

To: T10 Technical Committee
From: Rob Elliott, HP (elliott@hp.com)
Date: 17 May 2004
Subject: 04-115r2 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.

Revision 2 (17 May 2004) Incorporating comments from the May 2004 CAP WG. Moved invalid dword handling topics #12/13/14 to 04-167. Moved RRDY/CREDIT BLOCKED handling topic #17 to 04-168.

Related documents

sas1r04 - Serial Attached SCSI 1.1 revision 4

04-131 SAS-1.1 Miscellaneous physical layer changes

04-167 SAS-1.1 Invalid dword handling

04-168 SAS-1.1 RRDY and CREDIT BLOCKED clarifications

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 (dropped)
- 11 Clause 7.2 - expanders need not enforce SATA_CONT rules
- 12 Clause 7.2 - invalid dword changes - moved to 04-167
- 13 Clause 7.2 - invalid dword changes - moved to 04-167
- 14 Clause 7.2 - invalid dword changes - moved to 04-167
- 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 - moved to 04-168
- 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.

Editor's Note 1: May status: OK, with addition of note that each STP/SATA bridge has a unique address

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

...

The updated figure follows:

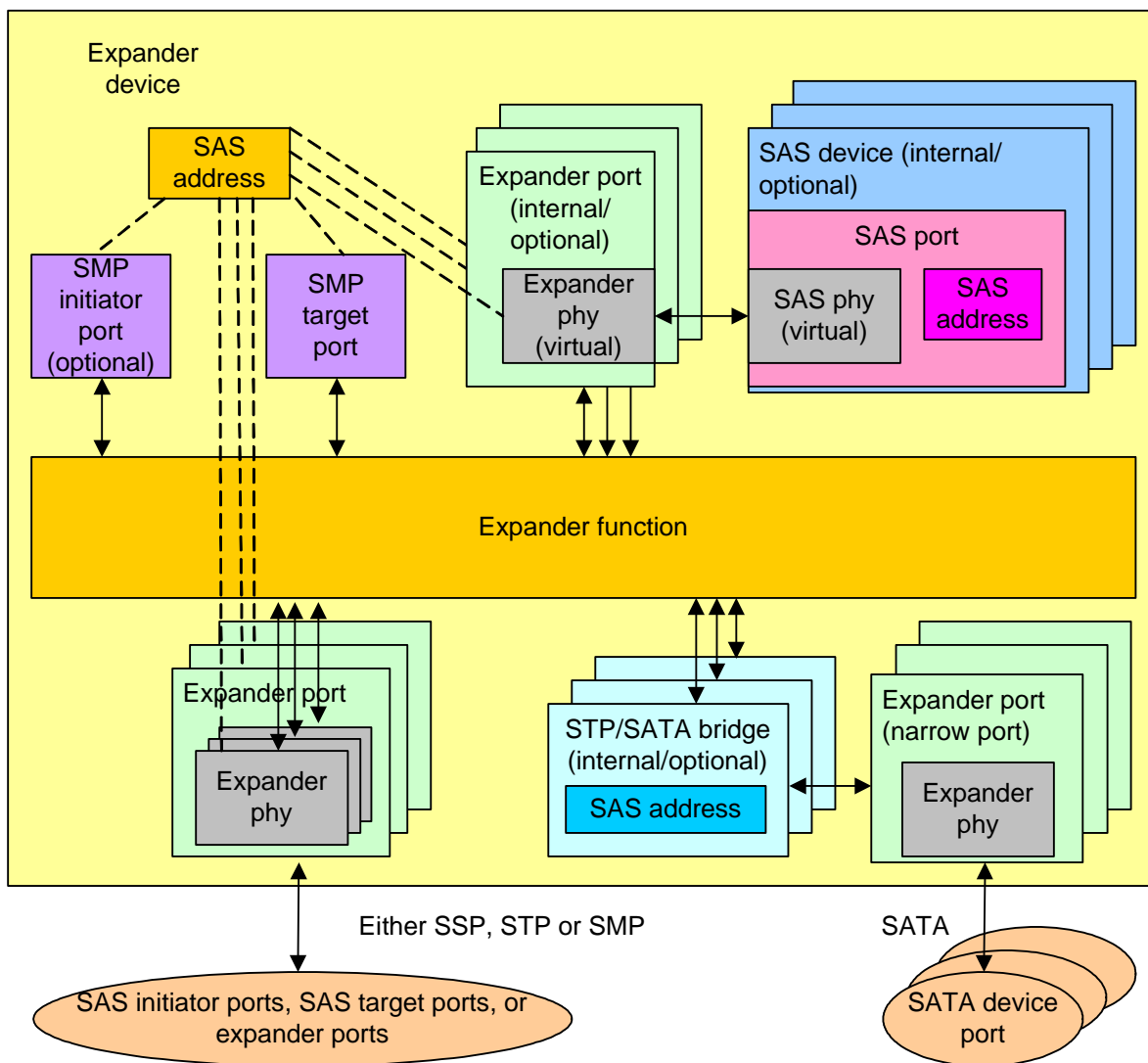


Figure 1 — Expander device [updated figure 12]

4.6.2 Expander ports

An external expander port contains one or more physical phys (see 4.1.2). Each expander phy contains an expander link layer with an XL state machine and an SL_IR state machine. The expander link layers within an expander port request and respond to connection requests independently.

An internal expander port contains a virtual phy with an expander link layer and a protocol-specific transport layer (e.g., to provide access to a SCSI enclosure services device as an SSP target).

Each expander device shall include one internal SMP port using the expander device's SAS address.

Any additional internal SAS ports shall be inside SAS devices contained in the expander device, and thus have SAS addresses different from that of the expander device. These SAS ports shall be attached to internal expander ports with virtual phys.

[Each STP/SATA bridge shall have a unique SAS address.](#)

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.

Editor's Note 2: May status: OK

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.

[Editor's Note 3: May status: OK, adding mode page comment and changing i.e. to e.g. in 10.4.3.2](#)

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). In a SAS device containing an SSP target port, phy identifiers shall be less than the value of the NUMBER OF PHYS field in the Protocol-Specific Port mode page for SAS SSP - Phy Control And Discover subpage.

10.4.3.2 SMP function response frame format

...

The FUNCTION RESULT field is defined in table 1.

Table 1 — Function results

Code	Name	SMP function(s)	Description
10h	PHY DOES NOT EXIST	DISCOVER, REPORT PHY ERROR LOG, REPORT PHY SATA, REPORT ROUTE INFORMATION, CONFIGURE ROUTE INFORMATION, PHY CONTROL	The phy specified by the PHY IDENTIFIER field in the SMP request frame does not exist (i.e. , <u>e.g.</u> , the value is not within the range of zero to the value of the NUMBER OF PHYS field reported in the REPORT GENERAL function). The ADDITIONAL RESPONSE BYTES field may be present but shall be ignored.
All others	Reserved		

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.

Editor's Note 4: May status: OK

4.6.6.3 ECM interface

...

Table 2 describes the responses from an expander phy to the ECM.

Table 2 — Expander phy to ECM responses

Message	Description
Phy Status (Partial Pathway)	Response meaning that an expander phy: <ol style="list-style-type: none"> 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) <u>the last AIP received or transmitted is not AIP (WAITING ON PARTIAL)</u>); or 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: <ol style="list-style-type: none"> 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) <u>the last AIP received or transmitted is AIP (WAITING ON PARTIAL)</u>); or 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: <ol style="list-style-type: none"> is being used for a connection (i.e., the expander phy is in the XL7:Connected or XL8:Close_Wait state); or 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.

[Editor's Note 5: May status: WG requested several changes](#)

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 3 lists the primitives used only inside STP connections and on SATA physical links. [These are called STP primitives.](#)

Table 3 — 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 4 summarizes the types of primitive sequences.

Table 4 — Primitive sequences

Primitive sequence type	Number of times to transmit <u>Number of times the transmitter transmits the primitive to transmit the primitive sequence</u>	Number of times received to detect <u>Number of times the receiver receives the primitive to detect the primitive sequence</u>
Single	1	1
Repeated	2 <u>1 or more</u>	1
<u>Continued</u>	<u>2 followed by SATA_CONT</u>	<u>1</u>
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

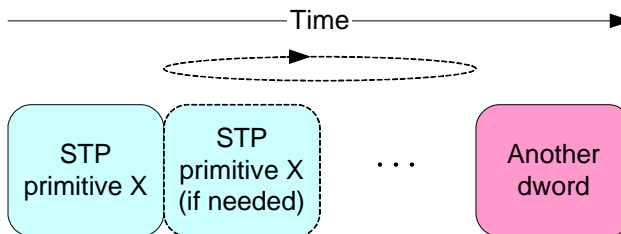
Primitives labeled as single primitive sequences (e.g., RRDY, SATA_SOF) ~~are sent one time~~ shall be transmitted one time to form a single primitive sequence.

Receivers count each single primitive sequence received as a distinct primitive sequence.

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.



NOTE: Another dword is a dword other than ALIGN, NOTIFY, or STP primitive X

Figure 2 — Transmitting a repeated primitive sequence

Receivers do not count the number of times a repeated primitive is received (i.e., receivers are simply in the state of receiving the primitive).

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

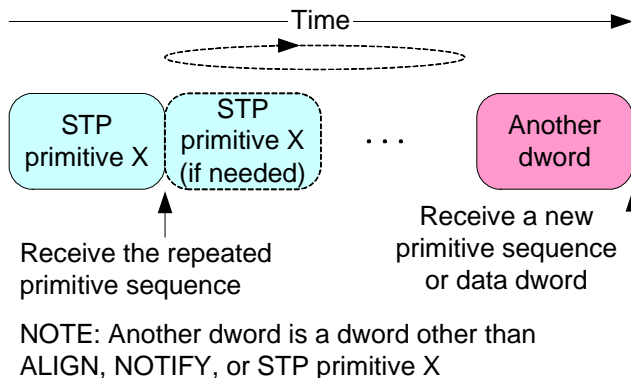


Figure 3 — Receiving a repeated primitive sequence

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.

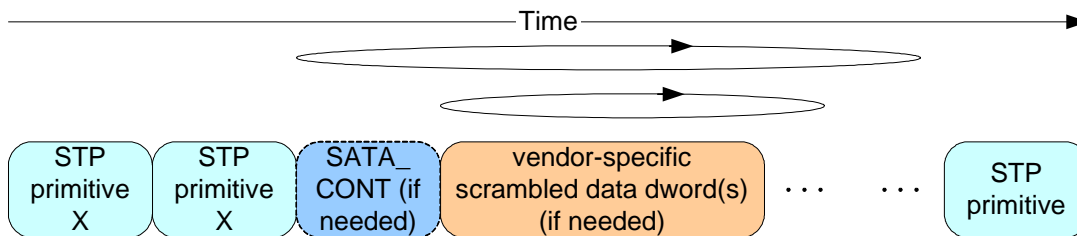


Figure 4 — Transmitting a continued primitive sequence [figure updated]

Receivers shall detect a continued primitive sequence after at least one primitive is received. The primitive may be followed by one or more of the same primitive. 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 (i.e., receivers are simply in the state of receiving the primitive).

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

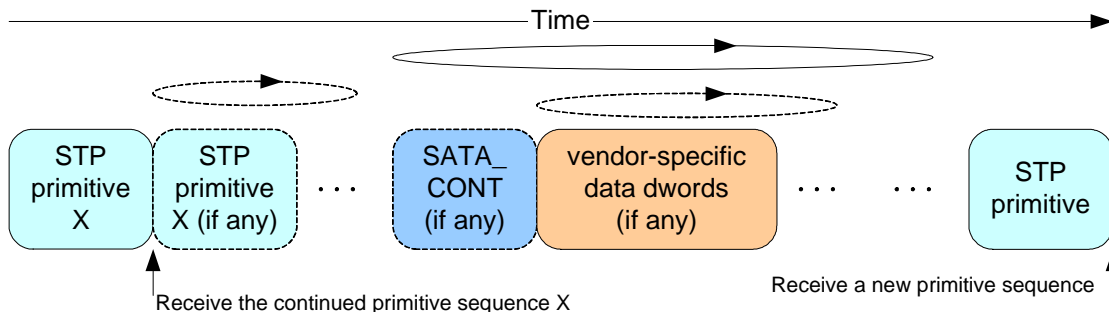


Figure 5 — Receiving a continued primitive sequence

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 5 lists the scrambling for different types of data dwords.

Table 5 — 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~~ primitives sequences may occur inside STP frames and they STP frame and the continued primitive sequence have independent scrambling patterns.

10 Clause 7.2 - allow sending two SATA_CONTs (dropped)

Dropped.

Editor's Note 6: May status: since previous transmit rules allow this, remain silent

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.

Editor's Note 7: May status: Change "are not required to" to "may or may not" and work on the note in 7.4

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 (i.e., receivers are simply in the state of receiving the primitive).

Figure 7 shows an example of receiving a continued primitive sequence.

...

[Expanders forwarding dwords may or may not 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.~~

[SATA_SYNC is a continued primitive sequence which may contain vendor-specific data dwords \(see 7.2.4.3\).](#)

Idle dwords are scrambled (see 7.6).

12 Clause 7.2 - invalid dword changes - moved to 04-167

Moved to 04-167.

13 Clause 7.2 - invalid dword changes - moved to 04-167

Moved to 04-167.

14 Clause 7.2 - invalid dword changes - moved to 04-167

Moved to 04-167.

15 Clause 7.12 - Expander arbitration rules

The word “available” is 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.

Editor's Note 8: May status: OK

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 9: note the “partial pathway” to “blocked partial pathway” change in c)

Editor's Note 10: 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).

Editor’s Note 11: May status: OK, be sure to delete the b) heading too

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 phys within the requested destination port contains ~~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 12: 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 - moved to 04-168

Moved to 04-168.

[Editor's Note 13: May status: WG wants to allow CREDIT_BLOCKED after RRDY](#)

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.

Editor's Note 14: May status: OK

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.

[Editor's Note 15: May status: OK, delete AIP from figure](#)

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

...

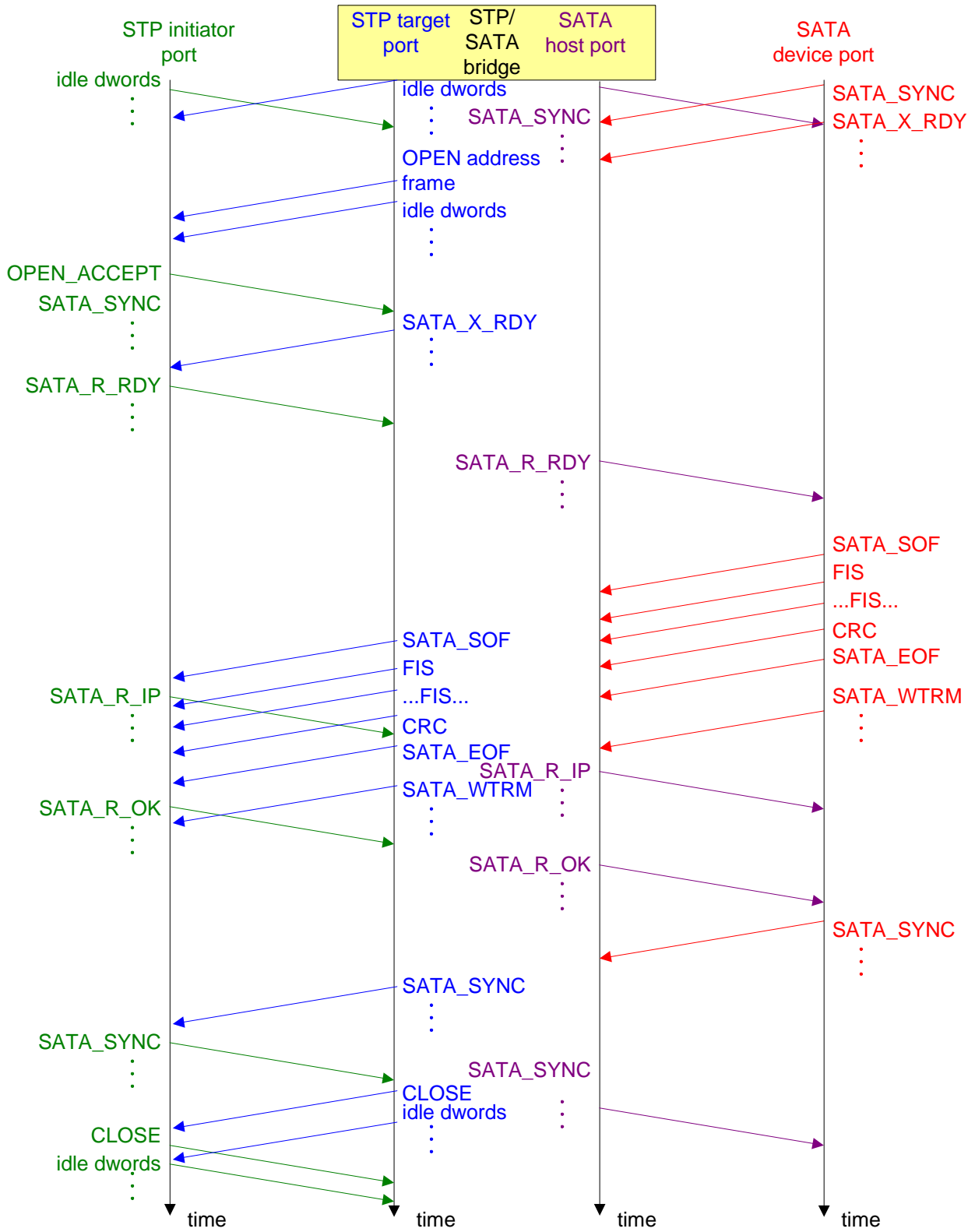


Figure 103 — STP target port opening an STP connection [\[modified\]](#)

[\[changed idle dwords to SATA_SYNC in what is now the top left\]](#)

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.

Editor's Note 16: May status: OK

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.

[Editor's Note 17: May status: Add some "\(if any\)"s to the table](#)

9.2.2.5.1 RESPONSE information unit overview

...

[Change the RESPONSE frame definition to:](#)

Table 6 — RESPONSE information unit

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

Table 7 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 ATA field.

Table 7 — DATAPRES field

Code	Name	Description	Reference
00b	NO DATA	No response data or sense 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.

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 initial Register FIS - don't accept it.

Editor's Note 18: May status: WG questioned how SATA port multipliers will handle the initiator register FISes. Per PM 1.1, the port multiplier returns its own signature as the first register FIS, like a drive. The host has to send a WRITE PORT MULTIPLIER (WRITE BUFFER) command to the control port (Fh) to enable each device port of the PM (setting the DET field in the device port's PSCR[2] register) before the PM starts communicating with the device. If the host does not do this quickly enough, however, and device 0 manages to send a COMINIT first, then the PM goes into a legacy support mode. It passes through the COMINIT and all subsequent traffic from device 0, leaving all the other device ports disabled. The expander will latch the device 0 signature since the COMINIT went through. Each STP initiator can still attempt to probe for a port multiplier by sending an SRST to the control port (Fh). Since the signature won't be latched in the expander each one has to do that individually. The SRST has no effect other than to return the signature, so there should be no problem with initiators clearing each others' port-enabled settings just by performing discovery.

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.

Editor's Note 19: May status: provide a replacement for the "ATTACHED SAS ADDRESS field is invalid" sentence being deleted.

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.

[An STP initiator port should not make a connection request to the attached SAS address until the ATTACHED DEVICE TYPE field is not 000b.](#)

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.

[Editor's Note 20: May status: OK](#)

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.

Editor's Note 21: May status: OK

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.

[Editor's Note 22: May status: Give the group more time to review](#)

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 to 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 to 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 to 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.