions of the DONE primitive indicating additional information about why the SSP connection is being closed:

a) DONE (NORMAL) indicates normal completion; the transmitter has no more SSP frames to transmit;
b) DONE (CREDIT TIMEOUT) indicates the transmitter still has SSP frames to transmit, but did not receive an RRDY granting frame credit within 1 ms or received a CREDIT_BLOCKED; and

c) DONE (ACK/NAK TIMEOUT) indicates the transmitter transmitted an SSP frame but did not receive the corresponding ACK or NAK within 1 ms. As a result, the ACK/NAK count is not balanced and the transmitter is going to transmit a BREAK in 1 ms unless the recipient replies with DONE and the connection is closed.

After transmitting DONE, the transmitting phy initializes and starts a 1 ms DONE Timeout timer (see 7.16.7.5).

After transmitting DONE, the transmitting phy shall not transmit any more SSP frames during this connection.

However, the phy may transmit ACK, NAK, RRDY, and CREDIT_BLOCKED after transmitting DONE if the other phy is still transmitting SSP frames in the reverse direction. Once an SSP phy has both transmitted and received DONE, it shall close the connection by transmitting CLOSE (NORMAL) (see 7.12.7).

...

7.16.7.4 SSP_TCM (transmit frame credit monitor) state machine

The SSP_TCM state machine's function is to ensure that credit is available ~~from the originator~~ before a frame is transmitted. This state machine consists of one state.

This state machine shall keep track of the number of transmit frame credits available ~~received versus the number of transmit frame credits used~~. This state machine adds one transmit frame credit for each RRDY Received message received and subtracts one transmit frame credit for each Tx Credit Used message received. ~~This state machine shall remember any~~ The CREDIT_BLOCKED Received message ~~that is received~~ indicates that transmit frame credit is blocked.

When transmit frame credit is available, this state machine shall send the Tx Credit Status (~~Credit~~ Available) message to the SSP_TF2:Tx_Wait state.

When transmit frame credit is not available and transmit frame credit is not blocked, this state machine shall send the Tx Credit Status (~~Credit~~ Not Available) message to the SSP_TF2:Tx_Wait state.

When transmit frame credit is not available and transmit frame credit is blocked, this state machine shall send the Tx Credit Status (~~Credit~~ Blocked) message to the SSP_TF2:Tx_Wait state.

When this state machine receives an Enable Disable SSP (Enable) message, Request Close message, or Request Break message transmit frame credit shall be set to not available and transmit frame credit shall not be blocked.

7.16.7.6.3 SSP_TF2:Tx_Wait state

7.16.7.6.3.1 State description

This state monitors the Tx Balance Status message and the Tx Credit Status message to ensure that frames are transmitted and connections are closed at the proper time.

If this state is entered from the SSP_TF1:Connected_Idle state with a Transmit Frame Balance Required argument or a Transmit Frame Balance Not Required argument, and:

a) if the last Tx Credit Status message received had an argument of Not Available this state shall initialize and start the Credit Timeout timer; or
b) if the last Tx Credit Status message had an argument other than Not Available this state shall stop the Credit Timeout timer.

7.16.7.6.3.2 Transition SSP_TF2:Tx_Wait to SSP_TF3:Indicate_Frame_Tx

This transition shall occur if this state was entered from the SSP_TF1:Connected_Idle state with an argument of Transmit Frame Balance Required if:

a) the last Tx Balance Status message received had an argument of Balanced; and
b) the last Tx Credit Status message received had an argument of ~~Credit~~ Available.

This transition shall occur if this state was entered from the SSP_TF1:Connected_Idle state with an argument of Transmit Frame Balance Not Required and if the last Tx Credit Status message received had an argument of ~~Credit~~ Available.

This transition shall occur after sending a Tx Credit Used message to the SSP_TCM state machine.

7.16.7.6.3.3 Transition SSP_TF2:Tx_Wait to SSP_TF4:Indicate_DONE_Tx

This transition shall occur and include an ACK/NAK Timeout argument if an ACK/NAK Timeout message is received.

This transition shall occur and include a Close Connection argument if:

a) this state was entered from the SSP_TF1:Connected_Idle state with an argument of Close Connection; and
b) the last Tx Balance Status message received had an argument of Balanced.

This transition shall occur and include a Credit Timeout argument if:

a) this state was entered from the SSP_TF1:Connected_Idle state with a Transmit Frame Balance Required argument or a Transmit Frame Balance Not Required argument;
b) the Credit Timeout timer expired before a Tx Credit Status message was received with an argument of Available, or the last Tx Credit Status message received had an argument of Blocked;
c) a Tx Balance Status message was received with an argument of Balanced (i.e., the Credit Timeout argument shall not be included in this transition for this reason unless the ACK/NAK count is balanced); and
d) an ACK/NAK Timeout message was not received.

7.16.7.7 SSP_RF (receive frame control) state machine

The SSP_RF state machine's function is to receive frames and determine whether or not those frames were received successfully. This state machine consists of one state.

This state machine:

a) checks the frame to determine if the frame should be accepted or discarded;
b) checks the frame to determine if an ACK or NAK should be transmitted; and
c) sends a Frame Received confirmation to the port layer.

The frame (i.e., all the dwords between an SOF and EOF) shall be discarded if any of the following conditions are true:

a) the number of data dwords between the SOF and EOF is less than 7;
b) the number of data dwords after the SOF is greater than 263 data dwords;
c) the Rx Credit Status (~~Credit~~ Exhausted) message is received; or
d) the DONE Received message is received.

If consecutive SOF Received messages are received without an intervening EOF Received message (i.e., SOF, data dwords, SOF, data dwords, and EOF instead of SOF, data dwords, EOF, SOF, data dwords, and EOF) then this state machine shall discard all dwords between those SOFs.

If the frame is discarded then no further action is taken by this state machine. If the frame is not discarded then this state machine shall:

a) send a Frame Received message to the SSP_RCM state machine; and
b) send a Frame Received message to the SSP_RIM state machine;

7.16.7.8 SSP_RCM (receive frame credit monitor) state machine

The SSP_RCM state machine's function is to ensure that there was credit given to the originator for every frame that is received. This state machine consists of one state.

This state machine monitors the receiver's resources and keeps track of the number of RRDYs transmitted versus the number of frames received.

Any time resources are released or become available, if this state machine has not sent the Rx Credit Control (Blocked) message to the SSP_TC state machine and the SSP_D state machine, this state machine shall send the Rx Credit Control (Available) message to the SSP_TC state machine. This state machine shall only send the Rx Credit Control (Available) message to the SSP_TC state machine after frame receive resources become available. The specifications for when or how resources become available is outside the scope of this standard.

This state machine may send the Rx Credit Control (Blocked) message to the SSP_TC state machine and the SSP_D state machine indicating that no resources are available and no more receive frame credit is going to be sent during this connection. After sending the Rx Credit Control (Blocked) message to the SSP_TC state machine and the SSP_D state machine, this state machine shall not send the Rx Credit Control (Available) message to the SSP_TC state machine or the SSP_D state machine for the duration of the current connection. The Rx Credit Control (Blocked) message should be sent to the SSP_TC state machine and the SSP_D state machine when no further receive frame credit is going to become available within a credit timeout (i.e., less than 1 ms).

This state machine shall indicate through the Rx Credit Control message only the amount of resources available to handle received frames (e.g., if this state machine has resources for 5 frames the maximum number of Rx Credit Control requests with the Available argument outstanding is 5).

This state machine shall use the Credit Transmitted message to keep track of the number of RRDYs transmitted. This state machine shall use the Frame Received message to keep a track of the number of frames received.

Any time the number of Credit Transmitted messages received exceeds the number of Frame Received messages received this state machine shall send a Rx Credit Status (~~Credit~~ Extended) message to the SSP_RF state machine and the SSP_D state machine.

Any time the number of Credit Transmitted messages received equals the number of Frame Received messages received this state machine shall send a Rx Credit Status (~~Credit~~ Exhausted) message to the SSP_RF state machine and the SSP_D state machine.

If this state machine receives an Enable Disable SSP (Enable) message, Request Close message, or Request Break message the frame receive resources shall be initialized to the no credit value for the current connection.

7.16.7.10 SSP_TC (transmit credit control) state machine

The SSP_TC state machine's function is to control the sending of requests to transmit an RRDY or CREDIT_BLOCKED. This state machine consists of one state.

Any time this state machine receives a Rx Credit Control (Available) message it shall send a number of Transmit RRDY (Normal) messages to the SSP transmitter as indicated by the amount of resources available to handle received frames (e.g., if the Available argument indicates 5 RRDYs are to be transmitted this state machine sends 5 Transmit RRDY (Normal) messages to the SSP transmitter).

Any time this state machine receives a RRDY Transmitted message it shall send a Credit Transmitted message to the SSP_RCM state machine.

Any time this state machine receives a Rx Credit Control (Blocked) message it shall send a Transmit CREDIT_BLOCKED message to the SSP transmitter.

18 Clause 7.17 - clarify LINK RESET does not clear affiliation

A link reset sequence caused by a LINK RESET phy operation does not clear an affiliation. This is not mentioned in the affiliation clearing list.

7.17.4 Affiliations

Coherent access to the SATA task file registers shall be provided for each STP initiator port. STP target ports that do not track all commands by the STP initiator ports' SAS addresses shall implement affiliations to provide coherency. STP target ports that track all commands by the STP initiator ports' SAS addresses shall not implement affiliations.

An affiliation is a state entered by an STP target port where it refuses to accept connection requests from STP initiator ports other than the one that has established an affiliation.

An STP target port that supports affiliations shall establish an affiliation whenever it accepts a connection request. When an affiliation is established, the STP target port shall reject all subsequent connection requests from other STP initiator ports with OPEN_REJECT (STP RESOURCES BUSY).

An STP target port shall maintain an affiliation until any of the following occurs:

a) power on;
b) the SAS target device receives an SMP PHY CONTROL request specifying the phy with the affiliation and specifying a phy operation of HARD RESET (see 10.4.3.10) from any SMP initiator port;
c) the SAS target device receives an SMP PHY CONTROL request specifying the phy with the affiliation and specifying a phy operation of CLEAR AFFILIATION from the same SAS initiator port that has the affiliation;
d) a connection to the phy with the affiliation is closed with CLOSE (CLEAR AFFILIATION); or
a) the STP target port is part of a STP/SATA bridge and a link reset sequence is begun on the SATA physical link that was not requested by an SMP PHY CONTROL request specifying the phy and specifying a phy operation of LINK RESET (see 10.4.3.10).

19 Clause 7.17 - send SATA_SYNC immediately after OPEN_ACCEPT

Figure 103 shows after sending OPEN_ACCEPT, an STP initiator continues to send idle dwords before sending SATA_R_RDY. Section 7.17.8 explicitly allows this.

Since it had to decode the protocol to choose to send the OPEN_ACCEPT, it should be ready to start sending SATA_SYNC and should not need to insert any more idle dwords. Require both STP targets and STP initiators do so.

Also, redraw figure 103 so it flows left to right rather than right to left.

7.17.5 Opening an STP connection

If no STP connection exists when the SATA host port in an STP/SATA bridge receives a SATA_X_RDY from the attached SATA device port, the STP target port in the STP/SATA bridge shall establish an STP connection to the appropriate STP initiator port before it transmits a SATA_R_RDY to the SATA device.

Wide STP initiator ports shall not request more than one connection at a time to an STP target port. Wide STP target ports shall not request more than one connection at a time to an STP initiator port.

While a wide STP target port is waiting for a response to a connection request or has established a connection to an STP initiator port, it shall:

a) reject incoming connection requests from that STP initiator port with OPEN_REJECT (RETRY); and
b) if affiliations are supported, reject incoming connection requests from other STP initiator ports with OPEN_REJECT (STP RESOURCES BUSY),

While a wide STP initiator port is waiting for a response to a connection request to an STP target port, it shall not reject an incoming connection request from that STP target port because of its outgoing connection request. It may reject incoming connection requests for other reasons (see 7.2.5.11).

The first dword that an STP phy sends inside an STP connection after OPEN_ACCEPT that is not an ALIGN or NOTIFY shall be an STP primitive (e.g., SATA_SYNC).

7.17.7 STP connection management examples

...

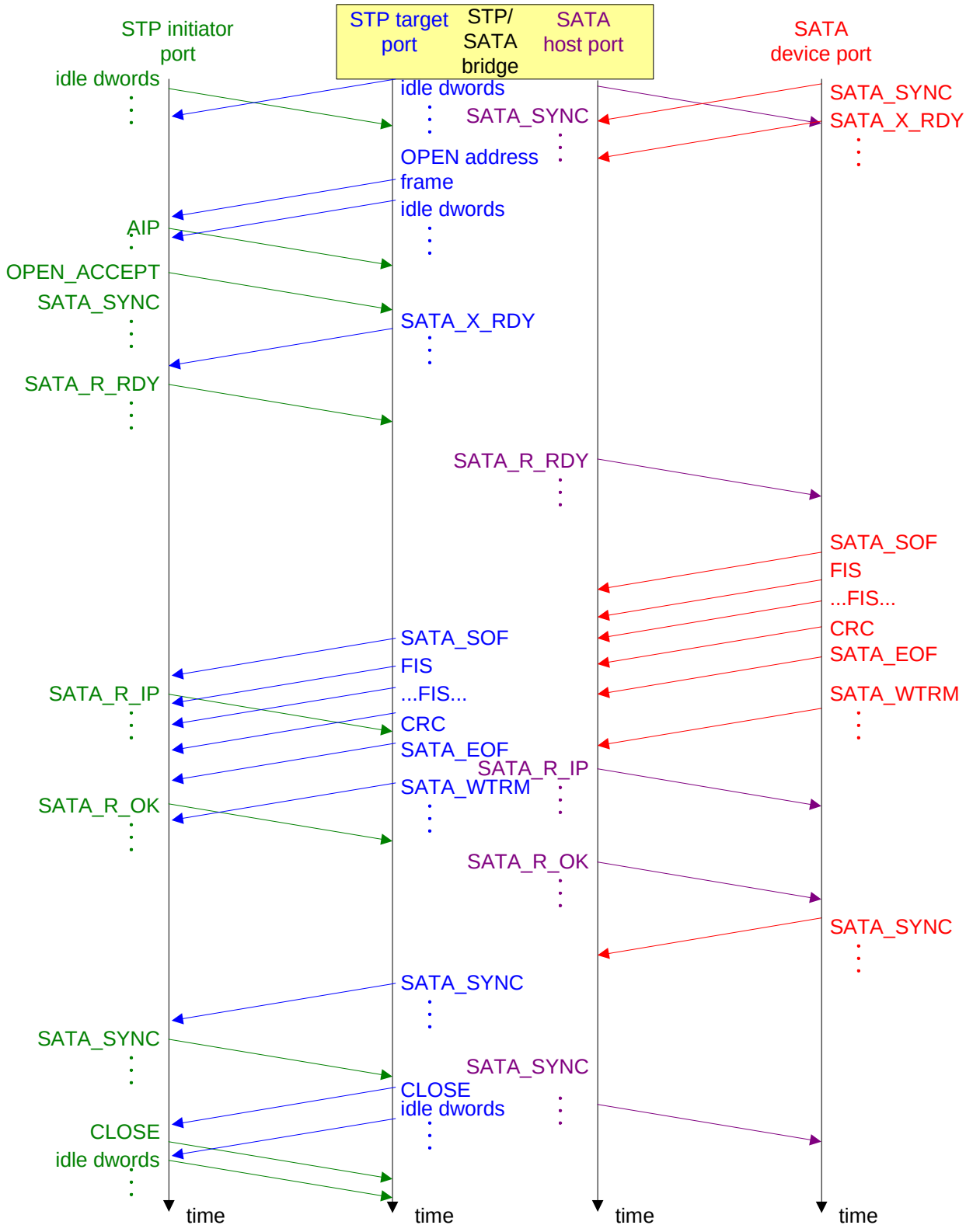

Figure 103 — STP target port opening an STP connection [modified]

[changed idle dwords to SATA_SYNC in what is now the top left]

7.17.8 STP (link layer for STP phys) state machines

The STP link layer uses the SATA link layer state machines (see ATA/ATAPI-7 V3), modified to:

a) communicate with the port layer rather than directly with the transport layer;
b) interface with the SL state machines for connection management (e.g., to select when to open and close STP connections, and to tolerate idle dwords between an OPEN address frame ~~or an OPEN_ACCEPT~~ and the first SATA primitive); and
c) implement affiliations (see 7.17.4).

These modifications are not described in this standard.

20 Clause 7.18 - SMP receipt of consecutive SOFs

If the SMP state machine receives a second SOF before receiving the first EOF, it's not clear if the frame reception silently restarts or if this is treated as an error. It should be treated like SSP - ignore all the dwords from the first to second SOF.

7.18.4.3.4 SMP_IP3:Receive_Frame state

This state checks the SMP response frame and determines if the SMP response frame was successfully received (e.g., no CRC error).

If the SMP response frame is received with a CRC error, this state shall send a Frame Received (SMP Failure) confirmation to the port layer.

If the number of dwords between the SOF and EOF of the SMP response frame is less than 2, or the number of dwords after an SOF is greater than 258, this state shall send a Frame Received (SMP Failure) confirmation to the port layer. If the SMP response frame is received with no CRC error and the SMP response frame is valid, this state shall:

a) send a Frame Received confirmation to the port layer; and
b) send a Request Close message to the SL state machines (see 7.14).

If consecutive SOF Received messages are received without an intervening EOF Received message (i.e., SOF, data dwords, SOF, data dwords, and EOF instead of SOF, data dwords, EOF, SOF, data dwords, and EOF) then this state machine shall discard all dwords between those SOFs.

If an SMP Transmit Break request is received, this state shall send a Request Break message to the SL state machines and this state machine shall terminate.

This state shall request idle dwords be transmitted by repeatedly sending Transmit Idle Dword messages to the SMP transmitter.

7.18.4.4.2 SMP_TP1:Receive_Frame state

7.18.4.4.2.1 State description

This state waits for an SMP frame and determines if the SMP frame was successfully received (e.g., no CRC error).

If an SMP frame is received, this state shall send a Request Break message to the SL state machines (see 7.14) and this state machine shall terminate if:

a) the SMP frame has a CRC error;
b) the number of data dwords between the SOF and EOF is less than 2; or
c) the number of data dwords after the SOF is greater than 258.

Otherwise, this state shall send a Frame Received confirmation to the port layer.

If consecutive SOF Received messages are received without an intervening EOF Received message (i.e., SOF, data dwords, SOF, data dwords, and EOF instead of SOF, data dwords, EOF, SOF, data dwords, and EOF) then this state machine shall discard all dwords between those SOFs.

This state shall request idle dwords be transmitted by repeatedly sending Transmit Idle Dword messages to the SMP transmitter.

21 Clause 8 - handling phy loss during frame transmission

If a phy goes down during a connection, the link layer sends up a Phy Disabled confirmation. If the port layer sees this while in the Connected state after it sends a frame, before receiving an ACK or NAK, it needs to send Transmission (Connection Lost Without ACK/NAK) upstream, just as if the connection were closed at that time with BREAK.

If it sees this during sending a frame, it needs to send a Retry Frame upstream, just as if the connection were closed at that time with BREAK.

8.2.3.4 PL_PM3:Connected state

8.2.3.4.1 PL_PM3:Connected state description

If this state receives a Close Connection message from the PL_OC state machine, then this state shall send a Close Connection request to the link layer.

If this state receives a Connection Closed confirmation or a Phy Disabled confirmation after sending a Transmission Status (Frame Transmitted) confirmation, but before this state receives an ACK Received or NAK Received confirmation, then this state shall send a Transmission Status (Connection Lost Without ACK/NAK) confirmation to the transport layer.

If this state receives a Connection Closed confirmation or a Phy Disabled confirmation after sending a Transmit Frame request but before receiving a Frame Transmitted confirmation, then this state shall send a Retry Frame message to the PL_OC state machine.

...

22 Clause 9.2 - combine Sense Data and Response Data fields

In the RESPONSE frame, the SENSE DATA and RESPONSE DATA fields cannot both be present at the same time due to how DATAPRES is defined. The description is confusing because it shows both fields. If both were somehow present, the row numbers would not be correct for the SENSE DATA field. Combine them into one field.

9.2.2.5.1 RESPONSE information unit overview

...

Change the RESPONSE frame definition to:

Table 7 — RESPONSE information unit

Byte\Bit	7	6	5	4	3	2	1	0
0	Reserved							
9								
10	Reserved						DATAPRES	
11	STATUS							
12	Reserved							
15								
16	(MSB)	SENSE DATA LENGTH (n bytes)						
19								(LSB)
20	(MSB)	RESPONSE DATA LENGTH (m bytes)						
23								(LSB)
~~20~~	~~RESPONSE DATA~~							
~~23~~								
~~24~~	~~SENSE DATA~~							
~~23+m~~								
24	SENSE OR RESPONSE DATA							
23+m or 23+n								

Table 8 defines the DATAPRES field, which indicates the format and content of the STATUS field, SENSE DATA LENGTH field, RESPONSE DATA LENGTH field, ~~RESPONSE DATA field,~~ and SENSE OR RESPONSE DATA field.

Table 8 — DATAPRES field

Code	Name	Description	Reference
00b	NO DATA	No data present	
01b	RESPONSE_DATA	Response data present	
10b	SENSE_DATA	Sense data present	
11b	Reserved		

The SSP target port shall return a RESPONSE frame with the DATAPRES field set to NO_DATA if a command completes without sense data to return.

The SSP target port shall return a RESPONSE frame with the DATAPRES field set to RESPONSE_DATA in response to every TASK frame and in response to errors that occur while the transport layer is processing a COMMAND frame.

The SSP target port shall return a RESPONSE frame with the DATAPRES field set to SENSE_DATA if a command completes with sense data to return (e.g., CHECK CONDITION status).

If the DATAPRES field is set to a reserved value, then the SSP initiator port shall discard the RESPONSE frame.

9.2.2.5.2 RESPONSE information unit NO_DATA format

If the DATAPRES field is set to NO_DATA, then:

a) the STATUS field shall contain the status code for a command that has ended (see SAM-3 for a list of status codes);
b) the SENSE DATA LENGTH field and the RESPONSE DATA LENGTH field shall be set to zero and shall be ignored by the SSP initiator port; and
c) the SENSE OR RESPONSE DATA field ~~and the RESPONSE DATA field~~ shall not be present.

9.2.2.5.3 RESPONSE information unit RESPONSE_DATA format

If the DATAPRES field is set to RESPONSE_DATA, then:

a) the STATUS field and the SENSE DATA LENGTH field shall be set to zero and shall be ignored by the SSP initiator port;
~~b) the SENSE DATA field shall not be present;~~
c) the RESPONSE DATA LENGTH field shall be set to four. Other lengths are reserved for future standardization; and
d) the SENSE OR RESPONSE DATA field shall be ~~present~~contain response data.

Table 9 defines the response data contained in the SENSE OR RESPONSE DATA field, which contains information describing protocol failures detected during processing of a request received by the SSP target port. The SENSE OR RESPONSE DATA field shall ~~be present~~ contain response data if the SSP target port detects any of the

conditions described by a non-zero RESPONSE CODE value and shall be present for a RESPONSE frame sent in response to a TASK frame.

Table 9 — SENSE OR RESPONSE DATA **field containing response data**

Byte\Bit	7	6	5	4	3	2	1	0
0	Reserved							
1	Reserved							
2	Reserved							
3	RESPONSE CODE							

Table 10 defines the RESPONSE CODE field, which indicates the error condition or the completion status of a task management function. See 10.2.1.5 and 10.2.1.13 for the mapping of these response codes to SCSI service responses.

Table 10 — RESPONSE CODE **field**

Code	Description
00h	TASK MANAGEMENT FUNCTION COMPLETE [a]
02h	INVALID FRAME
04h	TASK MANAGEMENT FUNCTION NOT SUPPORTED [a]
05h	TASK MANAGEMENT FUNCTION FAILED [a]
08h	TASK MANAGEMENT FUNCTION SUCCEEDED [a]
09h	INVALID LOGICAL UNIT NUMBER [a]
All others	Reserved

[a] Only valid when responding to a TASK frame

9.2.2.5.4 RESPONSE information unit SENSE_DATA format

If the DATAPRES field is set to SENSE_DATA, then:

a) the STATUS field shall contain the status code for a command that has ended (see SAM-3 for a list of status codes);
b) the RESPONSE DATA LENGTH field shall be set to zero and shall be ignored by the initiator;
~~c) the RESPONSE DATA field shall not be present;~~
d) the SENSE DATA LENGTH field shall be set to a non-zero value indicating the number of bytes in the SENSE OR RESPONSE DATA field. The value in the SENSE DATA LENGTH field shall not be larger than 1 000 (see table 97); and
e) the SENSE OR RESPONSE DATA field shall contain sense data (see SAM-3).

The value in the SENSE DATA LENGTH field need not be a multiple of four. If it is not, the NUMBER OF FILL BYTES field in the SSP frame header is non-zero and fill bytes are present.

23 Clause 9.3 - handling multiple initial SATA FISes

Describe what expanders are supposed to do if the SATA drive tries to send a second frame after delivering the initiaal Register FIS - don't accept it.

9.3 STP transport layer

9.3.1 Initial FIS

A SATA device phy transmits a Register - Device to Host FIS after completing the link reset sequence. The expander device shall update a set of shadow registers with these contents and shall not deliver them to any STP initiator port. The STP initiator ports may read the shadow register contents using the SMP REPORT PHY SATA function (see 10.4.3.7).

After the Register - Device to Host FIS is accepted, if the SATA device sends a SATA_X_RDY before an affiliation is established, the expander device shall not send SATA_R_RDY.

24 Clause 10 - SATA spinup hold begins at COMSAS Timeout Detected

SATA spinup hold needs to begin after COMSAS Detect Timeout, not when the SATA OOB sequence completes, which is after the host sends COMWAKE. A SATA drive may start to spin up if it gets that far.

6.9 Spin-up

If a SAS target device receives COMSAS during the reset sequence, it shall not spin-up until allowed by the SA_PC state machine (see 10.2.8).

If a SAS target device supporting SATA does not receive COMSAS during the reset sequence, it shall follow SATA spin-up rules (see ATA/ATAPI-7 V3).

Expander devices attached to SATA devices may halt the automatic phy reset sequence after the COMSAS Detect Timeout (see 6.7) to delay spin-up; this is called SATA spinup hold. This is reported in the SMP DISCOVER function (see 10.4.3.5) and is released with the SMP PHY CONTROL function (see 10.4.3.10).

...

10.4.3.5 DISCOVER function

...

Table 156 — Negotiated physical link rate

3h: Phy is enabled; detected a SATA device and entered the SATA spinup hold state. The LINK RESET and HARD RESET operations in the SMP PHY CONTROL function (see 10.4.3.10) may be used to release the phy. This field shall be updated to this value at SATA spin-up hold time (see 6.9) (i.e., after the COMSAS Detect Timeout timer expires during the SATA OOB sequence ~~completes~~) if SATA spinup hold is supported.

Table 157 — ATTACHED SATA PORT SELECTOR **and** ATTACHED SATA DEVICE **bits**

b This bit shall be updated ~~after the SATA OOB sequence completes and before the SATA speed negotiation sequence begins (i.e.,~~ at SATA spin-up hold time (see 6.9)~~)~~.

...

~~If the ATTACHED DEVICE TYPE field is set to 000b (i.e., no device attached), the ATTACHED SAS ADDRESS field is invalid.~~ The ATTACHED SAS ADDRESS field shall be updated:

a) after the identification sequence completes, if a SAS device or expander device is attached; or
b) ~~after the SATA OOB sequence completes~~at SATA spin-up hold time (see 6.9), if a SATA device is attached.

25 Clause 10 - honor programmed link rate changes for LINK RESET/HARD RESET

Clarify that new PROGRAMMED MINIMUM/MAXIMUM PHYSICAL LINK RATE values are honored for the phy operation specified in the same PHY CONTROL request.

10.4.3.10 PHY CONTROL function

The PROGRAMMED MINIMUM PHYSICAL LINK RATE field specifies the minimum physical link rate the phy shall support during a link reset sequence (see 4.4.1). Table 171 defines the values for this field. If this field is changed along with a phy operation of LINK RESET or HARD RESET, that phy operation shall utilize the new value for this field.

The PROGRAMMED MAXIMUM PHYSICAL LINK RATE field specifies the maximum physical link rates the phy shall support during a link reset sequence (see 4.4.1).Table 171 defines the values for this field. If this field is changed along with a phy operation of LINK RESET or HARD RESET, that phy operation shall utilize the new value for this field.

26 Clause 10 - handling errors in programmed link rate changes

If the PROGRAMMED MINIMUM/MAXIMUM PHYSICAL LINK RATE fields are set to invalid values, what happens?

a) an error
b) no error; both current values maintained unchanged
c) no error erroneous value ignored (dangerous; could leave to minimum above maximum)

If an error is returned, does returning the error have priority over running the selected phy operation or does it prevent the phy operation from running?

Proposed: both values shall remain unchanged. Error may optionally be returned.

10.4.3.10 PHY CONTROL function

The PROGRAMMED MINIMUM PHYSICAL LINK RATE field specifies the minimum physical link rate the phy shall support during a link reset sequence (see 4.4.1). Table 171 defines the values for this field. If this field is changed along with a phy operation of LINK RESET or HARD RESET, that phy operation shall utilize the new value for this field.

The PROGRAMMED MAXIMUM PHYSICAL LINK RATE field specifies the maximum physical link rates the phy shall support during a link reset sequence (see 4.4.1).Table 171 defines the values for this field. If this field is changed along with a phy operation of LINK RESET or HARD RESET, that phy operation shall utilize the new value for this field.

If the PROGRAMMED MINIMUM PHYSICAL LINK RATE field or the PROGRAMMED MAXIMUM PHYSICAL LINK RATE field is set to an unsupported or reserved value, or the PROGRAMMED MINIMUM PHYSICAL LINK RATE field and PROGRAMMED MAXIMUM PHYSICAL LINK RATE field are set to an invalid combination of values (e.g., the minimum is greater than the maximum), the SMP target port shall not change either of their values and may return a function result of SMP FUNCTION FAILED in the response frame. If it does so, it shall not perform the requested phy operation.

27 Clause 7 - add confirmation when SL rejects an incoming OPEN

If the SL state machine is sending an OPEN and receives one, SL_CC1 decides whether to honor the incoming one or not based on AIP and crossing-on-the-wire priority. If it chooses to accept the inbound OPEN, it moves to SL_CC2. SL_CC2 may choose to send an OPEN_REJECT, however. It does not send any confirmation to the port layer if this happens.

This causes the original request to be lost. The port layer is not notified to retry it and the link layer does not retry it. It needs to go back to the port layer, which may or may not choose to retry that request immediately.

Suggested corrections:

a) SL_CC sends (new) Inbound Connection Rejected confirmation to PL_PM while in SL_CC2 if an incoming request is rejected
b) PL_PM sends Retry Open (Collided) to PL_OC if it receives Inbound Connection Rejected while in PL_PM2. (At other times it is ignored, so if the port layer had not been making a request through that phy, it won't care about incoming requests that are rejected by the link layer)
c) Include Retry Open (Collided) along with Retry Open (Opened By Other) for PL_OC handling of Retry Opens. In this case it is allowed to reuse AWT & PBC so no special clearing rules are required.

7.14.4.4 SL_CC2:Selected state

7.14.4.4.1 State description

This state completes the establishment of an SSP, SMP, or STP connection when an incoming connection request has won arbitration by sending a Transmit OPEN_ACCEPT message, or rejects opening a connection by sending a Transmit OPEN_REJECT message to the SL transmitter.

This state shall respond to an incoming OPEN address frame using the following rules:

1) If the OPEN address frame DESTINATION SAS ADDRESS field does not match the SAS address of this port, this state shall send a Transmit OPEN_REJECT (Wrong Destination) message to the SL transmitter;
2) If the OPEN address frame INITIATOR PORT bit, PROTOCOL field, FEATURES field, and/or INITIATOR CONNECTION TAG field are set to values that are not supported (e.g., a connection request from an SMP target port), this state shall send a Transmit OPEN_REJECT (Protocol Not Supported) message the SL transmitter;
3) If the OPEN address frame CONNECTION RATE field is set to a connection rate that is not supported, this state shall send a Transmit OPEN_REJECT (Connection Rate Not Supported) message the SL transmitter;
4) If the OPEN address frame PROTOCOL field is set to STP, the source SAS address is not that of the STP initiator port with an affiliation established or the source SAS address is not that of an STP initiator port with task file register set resources (see), this state shall send a Transmit OPEN_REJECT (STP Resources Busy) message the SL transmitter;
5) If an Accept_Reject Opens (Reject SSP) request, Accept_Reject Opens (Reject SMP) request, or Accept_Reject Opens (Reject STP) request is received and the requested protocol is the corresponding protocol, this state shall send a Transmit OPEN_REJECT (Retry) message to the SL transmitter;
6) If the requested protocol is SSP and this state has not received an Accept_Reject Opens (Reject SSP) request then this state shall send a Transmit OPEN_ACCEPT message to the SL transmitter and send a Connection Opened (SSP, Destination Opened) confirmation to the port layer;
7) If the requested protocol is SMP and this state has not received an Accept_Reject Opens (Reject SMP) request then this state shall send a Transmit OPEN_ACCEPT message to the SL transmitter and send a Connection Opened (SMP, Destination Opened) confirmation to the port layer; or
8) If the requested protocol is STP and this state has not received an Accept_Reject Opens (Reject STP) request then this state shall send a Transmit OPEN_ACCEPT message to the SL transmitter and send a Connection Opened (STP, Destination Opened) confirmation to the port layer.

If this state sends a Transmit OPEN_REJECT message to the SL transmitter, it shall also send a Inbound Connection Rejected confirmation to the port layer.

7.14.4.4.2 Transition SL_CC2:Selected to SL_CC0:Idle

This transition shall occur after this state sends a Transmit OPEN_REJECT message to the SL transmitter.

7.14.4.4.3 Transition SL_CC2:Selected to SL_CC3:Connected

This transition shall occur after sending a Connection Opened confirmation.

This transition shall include an Open SSP Connection, Open STP Connection, or Open SMP Connection argument based on the requested protocol.

7.14.4.4.4 Transition SL_CC2:Selected to SL_CC6:Break

This transition shall occur after a BREAK Received message is received.

8.22.3.2 PL_OC2:Overall_Control state establishing connections

This state receives Phy Enabled confirmations indicating when a phy is available.

This state receives Retry Open messages from a PL_PM state machine.

This state creates pending Tx Open messages based on pending Tx Frame messages and Retry Open messages. Pending Tx Open messages are sent to a PL_PM state machine as Tx Open messages.

If this state receives a Retry Open (Retry) message, then this state shall process the Retry Open message.

If this state receives a Retry Open (No Destination) or a Retry Open (Open Timeout Occurred) message and an I_T Nexus Loss timer has not been created for the destination SAS address (e.g., an SSP target port does not support the I_T NEXUS LOSS TIME field in the Protocol Specific Port mode page or the field is set to 0000h), then this state shall process the Retry Open message as either a Retry Open message or an Unable To Connect message. This selection is vendor-specific.

If this state receives a Retry Open (Pathway Blocked) message and an I_T Nexus Loss timer has not been created for the destination SAS address, then this state shall process the Retry Open message.

If this state receives a Retry Open (No Destination), Retry Open (Open Timeout Occurred), or Retry Open (Pathway Blocked) message, and an I_T Nexus Loss timer has been created for the destination SAS address with an initial value of FFFFh, then this state shall process the Retry Open message (i.e., the Retry Open message is never processed as an Unable to Connect message).

If this state receives a Retry Open (No Destination) or a Retry Open (Open Timeout Occurred) message, an I_T Nexus Loss timer has been created for the destination SAS address, and there is no connection established with the destination SAS address, then this state shall check the I_T Nexus Loss timer, and:

a) if the I_T Nexus Loss timer is not running and the I_T nexus loss time is not set to FFFFh, then this state shall start the timer;
b) if the I_T Nexus Loss timer is running, then this state shall not stop the timer; and
c) if the I_T Nexus Loss timer has expired, then this state shall process the Retry Open message as if it were an Unable To Connect message.

If this state receives a Retry Open (Pathway Blocked) message, an I_T Nexus Loss timer has been created for the destination SAS address, and there is no connection established with the destination SAS address, then this state shall check the I_T Nexus Loss timer, and:

a) if the I_T Nexus Loss timer is running, then this state shall not stop the timer; and
b) if the I_T Nexus Loss timer has expired, then this state shall process the Retry Open message as if it were an Unable To Connect message.

If this state receives a Retry Open (Retry) and an I_T Nexus Loss timer is running for the destination SAS address, then this state shall:

a) stop the I_T Nexus Loss timer (if the timer has been running); and
b) initialize the I_T Nexus Loss timer.

This state shall create a pending Tx Open message if:

a) this state has a pending Tx Frame message or has received a Retry Open message;

b) this state has fewer pending Tx Open messages than the number of PL_PM state machines (i.e., the number of phys in the port);
c) there is no pending Tx Open message for the destination SAS address; and
d) there is no connection established with the destination SAS address.

This state may create a pending Tx Open message if:

a) this state has a pending Tx Frame message, or this state has received a Retry Open message and has not processed the message by sending a confirmation; and
b) this state has fewer pending Tx Open messages than the number of PL_PM state machines.

This state shall have no more pending Tx Open messages than the number of PL_PM state machines.

If this state receives a Retry Open message and there are pending Tx Frame messages for which pending Tx Open messages have not been created, then this state should create a pending Tx Open message from the Retry Open message.

If this state does not create a pending Tx Open message from a Retry Open message (e.g., the current number of pending Tx Open messages equals the number of phys), then this state shall discard the Retry Open message. This state may create a new pending Tx Open message at a later time for the pending Tx Frame message that resulted in the Retry Open message.

If this state receives a Retry Open (Opened By Destination) message and the initiator and protocol arguments match those in the Tx Open messages that resulted in the Retry Open message, then this state may discard the Retry Open message and use the established connection to send pending Tx Frame messages as Tx Frame messages to the destination SAS address. If this state receives a Retry Open (Opened By Destination) message, then, if this state has a pending Tx Open slot available, this state may create a pending Tx Open message from the Retry Open message.

NOTE 5 - If a connection is established by another port as indicated by a Retry Open (Opened By Destination) message, credit may not be granted for frame transmission. In this case this state may create a pending Tx Open message from a Retry Open message in order to establish a connection where credit is granted.

This state shall send a pending Tx Open message as a Tx Open message to a PL_PM state machine that has an enabled phy and does not have a connection established. If there is more than one pending Tx Open message, this state should send a Tx Open message for the pending Tx Open message that has been pending for the longest time first.

If this state creates a pending Tx Open message from one of the following messages:

a) ~~a~~ Retry Open (Opened By Destination);
b) ~~a~~ Retry Open (Opened By Other);
d) Retry Open (Collided); or
a) ~~a~~ Retry Open (Pathway Blocked),

then this state shall:

a) create an Arbitration Wait Time timer for the pending Tx Open message;
b) set the Arbitration Wait Time timer for the pending Tx Open message to the arbitration wait time argument from the Retry Open message; and
c) start the Arbitration Wait Time timer for the pending Tx Open message.

When a pending Tx Open message is sent to a PL_PM state machine as a Tx Open message, the Tx Open message shall contain the following arguments to be used in an OPEN address frame:

a) initiator bit from the Transmit Frame request;
b) protocol from the Transmit Frame request;
c) connection rate from the Transmit Frame request;
d) initiator connection tag from the Transmit Frame request;
e) destination SAS address from the Transmit Frame request;
f) source SAS address from the Transmit Frame request;
g) pathway blocked count; and

h) arbitration wait time.

If this state creates a pending Tx Open message from one of the following:

a) a Transmit Frame request;
b) a Retry Open (No Destination) message;
c) a Retry Open (Open Timeout Occurred) message; or
d) a Retry Open (Retry) message,

then this state shall:

a) set the pathway blocked count argument in the Tx Open message to zero; and
b) set the arbitration wait time argument in the Tx Open message to zero or a vendor-specific value less than 8000h (see 7.12.3).

If a pending Tx Open message was created as the result this state receiving a Retry Open (Pathway Blocked) message, then this state shall set the pathway blocked count argument in the Tx Open message to the value of the pathway blocked count argument received with the message plus one, unless the pathway blocked count received with the argument is FFh.

If a pending Tx Open message was created as the result of this state receiving one of the following:

a) a Retry Open (Opened By Destination) message;
b) a Retry Open (Opened By Other) message;
e) a Retry Open (Collided) message; or
a) a Retry Open (Pathway Blocked) message;

then this state shall set the arbitration wait time argument in the Tx Open message to be the value from the Arbitration Wait Time timer created as a result of the Retry Open message.

After this state sends a Tx Open message, this state shall discard the pending Tx Open message from which the Tx Open messages was created. After this state discards a pending Tx Open message, this state may create a new pending Tx Open message.

If this state receives a Connection Opened message and the initiator and protocol arguments match those in any pending Tx Frame messages, then this state may use the established connection to send pending Tx Frame messages as Tx Frame messages to the destination SAS address.

8.2.3.3.3 PL_PM2:Req_Wait connection established

If this state receives a Connection Opened confirmation, then this state shall send a Connection Opened message to the PL_OC state machine.

If this state receives a Connection Opened confirmation and the confirmation was not in response to an Open Connection request from this state (i.e., the connection was established in response to an OPEN address frame from another device), then this state shall discard any Open Connection request and send a Retry Open message to the PL_OC state machine. If the Connection Opened confirmation was from the destination of the Open Connection request, then this state shall send a Retry Open (Opened By Destination) message. If the Connection Opened confirmation was from a destination other than the destination of the Open Connection request, then this state shall send a Retry Open (Opened By Other) message.

If this state receives a Incoming Connection Rejected confirmation after sending an Open Connection request, then this state shall discard the Open Connection request and send a Retry Open (Collided) message to the PL_OC state machine.

A Retry Open (Opened By Destination) ~~or~~, Retry Open (Opened By Other), or Retry Open (Collided) message shall contain the following arguments:

a) initiator bit set to the value received with the Tx Open message;
b) protocol set to the value received with the Tx Open message;
c) connection rate set to the value received with the Tx Open message;
d) initiator connection tag set to the value received with the Tx Open message;
e) destination SAS address set to the value received with the Tx Open message;

f) source SAS address set to the value received with the Tx Open message;
g) pathway blocked count argument set to the value received with the Tx Open message; and
h) arbitration wait time set to the value of the Arbitration Wait Time timer.