

---

**eb-2016-00611-002**

Document Date: 1/4/17

To: INCITS Members

Reply To: [Rachel Porter](#)

Subject: Public Review and Comments Register for the Approval of:

**INCITS 538-201x, Information technology - SAS Protocol Layer - 4**

**Due Date: The public review is from October 28, 2016 – December 27, 2016**

Action: The InterNational Committee for Information Technology Standards ([INCITS](#)) announces that the subject-referenced document(s) is being circulated for a 60-day public review and comment period. Comments received during this period will be considered and answered. Commenters who have objections/suggestions to this document should so indicate and include their reasons.

All comments should be forwarded not later than the date noted above to the following address:

INCITS Secretariat/ITI

1101 K Street NW - Suite 610

Washington DC 20005-3922

Email: [comments@standards.incits.org](mailto:comments@standards.incits.org) (preferred)

*This public review also serves as a call for patents and any other pertinent issues (copyrights, trademarks). Correspondence regarding intellectual property rights may be emailed to the INCITS Secretariat at [patents@itic.org](mailto:patents@itic.org).*

---

[Comment #2](#) – Received from Brad Besmer, Broadcom Limited

# **Working Draft American National Standard**

# **Project T10/BSR INCITS 538**

Revision 10  
15 September 2016

---

## **Information technology - SAS Protocol Layer - 4 (SPL-4)**

This is an internal working document of T10, a Technical Committee of Accredited Standards Committee INCITS (International Committee for Information Technology Standards). As such this is not a completed standard and has not been approved. The contents may be modified by the T10 Technical Committee. The contents are actively being modified by T10. This document is made available for review and comment only.

Permission is granted to members of INCITS, its technical committees, and their associated task groups to reproduce this document for the purposes of INCITS standardization activities without further permission, provided this notice is included. All other rights are reserved. Any duplication of this document for commercial or for-profit use is strictly prohibited.

T10 Technical Editors:

Tim Symons  
Microsemi Corporation  
8555 BaxterPlace  
Burnaby, BC V5A 4V7  
Canada

and

George Penokie  
Broadcom Limited  
4109 Manor View Dr NW  
Rochester, MN 55901-3109  
USA

Telephone:(604) 415 6000  
Email: Tim.Symons@Microsemi.com

Telephone: (507) 921-2495

---

**Reference number  
ISO/IEC 14776-264:201x  
BSR INCITS 538-201x**

## Points of Contact

### International Committee for Information Technology Standards (INCITS) T10 Technical Committee

T10 Chair  
Ralph O. Weber  
Western Digital Corp.  
18484 Preston Road, Suite 102, PMB 178  
Dallas, TX 75252  
USA

Telephone: (214) 912-1373  
Email: [Ralph.Weber@WDC.com](mailto:Ralph.Weber@WDC.com)

T10 Vice-Chair  
William Martin  
Samsung Semiconductor, Inc  
7213 Marblethorpe Drive  
Roseville, CA 95747-5925  
USA

Telephone: (916) 765-6875  
Email: [bill.martin@ssi.samsung.com](mailto:bill.martin@ssi.samsung.com)

T10 Web Site: <http://www.t10.org>

INCITS Secretariat  
Suite 610  
1101 K Street, NW  
Washington, DC 20005-7031  
USA

Telephone: (202) 737-8888  
Web site: <http://www.incits.org>  
Email: [incits@itic.org](mailto:incits@itic.org)

Information Technology Industry Council  
Web site: <http://www.itic.org>

Purchase INCITS Standards  
Web site: <http://www.incits.org/standards-information/purchase-standards-or-download-dpans>

American National Standard  
for Information Technology

# SAS Protocol Layer - 4 (SPL-4)

Secretariat  
**Information Technology Industry Council**

Approved mm.dd.yy

American National Standards Institute, Inc.

## ABSTRACT

This standard specifies three transport protocols used over the SAS interconnect specified in SAS-4, one to transport SCSI commands, another to transport Serial ATA commands to SATA devices, and a third to support interface management. This standard is intended to be used in conjunction with SAS standards, SCSI command set standards, and ATA command set standards.



## American National Standard

Approval of an American National Standard requires verification by ANSI that the requirements for due process, consensus, and other criteria for approval have been met by the standards developer. Consensus is established when, in the judgment of the ANSI Board of Standards Review, substantial agreement has been reached by directly and materially affected interests. Substantial agreement means much more than a simple majority, but not necessarily unanimity. Consensus requires that all views and objections be considered, and that effort be made towards their resolution.

The use of American National Standards is completely voluntary; their existence does not in any respect preclude anyone, whether he has approved the standards or not, from manufacturing, marketing, purchasing, or using products, processes, or procedures not conforming to the standards.

The American National Standards Institute does not develop standards and will in no circumstances give interpretation on any American National Standard. Moreover, no person shall have the right or authority to issue an interpretation of an American National Standard in the name of the American National Standards Institute. Requests for interpretations should be addressed to the secretariat or sponsor whose name appears on the title page of this standard.

**CAUTION NOTICE:** This American National Standard may be revised or withdrawn at any time. The procedures of the American National Standards Institute require that action be taken periodically to reaffirm, revise, or withdraw this standard. Purchasers of American National Standards may receive current information on all standards by calling or writing the American National Standards Institute.

**CAUTION:** The developers of this standard have requested that holders of patents that may be required for the implementation of the standard, disclose such patents to the publisher. However, neither the developers nor the publisher have undertaken a patent search in order to identify which patents, if any, may apply to this standard.

As of the date of publication of this standard, following calls for the identification of patents that may be required for the implementation of the standard, notice of one or more claims has been received.

By publication of this standard, no position is taken with respect to the validity of this claim or of any rights in connection therewith. The known patent holder(s) has (have), however, filed a statement of willingness to grant a license under these rights on reasonable and nondiscriminatory terms and conditions to applicants desiring to obtain such a license. Details may be obtained from the publisher.

No further patent search is conducted by the developer or the publisher in respect to any standard it processes. No representation is made or implied that licenses are not required to avoid infringement in the use of this standard.

Published by

**American National Standards Institute**

**25 West 43rd Street 4th floor, New York, New York 10036-7422**

Copyright © 2017 by Information Technology Industry Council (ITI). All rights reserved.

No part of this publication may be reproduced in any form, in an electronic retrieval system or otherwise, without prior written permission of ITI, 1101 K Street NW, Suite 610, Washington, DC 20005-7031.

Printed in the United States of America

## Contents

	Page
Foreword .....	xlviii
Introduction .....	lii
1 Scope .....	1
2 Normative references .....	3
3 Terms, definitions, symbols, abbreviations, keywords, and conventions .....	4
3.1 Terms and definitions .....	4
3.2 Symbols and abbreviations .....	29
3.2.1 Abbreviations .....	29
3.2.2 Units .....	32
3.2.3 Symbols .....	32
3.2.4 Mathematical operators .....	32
3.3 Keywords .....	33
3.4 Editorial conventions .....	34
3.5 Numeric and character conventions .....	34
3.5.1 Numeric conventions .....	34
3.5.2 Units of measure .....	35
3.5.3 Byte encoded character strings conventions .....	36
3.6 UML notation conventions .....	36
3.6.1 Notation conventions overview .....	36
3.6.2 Constraint and note conventions .....	36
3.6.3 Class diagram conventions .....	38
3.6.4 Object diagram conventions .....	42
3.7 State machine conventions .....	44
3.7.1 State machine conventions overview .....	44
3.7.2 Transitions .....	45
3.7.3 Messages, requests, indications, confirmations, responses, and event notifications .....	45
3.7.4 State machine counters, timers, and variables .....	45
3.7.5 State machine arguments .....	46
3.8 Bit and byte ordering .....	46
3.9 Notation for procedures and functions .....	47
4 General .....	48
4.1 Architecture .....	48
4.1.1 Architecture overview .....	48
4.1.2 Physical links and phys .....	50
4.1.3 Logical links .....	54
4.1.4 Narrow ports and wide ports .....	54
4.1.5 Application clients and device servers .....	58
4.1.6 SAS devices .....	58
4.1.7 Expander devices .....	59
4.1.8 Service delivery subsystem .....	61
4.1.9 Domains .....	61
4.1.10 Expander device topologies .....	64
4.1.10.1 Expander device topology overview .....	64
4.1.10.2 Expander device topologies .....	65
4.1.11 Pathways .....	67
4.1.12 Connections .....	67
4.1.13 Persistent connections .....	69
4.1.13.1 Persistent connection operation .....	69
4.1.13.2 Persistent connection support .....	70
4.1.14 Advancing credit .....	70
4.1.15 Broadcasts .....	70

4.2 Names and identifiers .....	73
4.2.1 Names and identifiers overview .....	73
4.2.2 NAA IEEE Registered format identifier .....	74
4.2.3 NAA Locally Assigned format identifier .....	75
4.2.4 SAS address .....	75
4.2.5 Hashed SAS addresses .....	75
4.2.6 Device names and expander device SAS addresses .....	76
4.2.7 Device names for SATA devices with world wide names .....	77
4.2.8 Port names .....	77
4.2.9 Port identifiers and SAS port SAS addresses .....	77
4.2.10 Phy identifiers .....	78
4.3 State machines .....	79
4.3.1 State machine overview .....	79
4.3.2 Transmit data path .....	80
4.3.3 Receive data path .....	85
4.3.3.1 Receive data path while in the SAS dword mode .....	85
4.3.3.2 Receive data path while in the SAS packet mode .....	88
4.3.4 State machines and SAS Device, SAS Port, and SAS Phy classes .....	92
4.4 Events .....	94
4.4.1 Reset sequences .....	94
4.4.2 Hard reset .....	96
4.4.2.1 Hard reset overview .....	96
4.4.2.2 Additional hard reset processing by SAS ports .....	96
4.4.2.3 Additional hard reset processing by expander ports .....	96
4.4.3 I_T nexus loss .....	96
4.4.4 Power loss expected .....	97
4.5 Expander device model .....	98
4.5.1 Expander device model overview .....	98
4.5.2 Expander ports .....	99
4.5.3 Expander connection manager (ECM) .....	100
4.5.4 Expander connection router (ECR) .....	100
4.5.5 Broadcast propagation processor (BPP) .....	101
4.5.6 Expander device interfaces .....	101
4.5.6.1 Expander device interface overview .....	101
4.5.6.2 Expander device interfaces detail .....	103
4.5.6.3 ECM interface .....	104
4.5.6.4 ECR interface .....	106
4.5.6.5 BPP interface .....	108
4.5.7 Expander device routing .....	109
4.5.7.1 Routing attributes and routing methods .....	109
4.5.7.2 Expander device topology routing attribute restrictions .....	110
4.5.7.3 Connection request routing .....	110
4.5.7.4 Expander route table .....	110
4.5.7.4.1 Expander route table overview .....	110
4.5.7.4.2 Phy-based expander route table .....	111
4.5.7.4.3 Expander-based expander route table .....	112
4.5.8 Expander device reduced functionality .....	112
4.5.9 Broadcast (Expander) handling .....	113
4.6 Discover process .....	113
4.6.1 Discover process overview .....	113
4.6.2 Starting the discover process (Broadcast (Change) handling) .....	113
4.6.3 Discover process traversal .....	114
4.6.4 Discover process in a self-configuring expander device .....	116
4.6.5 Enabling multiplexing .....	117
4.7 Configuration subprocess .....	117
4.7.1 Configuration subprocess overview .....	117
4.7.2 Allowed expander device topologies .....	117

4.7.3 Externally configurable expander device route table optimization .....	119
4.7.4 Externally configurable expander device expander route index order .....	120
4.8 Zoning .....	128
4.8.1 Zoning overview .....	128
4.8.2 Zoning expander device requirements .....	132
4.8.3 Zoning operation .....	135
4.8.3.1 Zone phy information .....	135
4.8.3.2 Zone groups .....	138
4.8.3.3 Zone permission table .....	138
4.8.3.4 Zoning expander route table .....	140
4.8.3.5 Source zone group and destination zone group determination .....	141
4.8.4 Zone phy information and link reset sequences .....	142
4.8.5 Broadcast processing in a zoning expander device with zoning enabled .....	145
4.8.6 Zone configuration .....	146
4.8.6.1 Zone configuration overview .....	146
4.8.6.2 Lock step .....	146
4.8.6.3 Load step .....	147
4.8.6.4 Activate step .....	148
4.8.6.5 Unlock step .....	148
4.8.6.6 Zone lock inactivity timer .....	149
4.8.6.7 Enable a zoning expander device .....	149
4.9 SAS device and expander device power conditions .....	149
4.10 Phy power conditions .....	150
4.10.1 Low phy power conditions .....	150
4.10.1.1 Low phy power conditions overview .....	150
4.10.1.2 Active phy power condition .....	150
4.10.1.3 Partial phy power condition .....	150
4.10.1.4 Slumber phy power condition .....	150
4.10.1.5 End device low phy power conditions .....	151
4.10.1.6 Expander device low phy power conditions .....	151
4.10.2 SATA phy power conditions .....	152
4.11 Phy test functions .....	152
4.11.1 Phy test functions overview .....	152
4.11.2 Transmit pattern phy test function .....	153
4.12 Phy events .....	153
4.13 Using POWER DISABLE signal to create a power on event .....	158
4.13.1 Using POWER DISABLE signal to create a power on event overview .....	158
4.13.2 Discovering POWER DISABLE signal support .....	158
4.13.3 Using a management device server to control the POWER DISABLE signal .....	159
4.14 Management application client model for APTA .....	159
5 Phy layer .....	160
5.1 Phy layer overview .....	160
5.2 8b10b coding .....	160
5.2.1 8b10b coding overview .....	160
5.2.2 8b10b coding notation conventions .....	160
5.3 Character encoding and decoding .....	162
5.3.1 Introduction .....	162
5.3.2 Bit transmission order .....	162
5.3.3 Character transmission order .....	162
5.3.4 Frame transmission order .....	162
5.3.5 Running disparity (RD) .....	162
5.3.6 Data characters .....	163
5.3.7 Control characters .....	169
5.3.8 Encoding characters in the transmitter .....	170
5.3.9 Decoding characters in the receiver .....	170
5.4 SAS dword mode bit order .....	171

5.5 SPL packet .....	173
5.5.1 SPL packet overview .....	173
5.5.2 SPL packet format .....	174
5.5.3 SPL packet payload that contains a scrambled idle segment.....	177
5.5.4 SPL packet payload that contains a primitive segment .....	177
5.5.5 SPL packet payload that contains an SPL frame segment.....	184
5.5.6 SPL packet payload that contains an idle dword segment .....	184
5.5.7 Forward error correction .....	185
5.5.7.1 Forward error correction overview.....	185
5.5.7.2 Forward error correction encoding .....	187
5.5.7.3 Forward error correction decoding .....	190
5.6 Dwords, primitives, binary primitives, extended binary primitives, data dwords, SPL frame segments, and invalid dwords .....	192
5.7 Out of band (OOB) signals .....	192
5.7.1 OOB signals overview.....	192
5.7.2 Transmission of OOB signals .....	193
5.7.3 Receiver detection of OOB signals .....	194
5.7.4 SATA port selection signal.....	196
5.7.5 Phy power conditions.....	196
5.8 Phy capabilities bits .....	196
5.9 BMC coding .....	201
5.9.1 BMC coding overview .....	201
5.9.2 TTIU bit cell encoding in the transmitter .....	202
5.9.3 TTIU bit transmission order.....	203
5.9.4 TTIU bit cell decoding in the receiver.....	203
5.10 Train_Tx-SNW TTIUs .....	204
5.10.1 Train_Tx-SNW TTIU format .....	204
5.10.2 Control/Status TTIU .....	205
5.10.3 Error Response TTIU.....	209
5.11 Phy reset sequences .....	212
5.11.1 Phy reset sequences overview .....	212
5.11.2 SATA phy reset sequence .....	213
5.11.2.1 SATA OOB sequence .....	213
5.11.2.2 SATA speed negotiation sequence .....	213
5.11.3 SAS to SATA phy reset sequence .....	214
5.11.4 SAS to SAS phy reset sequence .....	215
5.11.4.1 SAS OOB sequence.....	215
5.11.4.2 SAS speed negotiation sequence .....	218
5.11.4.2.1 SAS speed negotiation sequence overview .....	218
5.11.4.2.2 SAS speed negotiation sequence timing specifications .....	219
5.11.4.2.3 Speed negotiation window (SNW) definitions.....	220
5.11.4.2.3.1 SNW definitions overview .....	220
5.11.4.2.3.2 SNW-1, SNW-2, and Final-SNW.....	221
5.11.4.2.3.3 SNW-3.....	222
5.11.4.2.3.4 Train_Tx-SNW .....	224
5.11.4.2.3.4.1 Phy's transmitter initial condition .....	224
5.11.4.2.3.4.2 Transmitter training.....	224
5.11.4.2.3.4.3 Pattern marker.....	226
5.11.4.2.3.4.4 Pattern marker detection while in SAS dword mode .....	227
5.11.4.2.3.4.5 Pattern marker detection while in SAS packet mode .....	228
5.11.4.2.3.5 Train_Rx-SNW while in SAS dword mode .....	229
5.11.4.2.3.6 Train_Rx-SNW while in SAS packet mode .....	231
5.11.4.2.4 SAS speed negotiation sequence.....	234
5.11.4.2.5 SAS speed negotiation sequence examples .....	235
5.11.4.2.6 Train_Tx pattern sequence.....	243
5.11.4.2.6.1 Train_Tx pattern sequence overview .....	243
5.11.4.2.6.2 Train_Tx pattern initial sequence .....	244

5.11.4.2.6.3 Train_Tx pattern handshake sequence.....	247
5.11.4.2.6.3.1 Train_Tx pattern handshake sequence overview .....	247
5.11.4.2.6.3.2 Attached phy's receiver increment or decrement request .....	247
5.11.4.2.6.3.3 Attached phy's receiver reference_1, reference_2, or no_equalization request ....	250
5.11.4.2.6.4 Train_Tx pattern completion sequence .....	252
5.11.4.2.6.5 Invalid TTIU sequence .....	255
5.11.4.3 Multiplexing sequence .....	256
5.11.5 Phy reset sequence after devices are attached .....	257
5.12 APTA .....	258
5.13 Phy power condition sequences .....	259
5.13.1 Transitioning from the active phy power condition to a low phy power condition .....	259
5.13.2 Transitioning from a low phy power condition to the active phy power condition .....	260
5.13.3 Events during low phy power condition .....	260
5.14 SP (phy layer) state machine .....	262
5.14.1 SP state machine overview .....	262
5.14.2 SP transmitter and SP receiver .....	265
5.14.3 OOB sequence states .....	269
5.14.3.1 OOB sequence states overview .....	269
5.14.3.2 SP0:OOB_COMINIT state .....	270
5.14.3.2.1 State description .....	270
5.14.3.2.2 Transition SP0:OOB_COMINIT to SP1:OOB_AwaitCOMX .....	271
5.14.3.2.3 Transition SP0:OOB_COMINIT to SP3:OOB_AwaitCOMINIT_Sent .....	271
5.14.3.2.4 Transition SP0:OOB_COMINIT to SP4:OOB_COMSAS .....	271
5.14.3.3 SP1:OOB_AwaitCOMX state .....	271
5.14.3.3.1 State description .....	271
5.14.3.3.2 Transition SP1:OOB_AwaitCOMX to SP0:OOB_COMINIT .....	271
5.14.3.3.3 Transition SP1:OOB_AwaitCOMX to SP4:OOB_COMSAS .....	272
5.14.3.4 SP2:OOB_NoCOMSASTimeout state .....	272
5.14.3.4.1 State description .....	272
5.14.3.4.2 Transition SP2:OOB_NoCOMSASTimeout to SP0:OOB_COMINIT .....	272
5.14.3.4.3 Transition SP2:OOB_NoCOMSASTimeout to SP4:OOB_COMSAS .....	272
5.14.3.5 SP3:OOB_AwaitCOMINIT_Sent state .....	272
5.14.3.5.1 State description .....	272
5.14.3.5.2 Transition SP3:OOB_AwaitCOMINIT_Sent to SP4:OOB_COMSAS .....	272
5.14.3.6 SP4:OOB_COMSAS state .....	272
5.14.3.6.1 State description .....	272
5.14.3.6.2 Transition SP4:OOB_COMSAS to SP5:OOB_AwaitCOMSAS_Sent .....	273
5.14.3.6.3 Transition SP4:OOB_COMSAS to SP6:OOB_AwaitNoCOMSAS .....	273
5.14.3.6.4 Transition SP4:OOB_COMSAS to SP7:OOB_AwaitCOMSAS .....	273
5.14.3.7 SP5:OOB_AwaitCOMSAS_Sent state .....	273
5.14.3.7.1 State description .....	273
5.14.3.7.2 Transition SP5:OOB_AwaitCOMSAS_Sent to SP6:OOB_AwaitNoCOMSAS .....	273
5.14.3.8 SP6:OOB_AwaitNoCOMSAS state .....	274
5.14.3.8.1 State description .....	274
5.14.3.8.2 Transition SP6:OOB_AwaitNoCOMSAS to SP0:OOB_COMINIT .....	274
5.14.3.8.3 Transition SP6:OOB_AwaitNoCOMSAS to SP8:SAS_Start .....	274
5.14.3.9 SP7:OOB_AwaitCOMSAS state .....	274
5.14.3.9.1 State description .....	274
5.14.3.9.2 Transition SP7:OOB_AwaitCOMSAS to SP2:OOB_NoCOMSASTimeout .....	274
5.14.3.9.3 Transition SP7:OOB_AwaitCOMSAS to SP6:OOB_AwaitNoCOMSAS .....	274
5.14.3.9.4 Transition SP7:OOB_AwaitCOMSAS to SP16:SATA_COMWAKE .....	274
5.14.3.9.5 Transition SP7:OOB_AwaitCOMSAS to SP26:SATA_SpinupHold .....	275
5.14.4 SAS speed negotiation states .....	275
5.14.4.1 SAS speed negotiation states overview .....	275
5.14.4.2 Negotiation idle .....	275
5.14.4.3 SP8:SAS_Start state .....	278
5.14.4.3.1 State description .....	278

5.14.4.3.2 Transition SP8:SAS_Start to SP0:OOB_COMINIT .....	278
5.14.4.3.3 Transition SP8:SAS_Start to SP1:OOB_AwaitCOMX .....	279
5.14.4.3.4 Transition SP8:SAS_Start to SP9:SAS_WindowNotSupported .....	279
5.14.4.3.5 Transition SP8:SAS_Start to SP10:SAS_AwaitALIGN .....	279
5.14.4.3.6 Transition SP8:SAS_Start to SP27:SAS_Settings .....	279
5.14.4.4 SP9:SAS_WindowNotSupported state .....	279
5.14.4.4.1 State description .....	279
5.14.4.4.2 Transition SP9:SAS_WindowNotSupported to SP0:OOB_COMINIT .....	279
5.14.4.4.3 Transition SP9:SAS_WindowNotSupported to SP14:SAS_Fail .....	279
5.14.4.5 SP10:SAS_AwaitALIGN state .....	279
5.14.4.5.1 State description .....	279
5.14.4.5.2 Transition SP10:SAS_AwaitALIGN to SP0:OOB_COMINIT .....	280
5.14.4.5.3 Transition SP10:SAS_AwaitALIGN to SP11:SAS_AwaitALIGN1 .....	280
5.14.4.5.4 Transition SP10:SAS_AwaitALIGN to SP12:SAS_AwaitSNW .....	280
5.14.4.5.5 Transition SP10:SAS_AwaitALIGN to SP14:SAS_Fail .....	280
5.14.4.6 SP11:SAS_AwaitALIGN1 state .....	280
5.14.4.6.1 State description .....	280
5.14.4.6.2 Transition SP11:SAS_AwaitALIGN1 to SP0:OOB_COMINIT .....	280
5.14.4.6.3 Transition SP11:SAS_AwaitALIGN1 to SP12:SAS_AwaitSNW .....	280
5.14.4.6.4 Transition SP11:SAS_AwaitALIGN1 to SP14:SAS_Fail .....	280
5.14.4.7 SP12:SAS_AwaitSNW state .....	281
5.14.4.7.1 State description .....	281
5.14.4.7.2 Transition SP12:SAS_AwaitSNW to SP0:OOB_COMINIT .....	281
5.14.4.7.3 Transition SP12:SAS_AwaitSNW to SP13:SAS_Pass .....	281
5.14.4.8 SP13:SAS_Pass state .....	281
5.14.4.8.1 State description .....	281
5.14.4.8.2 Transition SP13:SAS_Pass to SP0:OOB_COMINIT .....	281
5.14.4.8.3 Transition SP13:SAS_Pass to SP8:SAS_Start .....	281
5.14.4.8.4 Transition SP13:SAS_Pass to SP15:SAS_PHY_Ready .....	282
5.14.4.9 SP14:SAS_Fail state .....	282
5.14.4.9.1 State description .....	282
5.14.4.9.2 Transition SP14:SAS_Fail to SP1:OOB_AwaitCOMX .....	282
5.14.4.9.3 Transition SP14:SAS_Fail to SP8:SAS_Start .....	282
5.14.4.10 SP15:SAS_PHY_Ready state .....	282
5.14.4.10.1 State description .....	282
5.14.4.10.2 Transition SP15:SAS_PHY_Ready to SP0:OOB_COMINIT .....	283
5.14.4.10.3 Transition SP15:SAS_PHY_Ready to SP31:SAS_PS_Low_Phy_Power .....	283
5.14.4.11 SP27:SAS_Settings state .....	283
5.14.4.11.1 State description .....	283
5.14.4.11.2 Transition SP27:SAS_Settings to SP0:OOB_COMINIT .....	284
5.14.4.11.3 Transition SP27:SAS_Settings to SP1:OOB_AwaitCOMX .....	284
5.14.4.11.4 Transition SP27:SAS_Settings to SP8:SAS_Start .....	284
5.14.4.11.5 Transition SP27:SAS_Settings to SP28:SAS_TrainSetup .....	284
5.14.4.12 SP28:SAS_TrainSetup .....	284
5.14.4.12.1 State description .....	284
5.14.4.12.2 Transition SP28:SAS_TrainSetup to SP0:OOB_COMINIT .....	285
5.14.4.12.3 Transition SP28:SAS_TrainSetup to SP29:SAS_Train_Rx .....	285
5.14.4.12.4 Transition SP28:SAS_TrainSetup to SP34:SAS_Train_Tx .....	285
5.14.4.13 SP34:SAS_Train_Tx state .....	285
5.14.4.13.1 State description .....	285
5.14.4.13.2 Transition SP34:SAS_Train_Tx to SP1:OOB_AwaitCOMX .....	286
5.14.4.13.3 Transition SP34:SAS_Train_Tx to SP28:SAS_TrainSetup .....	286
5.14.4.13.4 Transition SP34:SAS_Train_Tx to SP29:SAS_Train_Rx .....	286
5.14.4.14 SP29:SAS_Train_Rx state .....	286
5.14.4.14.1 State description .....	286
5.14.4.14.2 Transition SP29:SAS_Train_Rx to SP0:OOB_COMINIT .....	287
5.14.4.14.3 Transition SP29:SAS_Train_Rx to SP1:OOB_AwaitCOMX .....	287

5.14.4.14.4 Transition SP29:SAS_Train_Rx to SP28:SAS_TrainSetup .....	287
5.14.4.14.5 Transition SP29:SAS_Train_Rx to SP30:SAS_TrainingDone .....	287
5.14.4.15 SP30:SAS_TrainingDone state .....	288
5.14.4.15.1 State description .....	288
5.14.4.15.2 Transition SP30:SAS_TrainingDone to SP0:OOB_COMINIT .....	288
5.14.4.15.3 Transition SP30:SAS_TrainingDone to SP1:OOB_AwaitCOMX .....	288
5.14.4.15.4 Transition SP30:SAS_TrainingDone to SP28:SAS_TrainSetup .....	288
5.14.4.15.5 Transition SP30:SAS_TrainingDone to SP15:SAS_PHY_Ready .....	288
5.14.5 SAS phy power conditions states .....	289
5.14.5.1 SAS phy power conditions states overview .....	289
5.14.5.2 SP31:SAS_PS_Low_Phy_Power state .....	290
5.14.5.2.1 State description .....	290
5.14.5.2.2 Transition SP31:SAS_PS_Low_Phy_Power to SP0:OOB_COMINIT .....	291
5.14.5.2.3 Transition SP31:SAS_PS_Low_Phy_Power to SP32:SAS_PS_ALIGN0 .....	291
5.14.5.2.4 Transition SP31:SAS_PS_Low_Phy_Power to SP35:SAS_PS_Sync .....	291
5.14.5.3 SP32:SAS_PS_ALIGN0 state .....	291
5.14.5.3.1 State description .....	291
5.14.5.3.2 Transition SP32:SAS_PS_ALIGN0 state to SP0:OOB_COMINIT .....	292
5.14.5.3.3 Transition SP32:SAS_PS_ALIGN0 to SP33:SAS_PS_ALIGN1 .....	292
5.14.5.4 SP33:SAS_PS_ALIGN1 state .....	292
5.14.5.4.1 State description .....	292
5.14.5.4.2 Transition SP33:SAS_PS_ALIGN1 state to SP0:OOB_COMINIT .....	292
5.14.5.4.3 Transition SP33:SAS_PS_ALIGN1 state to SP15:SAS_PHY_Ready .....	292
5.14.5.5 SP35:SAS_PS_Sync state .....	292
5.14.5.5.1 State description .....	292
5.14.5.5.2 Transition SP35:SAS_PS_SYNC state to SP0:OOB_COMINIT .....	293
5.14.5.5.3 Transition SP35:SAS_PS_SYNC state to SP15:SAS_PHY_Ready .....	293
5.14.6 SATA host emulation states .....	293
5.14.6.1 SATA host emulation states overview .....	293
5.14.6.2 SP16:SATA_COMWAKE state .....	295
5.14.6.2.1 State description .....	295
5.14.6.2.2 Transition SP16:SATA_COMWAKE to SP0:OOB_COMINIT .....	295
5.14.6.2.3 Transition SP16:SATA_COMWAKE to SP17:SATA_AwaitCOMWAKE .....	295
5.14.6.3 SP17:SATA_AwaitCOMWAKE state .....	295
5.14.6.3.1 State description .....	295
5.14.6.3.2 Transition SP17:SATA_AwaitCOMWAKE to SP0:OOB_COMINIT .....	295
5.14.6.3.3 Transition SP17:SATA_AwaitCOMWAKE to SP18:SATA_AwaitNoCOMWAKE .....	295
5.14.6.4 SP18:SATA_AwaitNoCOMWAKE state .....	295
5.14.6.4.1 State description .....	295
5.14.6.4.2 Transition SP18:SATA_AwaitNoCOMWAKE to SP0:OOB_COMINIT .....	295
5.14.6.4.3 Transition SP18:SATA_AwaitNoCOMWAKE to SP19:SATA_AwaitALIGN .....	295
5.14.6.5 SP19:SATA_AwaitALIGN state .....	296
5.14.6.5.1 State description .....	296
5.14.6.5.2 Transition SP19:SATA_AwaitALIGN to SP0:OOB_COMINIT .....	296
5.14.6.5.3 Transition SP19:SATA_AwaitALIGN to SP20:SATA_AdjustSpeed .....	296
5.14.6.6 SP20:SATA_AdjustSpeed state .....	296
5.14.6.6.1 State description .....	296
5.14.6.6.2 Transition SP20:SATA_AdjustSpeed to SP0:OOB_COMINIT .....	296
5.14.6.6.3 Transition SP20:SATA_AdjustSpeed to SP21:SATA_TransmitALIGN .....	297
5.14.6.7 SP21:SATA_TransmitALIGN state .....	297
5.14.6.7.1 State description .....	297
5.14.6.7.2 Transition SP21:SATA_TransmitALIGN to SP0:OOB_COMINIT .....	297
5.14.6.7.3 Transition SP21:SATA_TransmitALIGN to SP22:SATA_PHY_Ready .....	297
5.14.6.8 SP22:SATA_PHY_Ready state .....	297
5.14.6.8.1 State description .....	297
5.14.6.8.2 Transition SP22:SATA_PHY_Ready to SP0:OOB_COMINIT .....	297
5.14.6.8.3 Transition SP22:SATA_PHY_Ready to SP23:SATA_PM_Partial .....	298



5.14.6.8.4 Transition SP22:SATA_PHY_Ready to SP24:SATA_PM_Slumber .....	298
5.14.6.9 SP23:SATA_PM_Partial state .....	298
5.14.6.9.1 State description .....	298
5.14.6.9.2 Transition SP23:SATA_PM_Partial to SP0:OOB_COMINIT .....	298
5.14.6.9.3 Transition SP23:SATA_PM_Partial to SP16:SATA_COMWAKE .....	298
5.14.6.9.4 Transition SP23:SATA_PM_Partial to SP19:SATA_AwaitALIGN .....	298
5.14.6.10 SP24:SATA_PM_Slumber state .....	298
5.14.6.10.1 State description .....	298
5.14.6.10.2 Transition SP24:SATA_PM_Slumber to SP0:OOB_COMINIT .....	298
5.14.6.10.3 Transition SP24:SATA_PM_Slumber to SP16:SATA_COMWAKE .....	299
5.14.6.10.4 Transition SP24:SATA_PM_Slumber to SP19:SATA_AwaitALIGN .....	299
5.14.7 SATA port selector state SP25:SATA_PortSel .....	299
5.14.7.1 State description .....	299
5.14.7.2 Transition SP25:SATA_PortSel to SP1:OOB_AwaitCOMX .....	299
5.14.8 SATA spinup hold state SP26:SATA_SpinupHold .....	300
5.14.8.1 State description .....	300
5.14.8.2 Transition SP26:SATA_SpinupHold to SP0:OOB_COMINIT .....	300
5.15 SP_DWS (phy layer dword synchronization) state machine .....	300
5.15.1 SP_DWS state machine overview .....	300
5.15.2 SP_DWS receiver .....	303
5.15.3 SP_DWS0:AcquireSync state .....	303
5.15.3.1 State description .....	303
5.15.3.2 Transition SP_DWS0:AcquireSync to SP_DWS1:Valid1 .....	304
5.15.4 SP_DWS1:Valid1 state .....	304
5.15.4.1 State description .....	304
5.15.4.2 Transition SP_DWS1:Valid1 to SP_DWS0:AcquireSync .....	304
5.15.4.3 Transition SP_DWS1:Valid1 to SP_DWS2:Valid2 .....	304
5.15.5 SP_DWS2:Valid2 state .....	304
5.15.5.1 State description .....	304
5.15.5.2 Transition SP_DWS2:Valid2 to SP_DWS0:AcquireSync .....	304
5.15.5.3 Transition SP_DWS2:Valid2 to SP_DWS3:SyncAcquired .....	304
5.15.6 SP_DWS3:SyncAcquired state .....	305
5.15.6.1 State description .....	305
5.15.6.2 Transition SP_DWS3:SyncAcquired to SP_DWS0:AcquireSync .....	305
5.15.6.3 Transition SP_DWS3:SyncAcquired to SP_DWS4:Lost1 .....	305
5.15.7 SP_DWS4:Lost1 state .....	305
5.15.7.1 State description .....	305
5.15.7.2 Transition SP_DWS4:Lost1 to SP_DWS0:AcquireSync .....	305
5.15.7.3 Transition SP_DWS4:Lost1 to SP_DWS5:Lost1Recovered .....	305
5.15.7.4 Transition SP_DWS4:Lost1 to SP_DWS6:Lost2 .....	305
5.15.8 SP_DWS5:Lost1Recovered state .....	306
5.15.8.1 State description .....	306
5.15.8.2 Transition SP_DWS5:Lost1Recovered to SP_DWS0:AcquireSync .....	306
5.15.8.3 Transition SP_DWS5:Lost1Recovered to SP_DWS3:SyncAcquired .....	306
5.15.8.4 Transition SP_DWS5:Lost1Recovered to SP_DWS6:Lost2 .....	306
5.15.9 SP_DWS6:Lost2 state .....	306
5.15.9.1 State description .....	306
5.15.9.2 Transition SP_DWS6:Lost2 to SP_DWS0:AcquireSync .....	306
5.15.9.3 Transition SP_DWS6:Lost2 to SP_DWS7:Lost2Recovered .....	306
5.15.9.4 Transition SP_DWS6:Lost2 to SP_DWS8:Lost3 .....	306
5.15.10 SP_DWS7:Lost2Recovered state .....	307
5.15.10.1 State description .....	307
5.15.10.2 Transition SP_DWS7:Lost2Recovered to SP_DWS0:AcquireSync .....	307
5.15.10.3 Transition SP_DWS7:Lost2Recovered to SP_DWS4:Lost1 .....	307
5.15.10.4 Transition SP_DWS7:Lost2Recovered to SP_DWS8:Lost3 .....	307
5.15.11 SP_DWS8:Lost3 state .....	307
5.15.11.1 State description .....	307

5.15.11.2 Transition SP_DWS8:Lost3 to SP_DWS0:AcquireSync .....	307
5.15.11.3 Transition SP_DWS8:Lost3 to SP_DWS9:Lost3Recovered .....	307
5.15.12 SP_DWS9:Lost3Recovered state .....	307
5.15.12.1 State description .....	307
5.15.12.2 Transition SP_DWS9:Lost3Recovered to SP_DWS0:AcquireSync .....	308
5.15.12.3 Transition SP_DWS9:Lost3Recovered to SP_DWS6:Lost2 .....	308
5.16 SP_PS (phy layer SPL packet synchronization) state machine .....	308
5.16.1 SP_PS state machine overview .....	308
5.16.2 SP_PS receiver .....	310
5.16.3 SP_PS0:AcquireSync state .....	314
5.16.3.1 State description .....	314
5.16.3.2 Transition SP_PS0:AcquireSync to SP_PS1:Valid1 .....	314
5.16.4 SP_PS1:Valid1 state .....	314
5.16.4.1 State description .....	314
5.16.4.2 Transition SP_PS1:Valid1 to SP_PS0:AcquireSync .....	314
5.16.4.3 Transition SP_PS1:Valid1 to SP_PS2:SyncAcquired .....	314
5.16.5 SP_PS2:SyncAcquired state .....	315
5.16.5.1 State description .....	315
5.16.5.2 Transition SP_PS2:SyncAcquired to SP_PS0:AcquireSync .....	315
5.16.5.3 Transition SP_PS2:SyncAcquired to SP_PS3:Lost1 .....	315
5.16.6 SP_PS3:Lost1 state .....	315
5.16.6.1 State description .....	315
5.16.6.2 Transition SP_PS3:Lost1 to SP_PS0:AcquireSync .....	315
5.16.6.3 Transition SP_PS3:Lost1 to SP_PS5:Lost2 .....	315
5.16.6.4 Transition SP_PS3:Lost1 to SP_PS4:LostRecovered .....	315
5.16.7 SP_PS4:LostRecovered state .....	315
5.16.7.1 State description .....	315
5.16.7.2 Transition SP_PS4:LostRecovered to SP_PS0:AcquireSync .....	315
5.16.7.3 Transition SP_PS4:LostRecovered to SP_PS2:SyncAcquired .....	316
5.16.7.4 Transition SP_PS4:LostRecovered to SP_PS3:Lost1 .....	316
5.16.8 SP_PS5:Lost2 state .....	316
5.16.8.1 State description .....	316
5.16.8.2 Transition SP_PS5:Lost2 to SP_PS0:AcquireSync .....	316
5.16.8.3 Transition SP_PS5:Lost2 to SP_PS6:Lost3 .....	316
5.16.9 SP_PS6:Lost3 state .....	316
5.16.9.1 State description .....	316
5.16.9.2 Transition SP_PS6:Lost3 to SP_PS0:AcquireSync .....	316
5.16.9.3 Transition SP_PS6:Lost3 to SP_PS4:LostRecovered .....	316
5.17 SP_ReSync (phy layer resynchronization) state machine .....	317
5.17.1 SP_ReSync state machine overview .....	317
5.17.2 SP_ReSync0:Start .....	318
5.17.2.1 State description .....	318
5.17.2.2 Transition SP_ReSync0:Start to SP_ReSync1:Request .....	318
5.17.2.3 Transition SP_ReSync0:Start to SP_ReSync2:Response .....	318
5.17.3 SP_ReSync1:Request .....	319
5.17.3.1 Transition SP_ReSync1 to SP_ReSync0:Start .....	319
5.17.3.2 Transition SP_ReSync1 to SP_ReSync2:Response .....	319
5.17.4 SP_ReSync2:Response state .....	319
5.17.4.1 State description .....	319
5.17.4.2 Transition SP_ReSync2:Response state to SP_ReSync0:Start .....	320
5.18 PTT (phy layer transmitter training) state machines .....	320
5.18.1 PTT state machines overview .....	320
5.18.2 SP transmitter additions for transmitter training .....	320
5.18.2.1 SP transmitter additions for transmitter training overview .....	320
5.18.2.2 TTU transmit setup .....	321
5.18.2.3 No_equalization, reference_1, and reference_2 coefficient settings request .....	321
5.18.2.4 Coefficient limits .....	321

5.18.2.5 Coefficient request result of update complete .....	321
5.18.2.5.1 Coefficient request processing .....	321
5.18.2.5.2 Coefficient adjustment completes .....	322
5.18.2.5.3 No coefficient adjustment .....	322
5.18.2.6 Coefficient request result of maximum .....	322
5.18.2.6.1 Coefficient request processing .....	322
5.18.2.6.2 Coefficient adjustment completes .....	323
5.18.2.6.3 No coefficient adjustment .....	323
5.18.2.7 Coefficient request result of minimum .....	323
5.18.2.7.1 Coefficient request processing .....	323
5.18.2.7.2 Coefficient adjustment completes .....	323
5.18.2.7.3 No coefficient adjustment .....	323
5.18.3 SP receiver additions for transmitter training .....	324
5.18.4 PTT_T (phy layer transmitter training transmit pattern) state machine .....	325
5.18.4.1 PTT_T state machine overview .....	325
5.18.4.2 PTT_T0:Idle state .....	327
5.18.4.2.1 State description .....	327
5.18.4.2.2 Transition PTT_T0:Idle to PTT_T1:Initialize .....	327
5.18.4.3 PTT_T1:Initialize state .....	327
5.18.4.3.1 State description .....	327
5.18.4.3.2 Transition PTT_T1:Initialize to PTT_T0:Idle .....	327
5.18.4.3.3 Transition PTT_T1:Initialize to PTT_T2:Tx_Training .....	328
5.18.4.4 PTT_T2:Tx_Training state .....	328
5.18.4.4.1 State description .....	328
5.18.4.4.2 Entry conditions .....	328
5.18.4.4.3 Control word and status word mappings .....	329
5.18.4.4.4 Error message handling .....	330
5.18.4.4.5 Resetting attached phy's transmitter .....	330
5.18.4.4.6 Local phy's transmitter and attached phy's transmitter training completed .....	331
5.18.4.4.7 Transition PTT_T2:Tx_Training to PTT_T0:Idle .....	331
5.18.4.4.8 Transition PTT_T2:Tx_Training to PTT_T3:Local_Tx_Training .....	332
5.18.4.5 PTT_T3:Local_Tx_Training state .....	332
5.18.4.5.1 State description .....	332
5.18.4.5.2 Entry conditions .....	332
5.18.4.5.3 Status word mappings .....	332
5.18.4.5.4 Local phy's transmitter and attached phy's transmitter training completed .....	332
5.18.4.5.5 Error message handling .....	333
5.18.4.5.6 Transition PTT_T3:Local_Tx_Training to PTT_T0:Idle .....	333
5.18.5 PTT_R (phy layer transmitter training receive pattern) state machine .....	333
5.18.5.1 PTT_R state machine overview .....	333
5.18.5.2 PTT_R0:Idle state .....	335
5.18.5.2.1 State description .....	335
5.18.5.2.2 Transition PTT_R0:Idle to PTT_R1:Initialize .....	335
5.18.5.3 PTT_R1:Initialize state .....	335
5.18.5.3.1 State description .....	335
5.18.5.3.2 Transition PTT_R1:Initialize to PTT_R0:Idle .....	335
5.18.5.3.3 Transition PTT_R1:Initialize to PTT_R2:Receive_Train_Tx_Pattern .....	335
5.18.5.4 PTT_R2:Receive_Train_Tx_Pattern state .....	335
5.18.5.4.1 State description .....	335
5.18.5.4.2 Transition PTT_R2:Receive_Train_Tx_Pattern to PTT_R0:Idle .....	341
5.18.5.4.3 Transition PTT_R2:Receive_Train_Tx_Pattern to PTT_R1:Initialize .....	341
5.18.6 PTT_SC (phy layer transmitter training set transmitter coefficient) state machines .....	341
5.18.6.1 PTT_SC (phy layer transmitter training set transmitter coefficient) state machines overview .....	341
5.18.6.2 PTT_SC1 state machine overview .....	343
5.18.6.3 PTT_SC1_0:Idle state .....	343
5.18.6.3.1 State description .....	343
5.18.6.3.2 Transition PTT_SC1_0:Idle to PTT_SC1_1:Wait_Inc_Dec .....	343

5.18.6.4 PTT_SC1_1:Wait_Inc_Dec state .....	343
5.18.6.4.1 State description .....	343
5.18.6.4.2 Transition PTT_SC1_1:Wait_Inc_Dec to PTT_SC1_0:Idle .....	343
5.18.6.4.3 Transition PTT_SC1_1:Wait_Inc_Dec to PTT_SC1_2:Set_Coefficient.....	343
5.18.6.5 PTT_SC1_2:Set_Coefficient state .....	344
5.18.6.5.1 State description .....	344
5.18.6.5.2 Transition PTT_SC1_2:Set_Coefficient to PTT_SC1_0:Idle .....	345
5.18.6.5.3 Transition PTT_SC1_2:Set_Coefficient to PTT_SC1_3:Wait_Hold .....	345
5.18.6.6 PTT_SC1_3:Wait_Hold state .....	345
5.18.6.6.1 State description .....	345
5.18.6.6.2 Transition PTT_SC1_3:Wait_Hold to PTT_SC1_0:Idle .....	345
5.18.6.6.3 Transition PTT_SC1_3:Wait_Hold to PTT_SC1_1:Wait_Inc_Dec .....	345
5.18.7 PTT_SC2 (phy layer transmitter training set transmitter coefficient 2) state machine .....	345
5.18.8 PTT_SC3 (phy layer transmitter training set transmitter coefficient 3) state machine .....	346
5.18.9 PTT_GC (phy layer transmitter training get transmitter coefficient) state machines.....	346
5.18.9.1 PTT_GC (phy layer transmitter training get transmitter coefficient) state machines overview	346
5.18.9.2 PTT_GC1 state machine.....	347
5.18.9.3 PTT_GC1_0:Idle state.....	348
5.18.9.3.1 State description .....	348
5.18.9.3.2 Transition PTT_GC1_0:Idle to PTT_GC1_1:Get_Coefficient .....	348
5.18.9.4 PTT_GC1_1:Get_Coefficient state.....	348
5.18.9.4.1 State description .....	348
5.18.9.4.2 Transition PTT_GC1_1:Get_Coefficient to PTT_GC1_0:Idle .....	348
5.18.9.4.3 Transition PTT_GC1_1:Get_Coefficient to PTT_GC1_2:Wait_Restart .....	348
5.18.9.5 PTT_GC1_2:Wait_Restart state.....	349
5.18.9.5.1 State description .....	349
5.18.9.5.2 Transition PTT_GC1_2:Wait_Restart to PTT_GC1_0:Idle .....	349
5.18.10 PTT_GC2 (phy layer transmitter training get transmitter coefficient 2) state machine .....	349
5.18.11 PTT_GC3 (phy layer transmitter training get transmitter coefficient 3) state machine .....	349
5.18.12 PTT_PL (phy layer transmitter training pattern lock) state machine .....	350
5.18.12.1 PTT_PL state machine overview.....	350
5.18.12.2 PTT_PL0:Idle state.....	351
5.18.12.2.1 State description .....	351
5.18.12.2.2 Transition PTT_PL0:Idle to PTT_PL1:Acquire_Lock .....	352
5.18.12.3 PTT_PL1:Acquire_Lock state.....	352
5.18.12.3.1 State description .....	352
5.18.12.3.2 Transition PTT_PL1:Acquire_Lock to PTT_PL2:Valid .....	352
5.18.12.4 PTT_PL2:Valid state .....	352
5.18.12.4.1 State description .....	352
5.18.12.4.2 Transition PTT_PL2:Valid to PTT_PL1:Acquire_Lock.....	352
5.18.12.4.3 Transition PTT_PL2:Valid to PTT_PL3:Lock_Acquired .....	352
5.18.12.5 PTT_PL3:Lock_Acquired state.....	352
5.18.12.5.1 State description .....	352
5.18.12.5.2 Transition PTT_PL3:Lock_Acquired to PTT_PL4:Lost1 .....	352
5.18.12.6 PTT_PL4:Lost1 state.....	353
5.18.12.6.1 State description .....	353
5.18.12.6.2 Transition PTT_PL4:Lost1 to PTT_PL3:Lock_Acquired .....	353
5.18.12.6.3 Transition PTT_PL4:Lost1 to PTT_PL5:Lost2 .....	353
5.18.12.7 PTT_PL5:Lost2 state.....	353
5.18.12.7.1 State description .....	353
5.18.12.7.2 Transition PTT_PL5:Lost2 to PTT_PL3:Lock_Acquired .....	353
5.18.12.7.3 Transition PTT_PL5:Lost2 to PTT_PL6:Lost3 .....	353
5.18.12.8 PTT_PL6:Lost3 state.....	353
5.18.12.8.1 State description .....	353
5.18.12.8.2 Transition PTT_PL6:Lost3 to PTT_PL3:Lock_Acquired .....	353
5.18.12.8.3 Transition PTT_PL6:Lost3 to PTT_PL7:Lost4 .....	353
5.18.12.9 PTT_PL7:Lost4 state.....	353

5.18.12.9.1 State description .....	353
5.18.12.9.2 Transition PTT_PL7:Lost4 to PTT_PL3:Lock_Acquired .....	354
5.18.12.9.3 Transition PTT_PL7:Lost4 to PTT_PL1:Acquire_Lock .....	354
5.19 PAPTA (phy layer active phy transmitter adjustment) state machines .....	354
5.19.1 PAPTA state machines overview .....	354
5.19.2 SP transmitter additions for APTA .....	354
5.19.2.1 SP transmitter additions for APTA overview .....	354
5.19.2.2 SP transmitter sends APTA binary primitives messages when messages are received .....	355
5.19.2.3 APTA Coefficient limits .....	356
5.19.2.3.1 APTA Coefficient limits overview .....	356
5.19.2.3.2 APTA Coefficient request result of updated .....	356
5.19.2.3.2.1 APTA Coefficient request processing .....	356
5.19.2.3.2.2 APTA Coefficient adjustment completes .....	356
5.19.2.3.3 APTA Coefficient request result of maximum .....	356
5.19.2.3.3.1 APTA Coefficient request processing .....	356
5.19.2.3.3.2 APTA Coefficient adjustment completes .....	357
5.19.2.3.3.3 APTA No coefficient adjustment .....	357
5.19.2.3.4 APTA Coefficient request result of minimum .....	357
5.19.2.3.4.1 APTA Coefficient request processing .....	357
5.19.2.3.4.2 APTA Coefficient adjustment completes .....	357
5.19.2.3.4.3 APTA No coefficient adjustment .....	358
5.19.3 SP receiver additions for APTA .....	358
5.19.3.1 SP receiver additions for APTA overview .....	358
5.19.3.2 SP receiver messages when APTA binary primitives are received .....	358
5.19.3.3 SP receiver messages that tune the attached SP transmitter .....	359
5.19.4 PAPTA_A_L (phy layer attached SP receiver adjusts the local SP transmitter coefficients) state machine .....	360
5.19.4.1 PAPTA_A_L state machine overview .....	360
5.19.4.2 PAPTA_A_L0:Idle state .....	361
5.19.4.2.1 State description .....	361
5.19.4.2.2 Transition PAPTA_A_L0:Idle to PAPTA_A_L1:Wait_for_Start .....	362
5.19.4.3 PAPTA_A_L1:Wait_For_Start state .....	362
5.19.4.3.1 State description .....	362
5.19.4.3.2 Transition PAPTA_A_L1:Wait_For_Start to PAPTA_A_L0:Idle .....	362
5.19.4.3.3 Transition PAPTA_A_L1:Wait_For_Start to PAPTA_A_L2:Adjust_Local_Transmitter .....	362
5.19.4.4 PAPTA_A_L2:Adjust_Local_Transmitter state .....	362
5.19.4.4.1 State description .....	362
5.19.4.4.2 Transition PAPTA_A_L2:Adjust_Local_Transmitter to PAPTA_A_L0:Idle .....	362
5.19.4.4.3 Transition PAPTA_A_L2:Adjust_Local_Transmitter to PAPTA_A_L1:Wait_For_Start .....	362
5.19.5 PAPTA_L_A (phy layer local SP receiver adjusts the attached SP transmitter coefficients) state machine .....	363
5.19.5.1 PAPTA_L_A state machine overview .....	363
5.19.5.2 PAPTA_L_A0:Initialize state .....	364
5.19.5.2.1 State description .....	364
5.19.5.2.2 Transition from PAPTA_L_A0:Initialize to PAPTA_L_A1:Start .....	365
5.19.5.3 PAPTA_L_A1:Start state .....	365
5.19.5.3.1 State description .....	365
5.19.5.3.2 Transition from PAPTA_L_A1:Start to PAPTA_L_A0:Initialize .....	365
5.19.5.3.3 Transition from PAPTA_L_A1:Start to PAPTA_L_A2:Adjust_Attached_Transmitter .....	365
5.19.5.4 PAPTA_L_A2:Adjust_Attached_Transmitter state .....	365
5.19.5.4.1 State description .....	365
5.19.5.4.2 Transition from PAPTA_L_A2:Adjust_Attached_Transmitter to PAPTA_L_A0:Initialize ...	367
5.19.6 PAPTA_TC (phy layer SP receiver management of attached SP transmitter coefficient adjustments) state machines .....	367
5.19.6.1 PAPTA_TC state machines overview .....	367
5.19.6.2 PAPTA_TC1 state machine .....	368
5.19.6.2.1 PAPTA_TC1 state machine overview .....	368

5.19.6.2.2 PAPT <sub>A</sub> _TC1_0:Idle state .....	369
5.19.6.2.2.1 State description .....	369
5.19.6.2.2.2 Transition PAPT <sub>A</sub> _TC1_0:Idle to PAPT <sub>A</sub> _TC1_1:Request_Change .....	369
5.19.6.2.3 PAPT <sub>A</sub> _TC1_1:Request_Change state .....	369
5.19.6.2.3.1 State description .....	369
5.19.6.2.3.2 Transition PAPT <sub>A</sub> _TC1_1:Request_Change to PAPT <sub>A</sub> _TC0:Idle .....	369
5.19.6.3 PAPT <sub>A</sub> _TC2 state machine .....	370
5.19.6.4 PAPT <sub>A</sub> _TC3 state machine .....	370
5.19.6.5 PAPT <sub>A</sub> _TC1_2 state machine .....	370
5.19.6.6 PAPT <sub>A</sub> _TC2_3 state machine .....	371
5.20 Multiplexing.....	371
5.21 Spinup .....	372
6 Link layer.....	374
6.1 Link layer overview .....	374
6.2 Primitives .....	374
6.2.1 Primitives overview .....	374
6.2.2 Primitive summary .....	375
6.2.3 Primitive encodings.....	384
6.2.4 Primitive sequences.....	389
6.2.4.1 Primitive sequences overview .....	389
6.2.4.2 Single primitive sequence .....	389
6.2.4.3 Repeated primitive sequence.....	390
6.2.4.4 Continued primitive sequence .....	390
6.2.4.5 Extended primitive sequence .....	391
6.2.4.6 Triple primitive sequence .....	392
6.2.4.7 Redundant primitive sequence.....	394
6.2.5 Deletable primitives.....	396
6.2.5.1 ALIGN.....	396
6.2.5.2 MUX (Multiplex).....	397
6.2.5.3 NOTIFY .....	398
6.2.5.3.1 NOTIFY overview .....	398
6.2.5.3.2 NOTIFY (ENABLE SPINUP).....	399
6.2.5.3.3 NOTIFY (POWER LOSS EXPECTED).....	399
6.2.5.4 OOB_IDLE .....	400
6.2.6 Primitives not specific to type of connections .....	400
6.2.6.1 AIP (Arbitration in progress).....	400
6.2.6.2 BREAK .....	401
6.2.6.3 BREAK_REPLY .....	401
6.2.6.4 BROADCAST .....	401
6.2.6.5 CLOSE .....	402
6.2.6.5.1 CLOSE overview .....	402
6.2.6.5.2 CLOSE primitive parameter .....	403
6.2.6.5.3 CLOSE primitive parameter fields when being set from an open address frame .....	403
6.2.6.5.4 CLOSE primitive parameter fields when being set from a received CLOSE with a primitive parameter .....	404
6.2.6.6 EOAF (End of address frame).....	404
6.2.6.7 ERROR .....	404
6.2.6.8 HARD_RESET .....	404
6.2.6.9 OPEN_ACCEPT.....	404
6.2.6.10 OPEN_REJECT .....	404
6.2.6.10.1 OPEN_REJECT overview .....	404
6.2.6.10.2 OPEN_REJECT retry-class primitive parameter .....	407
6.2.6.11 PS_ACK .....	408
6.2.6.12 PS_NAK .....	408
6.2.6.13 PS_REQ.....	408
6.2.6.14 PWR_ACK.....	408

6.2.6.15 PWR_DONE.....	408
6.2.6.16 PWR_GRANT .....	409
6.2.6.17 PWR_REQ .....	409
6.2.6.18 SOAF (Start of address frame).....	409
6.2.6.19 TRAIN.....	409
6.2.6.20 TRAIN_DONE .....	409
6.2.7 Primitives used only inside SSP and SMP connections .....	409
6.2.7.1 ACK (Acknowledgement) .....	409
6.2.7.2 CREDIT_BLOCKED .....	409
6.2.7.3 DONE .....	409
6.2.7.4 EOF (End of frame).....	410
6.2.7.5 EXTEND_CONNECTION.....	410
6.2.7.6 NAK (Negative acknowledgement) .....	410
6.2.7.7 RRDY (Receiver ready).....	411
6.2.7.8 SOF (Start of frame).....	411
6.2.8 Primitives used only inside STP connections and on SATA physical links .....	411
6.2.8.1 SATA_ERROR.....	411
6.2.8.2 SATA_PMACK, SATA_PMNAK, SATA_PMREQ_P, and SATA_PMREQ_S (Power management acknowledgements and requests) .....	412
6.2.8.3 SATA_HOLD and SATA_HOLD_A (Hold and hold acknowledge).....	412
6.2.8.4 SATA_R_RDY and SATA_X_RDY (Receiver ready and transmitter ready).....	412
6.2.8.5 SATA_EOF (End of frame).....	412
6.2.8.6 Other primitives used inside STP connections and on SATA physical links .....	412
6.3 Binary primitives .....	413
6.3.1 Binary primitives overview .....	413
6.3.2 Binary primitive codes .....	416
6.3.3 Binary primitive sequences .....	417
6.3.3.1 Binary primitive sequences overview .....	417
6.3.3.2 Single binary primitive sequence.....	417
6.3.4 Deletable binary primitives .....	418
6.3.4.1 APTA_ADJUST .....	418
6.3.4.2 APTA_COEFFICIENT_1 .....	418
6.3.4.3 APTA_COEFFICIENT_2 .....	419
6.3.4.4 APTA_COEFFICIENT_3 .....	419
6.3.4.5 APTA_COEFFICIENT_1_2 .....	420
6.3.4.6 APTA_COEFFICIENT_2_3 .....	420
6.3.5 Binary primitives used only inside SSP and STP connections .....	421
6.3.5.1 B_EOF (binary end of frame) .....	421
6.4 Extended binary primitives .....	423
6.4.1 Deletable extended binary primitives .....	423
6.4.2 Extended binary primitive codes .....	425
6.4.3 Extended binary primitive sequences .....	426
6.4.3.1 Extended binary primitive sequences overview .....	426
6.4.3.2 Single extended binary primitive sequence .....	426
6.4.4 Deletable extended binary primitives .....	426
6.4.4.1 PACKET_SYNC .....	426
6.4.4.2 PACKET_SYNC_LOST.....	426
6.4.4.3 LINK_RATE_MANAGEMENT .....	426
6.4.5 Extended binary primitives not specific to type of connections.....	427
6.4.5.1 END_TRAIN .....	427
6.5 Physical link rate tolerance management.....	427
6.5.1 Physical link rate tolerance management overview .....	427
6.5.2 Phys originating dwords while in the SAS dword mode .....	430
6.5.3 Phys originating SPL packets while in the SAS packet mode .....	431
6.5.4 Expander phys forwarding dwords and deletable extended binary primitives .....	431
6.6 Idle physical links.....	432
6.7 CRC.....	432

6.7.1 CRC overview .....	432
6.7.2 CRC generation .....	434
6.7.3 CRC checking .....	436
6.8 Scrambling.....	437
6.8.1 Scrambling overview .....	437
6.8.2 Scrambling while in the SAS dword mode .....	438
6.8.3 Scrambling while in the SAS packet mode .....	439
6.9 Bit order of CRC and scrambler .....	440
6.9.1 Bit order of CRC and scrambler while in the SAS dword mode .....	440
6.9.2 Bit order of CRC and scrambler while in the SAS packet mode .....	444
6.10 Address frames .....	445
6.10.1 Address frames overview.....	445
6.10.2 IDENTIFY address frame.....	447
6.10.3 OPEN address frame.....	451
6.11 Link reset sequence .....	455
6.11.1 Link reset sequence overview.....	455
6.11.2 Expander device handling of link reset sequences .....	458
6.12 SL_IR (link layer identification and hard reset) state machines.....	458
6.12.1 SL_IR state machines overview.....	458
6.12.2 SL_IR transmitter and receiver .....	460
6.12.3 SL_IR_TIR (transmit IDENTIFY or HARD_RESET) state machine .....	460
6.12.3.1 SL_IR_TIR state machine overview .....	460
6.12.3.2 SL_IR_TIR1:Idle state .....	461
6.12.3.2.1 State description .....	461
6.12.3.2.2 Transition SL_IR_TIR1:Idle to SL_IR_TIR2:Transmit_Identify .....	461
6.12.3.2.3 Transition SL_IR_TIR1:Idle to SL_IR_TIR3:Transmit_Hard_Reset .....	461
6.12.3.3 SL_IR_TIR2:Transmit_Identify state .....	461
6.12.3.3.1 State description .....	461
6.12.3.3.2 Transition SL_IR_TIR2:Transmit_Identify to SL_IR_TIR4:Completed .....	461
6.12.3.4 SL_IR_TIR3:Transmit_Hard_Reset state.....	461
6.12.3.4.1 State description .....	461
6.12.3.4.2 Transition SL_IR_TIR3:Transmit_Hard_Reset to SL_IR_TIR4:Completed .....	461
6.12.3.5 SL_IR_TIR4:Completed state .....	462
6.12.4 SL_IR_RIF (receive IDENTIFY address frame) state machine .....	462
6.12.4.1 SL_IR_RIF state machine overview .....	462
6.12.4.2 SL_IR_RIF1:Idle state .....	462
6.12.4.2.1 State description .....	462
6.12.4.2.2 Transition SL_IR_RIF1:Idle to SL_IR_RIF2:Receive_Identify_Frame.....	462
6.12.4.3 SL_IR_RIF2:Receive_Identify_Frame state.....	462
6.12.4.3.1 State description .....	462
6.12.4.3.2 Transition SL_IR_RIF2:Receive_Identify_Frame to SL_IR_RIF3:Completed .....	463
6.12.4.4 SL_IR_RIF3:Completed state .....	463
6.12.5 SL_IR_IRC (identification and hard reset control) state machine .....	463
6.12.5.1 SL_IR_IRC state machine overview.....	463
6.12.5.2 SL_IR_IRC1:Idle state.....	463
6.12.5.2.1 State description .....	463
6.12.5.2.2 Transition SL_IR_IRC1:Idle to SL_IR_IRC2:Wait.....	463
6.12.5.3 SL_IR_IRC2:Wait state .....	464
6.12.5.3.1 State description .....	464
6.12.5.3.2 Transition SL_IR_IRC2:Wait to SL_IR_IRC3:Completed .....	464
6.12.5.4 SL_IR_IRC3:Completed state .....	464
6.13 Entering a low phy power condition.....	465
6.14 Power control and SL_P (link layer power control) state machines .....	465
6.14.1 Power source device.....	465
6.14.2 Power consumer device.....	466
6.14.3 NOTIFY (ENABLE SPINUP) usage .....	466
6.14.4 SL_P_S (link layer power source device) state machine.....	467



6.14.4.1 SL_P_S state machine overview.....	467
6.14.4.2 SL_P_S transmitter and SL_P_S receiver .....	468
6.14.4.3 SL_P_S_1:Idle state.....	469
6.14.4.3.1 State description .....	469
6.14.4.3.2 Transition SL_P_S_1:Idle to SL_P_S_2:Wait_Grant.....	469
6.14.4.4 SL_P_S_2:Wait_Grant state .....	469
6.14.4.4.1 State description .....	469
6.14.4.4.2 Transition SL_P_S_2:Wait_Grant to SL_P_S_1:Idle.....	470
6.14.4.4.3 Transition SL_P_S_2:Wait_Grant to SL_P_S_3:Wait_Done.....	470
6.14.4.5 SL_P_S_3:Wait_Done state.....	470
6.14.4.5.1 State description .....	470
6.14.4.5.2 Transition SL_P_S_3:Wait_Done to SL_P_S_1:Idle .....	471
6.14.5 SL_P_C (link layer power consumer device) state machine.....	471
6.14.5.1 SL_P_C state machine overview .....	471
6.14.5.2 SL_P_C receiver .....	474
6.14.5.3 SL_P_C_1:Idle state .....	474
6.14.5.3.1 State description .....	474
6.14.5.3.2 Transition SL_P_C_1:Idle to SL_P_C_2:Request_Power.....	474
6.14.5.4 SL_P_C_2:Request_Power state.....	474
6.14.5.4.1 State description .....	474
6.14.5.4.2 Transition SL_P_C_2:Request_Power to SL_P_C_1:Idle.....	475
6.14.5.4.3 Transition SL_P_C_2:Request_Power to SL_P_C_3:Wait_Grant .....	475
6.14.5.4.4 Transition SL_P_C_2:Request_Power to SL_P_C_4:Wait_Done .....	475
6.14.5.5 SL_P_C_3:Wait_Grant state .....	475
6.14.5.5.1 State description .....	475
6.14.5.5.2 Transition SL_P_C_3:Wait_Grant to SL_P_C_1:Idle .....	476
6.14.5.5.3 Transition SL_P_C_3:Wait_Grant to SL_P_C_4:Wait_Done .....	476
6.14.5.6 SL_P_C_4:Wait_Done state .....	476
6.14.5.6.1 State description .....	476
6.14.5.6.2 Transition SL_P_C_4:Wait_Done to SL_P_C_1:Idle.....	476
6.15 SAS domain changes (Broadcast (Change) usage).....	477
6.16 Connections.....	478
6.16.1 Connections overview.....	478
6.16.2 Opening a connection .....	478
6.16.2.1 Connection request .....	478
6.16.2.2 Results of a connection request .....	480
6.16.3 SMP frame priority .....	480
6.16.4 Arbitration fairness .....	480
6.16.5 Arbitration inside an expander device.....	482
6.16.5.1 Expander logical phy arbitration requirements .....	482
6.16.5.2 ECM arbitration requirements .....	482
6.16.5.2.1 ECM arbitration requirements overview.....	482
6.16.5.2.2 Arbitrating confirmations .....	484
6.16.5.2.3 Arb Won confirmation .....	484
6.16.5.2.4 Arb Lost confirmation.....	485
6.16.5.2.5 Arb Reject confirmation .....	485
6.16.5.3 Arbitration status .....	486
6.16.5.4 Partial Pathway Timeout timer .....	486
6.16.5.5 Pathway recovery.....	487
6.16.6 BREAK handling .....	487
6.16.7 Aborting a connection request.....	488
6.16.8 Expander device request for an SSP connection close .....	490
6.16.9 Closing a connection.....	490
6.16.10 Expander device closing a connection.....	491
6.16.11 Breaking a connection .....	492
6.17 Rate matching .....	493
6.17.1 Rate matching overview.....	493

6.17.2 Rate matching while in the SAS dword mode .....	493
6.17.3 Rate matching while in the SAS packet mode .....	495
6.18 SL (link layer for SAS logical phys) state machines .....	497
6.18.1 SL state machines overview .....	497
6.18.2 SL transmitter and receiver .....	500
6.18.3 SL_RA (receive OPEN address frame) state machine .....	502
6.18.4 SL_CC (connection control) state machine .....	502
6.18.4.1 SL_CC state machine overview .....	502
6.18.4.2 SL_CC0:Idle state .....	504
6.18.4.2.1 State description .....	504
6.18.4.2.2 Transition SL_CC0:Idle to SL_CC1:ArbSel .....	506
6.18.4.2.3 Transition SL_CC0:Idle to SL_CC2:Selected .....	506
6.18.4.2.4 Transition SL_CC0:Idle to SL_CC8:PS_Request .....	507
6.18.4.2.5 Transition SL_CC0:Idle to SL_CC9:PS_Quiet .....	507
6.18.4.3 SL_CC1:ArbSel state .....	507
6.18.4.3.1 State description .....	507
6.18.4.3.2 Transition SL_CC1:ArbSel to SL_CC0:Idle .....	509
6.18.4.3.3 Transition SL_CC1:ArbSel to SL_CC2:Selected .....	509
6.18.4.3.4 Transition SL_CC1:ArbSel to SL_CC3:Connected .....	510
6.18.4.3.5 Transition SL_CC1:ArbSel to SL_CC5:BreakWait .....	510
6.18.4.3.6 Transition SL_CC1:ArbSel to SL_CC6:Break .....	511
6.18.4.4 SL_CC2:Selected state .....	511
6.18.4.4.1 State description .....	511
6.18.4.4.2 Transition SL_CC2:Selected to SL_CC0:Idle .....	512
6.18.4.4.3 Transition SL_CC2:Selected to SL_CC3:Connected .....	512
6.18.4.4.4 Transition SL_CC2:Selected to SL_CC5:BreakWait .....	512
6.18.4.4.5 Transition SL_CC2:Selected to SL_CC6:Break .....	512
6.18.4.5 SL_CC3:Connected state .....	512
6.18.4.5.1 State description .....	512
6.18.4.5.2 Transition SL_CC3:Connected to SL_CC4:DisconnectWait .....	513
6.18.4.5.3 Transition SL_CC3:Connected to SL_CC5:BreakWait .....	513
6.18.4.5.4 Transition SL_CC3:Connected to SL_CC6:Break .....	513
6.18.4.5.5 Transition SL_CC3:Connected to SL_CC7:CloseSTP .....	514
6.18.4.6 SL_CC4:DisconnectWait state .....	514
6.18.4.6.1 State description .....	514
6.18.4.6.2 Transition SL_CC4:DisconnectWait to SL_CC0:Idle .....	514
6.18.4.6.3 Transition SL_CC4:DisconnectWait to SL_CC5:BreakWait .....	514
6.18.4.6.4 Transition SL_CC4:DisconnectWait to SL_CC6:Break .....	514
6.18.4.7 SL_CC5:BreakWait state .....	515
6.18.4.7.1 State description .....	515
6.18.4.7.2 Transition SL_CC5:BreakWait to SL_CC0:Idle .....	515
6.18.4.8 SL_CC6:Break state .....	515
6.18.4.8.1 State description .....	515
6.18.4.8.2 Transition SL_CC6:Break to SL_CC0:Idle .....	516
6.18.4.9 SL_CC7:CloseSTP state .....	516
6.18.4.9.1 State description .....	516
6.18.4.9.2 Transition SL_CC7:CloseSTP to SL_CC0:Idle .....	516
6.18.4.10 SL_CC8:PS_Request state .....	516
6.18.4.10.1 State description .....	516
6.18.4.10.2 Transition SL_CC8:PS_Request to SL_CC9:PS_Quiet .....	517
6.18.4.10.3 Transition SL_CC8:PS_Request to SL_CC0:Idle .....	517
6.18.4.10.4 Transition SL_CC8:PS_Request to SL_CC2:Selected .....	517
6.18.4.10.5 Transition SL_CC8:PS_Request to SL_CC6:Break .....	517
6.18.4.11 SL_CC9:PS_Quiet state .....	517
6.18.4.11.1 State description .....	517
6.18.4.11.2 Transition SL_CC9:PS_Quiet to SL_CC0:Idle .....	518
6.18.4.11.3 Transition SL_CC9:PS_Quiet to SL_CC1:ArbSel .....	518

6.19 XL (link layer for expander logical phys) state machine .....	518
6.19.1 XL state machine overview .....	518
6.19.2 XL transmitter and receiver .....	525
6.19.3 XL0:Idle state .....	527
6.19.3.1 State description .....	527
6.19.3.2 Transition XL0:Idle to XL1:Request_Path .....	528
6.19.3.3 Transition XL0:Idle to XL5:Forward_Open .....	529
6.19.3.4 Transition XL0:Idle to XL11:PS_Request .....	529
6.19.3.5 Transition XL0:Idle to XL12:PS_Quiet .....	530
6.19.4 XL1:Request_Path state .....	530
6.19.4.1 State description .....	530
6.19.4.2 Transition XL1:Request_Path to XL0:Idle .....	531
6.19.4.3 Transition XL1:Request_Path to XL2:Request_Open .....	531
6.19.4.4 Transition XL1:Request_Path to XL5:Forward_Open .....	531
6.19.4.5 Transition XL1:Request_Path to XL9:Break .....	532
6.19.5 XL2:Request_Open state .....	532
6.19.5.1 State description .....	532
6.19.5.2 Transition XL2:Request_Open to XL3:Open_Confirm_Wait .....	532
6.19.6 XL3:Open_Confirm_Wait state .....	533
6.19.6.1 State description .....	533
6.19.6.2 Transition XL3:Open_Confirm_Wait to XL0:Idle .....	533
6.19.6.3 Transition XL3:Open_Confirm_Wait to XL1:Request_Path .....	534
6.19.6.4 Transition XL3:Open_Confirm_Wait to XL5:Forward_Open .....	534
6.19.6.5 Transition XL3:Open_Confirm_Wait to XL7:Connected .....	534
6.19.6.6 Transition XL3:Open_Confirm_Wait to XL9:Break .....	534
6.19.6.7 Transition XL3:Open_Confirm_Wait to XL10:Break_Wait .....	534
6.19.7 XL5:Forward_Open state .....	534
6.19.7.1 State description .....	534
6.19.7.2 Transition XL5:Forward_Open to XL6:Open_Response_Wait .....	535
6.19.8 XL6:Open_Response_Wait state .....	535
6.19.8.1 State description .....	535
6.19.8.2 Transition XL6:Open_Response_Wait to XL0:Idle .....	538
6.19.8.3 Transition XL6:Open_Response_Wait to XL1:Request_Path .....	538
6.19.8.4 Transition XL6:Open_Response_Wait to XL2:Request_Open .....	538
6.19.8.5 Transition XL6:Open_Response_Wait to XL7:Connected .....	538
6.19.8.6 Transition XL6:Open_Response_Wait to XL9:Break .....	538
6.19.8.7 Transition XL6:Open_Response_Wait to XL10:Break_Wait .....	538
6.19.9 XL7:Connected state .....	538
6.19.9.1 State description .....	538
6.19.9.2 Transition XL7:Connected to XL8:Close_Wait .....	540
6.19.9.3 Transition XL7:Connected to XL9:Break .....	540
6.19.9.4 Transition XL7:Connected to XL10:Break_Wait .....	540
6.19.10 XL8:Close_Wait state .....	540
6.19.10.1 State description .....	540
6.19.10.2 Transition XL8:Close_Wait to XL0:Idle .....	543
6.19.10.3 Transition XL8:Close_Wait to XL9:Break .....	543
6.19.10.4 Transition XL8:Close_Wait to XL10:Break_Wait .....	543
6.19.11 XL9:Break state .....	543
6.19.11.1 State description .....	543
6.19.11.2 Transition XL9:Break to XL0:Idle .....	544
6.19.12 XL10:Break_Wait state .....	544
6.19.12.1 State description .....	544
6.19.12.2 Transition XL10:Break_Wait to XL0:Idle .....	544
6.19.13 XL11:PS_Request state .....	544
6.19.13.1 State description .....	544
6.19.13.2 Transition XL11:PS_Request to XL12:PS_Quiet .....	545
6.19.13.3 Transition XL11:PS_Request to XL0:Idle .....	545

6.19.13.4 Transition XL11:PS_Request to XL1:Request_Path.....	545
6.19.13.5 Transition XL11:PS_Request to XL9:Break .....	546
6.19.14 XL12:PS_Quiet state .....	546
6.19.14.1 State description.....	546
6.19.14.2 Transition XL12:PS_Quiet to XL0:Idle.....	546
6.20 SSP link layer .....	546
6.20.1 Opening an SSP connection .....	546
6.20.2 Full duplex.....	547
6.20.3 SSP frame transmission and reception .....	547
6.20.3.1 SSP frame transmission and reception overview.....	547
6.20.3.2 SSP frame transmission and reception while in the SAS dword mode .....	548
6.20.3.3 SSP frame transmission and reception while in SAS packet mode .....	549
6.20.3.4 SSP frame and SMP frame transmission and reception while in SAS packet mode .....	550
6.20.4 SSP flow control.....	551
6.20.5 Extending an SSP connection with persistent connections .....	552
6.20.6 Interlocked frames .....	552
6.20.7 Breaking an SSP connection .....	554
6.20.8 Closing an SSP connection .....	554
6.20.9 SSP (link layer for SSP phys) state machines .....	556
6.20.9.1 SSP state machines overview .....	556
6.20.9.2 SSP transmitter and receiver .....	559
6.20.9.3 SSP_TIM (transmit interlocked frame monitor) state machine.....	561
6.20.9.4 SSP_TCM (transmit frame credit monitor) state machine.....	562
6.20.9.5 SSP_D (DONE control) state machine.....	562
6.20.9.6 SSP_TF (transmit frame control) state machine .....	564
6.20.9.6.1 SSP_TF state machine overview.....	564
6.20.9.6.2 SSP_TF1:Connected_Idle state .....	564
6.20.9.6.2.1 State description .....	564
6.20.9.6.2.2 Transition SSP_TF1:Connected_Idle to SSP_TF2:Tx_Wait.....	564
6.20.9.6.2.3 Transition SSP_TF1:Connected_Idle to SSP_TF4:Transmit_DONE.....	564
6.20.9.6.3 SSP_TF2:Tx_Wait state .....	565
6.20.9.6.3.1 State description .....	565
6.20.9.6.3.2 Transition SSP_TF2:Tx_Wait to SSP_TF3:Transmit_Frame.....	565
6.20.9.6.3.3 Transition SSP_TF2:Tx_Wait to SSP_TF4:Transmit_DONE.....	565
6.20.9.6.4 SSP_TF3:Transmit_Frame state .....	566
6.20.9.6.4.1 State description .....	566
6.20.9.6.4.2 Transition SSP_TF3:Transmit_Frame to SSP_TF1:Connected_Idle.....	566
6.20.9.6.5 SSP_TF4:Transmit_DONE state .....	566
6.20.9.7 SSP_RF (receive frame control) state machine .....	566
6.20.9.8 SSP_RCM (receive frame credit monitor) state machine.....	567
6.20.9.9 SSP_RIM (receive interlocked frame monitor) state machine.....	568
6.20.9.10 SSP_TC (transmit credit control) state machine .....	569
6.20.9.11 SSP_TAN (transmit ACK/NAK control) state machine .....	569
6.20.9.12 SSP_EM (establish and manage persistent connection) state machine.....	569
6.20.9.12.1 SSP_EM state machine overview.....	569
6.20.9.12.2 SSP_EM1:Establish state.....	570
6.20.9.12.3 Transition SSP_EM1:Establish to SSP_EM2:Manage .....	570
6.20.9.12.4 SSP_EM2:Manage .....	570
6.20.9.12.5 Transition SSP_EM2:Manage to SSP_EM1:Establish .....	571
6.21 STP link layer .....	571
6.21.1 STP frame transmission and reception overview.....	571
6.21.2 STP frame transmission and reception while in the SAS dword mode .....	572
6.21.3 STP frame transmission and reception while in the SAS packet mode .....	572
6.21.4 STP flow control.....	574
6.21.4.1 STP flow control overview .....	574
6.21.4.2 SATA frame buffering.....	574
6.21.4.3 STP flow control buffer size.....	575

6.21.4.4 STP flow control example.....	577
6.21.4.5 STP insufficient buffer support .....	579
6.21.5 Continued primitive sequence.....	579
6.21.6 Affiliations.....	580
6.21.7 Opening an STP connection .....	582
6.21.8 Closing an STP connection.....	585
6.21.9 STP connection management examples .....	585
6.21.10 STP (link layer for STP phys) state machines .....	588
6.21.11 SMP target port support.....	588
6.22 SMP link layer.....	588
6.22.1 SMP frame transmission and reception .....	588
6.22.1.1 SMP frame transmission and reception while in SAS dword mode .....	588
6.22.2 SMP frame transmission and reception while in the SAS packet mode .....	589
6.22.3 SMP flow control .....	590
6.22.4 Opening an SMP connection .....	590
6.22.5 Closing an SMP connection.....	590
6.22.6 SMP (link layer for SMP phys) state machines.....	590
6.22.6.1 SMP state machines overview .....	590
6.22.6.2 SMP transmitter and receiver.....	590
6.22.6.3 SMP_IP (link layer for SMP initiator phys) state machine .....	591
6.22.6.3.1 SMP_IP state machine overview .....	591
6.22.6.3.2 SMP_IP1:Idle state .....	592
6.22.6.3.2.1 State description .....	592
6.22.6.3.2.2 Transition SMP_IP1:Idle to SMP_IP2:Transmit_Frame .....	593
6.22.6.3.3 SMP_IP2:Transmit_Frame state .....	593
6.22.6.3.3.1 State description .....	593
6.22.6.3.3.2 Transition SMP_IP2:Transmit_Frame to SMP_IP3:Receive_Frame .....	593
6.22.6.3.4 SMP_IP3:Receive_Frame state .....	593
6.22.6.4 SMP_TP (link layer for SMP target phys) state machine .....	594
6.22.6.4.1 SMP_TP state machine overview .....	594
6.22.6.4.2 SMP_TP1:Receive_Frame state .....	595
6.22.6.4.2.1 State description .....	595
6.22.6.4.2.2 Transition SMP_TP1:Receive_Frame to SMP_TP2:Transmit_Frame .....	595
6.22.6.4.3 SMP_TP2:Transmit_Frame state .....	595
7 Port layer.....	596
7.1 Port layer overview .....	596
7.2 Port layer state machines .....	596
7.2.1 Port layer state machines overview .....	596
7.2.2 PL_OC (port layer overall control) state machine .....	598
7.2.2.1 PL_OC state machine overview .....	598
7.2.2.2 PL_OC1:Idle state .....	600
7.2.2.2.1 PL_OC1:Idle state description .....	600
7.2.2.2.2 Transition PL_OC1:Idle to PL_OC2:Overall_Control.....	601
7.2.2.3 PL_OC2:Overall_Control state.....	601
7.2.2.3.1 PL_OC2:Overall_Control state overview .....	601
7.2.2.3.2 PL_OC2: Non-connection specific confirmations and requests.....	601
7.2.2.3.2.1 PL_OC2: Transmit Frame request .....	601
7.2.2.3.2.2 PL_OC2: HARD_RESET Received confirmation.....	602
7.2.2.3.2.3 PL_OC2: NOTIFY Received (Power Loss Expected) confirmation .....	602
7.2.2.3.2.4 PL_OC2: Phy Disabled confirmation.....	602
7.2.2.3.2.5 PL_OC2: Start I_T Nexus Loss Timer request.....	603
7.2.2.3.3 PL_OC2:Overall_Control state establishing connections .....	603
7.2.2.3.4 PL_OC2:Overall_Control state connection established.....	607
7.2.2.3.5 PL_OC2:Overall_Control state unable to establish a connection — Unable To Connect message.....	607
7.2.2.3.6 PL_OC2:Overall_Control state unable to establish a connection — Unable To	

Connect message - Retry Open message processed as an Unable To Connect message .....	608
7.2.2.3.7 PL_OC2:Overall_Control state unable to establish a connection — I_T Nexus	
Loss timer expires .....	608
7.2.2.3.8 PL_OC2:Overall_Control state - I_T nexus loss event .....	608
7.2.2.3.9 PL_OC2:Overall_Control state connection management .....	608
7.2.2.3.10 PL_OC2:Overall_Control state frame transmission .....	610
7.2.2.3.11 PL_OC2:Overall_Control state frame transmission cancellations .....	611
7.2.2.3.12 Transition PL_OC2:Overall_Control to PL_OC1:Idle .....	611
7.2.3 PL_PM (port layer phy manager) state machine .....	612
7.2.3.1 PL_PM state machine overview .....	612
7.2.3.2 PL_PM1:Idle state .....	615
7.2.3.2.1 PL_PM1:Idle state description .....	615
7.2.3.2.2 Transition PL_PM1:Idle to PL_PM2:Req_Wait .....	615
7.2.3.2.3 Transition PL_PM1:Idle to PL_PM3:Connected .....	615
7.2.3.3 PL_PM2:Req_Wait state .....	615
7.2.3.3.1 PL_PM2:Req_Wait state overview .....	615
7.2.3.3.2 PL_PM2:Req_Wait establishing a connection .....	615
7.2.3.3.3 PL_PM2:Req_Wait connection established .....	616
7.2.3.3.4 PL_PM2:Req_Wait unable to establish a connection .....	616
7.2.3.3.5 PL_PM2:Req_Wait connection management .....	618
7.2.3.3.6 Transition PL_PM2:Req_Wait to PL_PM1:Idle .....	618
7.2.3.3.7 Transition PL_PM2:Req_Wait to PL_PM3:Connected .....	618
7.2.3.3.8 Transition PL_PM2:Req_Wait to PL_PM4:Wait_For_Close .....	618
7.2.3.4 PL_PM3:Connected state .....	618
7.2.3.4.1 PL_PM3:Connected state description .....	618
7.2.3.4.2 Transition PL_PM3:Connected to PL_PM1:Idle .....	622
7.2.3.5 PL_PM4:Wait_For_Close state .....	622
7.2.3.5.1 PL_PM4:Wait_For_Close state description .....	622
7.2.3.5.2 Transition PL_PM4:Wait_For_Close to PL_PM1:Idle .....	623
8 Transport layer .....	624
8.1 Transport layer overview .....	624
8.2 SSP transport layer .....	625
8.2.1 SSP frame format .....	625
8.2.2 Information units .....	629
8.2.2.1 COMMAND frame - Command information unit .....	629
8.2.2.2 TASK frame - Task Management Function information unit .....	630
8.2.2.3 XFER_RDY frame - Transfer Ready information unit .....	633
8.2.2.4 DATA frame - Data information unit .....	634
8.2.2.5 RESPONSE frame - Response information unit .....	636
8.2.2.5.1 RESPONSE frame - Response information unit overview .....	636
8.2.2.5.2 Response information unit - NO_DATA format .....	637
8.2.2.5.3 Response information unit - RESPONSE_DATA format .....	637
8.2.2.5.4 Response information unit - SENSE_DATA format .....	638
8.2.3 Sequences of SSP frames .....	639
8.2.3.1 Sequences of SSP frames overview .....	639
8.2.3.2 Task management function sequence of SSP frames .....	640
8.2.3.3 Non-data command sequence of SSP frames .....	640
8.2.3.4 Write command sequence of SSP frames .....	641
8.2.3.5 Read command sequence of SSP frames .....	641
8.2.3.6 Bidirectional command sequence of SSP frames .....	642
8.2.4 SSP transport layer handling of link layer errors .....	642
8.2.4.1 SSP transport layer handling of link layer errors overview .....	642
8.2.4.2 COMMAND frame - handling of link layer errors .....	643
8.2.4.3 TASK frame - handling of link layer errors .....	644
8.2.4.4 XFER_RDY frame - handling of link layer errors .....	644
8.2.4.4.1 XFER_RDY frame overview .....	644

8.2.4.4.2 XFER_RDY frame with transport layer retries enabled .....	644
8.2.4.4.3 XFER_RDY frame with transport layer retries disabled.....	645
8.2.4.5 Read DATA frame - handling of link layer errors.....	645
8.2.4.5.1 Read DATA frame overview .....	645
8.2.4.5.2 Read DATA frame with transport layer retries enabled .....	645
8.2.4.5.3 Read DATA frame with transport layer retries disabled.....	646
8.2.4.6 Write DATA frame - handling of link layer errors .....	646
8.2.4.6.1 Write DATA frame overview .....	646
8.2.4.6.2 Write DATA frame with transport layer retries enabled .....	646
8.2.4.6.3 Write DATA frame with transport layer retries disabled.....	647
8.2.4.7 RESPONSE frame - handling of link layer errors .....	647
8.2.5 SSP transport layer error handling summary .....	647
8.2.5.1 SSP transport layer error handling summary introduction.....	647
8.2.5.2 SSP initiator port transport layer error handling summary .....	647
8.2.5.3 SSP target port transport layer error handling summary.....	648
8.2.6 ST (transport layer for SSP ports) state machines .....	650
8.2.6.1 ST state machines overview .....	650
8.2.6.2 ST_I (transport layer for SSP initiator ports) state machines .....	650
8.2.6.2.1 ST_I state machines overview .....	650
8.2.6.2.2 ST_IFR (initiator frame router) state machine .....	652
8.2.6.2.2.1 ST_IFR state machine overview .....	652
8.2.6.2.2.2 Processing transport protocol service requests .....	652
8.2.6.2.2.3 Processing Frame Received confirmations.....	653
8.2.6.2.2.4 Processing Transmission Complete and Reception Complete messages.....	654
8.2.6.2.2.5 Processing miscellaneous requests .....	655
8.2.6.2.3 ST_ITS (initiator transport server) state machine .....	656
8.2.6.2.3.1 ST_ITS state machine overview .....	656
8.2.6.2.3.2 ST_ITS1:Initiator_Start state .....	657
8.2.6.2.3.2.1 State description .....	657
8.2.6.2.3.2.2 Transition ST_ITS1:Initiator_Start to ST_ITS3:Prepare_Command.....	657
8.2.6.2.3.2.3 Transition ST_ITS1:Initiator_Start to ST_ITS4:Prepare_Task .....	657
8.2.6.2.3.3 ST_ITS2:Initiator_Send_Frame state.....	657
8.2.6.2.3.3.1 Transition ST_ITS2:Initiator_Send_Frame to ST_ITS1:Initiator_Start .....	661
8.2.6.2.3.3.2 Transition ST_ITS2:Initiator_Send_Frame to ST_ITS5:Prepare_Data_Out.....	662
8.2.6.2.3.3.3 Transition ST_ITS2:Initiator_Send_Frame to ST_ITS6:Receive_Data_In .....	662
8.2.6.2.3.4 ST_ITS3:Prepare_Command state .....	663
8.2.6.2.3.4.1 State description .....	663
8.2.6.2.3.4.2 Transition ST_ITS3:Prepare_Command to ST_ITS2:Initiator_Send_Frame .....	663
8.2.6.2.3.5 ST_ITS4:Prepare_Task state.....	663
8.2.6.2.3.5.1 State description .....	663
8.2.6.2.3.5.2 Transition ST_ITS4:Prepare_Task to ST_ITS2:Initiator_Send_Frame .....	664
8.2.6.2.3.6 ST_ITS5:Prepare_Data_Out state .....	664
8.2.6.2.3.6.1 State description .....	664
8.2.6.2.3.6.2 Transition ST_ITS5:Prepare_Data_Out to ST_ITS2:Initiator_Send_Frame .....	665
8.2.6.2.3.7 ST_ITS6:Receive_Data_In state.....	665
8.2.6.2.3.7.1 State description .....	665
8.2.6.2.3.7.2 Transition ST_ITS6:Receive_Data_In to ST_ITS1:Initiator_Start .....	666
8.2.6.2.3.7.3 Transition ST_ITS6:Receive_Data_In to ST_ITS2:Initiator_Send_Frame .....	666
8.2.6.3 ST_T (transport layer for SSP target ports) state machines .....	667
8.2.6.3.1 ST_T state machines overview.....	667
8.2.6.3.2 ST_TFR (target frame router) state machine.....	669
8.2.6.3.2.1 ST_TFR state machine overview .....	669
8.2.6.3.2.2 Processing Frame Received confirmations.....	669
8.2.6.3.2.3 Processing transport protocol service requests and responses.....	671
8.2.6.3.2.4 Processing miscellaneous requests and confirmations .....	675
8.2.6.3.3 ST_TTS (target transport server) state machine .....	675
8.2.6.3.3.1 ST_TTS state machine overview .....	675

8.2.6.3.3.2 ST_TTS1:Target_Start state .....	676
8.2.6.3.3.2.1 State description .....	676
8.2.6.3.3.2.2 Transition ST_TTS1:Target_Start to ST_TTS3:Prepare_Data_In .....	677
8.2.6.3.3.2.3 Transition ST_TTS1:Target_Start to ST_TTS4:Prepare_Xfer_Rdy .....	677
8.2.6.3.3.2.4 Transition ST_TTS1:Target_Start to ST_TTS5:Receive_Data_Out .....	677
8.2.6.3.3.2.5 Transition ST_TTS1:Target_Start to ST_TTS6:Prepare_Response .....	677
8.2.6.3.3.3 ST_TTS2:Target_Send_Frame state .....	677
8.2.6.3.3.3.1 State description .....	677
8.2.6.3.3.3.2 Transition ST_TTS2:Target_Send_Frame to ST_TTS1:Target_Start .....	681
8.2.6.3.3.3.3 Transition ST_TTS2:Target_Send_Frame to ST_TTS3:Prepare_Data_In .....	681
8.2.6.3.3.3.4 Transition ST_TTS2:Target_Send_Frame to ST_TTS5:Receive_Data_Out .....	681
8.2.6.3.3.4 ST_TTS3:Prepare_Data_In state .....	682
8.2.6.3.3.4.1 State description .....	682
8.2.6.3.3.4.2 Transition ST_TTS3:Prepare_Data_In to ST_TTS2:Target_Send_Frame .....	683
8.2.6.3.3.5 ST_TTS4:Prepare_Xfer_Rdy state .....	683
8.2.6.3.3.5.1 State description .....	683
8.2.6.3.3.5.2 Transition ST_TTS4:Prepare_Xfer_Rdy to ST_TTS2:Target_Send_Frame .....	684
8.2.6.3.3.6 ST_TTS5:Receive_Data_Out state .....	684
8.2.6.3.3.6.1 State description .....	684
8.2.6.3.3.6.2 Transition ST_TTS5:Receive_Data_Out to ST_TTS1:Target_Start .....	685
8.2.6.3.3.6.3 Transition ST_TTS5:Receive_Data_Out to ST_TTS4:Prepare_Xfer_Rdy .....	685
8.2.6.3.3.7 ST_TTS6:Prepare_Response state .....	686
8.2.6.3.3.7.1 State description .....	686
8.2.6.3.3.7.2 Transition ST_TTS6:Prepare_Response to ST_TTS2:Target_Send_Frame .....	687
8.3 STP transport layer .....	687
8.3.1 Initial FIS .....	687
8.3.2 BIST Activate FIS .....	688
8.3.3 TT (transport layer for STP ports) state machines .....	688
8.4 SMP transport layer .....	688
8.4.1 SMP transport layer overview .....	688
8.4.2 SMP_REQUEST frame .....	689
8.4.3 SMP_RESPONSE frame .....	689
8.4.4 Sequence of SMP frames .....	690
8.4.5 MT (transport layer for SMP ports) state machines .....	690
8.4.5.1 SMP transport layer state machines overview .....	690
8.4.5.2 MT_IP (transport layer for SMP initiator ports) state machine .....	690
8.4.5.2.1 MT_IP state machine overview .....	690
8.4.5.2.2 MT_IP1:Idle state .....	691
8.4.5.2.2.1 State description .....	691
8.4.5.2.2.2 Transition MT_IP1:Idle to MT_IP2:Send .....	691
8.4.5.2.3 MT_IP2:Send state .....	692
8.4.5.2.3.1 State description .....	692
8.4.5.2.3.2 Transition MT_IP2:Send to MT_IP1:Idle .....	692
8.4.5.2.3.3 Transition MT_IP2:Send to MT_IP3:Receive .....	692
8.4.5.2.4 MT_IP3:Receive state .....	692
8.4.5.2.4.1 State description .....	692
8.4.5.2.4.2 Transition MT_IP3:Receive to MT_IP1:Idle .....	692
8.4.5.3 MT_TP (transport layer for SMP target ports) state machine .....	693
8.4.5.3.1 MT_TP state machine overview .....	693
8.4.5.3.2 MT_TP1:Idle state .....	693
8.4.5.3.2.1 State description .....	693
8.4.5.3.2.2 Transition MT_TP1:Idle to MT_TP2:Respond .....	694
8.4.5.3.3 MT_TP2:Respond state .....	694
8.4.5.3.3.1 State description .....	694
8.4.5.3.3.2 Transition MT_TP2:Respond to MT_TP1:Idle .....	694
9 Application layer .....	695



9.1 Application layer overview .....	695
9.2 SCSI application layer .....	695
9.2.1 SCSI transport protocol services .....	695
9.2.1.1 SCSI transport protocol services overview .....	695
9.2.1.2 Send SCSI Command SCSI transport protocol service .....	697
9.2.1.3 SCSI Command Received SCSI transport protocol service .....	698
9.2.1.4 Send Command Complete SCSI transport protocol service .....	699
9.2.1.5 Command Complete Received SCSI transport protocol service .....	700
9.2.1.6 Send Data-In SCSI transport protocol service .....	701
9.2.1.7 Data-In Delivered SCSI transport protocol service .....	702
9.2.1.8 Receive Data-Out SCSI transport protocol service .....	702
9.2.1.9 Data-Out Received SCSI transport protocol service .....	703
9.2.1.10 Terminate Data Transfer SCSI transport protocol service request .....	704
9.2.1.11 Data Transfer Terminated SCSI transport protocol service confirmation .....	704
9.2.1.12 Send Task Management Request SCSI transport protocol service .....	705
9.2.1.13 Task Management Request Received SCSI transport protocol service .....	706
9.2.1.14 Task Management Function Executed SCSI transport protocol service .....	706
9.2.1.15 Received Task Management Function Executed SCSI transport protocol service .....	707
9.2.2 SCSI application client error handling .....	708
9.2.3 SCSI device server error handling .....	709
9.2.3.1 SCSI Command Received () error handling .....	709
9.2.3.2 Data-Out Received () error handling .....	710
9.2.4 Task router and task manager error handling .....	710
9.2.5 SCSI transport protocol services for event notifications .....	710
9.2.6 SCSI commands .....	711
9.2.6.1 INQUIRY command .....	711
9.2.6.2 LOG SELECT and LOG SENSE commands .....	711
9.2.6.3 MODE SELECT and MODE SENSE commands .....	711
9.2.6.4 SEND DIAGNOSTIC and RECEIVE DIAGNOSTIC RESULTS commands .....	711
9.2.6.5 START STOP UNIT command .....	711
9.2.7 SCSI mode parameters .....	712
9.2.7.1 SCSI mode parameters overview .....	712
9.2.7.2 Disconnect-Reconnect mode page .....	712
9.2.7.2.1 Disconnect-Reconnect mode page overview .....	712
9.2.7.2.2 BUS INACTIVITY LIMIT field .....	714
9.2.7.2.3 CONNECT TIME LIMIT field .....	714
9.2.7.2.4 MAXIMUM BURST SIZE field .....	714
9.2.7.2.5 FIRST BURST SIZE field .....	715
9.2.7.3 Protocol Specific Logical Unit mode page .....	715
9.2.7.4 Protocol Specific Port mode page .....	716
9.2.7.5 Phy Control And Discover mode page .....	718
9.2.7.6 Shared Port Control mode page .....	722
9.2.7.7 Enhanced Phy Control mode page .....	723
9.2.8 SCSI log parameters .....	726
9.2.8.1 Protocol Specific Port log page .....	726
9.2.8.2 Protocol Specific Port log parameter for SAS target ports .....	728
9.2.9 SCSI diagnostic parameters .....	731
9.2.9.1 SCSI diagnostic parameters overview .....	731
9.2.9.2 Protocol Specific diagnostic page .....	732
9.2.9.3 Enclosure Control diagnostic page .....	738
9.2.9.4 Enclosure Status diagnostic page .....	739
9.2.9.5 Additional Element Status diagnostic page .....	739
9.2.10 SCSI power conditions .....	739
9.2.10.1 SCSI power conditions overview .....	739
9.2.10.2 SA_PC (SCSI application layer power condition) state machine .....	739
9.2.10.2.1 SA_PC state machine overview .....	739
9.2.10.2.2 SA_PC_0:Powered_On state .....	743

9.2.10.2.2.1 State description .....	743
9.2.10.2.2.2 Transition SA_PC_0:Powered_On to SA_PC_4:Active_Wait.....	743
9.2.10.2.2.3 Transition SA_PC_0:Powered_On to SA_PC_8:Stopped .....	743
9.2.10.2.3 SA_PC_1:Active state .....	743
9.2.10.2.3.1 State description .....	743
9.2.10.2.3.2 Transition SA_PC_1:Active to SA_PC_5:Wait_Idle .....	743
9.2.10.2.3.3 Transition SA_PC_1:Active to SA_PC_6:Wait_Standby.....	743
9.2.10.2.3.4 Transition SA_PC_1:Active to SA_PC_10:Wait_Stopped .....	743
9.2.10.2.4 SA_PC_2:Idle state .....	743
9.2.10.2.4.1 State description .....	743
9.2.10.2.4.2 Transition SA_PC_2:Idle to SA_PC_4:Active_Wait .....	743
9.2.10.2.4.3 Transition SA_PC_2:Idle to SA_PC_5:Wait_Idle .....	744
9.2.10.2.4.4 Transition SA_PC_2:Idle to SA_PC_6:Wait_Standby.....	744
9.2.10.2.4.5 Transition SA_PC_2:Idle to SA_PC_7:Idle_Wait .....	744
9.2.10.2.4.6 Transition SA_PC_2:Idle to SA_PC_10:Wait_Stopped .....	744
9.2.10.2.5 SA_PC_3:Standby state .....	744
9.2.10.2.5.1 State description .....	744
9.2.10.2.5.2 Transition SA_PC_3:Standby to SA_PC_4:Active_Wait.....	744
9.2.10.2.5.3 Transition SA_PC_3:Standby to SA_PC_6:Wait_Standby .....	744
9.2.10.2.5.4 Transition SA_PC_3:Standby to SA_PC_7:Idle_Wait.....	744
9.2.10.2.5.5 Transition SA_PC_3:Standby to SA_PC_9: Standby_Wait .....	744
9.2.10.2.5.6 Transition SA_PC_3:Standby to SA_PC_10:Wait_Stopped .....	745
9.2.10.2.6 SA_PC_4:Active_Wait state .....	745
9.2.10.2.6.1 State description .....	745
9.2.10.2.6.2 Transition SA_PC_4:Active_Wait to SA_PC_1:Active .....	747
9.2.10.2.7 SA_PC_5:Wait_Idle state .....	747
9.2.10.2.7.1 SA_PC_5:Wait_Idle state description .....	747
9.2.10.2.7.2 Transition SA_PC_5:Wait_Idle to SA_PC_2:Idle .....	747
9.2.10.2.8 SA_PC_6:Wait_Standby state.....	747
9.2.10.2.8.1 SA_PC_6:Wait_Standby state description.....	747
9.2.10.2.8.2 Transition SA_PC_6:Wait_Standby to SA_PC_3:Standby .....	747
9.2.10.2.9 SA_PC_7:Idle_Wait state .....	747
9.2.10.2.9.1 State description .....	747
9.2.10.2.9.2 Transition SA_PC_7:Idle_Wait to SA_PC_2:Idle .....	749
9.2.10.2.10 SA_PC_8:Stopped state.....	749
9.2.10.2.10.1 State description .....	749
9.2.10.2.10.2 Transition SA_PC_8:Stopped to SA_PC_4:Active_Wait .....	749
9.2.10.2.10.3 Transition SA_PC_8:Stopped to SA_PC_7:Idle_Wait .....	749
9.2.10.2.10.4 Transition SA_PC_8:Stopped to SA_PC_9:Standby_Wait .....	749
9.2.10.2.11 SA_PC_9:Standby_Wait state.....	749
9.2.10.2.11.1 SA_PC_9:Standby_Wait state description.....	749
9.2.10.2.11.2 Transition SA_PC_9:Standby_Wait to SA_PC_3:Standby .....	749
9.2.10.2.12 SA_PC_10:Wait_Stopped state.....	749
9.2.10.2.12.1 SA_PC_10:Wait_Stopped state description.....	749
9.2.10.2.12.2 Transition SA_PC_10:Wait_Stopped to SA_PC_8:Stopped .....	749
9.2.11 SCSI vital product data (VPD) .....	750
9.2.11.1 SCSI vital product data (VPD) overview.....	750
9.2.11.2 Device Identification VPD page.....	750
9.2.11.3 Protocol Specific Logical Unit Information VPD page .....	751
9.2.11.4 Protocol Specific Port Information VPD page.....	753
9.3 ATA application layer.....	756
9.4 Management application layer.....	756
9.4.1 READY LED signal behavior .....	756
9.4.2 Management protocol services .....	758
9.4.3 SMP functions .....	758
9.4.3.1 SMP functions overview .....	758
9.4.3.2 SMP function request frame format.....	761

9.4.3.2.1 SMP function request frame format overview .....	761
9.4.3.2.2 SMP FRAME TYPE field .....	761
9.4.3.2.3 FUNCTION field .....	761
9.4.3.2.4 ALLOCATED RESPONSE LENGTH field .....	761
9.4.3.2.5 REQUEST LENGTH field.....	762
9.4.3.2.6 Additional request bytes .....	762
9.4.3.2.7 CRC field .....	763
9.4.3.3 SMP function response frame format.....	763
9.4.3.3.1 SMP function response frame format overview .....	763
9.4.3.3.2 SMP FRAME TYPE field .....	764
9.4.3.3.3 FUNCTION field .....	764
9.4.3.3.4 FUNCTION RESULT field .....	764
9.4.3.3.5 RESPONSE LENGTH field.....	772
9.4.3.3.6 Additional response bytes.....	772
9.4.3.3.7 CRC field .....	773
9.4.3.4 REPORT GENERAL function.....	773
9.4.3.5 REPORT MANUFACTURER INFORMATION function .....	781
9.4.3.6 REPORT SELF-CONFIGURATION STATUS function .....	784
9.4.3.6.1 REPORT SELF-CONFIGURATION STATUS function overview .....	784
9.4.3.6.2 REPORT SELF-CONFIGURATION STATUS request .....	784
9.4.3.6.3 REPORT SELF-CONFIGURATION STATUS response .....	786
9.4.3.6.4 Self-configuration status descriptor .....	788
9.4.3.7 REPORT ZONE PERMISSION TABLE function.....	790
9.4.3.7.1 REPORT ZONE PERMISSION TABLE function overview .....	790
9.4.3.7.2 REPORT ZONE PERMISSION TABLE request.....	791
9.4.3.7.3 REPORT ZONE PERMISSION TABLE response .....	793
9.4.3.7.4 Zone permission descriptor .....	794
9.4.3.8 REPORT ZONE MANAGER PASSWORD function.....	795
9.4.3.9 REPORT BROADCAST function .....	798
9.4.3.9.1 REPORT BROADCAST function overview.....	798
9.4.3.9.2 REPORT BROADCAST request .....	798
9.4.3.9.3 REPORT BROADCAST response.....	800
9.4.3.9.4 Broadcast descriptor .....	801
9.4.3.10 DISCOVER function .....	803
9.4.3.11 REPORT PHY ERROR LOG function.....	821
9.4.3.12 REPORT PHY SATA function .....	824
9.4.3.13 REPORT ROUTE INFORMATION function .....	828
9.4.3.14 REPORT PHY EVENT function .....	832
9.4.3.14.1 REPORT PHY EVENT function overview.....	832
9.4.3.14.2 REPORT PHY EVENT request .....	833
9.4.3.14.3 REPORT PHY EVENT response.....	834
9.4.3.14.4 Phy event descriptor .....	835
9.4.3.15 DISCOVER LIST function .....	836
9.4.3.15.1 DISCOVER LIST function overview.....	836
9.4.3.15.2 DISCOVER LIST request .....	837
9.4.3.15.3 DISCOVER LIST response.....	840
9.4.3.15.4 DISCOVER LIST response SHORT FORMAT descriptor .....	842
9.4.3.16 REPORT PHY EVENT LIST function.....	843
9.4.3.16.1 REPORT PHY EVENT LIST function overview .....	843
9.4.3.16.2 REPORT PHY EVENT LIST request.....	843
9.4.3.16.3 REPORT PHY EVENT LIST response .....	845
9.4.3.16.4 Phy event list descriptor.....	847
9.4.3.17 REPORT EXPANDER ROUTE TABLE LIST function .....	847
9.4.3.17.1 REPORT EXPANDER ROUTE TABLE LIST function overview.....	847
9.4.3.17.2 REPORT EXPANDER ROUTE TABLE LIST request .....	848
9.4.3.17.3 REPORT EXPANDER ROUTE TABLE LIST response.....	850
9.4.3.17.4 Expander route table descriptor.....	852

9.4.3.18 CONFIGURE GENERAL function .....	852
9.4.3.19 ENABLE DISABLE ZONING function .....	856
9.4.3.20 ZONED BROADCAST function .....	859
9.4.3.21 ZONE LOCK function .....	862
9.4.3.22 ZONE ACTIVATE function .....	865
9.4.3.23 ZONE UNLOCK function .....	866
9.4.3.24 CONFIGURE ZONE MANAGER PASSWORD function .....	868
9.4.3.25 CONFIGURE ZONE PHY INFORMATION function.....	871
9.4.3.25.1 CONFIGURE ZONE PHY INFORMATION function overview .....	871
9.4.3.25.2 CONFIGURE ZONE PHY INFORMATION request.....	872
9.4.3.25.3 Zone phy configuration descriptor .....	874
9.4.3.25.4 CONFIGURE ZONE PHY INFORMATION response .....	874
9.4.3.26 CONFIGURE ZONE PERMISSION TABLE function .....	875
9.4.3.26.1 CONFIGURE ZONE PERMISSION TABLE function overview .....	875
9.4.3.26.2 CONFIGURE ZONE PERMISSION TABLE request .....	876
9.4.3.26.3 Zone permission configuration descriptor .....	878
9.4.3.26.4 CONFIGURE ZONE PERMISSION TABLE response .....	879
9.4.3.27 CONFIGURE ROUTE INFORMATION function .....	880
9.4.3.28 PHY CONTROL function .....	883
9.4.3.29 PHY TEST FUNCTION function.....	892
9.4.3.30 CONFIGURE PHY EVENT function .....	897
9.4.3.30.1 CONFIGURE PHY EVENT function overview .....	897
9.4.3.30.2 CONFIGURE PHY EVENT request.....	898
9.4.3.30.3 Phy event configuration descriptor .....	899
9.4.3.30.4 CONFIGURE PHY EVENT response .....	900
Annex A (normative) Jitter tolerance patterns when SAS dword mode is enabled .....	901
A.1 Jitter tolerance pattern (JTPAT) .....	901
A.2 Compliant jitter tolerance pattern (CJTPAT) .....	901
A.3 Considerations for a phy transmitting JTPAT and CJTPAT .....	909
A.4 Considerations for a phy receiving JTPAT and CJTPAT .....	910
Annex B (informative) SAS to SAS phy reset sequence examples .....	911
Annex C (informative) CRC.....	916
C.1 CRC generator and checker implementation examples .....	916
C.2 CRC implementation in C .....	916
C.3 CRC implementation with XORs .....	918
C.4 CRC examples .....	919
Annex D (informative) Forward error correction encoding while in SAS packet mode .....	921
D.1 Forward error correction encoding overview.....	921
D.2 Forward error correction encoder implementation example .....	921
D.3 Reed Solomon code encoding function block diagram.....	921
D.4 Forward error correction encoder implementation in C.....	922
D.5 Example forward error correction encoder results .....	928
Annex E (informative) SAS address hashing .....	930
E.1 SAS address hashing overview .....	930
E.2 Hash collision probability.....	930
E.3 Hash generation.....	931
E.4 Hash implementation in C .....	931
E.5 Hash implementation with XORs .....	932
E.6 Hash examples .....	933
Annex F (informative) Scrambling.....	937
F.1 Scrambling while in SAS dword mode .....	937

F.1.1 SAS dword mode scrambler implementation example .....	937
F.1.2 SAS dword mode scrambler implementation in C .....	937
F.1.3 SAS dword mode scrambler implementation with XORs .....	938
F.1.4 SAS dword mode scrambler examples .....	939
F.2 Scrambling while in SAS packet mode .....	941
F.2.1 SAS packed mode scrambler implementation example .....	941
F.2.2 SAS packet mode 8-bit pattern generator implementation in C .....	941
F.2.3 SAS packet mode 8-bit pattern generator implementation block diagram .....	942
F.2.4 SAS packet mode 8-bit pattern generator output .....	942
Annex G (informative) ATA architectural notes .....	944
G.1 STP differences from SATA .....	944
G.2 STP differences from SATA .....	944
G.3 Affiliation policies .....	944
G.3.1 Affiliation policies overview .....	944
G.3.2 Affiliation policy for static STP initiator port to STP target port mapping .....	945
G.3.3 Affiliation policy with SATA queued commands and multiple STP initiator ports .....	945
G.3.4 Applicability of affiliation for STP target ports .....	945
G.4 SATA port selector considerations .....	945
G.5 SATA device not transmitting initial Register Device-to-Host FIS .....	946
Annex H (informative) Minimum deletable primitive and scrambled idle segment insertion rate summary ..	947
Annex I (informative) Zone permission configuration descriptor examples .....	950
Annex J (informative) SAS addressing .....	954
J.1 SAS addressing in SAS domains .....	954
J.2 Expander device SAS addresses .....	954
Annex K (informative) Expander device handling of connections .....	955
K.1 Expander device handling of connections overview .....	955
K.2 Connection request - OPEN_ACCEPT .....	957
K.3 Connection request - OPEN_REJECT by end device .....	958
K.4 Connection request - OPEN_REJECT by expander device .....	959
K.5 Connection request - arbitration lost .....	960
K.6 Connection request - backoff and retry .....	961
K.7 Connection request - backoff and reverse path .....	962
K.8 Connection close - single step .....	963
K.9 Connection close - simultaneous .....	964
K.10 BREAK handling during path arbitration when the BREAK_REPLY method is disabled .....	965
K.11 BREAK handling during connection when the BREAK_REPLY method is disabled .....	966
K.12 BREAK handling during path arbitration when the BREAK_REPLY method is enabled .....	967
K.13 BREAK handling during connection when BREAK_REPLY method is enabled .....	968
K.14 STP connection - originated by STP initiator port .....	969
K.15 STP connection - originated by STP target port in an STP SATA bridge .....	970
K.16 STP connection close - originated by STP initiator port .....	971
K.17 STP connection close - originated by STP target port in an STP SATA bridge .....	972
K.18 Connection request - XL1:Request_Path to XL5:Forward_Open transition .....	973
K.19 Pathway blocked and pathway recovery example .....	974
Annex L (informative) Primitive encoding, binary primitive coding, and extended binary primitive coding ...	976
L.1 Primitive encoding .....	976
L.2 Binary primitive coding .....	979
L.2.1 Binary primitive codes overview .....	979
L.2.2 Deletable binary primitives .....	979
L.2.3 Binary primitives for use outside SAS logical link connections .....	982
L.2.4 Binary primitives for use inside SAS logical link connections .....	983

L.2.5 Binary primitives for use inside and outside SAS logical link connections.....	985
L.2.6 Unassigned binary primitives .....	987
L.3 Extended binary primitive coding.....	990
Annex M (informative) Standards bodies contact information.....	993
Annex N (informative) Successful low phy power condition handshake sequence .....	994
Annex O (informative) Terminology mapping to SPL-3.....	997
Bibliography .....	998

## Tables

	Page
Table 1 – Numbering conventions .....	35
Table 2 – Comparison of decimal prefixes and binary prefixes .....	36
Table 3 – Constraint and note notation .....	37
Table 4 – Class diagram notation for classes .....	38
Table 5 – Multiplicity notation .....	39
Table 6 – Class diagram notation for associations .....	39
Table 7 – Class diagram notation for aggregations .....	40
Table 8 – Class diagram notation for generalizations .....	41
Table 9 – Class diagram notation for dependency .....	41
Table 10 – Object diagram notation for objects .....	42
Table 11 – Object diagram notation for link .....	42
Table 12 – Data dword containing a value .....	46
Table 13 – Data dword containing four one-byte fields .....	47
Table 14 – Logical links .....	54
Table 15 – Broadcast types .....	71
Table 16 – Names and identifiers .....	73
Table 17 – SCSI architecture model object attribute mapping .....	73
Table 18 – NAA IEEE Registered format .....	74
Table 19 – NAA Locally Assigned format .....	75
Table 20 – Hashed SAS address code parameters .....	76
Table 21 – Device name created from the IDENTIFY DEVICE world wide name .....	77
Table 22 – Expander logical phy to ECM requests .....	104
Table 23 – Expander logical phy to ECM responses .....	104
Table 24 – ECM to expander logical phy confirmations .....	105
Table 25 – Expander logical phy to ECR to expander logical phy requests and indications .....	106
Table 26 – Expander logical phy to ECR to expander logical phy responses and confirmations .....	107
Table 27 – Expander logical phy to BPP requests .....	108
Table 28 – BPP to expander logical phy indications .....	109
Table 29 – Routing attributes and routing methods .....	109
Table 30 – Expander route table types .....	111
Table 31 – Expander route table levels for externally configurable expander device R phy A .....	124
Table 32 – Expander route table levels for externally configurable expander device N .....	125
Table 33 – Expander route entries for externally configurable expander device E0 phy 1 .....	127
Table 34 – Expander route entries for externally configurable expander device F phy 0 .....	128
Table 35 – Zone manager password .....	132
Table 36 – Zone phy information .....	135
Table 37 – Zone phy information usage .....	136
Table 38 – Zone groups .....	138
Table 39 – Zone permission table .....	139
Table 40 – Zone permission table granting minimal permissions .....	140
Table 41 – Source zone group determination .....	142
Table 42 – Destination zone group determination .....	142
Table 43 – REQUESTED INSIDE ZPSDS bit and INSIDE ZPSDS PERSISTENT bit changes after a link reset sequence .....	143
Table 44 – ZONE GROUP field values if the ZONE GROUP PERSISTENT bit is set to one .....	144
Table 45 – Conditions that cause the ZONE GROUP field to be updated if the ZONE GROUP PERSISTENT bit is set to zero .....	145
Table 46 – PHY EVENT SOURCE field .....	154
Table 47 – Bit designations .....	161
Table 48 – Conversion from byte notation to character name example .....	161
Table 49 – Data characters .....	163
Table 50 – Control characters .....	169
Table 51 – Control character usage .....	170
Table 52 – Delayed code violation example .....	171
Table 53 – SPL packet .....	176

Table 54 – SPL PACKET HEADER field.....	176
Table 55 – SPL packet payload that contains a scrambled idle segment.....	177
Table 56 – Primitive parameter location within primitive segment .....	178
Table 57 – Primitive segment primitive data character placement.....	178
Table 58 – Primitive segment SPL packet payload containing primitives and binary primitives .....	178
Table 59 – Primitive segment SPL packet payload that contains primitives, binary primitives, and a 1-dword primitive parameter in second dword.....	179
Table 60 – Primitive segment SPL packet payload that contains primitives, binary primitives, and a 1-dword primitive parameter in fourth dword .....	179
Table 61 – Primitive segment SPL packet payload that contains primitives, binary primitives, and a 2-dword primitive parameter .....	180
Table 62 – Primitive segment SPL packet payload that contains a primitive or binary primitive, and 3-dword primitive parameter .....	180
Table 63 – Primitive segment SPL packet payload that contains an extended binary primitives .....	180
Table 64 – PRIMITIVE SYNCHRONIZE SELECT field .....	181
Table 65 – CONTROL1 field, CONTROL2 field, and CONTROL3 field .....	181
Table 66 – SPL packet payload that contains an SPL frame segment.....	184
Table 67 – SPL packet payload that contains an idle dword segment.....	185
Table 68 – Reed Solomon code notation and definitions.....	186
Table 69 – Ordering of parity and message symbols transmitted .....	189
Table 70 – SNW-3 phy capabilities .....	196
Table 71 – Requested logical link rate .....	197
Table 72 – Multiplexing negotiation.....	198
Table 73 – Supported settings bit priorities .....	200
Table 74 – Example SNW-3 phy capabilities values.....	201
Table 75 – Train_Tx-SNW TTIU bit.....	204
Table 76 – Train_Tx-SNW TTIU .....	205
Table 77 – PATTERN TYPE field.....	205
Table 78 – Control/Status TTIU .....	206
Table 79 – COEFFICIENT SETTINGS field .....	206
Table 80 – COEFFICIENT 1 REQUEST field, COEFFICIENT 2 REQUEST field, and COEFFICIENT 3 REQUEST field ..	207
Table 81 – Valid coefficient requests .....	208
Table 82 – COEFFICIENT 1 STATUS field, COEFFICIENT 2 STATUS field, and COEFFICIENT 3 STATUS field .....	209
Table 83 – Error Response TTIU .....	209
Table 84 – ERROR CODE field .....	211
Table 85 – Phy reset sequence timing specifications .....	212
Table 86 – SATA speed negotiation sequence timing specifications.....	214
Table 87 – SAS speed negotiation sequence timing specifications .....	219
Table 88 – SNW rates used in SNW-1, SNW-2, and Final-SNW.....	222
Table 89 – SNW-3 phy capabilities bit .....	223
Table 90 – Transmitter training pattern .....	225
Table 91 – Receiver training patterns while in SAS dword mode .....	230
Table 92 – SP state machine timers .....	264
Table 93 – SP state machine variables.....	265
Table 94 – Messages to SP transmitter and SP receiver at start of RCDT .....	278
Table 95 – SP_DWS state machine timers.....	302
Table 96 – SP_PS state machine timers .....	309
Table 97 – Mapping messages to the Training Control word.....	329
Table 98 – Mapping messages from PTT_SC1 state machine, PTT_SC2 state machine, and PTT_SC3 state machine to the Training Status word.....	329
Table 99 – Mapping Transmit Error Response message arguments to Error Response TTIU fields .....	330
Table 100 – Transmit Error Response message arguments sent to PTT_T state machine .....	336
Table 101 – Mapping the Training Status word to SP receiver messages .....	337
Table 102 – Mapping the Training Status word to PTT_GC1 state machine messages, PTT_GC2 state machine messages, and PTT_GC3 state machine messages .....	338
Table 103 – Mapping the Training Control word to PTT_SC1 state machine messages, PTT_SC2 state machine messages, and PTT_SC3 state machine messages .....	339



Table 104 – Mapping Coefficient Request byte to PTT_SC3 state machine message, PTT_SC2 state machine message, and PTT_SC1 state machine messages .....	340
Table 105 – Mapping messages to the PTT_T state machine.....	344
Table 106 – PTT_SC1 messages to substitute for PTT_SC2 messages .....	345
Table 107 – PTT_SC1 messages to substitute for PTT_SC3 messages .....	346
Table 108 – Mapping messages to the PTT_T state machine.....	348
Table 109 – PTT_GC1 messages to substitute for PTT_GC2 messages .....	349
Table 110 – PTT_GC1 messages to substitute for PTT_GC2 messages .....	349
Table 111 – SP transmitter binary primitive messages.....	355
Table 112 – SP receiver binary primitive messages.....	358
Table 113 – Mapping messages to the PAPTA_TC state machine .....	366
Table 114 – PAPTA_TC state machine timers .....	367
Table 115 – PAPTA_TC1 messages to substitute for PAPTA_TC2 messages.....	370
Table 116 – PAPTA_TC1 messages to substitute for PAPTA_TC3 messages.....	370
Table 117 – PAPTA_TC1 messages to substitute for PAPTA_TC1_2 messages.....	370
Table 118 – PAPTA_TC1 messages to substitute for PAPTA_TC2_3 messages.....	371
Table 119 – Primitive format .....	374
Table 120 – Deletable primitives.....	375
Table 121 – Primitives not specific to type of connection .....	376
Table 122 – Primitives used inside SSP and SMP connections .....	380
Table 123 – Primitives used inside STP connections and on SATA physical links.....	382
Table 124 – Primitive encoding for deletable primitives.....	384
Table 125 – Primitive encoding for primitives not specific to type of connection .....	385
Table 126 – Primitive encoding for primitives used only inside SSP and SMP connections .....	387
Table 127 – Primitive encoding for primitives used only inside STP connections and only on SATA physical links.....	388
Table 128 – Primitive sequences .....	389
Table 129 – ALIGN primitives .....	397
Table 130 – MUX primitives .....	398
Table 131 – NOTIFY primitives.....	399
Table 132 – AIP primitives .....	401
Table 133 – BROADCAST primitives.....	402
Table 134 – CLOSE primitives.....	402
Table 135 – CLOSE primitive parameter format.....	403
Table 136 – Abandon-class OPEN_REJECT primitives .....	405
Table 137 – Retry-class OPEN_REJECT primitives.....	406
Table 138 – OPEN_REJECT retry-class primitive parameter format .....	407
Table 139 – TIME SCALE field .....	408
Table 140 – PS_REQ primitives .....	408
Table 141 – DONE primitives.....	410
Table 142 – NAK primitives.....	411
Table 143 – RRDY primitives.....	411
Table 144 – Deletable binary primitives .....	413
Table 145 – Binary primitives used inside SSP connections and STP connections .....	415
Table 146 – Binary primitive codes for deletable binary primitives .....	416
Table 147 – Binary primitive codes for binary primitives only used inside SSP and STP connections.....	417
Table 148 – Binary primitive sequences .....	417
Table 149 – APTA_ADJUST binary primitives.....	418
Table 150 – APTA_COEFFICIENT_1 binary primitives.....	419
Table 151 – APTA_COEFFICIENT_2 binary primitives.....	419
Table 152 – APTA_COEFFICIENT_3 binary primitives.....	420
Table 153 – APTA_COEFFICIENT_1_2 binary primitives.....	420
Table 154 – APTA_COEFFICIENT_2_3 binary primitives.....	421
Table 155 – B_EOF binary primitives .....	422
Table 156 – Deletable extended binary primitives .....	423
Table 157 – Extended binary primitives not specific to type of connection.....	424
Table 158 – Extended binary primitive codes for deletable extended binary primitives.....	425

Table 159 – Extended binary primitive codes for extended binary primitives not specific to any type of connection.....	425
Table 160 – Extended binary primitive sequences .....	426
Table 161 – Physical link rate tolerance management deletable primitive insertion requirement.....	430
Table 162 – Physical link rate tolerance management for deletable extended binary primitive insertion requirement .....	431
Table 163 – CRC notation and definitions .....	433
Table 164 – Scrambling for different data dword types while in the SAS dword mode.....	438
Table 165 – Address frame format.....	446
Table 166 – ADDRESS FRAME TYPE field .....	446
Table 167 – IDENTIFY address frame format.....	447
Table 168 – SAS DEVICE TYPE field.....	448
Table 169 – REASON field .....	448
Table 170 – POWER CAPABLE field .....	450
Table 171 – OPEN address frame format.....	451
Table 172 – SAS PROTOCOL field.....	452
Table 173 – FEATURES field .....	452
Table 174 – CONNECTION RATE field.....	452
Table 175 – ARBITRATION WAIT TIME field .....	454
Table 176 – SL_IR_IRC state machine timers.....	458
Table 177 – PS_ACK pattern.....	465
Table 178 – SL_P_S state machine timers.....	467
Table 179 – SL_P_C state machine timers.....	472
Table 180 – Connection results of a connection request.....	480
Table 181 – Arbitration priority for OPEN address frames passing on a logical link .....	482
Table 182 – Arbitration priority for a Request Path request in the ECM .....	485
Table 183 – Pathway recovery priority .....	487
Table 184 – Results of aborting a connection request.....	488
Table 185 – Results of closing a connection.....	491
Table 186 – Results of breaking a connection .....	492
Table 187 – Rate matching deletable primitive insertion requirements while in the SAS dword mode .....	493
Table 188 – Rate matching deletable primitive insertion requirements while in the SAS packet mode.....	495
Table 189 – SL_CC state machine timers .....	504
Table 190 – SL_CC state machine variables.....	504
Table 191 – OPEN_REJECT Received message to Open Failed confirmation mapping .....	508
Table 192 – XL state machine timers.....	519
Table 193 – XL state machine variable .....	520
Table 194 – Extended fairness priority.....	541
Table 195 – Setting CLOSE primitive parameters and Delay Expander Forward Open Indication timer .....	542
Table 196 – SSP frame interlock requirements .....	552
Table 197 – SSP state machines timers .....	556
Table 198 – SSP state machines timers for persistent connections .....	556
Table 199 – STP link layer differences from SATA link layer during an STP connection .....	571
Table 200 – Affiliation policies.....	581
Table 201 – Affiliation context relative identifier example .....	582
Table 202 – PL_OC state machine timers .....	599
Table 203 – Confirmations from Unable To Connect messages .....	607
Table 204 – PL_PM state machine timers .....	612
Table 205 – Messages from Open Failed confirmations.....	617
Table 206 – SSP frame format.....	625
Table 207 – FRAME TYPE field .....	626
Table 208 – TLR CONTROL field for COMMAND frames .....	627
Table 209 – COMMAND frame - Command information unit.....	629
Table 210 – TASK ATTRIBUTE field .....	630
Table 211 – TASK frame - Task Management Function information unit .....	631
Table 212 – TASK MANAGEMENT FUNCTION field.....	632
Table 213 – XFER_RDY frame - Transfer Ready information unit .....	634

Table 214 – DATA frame - Data information unit .....	635
Table 215 – RESPONSE frame - Response information unit .....	636
Table 216 – DATAPRES field .....	637
Table 217 – RESPONSE DATA field .....	638
Table 218 – RESPONSE CODE field .....	638
Table 219 – Sequences of SSP frames .....	639
Table 220 – Confirmations sent to the SCSI application layer if a frame transmission error or reception error occurs .....	655
Table 221 – ST_ITS state machine variables .....	656
Table 222 – ST_ITS state machine arguments .....	657
Table 223 – Messages sent to the ST_IFR state machine .....	659
Table 224 – Transmission Complete messages for XFER_RDY frame verification failures .....	661
Table 225 – Reception Complete messages for read DATA frame verification failures .....	665
Table 226 – ST_T state machine timers .....	667
Table 227 – Task Management Function Executed Service Response argument mapping to Request (Send Transport Response) Service Response argument .....	673
Table 228 – Confirmations sent to the SCSI application layer .....	674
Table 229 – ST_TTS state machine variables .....	676
Table 230 – ST_TTS state machine arguments .....	676
Table 231 – Messages sent to the ST_TFR state machine .....	680
Table 232 – Additional messages sent to the ST_TFR state machine .....	681
Table 233 – Reception Complete message for write DATA frame verification failures .....	684
Table 234 – Request (Send Transport Response) message Service Response argument to RESPONSE frame RESPONSE DATA field mapping .....	687
Table 235 – SMP frame format .....	688
Table 236 – SMP FRAME TYPE field .....	688
Table 237 – SMP_REQUEST frame format .....	689
Table 238 – SMP_RESPONSE frame format .....	689
Table 239 – MT_TP time limits .....	693
Table 240 – Execute Command procedure call transport protocol services .....	696
Table 241 – Task management function procedure call transport protocol services .....	697
Table 242 – Send SCSI Command SCSI transport protocol service arguments .....	698
Table 243 – SCSI Command Received SCSI transport protocol service arguments .....	699
Table 244 – Send Command Complete SCSI transport protocol service arguments .....	700
Table 245 – Command Complete Received SCSI transport protocol service arguments .....	701
Table 246 – Send Data-In SCSI transport protocol service arguments .....	702
Table 247 – Data-In Delivered SCSI transport protocol service arguments .....	702
Table 248 – Receive Data-Out SCSI transport protocol service arguments .....	703
Table 249 – Data-Out Received SCSI transport protocol service arguments .....	704
Table 250 – Terminate Data Transfer SCSI transport protocol service arguments .....	704
Table 251 – Data Transfer Terminated SCSI transport protocol service arguments .....	705
Table 252 – Send Task Management Request SCSI transport protocol service arguments .....	705
Table 253 – Task Management Request Received SCSI transport protocol service arguments .....	706
Table 254 – Task Management Function Executed SCSI transport protocol service arguments .....	707
Table 255 – Received Task Management Function Executed SCSI transport protocol service arguments .....	708
Table 256 – Delivery Result to additional sense code mapping .....	710
Table 257 – SCSI transport protocol events .....	711
Table 258 – SSP target port mode pages .....	712
Table 259 – Disconnect-Reconnect mode page for SAS SSP .....	713
Table 260 – Protocol Specific Logical Unit mode page for SAS SSP .....	716
Table 261 – Protocol Specific Port mode page for SAS SSP .....	717
Table 262 – I_T NEXUS LOSS TIME field .....	718
Table 263 – Phy Control And Discover mode page .....	719
Table 264 – SAS phy mode descriptor .....	721
Table 265 – Shared Port Control mode page .....	722
Table 266 – Enhanced Phy Control mode page .....	724
Table 267 – Enhanced phy control mode descriptor .....	725

Table 268 – Protocol Specific Port log parameters .....	726
Table 269 – Protocol Specific Port log page for SAS SSP .....	727
Table 270 – Protocol Specific Port log parameter for SAS target ports .....	728
Table 271 – SAS phy log descriptor.....	729
Table 272 – SSP target port diagnostic pages.....	731
Table 273 – Diagnostic pages affected by zoning .....	732
Table 274 – Protocol Specific diagnostic page for SAS SSP.....	733
Table 275 – PHY TEST FUNCTION field .....	734
Table 276 – PHY TEST PATTERN field .....	735
Table 277 – PHY TEST FUNCTION SSC field.....	736
Table 278 – PHY TEST FUNCTION PHYSICAL LINK RATE field .....	736
Table 279 – PHY TEST PATTERN DWORDS CONTROL field.....	737
Table 280 – TWO_DWORDS phy test pattern examples .....	738
Table 281 – Summary of states in the SA_PC state machine .....	740
Table 282 – VPD pages with special requirements for SAS SSP .....	750
Table 283 – Device Identification VPD page designation descriptors for the SAS target port.....	750
Table 284 – Device Identification VPD page designation descriptors for the SAS target device.....	751
Table 285 – Protocol Specific Logical Unit Information VPD page for SAS SSP.....	752
Table 286 – Logical unit information descriptor for SAS SSP .....	753
Table 287 – Protocol Specific Port Information VPD page for SAS SSP .....	754
Table 288 – Port information descriptor for SAS SSP .....	755
Table 289 – SAS phy information descriptor for SAS SSP .....	756
Table 290 – READY LED signal behavior.....	757
Table 291 – SMP functions (FUNCTION field) .....	758
Table 292 – SMP request frame format.....	761
Table 293 – SMP response frame format .....	763
Table 294 – FUNCTION RESULT field .....	764
Table 295 – Function result priority.....	768
Table 296 – REPORT GENERAL request.....	773
Table 297 – REPORT GENERAL response .....	774
Table 298 – NUMBER OF ZONE GROUPS field.....	778
Table 299 – REPORT MANUFACTURER INFORMATION request.....	781
Table 300 – REPORT MANUFACTURER INFORMATION response .....	782
Table 301 – REPORT SELF-CONFIGURATION STATUS request.....	784
Table 302 – REPORT SELF-CONFIGURATION STATUS response .....	786
Table 303 – Self-configuration status descriptor .....	788
Table 304 – STATUS TYPE field .....	788
Table 305 – REPORT ZONE PERMISSION TABLE request .....	791
Table 306 – REPORT TYPE field .....	792
Table 307 – REPORT ZONE PERMISSION TABLE response .....	793
Table 308 – Zone permission descriptors .....	794
Table 309 – Zone permission descriptor for a source zone group (i.e., s) with 128 zone groups.....	794
Table 310 – Zone permission descriptor for a source zone group (i.e., s) with 256 zone groups.....	795
Table 311 – Zone permission descriptor bit requirements .....	795
Table 312 – REPORT ZONE MANAGER PASSWORD request .....	796
Table 313 – REPORT TYPE field .....	797
Table 314 – REPORT ZONE MANAGER PASSWORD response .....	797
Table 315 – REPORT BROADCAST request.....	798
Table 316 – REPORT BROADCAST response .....	800
Table 317 – Broadcast descriptor .....	801
Table 318 – BROADCAST REASON field for originated Broadcasts.....	802
Table 319 – DISCOVER request .....	803
Table 320 – DISCOVER response.....	804
Table 321 – ATTACHED SAS DEVICE TYPE field .....	807
Table 322 – NEGOTIATED LOGICAL LINK RATE field and NEGOTIATED PHYSICAL LINK RATE field.....	809
Table 323 – NEGOTIATED PHYSICAL LINK RATE field and NEGOTIATED LOGICAL LINK RATE field combinations based on multiplexing .....	810

Table 324 – ATTACHED SATA PORT SELECTOR and ATTACHED SATA DEVICE bits .....	811
Table 325 – PROGRAMMED MINIMUM PHYSICAL LINK RATE field and PROGRAMMED MAXIMUM PHYSICAL LINK RATE field .....	814
Table 326 – The HARDWARE MINIMUM PHYSICAL LINK RATE field and the HARDWARE MAXIMUM PHYSICAL LINK RATE field .....	814
Table 327 – ROUTING ATTRIBUTE field .....	815
Table 328 – PHY POWER CONDITION field.....	816
Table 329 – SAS POWER CAPABLE field.....	816
Table 330 – PWR_DIS SIGNAL field.....	817
Table 331 – PWR_DIS CONTROL CAPABLE field.....	817
Table 332 – ATTACHED DEVICE NAME field .....	818
Table 333 – SELF-CONFIGURATION STATUS field.....	819
Table 334 – SELF-CONFIGURATION LEVELS COMPLETED field .....	819
Table 335 – REPORT PHY ERROR LOG request .....	822
Table 336 – REPORT PHY ERROR LOG response .....	823
Table 337 – REPORT PHY SATA request .....	825
Table 338 – REPORT PHY SATA response.....	826
Table 339 – REPORT ROUTE INFORMATION request .....	829
Table 340 – REPORT ROUTE INFORMATION response.....	831
Table 341 – REPORT PHY EVENT request.....	833
Table 342 – REPORT PHY EVENT response .....	834
Table 343 – Phy event descriptor .....	835
Table 344 – DISCOVER LIST request.....	837
Table 345 – PHY FILTER field.....	838
Table 346 – DESCRIPTOR TYPE field .....	839
Table 347 – DISCOVER LIST response .....	840
Table 348 – SHORT FORMAT descriptor.....	842
Table 349 – REPORT PHY EVENT LIST request .....	843
Table 350 – REPORT PHY EVENT LIST response .....	845
Table 351 – Phy event list descriptor .....	847
Table 352 – REPORT EXPANDER ROUTE TABLE LIST request.....	848
Table 353 – REPORT EXPANDER ROUTE TABLE LIST response .....	850
Table 354 – Expander route table descriptor .....	852
Table 355 – CONFIGURE GENERAL request .....	853
Table 356 – STP SMP I_T NEXUS LOSS TIME field.....	855
Table 357 – CONFIGURE GENERAL response.....	856
Table 358 – ENABLE DISABLE ZONING request.....	857
Table 359 – SAVE field .....	858
Table 360 – ENABLE DISABLE ZONING field .....	858
Table 361 – ENABLE DISABLE ZONING response .....	859
Table 362 – ZONED BROADCAST request .....	860
Table 363 – BROADCAST TYPE field .....	861
Table 364 – ZONED BROADCAST response .....	862
Table 365 – ZONE LOCK request .....	863
Table 366 – ZONE LOCK response.....	864
Table 367 – ZONE ACTIVATE request.....	865
Table 368 – ZONE ACTIVATE response.....	866
Table 369 – ZONE UNLOCK request .....	867
Table 370 – ZONE UNLOCK response .....	868
Table 371 – CONFIGURE ZONE MANAGER PASSWORD request .....	869
Table 372 – SAVE field .....	870
Table 373 – CONFIGURE ZONE MANAGER PASSWORD response.....	870
Table 374 – CONFIGURE ZONE PHY INFORMATION request .....	872
Table 375 – SAVE field .....	873
Table 376 – Zone phy configuration descriptor .....	874
Table 377 – CONFIGURE ZONE PHY INFORMATION response .....	874
Table 378 – CONFIGURE ZONE PERMISSION TABLE request.....	876

Table 379 – NUMBER OF ZONE GROUPS field .....	877
Table 380 – SAVE field .....	877
Table 381 – Zone permission configuration descriptors .....	878
Table 382 – Zone permission configuration descriptor for source zone group for 128 zone groups .....	878
Table 383 – Zone permission configuration descriptor for source zone group for 256 zone groups .....	878
Table 384 – Zone permission configuration descriptor bit requirements .....	879
Table 385 – CONFIGURE ZONE PERMISSION TABLE response .....	879
Table 386 – CONFIGURE ROUTE INFORMATION request .....	881
Table 387 – CONFIGURE ROUTE INFORMATION response .....	882
Table 388 – PHY CONTROL request .....	884
Table 389 – PHY OPERATION field .....	886
Table 390 – PROGRAMMED MINIMUM PHYSICAL LINK RATE field and PROGRAMMED MAXIMUM PHYSICAL LINK RATE field .....	888
Table 391 – ENABLE SAS SLUMBER field .....	889
Table 392 – ENABLE SAS PARTIAL field .....	889
Table 393 – ENABLE SATA SLUMBER field .....	890
Table 394 – ENABLE SATA PARTIAL field .....	890
Table 395 – PWR_DIS CONTROL field .....	891
Table 396 – PHY CONTROL response .....	891
Table 397 – PHY TEST FUNCTION request .....	893
Table 398 – PHY TEST FUNCTION field .....	895
Table 399 – PHY TEST FUNCTION PHYSICAL LINK RATE field .....	896
Table 400 – PHY TEST FUNCTION response .....	896
Table 401 – CONFIGURE PHY EVENT request .....	898
Table 402 – Phy event configuration descriptor .....	899
Table 403 – CONFIGURE PHY EVENT response .....	900
Table A.1 – JTPAT for RD+ and RD- .....	901
Table A.2 – CJTPAT .....	903
Table A.3 – CJTPAT with fixed content .....	909
Table C.1 – CRC examples while SAS dword mode is enabled .....	919
Table C.2 – CRC examples while SAS packet mode is enabled .....	920
Table D.1 – Example forward error correction coding results .....	928
Table E.1 – Monte-Carlo simulation results .....	931
Table E.2 – Hash results for simple SAS addresses .....	933
Table E.3 – Hash results for realistic SAS addresses .....	933
Table E.4 – Hash results for a walking ones pattern .....	934
Table E.5 – Hash results for a walking zeros pattern .....	935
Table F.1 – SAS dword mode scrambler examples .....	939
Table F.2 – Initial SAS dword mode scrambler output .....	940
Table F.3 – 8-bit pattern generator values produced after initialization of scrambler by PACKET_SYNC ...	943
Table H.1 – Minimum deletable primitive insertion rate examples while in the SAS dword mode .....	948
Table H.2 – Minimum insertion rate examples while in the SAS packet mode .....	949
Table I.1 – Zone permission table example initial value .....	950
Table I.2 – CONFIGURE ZONE PERMISSION TABLE request example .....	951
Table I.3 – Zone permission table after processing first zone permission configuration descriptor .....	952
Table I.4 – Zone permission table after processing second zone permission configuration descriptor .....	953
Table K.1 – Column descriptions for connection examples .....	956
Table L.1 – Primitives with Hamming distance of at least 7 .....	976
Table L.2 – Primitives without Hamming distance of 7 .....	979
Table L.3 – Deletable binary primitives .....	979
Table L.4 – Binary primitives used outside SAS logical link connections .....	982
Table L.5 – Binary primitives used inside SAS logical link connections .....	983
Table L.6 – Binary primitives used inside and outside SAS logical link connections .....	985
Table L.7 – Unassigned binary primitives .....	987
Table L.8 – Extended binary primitives .....	990
Table M.1 – Standards bodies .....	993
Table O.1 – Terminology name mapping to SPL-3 .....	997

Table O.2 – Field name mapping to SPL-3..... 997

## Figures

	Page
Figure 0 – Organization of this standard .....	liv
Figure 1 – SCSI document relationships .....	1
Figure 2 – ATA document relationships .....	2
Figure 3 – Examples of association relationships in class diagrams .....	40
Figure 4 – Examples of aggregation relationships in class diagrams .....	40
Figure 5 – Example of generalization relationships in class diagrams .....	41
Figure 6 – Example of a dependency relationship in class diagrams .....	42
Figure 7 – Examples of link relationships for object diagrams .....	43
Figure 8 – State machine conventions .....	44
Figure 9 – SAS Domain class diagram .....	49
Figure 10 – Phy class diagram .....	51
Figure 11 – SAS phy object diagram .....	52
Figure 12 – Expander phy object diagram .....	53
Figure 13 – Ports (narrow ports and wide ports) .....	55
Figure 14 – Port class diagram .....	56
Figure 15 – Port object diagram .....	57
Figure 16 – SAS devices .....	59
Figure 17 – Expander device .....	60
Figure 18 – Domains .....	61
Figure 19 – SAS domain bridging to ATA domains .....	62
Figure 20 – SAS domains bridging to ATA domains with SATA port selectors .....	63
Figure 21 – Devices spanning SAS domains .....	64
Figure 22 – Single expander device topology example .....	65
Figure 23 – Multiple expander device topologies and routing methods .....	66
Figure 24 – Potential pathways .....	67
Figure 25 – Multiple connections on wide ports .....	69
Figure 26 – State machines for SAS devices .....	79
Figure 27 – State machines for expander devices .....	80
Figure 28 – Transmit data path in a SAS phy .....	81
Figure 29 – SSP link, port, SSP transport, and SCSI application layer state machines .....	82
Figure 30 – SMP link, port, SMP transport, and management application layer state machines .....	83
Figure 31 – STP link, port, STP transport, and ATA application layer state machines .....	84
Figure 32 – Transmit data path and state machines in an expander phy .....	85
Figure 33 – Receive data path in a SAS phy while in the SAS dword mode .....	87
Figure 34 – Receive data path in an expander phy while in the SAS dword mode .....	88
Figure 35 – Receive data path in a SAS phy while in the SAS packet mode .....	90
Figure 36 – Receive data path in an expander phy while in the SAS packet mode .....	91
Figure 37 – State machines and SAS Device classes .....	92
Figure 38 – State machines and Expander Device classes .....	93
Figure 39 – Reset terminology .....	95
Figure 40 – Expander device model .....	99
Figure 41 – Expander device interfaces .....	102
Figure 42 – Expander device interface detail .....	103
Figure 43 – Phy-based expander route table .....	111
Figure 44 – Expander-based expander route table .....	112
Figure 45 – Level-order traversal example .....	115
Figure 46 – Examples of invalid topologies .....	119
Figure 47 – Externally configurable expander device and table-to-table attachment .....	122
Figure 48 – Expander route index levels example .....	123
Figure 49 – Expander route index order example .....	126
Figure 50 – Zoning example .....	129
Figure 51 – One ZPSDS example .....	129
Figure 52 – Zone manager location examples .....	130
Figure 53 – Three ZPSDSes example .....	131
Figure 54 – Extending a ZPSDS example .....	133



Figure 55 – Overtaking a ZPSDS example .....	134
Figure 56 – Zoning expander route table .....	141
Figure 57 – SAS bit transmission logic .....	172
Figure 58 – SAS bit reception logic .....	173
Figure 59 – SPL packet formats with error correction information .....	175
Figure 60 – Examples of primitive segment alignment .....	183
Figure 61 – Forward error correction encoding and transmission .....	188
Figure 62 – Forward error correction reception and decoding .....	191
Figure 63 – OOB signal transmission .....	193
Figure 64 – OOB signal detection .....	195
Figure 65 – TTIU transmitter BMC encoding .....	202
Figure 66 – TTIU bit cell transmitter encoding .....	203
Figure 67 – TTIU bit cell receiver decoding .....	204
Figure 68 – SATA OOB sequence .....	213
Figure 69 – SATA speed negotiation sequence .....	213
Figure 70 – SAS to SATA OOB sequence .....	215
Figure 71 – SAS to SAS OOB sequence .....	217
Figure 72 – SNW-1, SNW-2, and Final-SNW .....	221
Figure 73 – SNW-3 .....	223
Figure 74 – Train_Tx-SNW while in the SAS dword mode .....	224
Figure 75 – Train_Tx-SNW while in the SAS packet mode .....	225
Figure 76 – Pattern marker transmission while in the SAS dword mode .....	226
Figure 77 – Pattern marker transmission while in the SAS packet mode .....	227
Figure 78 – Valid pattern marker detection while in SAS dword mode .....	228
Figure 79 – Valid pattern marker detection while in SAS packet mode .....	229
Figure 80 – Train_Rx-SNW while in SAS dword mode .....	230
Figure 81 – Train_Rx-SNW while in SAS packet mode .....	232
Figure 82 – SAS speed negotiation sequence SNW flowchart .....	234
Figure 83 – SAS speed negotiation sequence (both phys SNW-1 through Train_Rx-SNW with no Train_Tx-SNW) .....	236
Figure 84 – SAS speed negotiation sequence (both phys SNW-1 through Train_Rx-SNW with Train_Tx-SNW) .....	237
Figure 85 – SAS speed negotiation sequence (phy A: SNW-1 through SNW-3, phy B: SNW-2 only) .....	238
Figure 86 – SAS speed negotiation sequence (phy A: SNW-3 only, phy B: SNW-1 only) .....	239
Figure 87 – SAS speed negotiation sequence - phy reset problem in Final-SNW .....	240
Figure 88 – SAS speed negotiation sequence - phy reset problem in SNW-3 .....	241
Figure 89 – SAS speed negotiation sequence - phy reset problem in Train_Rx-SNW .....	242
Figure 90 – SAS speed negotiation sequence - multiple Train_Rx-SNWs .....	243
Figure 91 – Local phy achieves pattern lock before the attached phy achieves pattern lock .....	245
Figure 92 – Local phy achieves pattern lock after the attached phy achieves pattern lock .....	246
Figure 93 – Attached receiver handshake sequence (requesting two increments to coefficient 1) .....	248
Figure 94 – Attached receiver handshake sequence (requesting one decrement and one increment to coefficient 1) .....	249
Figure 95 – Handshake sequence to set local phy's receiver coefficients to no_equalization values (attached phy) .....	251
Figure 96 – Handshake sequence to set local phy's receiver coefficients to no_equalization values (local phy) .....	252
Figure 97 – Local phy's receiver indicates completion of training before the attached phy's receiver completes training .....	253
Figure 98 – Attached phy's receiver indicates completion of training before the local phy's receiver completes training .....	254
Figure 99 – Processing an invalid TTIU .....	256
Figure 100 – Hot-plug and the phy reset sequence .....	258
Figure 101 – SP transmitter adjustment procedure .....	259
Figure 102 – Transition to active phy power condition .....	260
Figure 103 – Hot plug and low phy power condition .....	261
Figure 104 – SP (phy layer) state machine - OOB sequence states .....	269

Figure 105 – SP (phy layer) state machine - SAS speed negotiation states .....	276
Figure 106 – SP (phy layer) state machine - SAS speed negotiation states for SNW-3 and Train_Rx-SNW and Train_Tx-SNW .....	277
Figure 107 – SP (phy layer) state machine - SAS phy power condition states .....	290
Figure 108 – SP (phy layer) state machine - SATA host emulation states .....	294
Figure 109 – SP (phy layer) state machine – SATA port selector state .....	299
Figure 110 – SP (phy layer) state machine - SATA spinup hold state .....	300
Figure 111 – SP_DWS (phy layer dword synchronization) state machine .....	302
Figure 112 – SP_PS (phy layer SPL packet synchronization) state machine .....	310
Figure 113 – SPL packet payload primitive decoding .....	312
Figure 114 – Unpacking SPL packet .....	313
Figure 115 – SP_ReSync (phy layer resynchronization) state machine .....	318
Figure 116 – PTT_T (phy layer transmitter training transmit pattern) state machine .....	326
Figure 117 – PTT_R (phy layer transmitter training receive pattern) state machine .....	334
Figure 118 – PTT_SC1, PTT_SC2, and PTT_SC3 (phy layer transmitter training set transmitter coefficient) state machines .....	342
Figure 119 – PTT_GC1, PTT_GC2, and PTT_GC3 (phy layer transmitter training get transmitter coefficient) state machines .....	347
Figure 120 – PTT_PL (phy layer transmitter training pattern lock) state machine .....	351
Figure 121 – PAPTA_A_L (phy layer attached SP receiver adjusts the local SP transmitter coefficients) state machine .....	361
Figure 122 – PAPTA_L_A (phy layer local SP receiver adjusts the attached SP transmitter coefficients) state machine .....	364
Figure 123 – PAPTA_TC (phy layer SP receiver management of attached SP transmitter coefficient adjustments) state machines .....	368
Figure 124 – Multiplexing disabled .....	371
Figure 125 – Multiplexing enabled .....	372
Figure 126 – Transmitting a repeated primitive sequence .....	390
Figure 127 – Receiving a repeated primitive sequence .....	390
Figure 128 – Extended primitive sequences while in the SAS dword mode .....	391
Figure 129 – Extended primitive sequences while in the SAS packet mode .....	392
Figure 130 – Triple primitive sequences while in the SAS dword mode .....	393
Figure 131 – Triple primitive sequences while in the SAS packet mode .....	394
Figure 132 – Redundant primitive sequences while in the SAS dword mode .....	395
Figure 133 – Redundant primitive sequences while in the SAS packet mode .....	396
Figure 134 – Elasticity buffer with phys in the SAS dword mode .....	428
Figure 135 – Elasticity buffer with phys in the SAS packet mode .....	429
Figure 136 – Address frame, SSP frame, and SMP frame CRC bit ordering .....	435
Figure 137 – STP frame CRC bit ordering .....	436
Figure 138 – Transmit path bit ordering while in the SAS dword mode .....	440
Figure 139 – Receive path bit ordering while in the SAS dword mode .....	441
Figure 140 – STP transmit path bit ordering .....	442
Figure 141 – STP receive path bit ordering .....	443
Figure 142 – Transmit path bit ordering while in the SAS packet mode .....	444
Figure 143 – Receive path bit ordering while in the SAS packet mode .....	445
Figure 144 – Address frame transmission .....	445
Figure 145 – Identification sequence .....	456
Figure 146 – Hard reset sequence .....	457
Figure 147 – SL_IR (link layer identification and hard reset) state machines .....	459
Figure 148 – Transitioning from the active phy power condition to a low phy power condition .....	465
Figure 149 – SL_P_S (link layer power source device) state machine .....	468
Figure 150 – SL_P_C (link layer power consumer device) state machine .....	473
Figure 151 – Example simultaneous connection recommendations for wide ports .....	479
Figure 152 – Aborting a connection request with a BREAK primitive sequence .....	489
Figure 153 – Connection request timeout example .....	490
Figure 154 – Closing a connection example .....	491
Figure 155 – Rate matching example while in the SAS dword mode .....	494

Figure 156 – Rate matching example while in the SAS packet mode .....	496
Figure 157 – SL (link layer for SAS logical phys) state machines (1 of 3) .....	498
Figure 158 – SL (link layer for SAS logical phys) state machines (2 of 3) .....	499
Figure 159 – SL (link layer for SAS logical phys) state machines (3 of 3) .....	500
Figure 160 – XL (link layer for expander logical phys) state machine (1 of 4) .....	521
Figure 161 – XL (link layer for expander logical phys) state machine (2 of 4) .....	522
Figure 162 – XL (link layer for expander logical phys) state machine (3 of 4) .....	523
Figure 163 – XL (link layer for expander logical phys) state machine (4 of 4) .....	524
Figure 164 – SSP frame transmission .....	548
Figure 165 – SSP frame transmission with no pad dword .....	549
Figure 166 – SSP frame transmission with one pad dword .....	549
Figure 167 – SSP frame transmission with two pad dwords .....	549
Figure 168 – SSP frame transmission with three pad dwords .....	549
Figure 169 – Unaligned SOF at start of SSP frame example .....	550
Figure 170 – Unaligned B_EOF (2) at end of SSP frame example .....	551
Figure 171 – Interlocked frames .....	553
Figure 172 – Non-interlocked frames with the same initiator port transfer tags .....	553
Figure 173 – Non-interlocked frames with different initiator port transfer tags .....	554
Figure 174 – Closing an SSP connection example .....	555
Figure 175 – SSP (link layer for SSP phys) state machines (1 of 3 - frame transmission) .....	557
Figure 176 – SSP (link layer for SSP phys) state machines (2 of 3 - frame reception) .....	558
Figure 177 – SSP (link layer for SSP phys) state machines (3 of 3 - persistent connection) .....	559
Figure 178 – STP frame transmission .....	572
Figure 179 – STP frame transmission with no pad dword .....	573
Figure 180 – STP frame transmission with one pad dword .....	573
Figure 181 – STP frame transmission with two pad dwords .....	573
Figure 182 – STP frame transmission with three pad dwords .....	573
Figure 183 – STP flow control .....	578
Figure 184 – Transmitting a continued primitive sequence while in the SAS dword mode .....	580
Figure 185 – Receiving a continued primitive sequence while in the SAS dword mode .....	580
Figure 186 – Example simultaneous connection recommendations for an expander device .....	584
Figure 187 – STP initiator port opening an STP connection while SAS dword mode is enabled .....	586
Figure 188 – STP target port opening an STP connection while SAS dword mode is enabled .....	587
Figure 189 – SMP frame transmission while in the SAS dword mode .....	588
Figure 190 – SMP frame transmission with no pad dword .....	589
Figure 191 – SMP frame transmission with one pad dword .....	589
Figure 192 – SMP frame transmission with two pad dword .....	589
Figure 193 – SMP frame transmission with three pad dword .....	589
Figure 194 – SMP_IP (link layer for SMP initiator phys) state machine .....	592
Figure 195 – SMP_TP (link layer for SMP target phys) state machine .....	594
Figure 196 – Port layer examples .....	597
Figure 197 – PL_OC (port layer overall control) state machine .....	600
Figure 198 – PL_PM (port layer phy manager) state machine (1 of 2) .....	613
Figure 199 – PL_PM (port layer phy manager) state machine (2 of 2) .....	614
Figure 200 – Task management function sequence of SSP frames .....	640
Figure 201 – Non-data command sequence of SSP frames .....	640
Figure 202 – Write command sequence of SSP frames .....	641
Figure 203 – Read command sequence of SSP frames .....	641
Figure 204 – Bidirectional command sequence of SSP frames .....	642
Figure 205 – ST_I (transport layer for SSP initiator ports) state machines .....	651
Figure 206 – ST_T (transport layer for SSP target ports) state machines .....	668
Figure 207 – Sequence of SMP frames .....	690
Figure 208 – MT_IP (transport layer for SMP initiator ports) state machine .....	691
Figure 209 – MT_TP (transport layer for SMP target ports) state machine .....	693
Figure 210 – SA_PC (SCSI application layer power condition) state machine for SAS .....	742
Figure A.1 – CJTPAT pre-scrambling .....	902
Figure B.1 – SAS speed negotiation sequence (phy A: SNW-1 only, phy B: SNW-1 only) .....	911

Figure B.2 – SAS speed negotiation sequence (phy A: SNW-1, SNW-2, phy B: SNW-1, SNW-2) .....	912
Figure B.3 – SAS speed negotiation sequence (phy A: SNW-1, SNW-2, and SNW-3, phy B: SNW-1 and SNW-2) .....	913
Figure B.4 – SAS speed negotiation sequence (phy A: SNW-2, SNW-3, phy B: SNW-1, SNW-2) .....	914
Figure B.5 – SAS speed negotiation sequence (phy A: SNW-1 only, phy B: SNW-2 only) .....	915
Figure C.1 – CRC generator example .....	916
Figure C.2 – CRC checker example .....	916
Figure D.1 – SAS packet mode forward error correction encoder .....	921
Figure D.2 – Reed Solomon linear feedback shift register implementation .....	922
Figure E.1 – BCH(69, 39, 9) code generator .....	931
Figure F.1 – SAS dword mode Scrambler .....	937
Figure F.2 – SAS packet mode scrambler .....	941
Figure F.3 – SAS packet mode 8-bit pattern generator .....	942
Figure K.1 – Example topology .....	955
Figure K.2 – Connection request - OPEN_ACCEPT .....	957
Figure K.3 – Connection request - OPEN_REJECT by end device .....	958
Figure K.4 – Connection request - OPEN_REJECT by expander device .....	959
Figure K.5 – Connection request - arbitration lost .....	960
Figure K.6 – Connection request - backoff and retry .....	961
Figure K.7 – Connection request - backoff and reverse path .....	962
Figure K.8 – Connection close - single step .....	963
Figure K.9 – Connection close - simultaneous .....	964
Figure K.10 – BREAK handling during path arbitration when the BREAK_REPLY method is disabled .....	965
Figure K.11 – BREAK handling during a connection when the BREAK_REPLY method is disabled .....	966
Figure K.12 – BREAK handling during path arbitration when the BREAK_REPLY method is enabled .....	967
Figure K.13 – BREAK handling during a connection when the BREAK_REPLY method is enabled .....	968
Figure K.14 – STP connection - originated by STP initiator port .....	969
Figure K.15 – STP connection - originated by STP target port in an STP SATA bridge .....	970
Figure K.16 – STP connection close - originated by STP initiator port .....	971
Figure K.17 – STP connection close - originated by STP target port in an STP SATA bridge .....	972
Figure K.18 – XL1:Request_Path to XL5:Forward_Open transition .....	973
Figure K.19 – Partial pathway recovery .....	974
Figure N.1 – Example of a requesting SAS device's sequencing of a successful request for entering a partial phy power condition .....	995
Figure N.2 – Example of a SAS device's sequencing for a successful request to enter a partial phy power condition .....	996

## Foreword

This foreword is not part of this standard.

This standard defines the three transport protocols that use the SAS interconnect (see SAS-4):

- a) Serial SCSI Protocol (SSP): a mapping of SCSI supporting multiple initiators and multiple targets;
- b) Serial ATA Tunneled Protocol (STP): a mapping of Serial ATA expanded to support multiple initiators and multiple targets; and
- c) Serial Management Protocol (SMP): a management protocol.

Requests for interpretation, suggestions for improvement and addenda, or defect reports are welcome. They should be sent to the INCITS Secretariat, International Committee for Information Technology Standards, Information Technology Industry Council, Suite 610, 1101 K Street, NW, Washington, DC 20005-7031.

This standard was processed and approved for submittal to ANSI by the International Committee for Information Technology Standards (INCITS). Committee approval of the standard does not necessarily imply that all committee members voted for approval. At the time it approved this standard, INCITS had the following members:

AIM Global Inc	IEEE
Adobe Systems Inc	Hewlett Packard Inc.
American National Standards Institute	Hewlett Packard Enterprise
Apple	IBM Corporation
Dell Inc.	Intel Corporation
Department of Commerce - NIST	Microsoft Corporation
Distributed Management Task Force (DMTF)	Oracle
Farance Inc.	Purdue University
Futurewei Technologies Inc	Telecommunications Industry Association (TIA)
GS1GO	United States Dept of Homeland Security

INCITS Technical Committee T10 on SCSI Storage Interfaces, which developed and reviewed this standard, had the following members:

Ralph O. Weber, Chair  
William Martin, Vice-Chair  
Curtis Stevens, Secretary

<i>Organization Represented</i>	<i>Name of Representative</i>
Amphenol Corporation .....	Gregory McSorley
	Brad Brubaker (Alt)
	David Chan (Alt)
	Paul Coddington (Alt)
	Zhineng Fan (Alt)
	Adrian Green (Alt)
	Donald Harper (Alt)
	Yifan Huang (Alt)
	Martin Li (Alt)
	Chris Lyon (Alt)
	Alex Persaud (Alt)
	Chansy Phommachanh (Alt)
	Michael Scholeno (Alt)
	Michael Wingard (Alt)
	CN Wong (Alt)
	Matt Wright (Alt)
Broadcom Limited .....	Brad Besmer
	Patrick Bashford (Alt)
	Srikiran Dravida (Alt)
	Jeffrey Gauvin (Alt)
	Rick Kutcipal (Alt)
	Bernhard Laschinsky (Alt)
	Mohammad Mobin (Alt)
	Robert Sheffield (Alt)
	James Smart (Alt)
	Jason Stuhlsatz (Alt)
	Pat Thaler (Alt)
	Bill Voorhees (Alt)
Brocade .....	David Peterson
	Scott Kipp (Alt)
	Steven Wilson (Alt)
Dell Inc. ....	David Black
	Mark Bokhan (Alt)
	Erin Bournival (Alt)
	George Ericson (Alt)
	Mickey Felton (Alt)
	Christopher Goonan (Alt)
	Gary Kotzur (Alt)
	Bill Lynn (Alt)
	Kevin Marks (Alt)
	Ash McCarty (Alt)
	Daniel Oelke (Alt)
	Marlon Ramroopsingh (Alt)
ENDL Texas .....	Ralph Weber

Foxconn Electronics.....	Fred Fons Gary Hsieh (Alt) Glenn Moore (Alt) Mike Shu (Alt) Miller Zhao (Alt)
Fujitsu America Inc.....	Kun Katsumata Osamu Kimura (Alt) Mark Malcolm (Alt) Gene Owens (Alt)
Hewlett Packard Enterprise.....	Curtis Ballard Wayne Bellamy (Alt) Chris Cheng (Alt) Rob Elliott (Alt) Joe Foster (Alt) Barry Olawsky (Alt) Neil Wanamaker (Alt) Han Wang (Alt) Jeff Wolford (Alt)
IBM Corporation .....	Kevin Butt Mike Osborne (Alt)
Intel Corporation.....	Chunfei Ye
Marvell Semiconductor Inc.....	Paul Wassenberg Wei Liu (Alt) Wei Zhou (Alt)
Micron Technology Inc .....	Carl Mies Jerry Barkley (Alt) Andrew Dunn (Alt) Roy Feng (Alt) Neal Galbo (Alt) Michael George (Alt) Alan Haffner (Alt) Daniel Hubbard (Alt) Sebastien Jean (Alt) Michael Selzler (Alt)
Microsemi.....	Tim Symons Sanjay Goyal (Alt) Vincent Hache (Alt) David Hong (Alt) Adnan Jiwani (Alt) Keith Shaw (Alt) Ariel Sibley (Alt) Gregory Tabor (Alt) Jeremiah Tussey (Alt) Rod Zavari (Alt)
Molex Inc.....	Jay Neer Alex Haser (Alt) Ed Poh (Alt) Michael Rost (Alt) Darian Schulz (Alt) Scott Sommers (Alt)
NetApp Inc .....	Frederick Knight Chris Fore (Alt) Jaimon George (Alt)

Oracle.....	Dennis Appleyard Jon Allen (Alt) Seth Goldberg (Alt) Hyon Kim (Alt) Martin Petersen (Alt) Phi Tran (Alt) Lee Wan-Hui (Alt)
QLogic.....	Craig Carlson
Quantum Corporation.....	Darryl Torske
Samsung Semiconductor Inc (SSI).....	William Martin Judy Brock (Alt) HeeChang Cho (Alt) KeunSoo Jo (Alt) Sung Lee (Alt) Bhavith M.P. (Alt) Truong Nguyen (Alt) Aishwarya Ravichandran (Alt)
Seagate Technology.....	Gerald Houlder Alvin Cox (Alt) Ian Davies (Alt) Neil Edmunds (Alt) Timothy Feldman (Alt) John Fleming (Alt) Jim Hatfield (Alt) Tony Kilwein (Alt) Parag Maharana (Alt) Alan Westbury (Alt) Judy Westby (Alt)
TE Connectivity.....	Dan Gorenc Tom Grzysiewicz (Alt) Kyle Klinger (Alt) Jeffery Mason (Alt) Joel Meyers (Alt) Andy Nowak (Alt) Eric Powell (Alt) Yasuo Sasaki (Alt)
Toshiba America Electronic Components Inc.....	Tom Friend Mark Carlson (Alt) Kambiz Esmaily (Alt) Don Harwood (Alt) Johanna Hernandez (Alt) Patrick Hery (Alt) Yuji Katori (Alt) Tom McGoldrick (Alt) James Welch (Alt) Scott Wright (Alt)
VMware Inc .....	Murali Rajagopal Deepak Babarjung (Alt) Patrick Dirks (Alt) Neil H. MacLean (Alt) Mike Panas (Alt) Ahmad Tawil (Alt)



Western Digital Corporation ..... Curtis Stevens  
 Joe Breher (Alt)  
 David Brewer (Alt)  
 Jorge Campello (Alt)  
 Frank Chu (Alt)  
 Marvin DeForest (Alt)  
 Kirill Dimitrov (Alt)  
 Jason Gao (Alt)  
 Michael Koffman (Alt)  
 Dave Landsman (Alt)  
 Larry McMillan (Alt)  
 Chet Mercado (Alt)  
 Nadesan Narenthiran (Alt)  
 Nathan Obr (Alt)  
 Christopher Reed (Alt)  
 Avraham Shimor (Alt)  
 Yoni Shternhell (Alt)

## Introduction

This standard defines the protocol layer of the Serial Attached SCSI (SAS) interconnect and the three transport protocols that use the SAS interconnect:

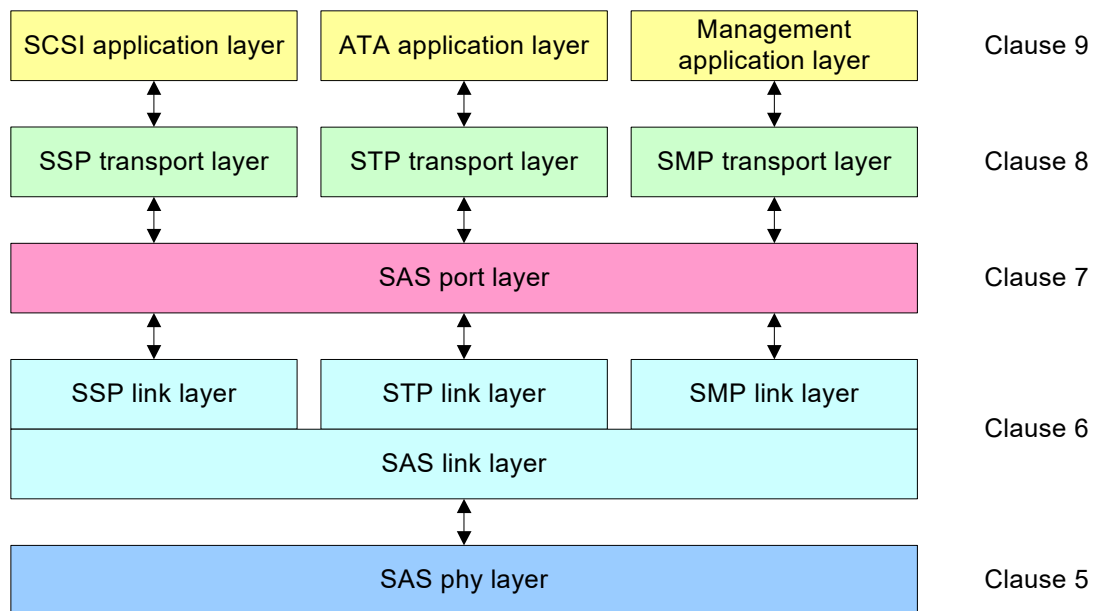
- a) Serial SCSI Protocol (SSP): a mapping of SCSI supporting multiple initiators and multiple targets;
- b) Serial ATA Tunneled Protocol (STP): a mapping of Serial ATA expanded to support multiple initiators and multiple targets; and
- c) Serial Management Protocol (SMP): a management protocol.

The standard is organized as follows:

- Clause 1 (Scope) describes the relationship of this standard to the SCSI and ATA families of standards.
- Clause 2 (Normative references) provides references to other standards and documents.
- Clause 3 (Terms, definitions, symbols, abbreviations, keywords, and conventions) defines terms and conventions used throughout this standard.
- Clause 4 (General) describes architecture, names and identifiers, state machines, resets, I\_T nexus loss, provides an expander device model, the discover process, the configuration subprocess, zoning, phy power conditions, phy test functions, and phy events.
- Clause 5 (Phy layer) describes the phy layer. It describes 8b10b encoding, 128b150b coding, bit order, out of band (OOB) signals, phy reset sequences, phy layer state machines, multiplexing, character encoding, character decoding, dwords, primitives, BMC coding, phy power conditions, and spinup.
- Clause 6 (Link layer) describes the link layer. It describes primitives, physical link rate tolerance management, idle physical links, CRC, scrambling, address frames, power control, the link reset sequence and its state machine, low phy power condition, SAS domain changes, connections, rate matching, link layer for SAS logical phys state machines and link layer for expander logical phys state machines, and SSP, STP, and SMP connection rules and link layer state machines.
- Clause 7 (Port layer) describes the port layer, which sits between one or more link layers and one or more transport layers. It includes port layer state machines.
- Clause 8 (Transport layer) describes the transport layer. It includes SSP, STP, and SMP frame definitions and transport layer state machines.
- Clause 9 (Application layer) describes the application layer. It describes SCSI transport protocol services, mode parameters, log parameters, diagnostic parameters, power conditions, error handling, and vital product data. It describes ATA application layer rules. It describes management application layer rules including READY LED signal behavior and SMP functions.

- Normative Annex A (Jitter tolerance patterns when SAS dword mode is enabled) provides information on methods the SAS protocol uses to control generation of JTPAT and CJTPAT.
- Informative Annex B (SAS to SAS phy reset sequence examples) provides additional phy reset sequence examples.
- Informative Annex C (CRC) provides information and example implementations of the CRC algorithm.
- Informative Annex D (Forward error correction encoding while in SAS packet mode) provides information and example implementations of the forward error correction encoding generated by a Reed Solomon code encoding function.
- Informative Annex E (SAS address hashing) provides information and example implementations of the hashing algorithm.
- Informative Annex F (Scrambling) provides information and example implementations of the scrambling algorithm.
- Informative Annex G (ATA architectural notes) describes ATA architectural differences from Serial ATA and Serial ATA II.
- Informative Annex H (Minimum deletable primitive and scrambled idle segment insertion rate summary) describes the minimum ALIGN and/or NOTIFY insertion rates for physical link rate tolerance management and rate matching.
- Informative Annex I (Zone permission configuration descriptor examples) provides examples of using multiple zone permission configuration descriptors in the SMP CONFIGURE ZONE PERMISSION TABLE function.
- Informative Annex J (SAS addressing) provides information on SAS addressing in SAS domains and expander device SAS addressing.
- Informative Annex K (Expander device handling of connections) describes expander device behavior in a variety of connection examples.
- Informative Annex L (Primitive encoding, binary primitive coding, and extended binary primitive coding) lists the primitive encodings available for future versions of this standard.
- Informative Annex M (Standards bodies contact information) lists the standards bodies contact information.
- Informative Annex N (Successful low phy power condition handshake sequence) contains an example of the sequencing required between attached phys to successfully enter into a partial phy power condition.
- Informative Annex O (Terminology mapping to SPL-3) lists the terminology mapping between this standard and SPL-3.
- Informative Bibliography lists a bibliography for this standard.

Figure 0 shows the organization of the layers of this standard.



**Figure 0 – Organization of this standard**

American National Standard  
for Information Technology -

SAS Protocol Layer - 4 (SPL-4)

1 Scope

The SCSI family of standards provides for many different transport protocols that define the rules for exchanging information between different SCSI devices. This standard defines the rules for exchanging information between SCSI devices using a serial interconnect. Other SCSI transport protocol standards define the rules for exchanging information between SCSI devices using other interconnects.

Figure 1 shows the relationship of this standard to the other standards and related projects in the SCSI family of standards.

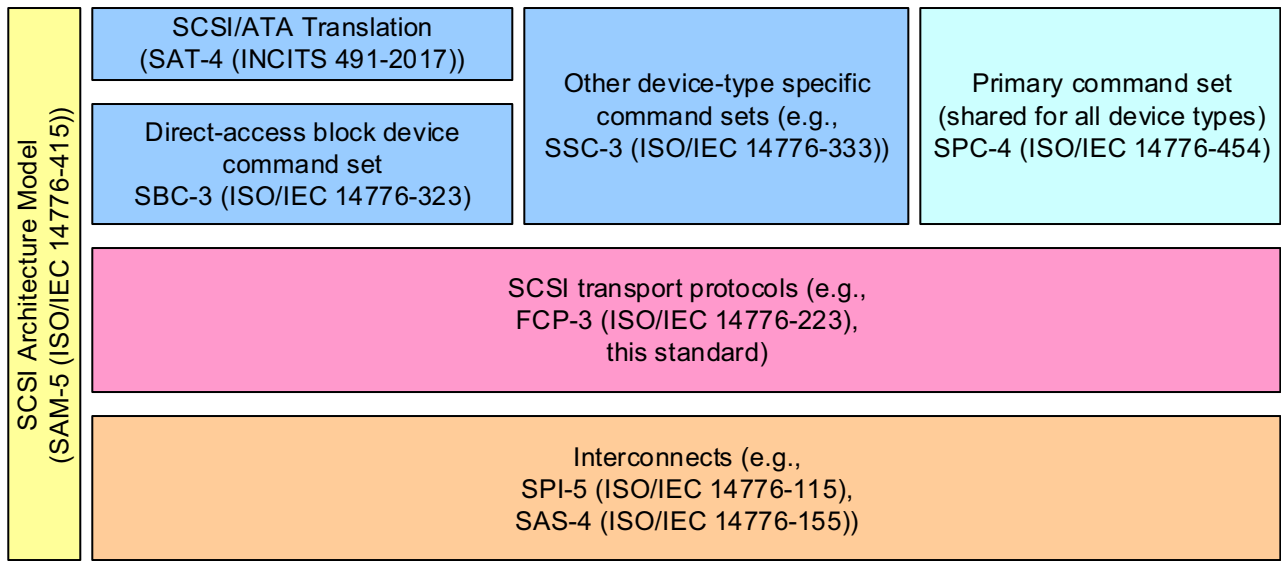
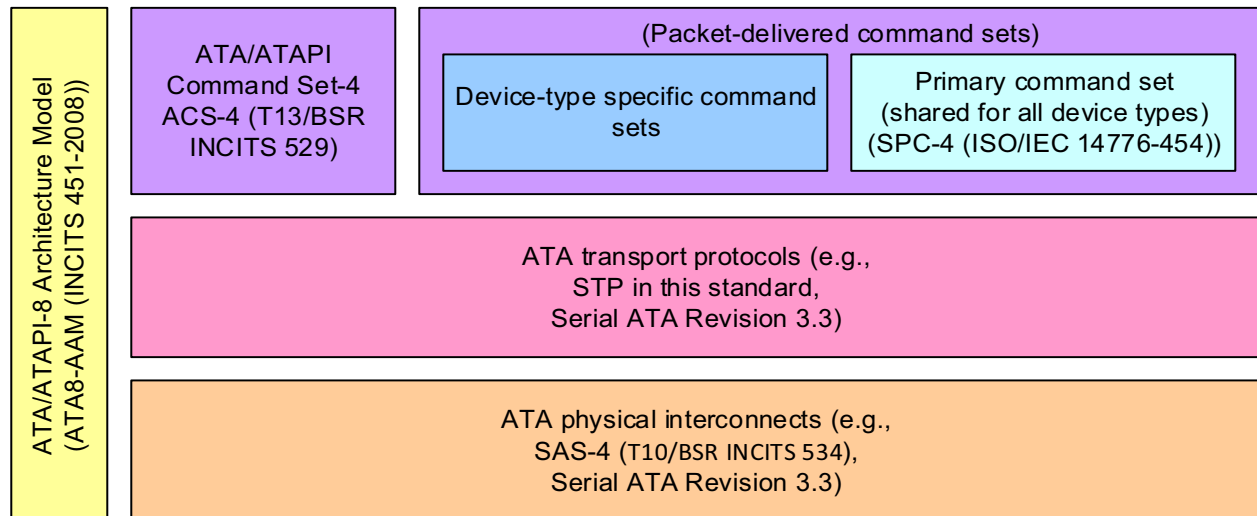


Figure 1 – SCSI document relationships

This standard also defines the rules for exchanging information between ATA hosts and ATA devices using the same serial interconnect. Other ATA transport protocol standards define the rules for exchanging information between ATA hosts and ATA devices using other interconnects.

Figure 2 shows the relationship of this standard to other standards and related projects in the ATA family of standards.



**Figure 2 – ATA document relationships**

Figure 1 and figure 2 show the general relationship of the documents to one another, and do not imply any hierarchy, protocol stack, or system architecture relationship.

These standards specify the interfaces, functions and operations necessary to ensure interoperability between conforming implementations. This standard is a functional description. Conforming implementations may employ any design technique that does not violate interoperability.

This standard makes obsolete the following concept from SPL-3:

- a) the TMC bit and the ETC bit of the Protocol Specific Port log parameter for SAS target ports.

## 2 Normative references

The following documents, in whole or in part, are normatively referenced in this document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

Additional availability contact information is provided in Annex M.

ISO/IEC 14776-151, *Serial Attached SCSI - 1.1 (SAS-1.1)*

ISO/IEC 14776-263, *Information technology – Small Computer System Interface (SCSI) – Part 263: SAS Protocol Layer - 3 (SPL-3) (INCITS 492-2015 under consideration)*

ISO/IEC 14776-323, *Information technology – Small Computer System Interface (SCSI) – Part 323: SCSI Block Commands - 3 (SBC-3)*

ISO/IEC 14776-454, *Information technology – Small Computer System Interface (SCSI) – Part 454: SCSI Primary Commands - 4 (SPC-4)*

INCITS 451-2008, *AT Attachment - 8 ATA/ATAPI Architecture Model (ATA8-AAM)*

INCITS 491-2017, *SCSI/ATA Translation - 4 (SAT-4)*

T10/BSR INCITS 502, *SCSI Primary Commands - 5 (SPC-5)* (planned as ISO/IEC 14776-455)

T10/BSR INCITS 515, *SCSI Architecture Model - 5 (SAM-5)* (planned as ISO/IEC 14776-415)

T10/BSR INCITS 518, *SCSI Enclosure Services - 3 (SES-3)* (planned as ISO/IEC 14776-373)

T13/BSR INCITS 529, *ATA Command Set - 4 (ACS-4)* (planned as ISO/IEC 17760-104)

T10/BSR INCITS 534, *Serial Attached SCSI - 4 (SAS-4)* (planned as ISO/IEC 14776-155)

For information on the current status of the listed documents or regarding availability, contact the indicated organization.

*Serial ATA Revision 3.3 (SATA)*. 2-February-2016

NOTE 1 - For information on the current status of Serial ATA documents, contact the Serial ATA International Organization (see <http://www.sata-io.org>).

SFF-8485, *Serial GPIO (SGPIO) Bus*

NOTE 2 - For more information on the current status of SFF documents, contact the Storage Networking Industry Association (SNIA) (see [www.snia.org/sff](http://www.snia.org/sff)).

## 3 Terms, definitions, symbols, abbreviations, keywords, and conventions

### 3.1 Terms and definitions

#### 3.1.1 8b10b coding

coding scheme that represents an 8-bit byte (i.e., a control byte or data byte) as a 10-bit character (i.e., a control character or data character)

Note 1 to entry: See 5.2.

#### 3.1.2 8b10b encoding

method of encoding an 8-bit byte (i.e., a control byte or data byte) into a 10-bit character (i.e., a control character or data character)

Note 1 to entry: See 5.2.

#### 3.1.3 10b8b decoding

method for decoding a 10-bit character (i.e., a control character or data character) into an 8-bit byte (i.e., a control byte or data byte)

Note 1 to entry: See 5.2.

#### 3.1.4 active cable assembly

cable assembly (see SAS-4) that requires power for internal circuitry used in the transmission of the signal through the cable assembly

Note 1 to entry: See SAS-4.

#### 3.1.5 active phy power condition

normal power condition for a SAS phy or expander phy

Note 1 to entry: See 4.10.1.2.

#### 3.1.6 active phy transmitter adjustment

adjustment of the SP transmitter coefficients without causing a link reset sequence or a link layer error

#### 3.1.7 active zone manager

zone manager (see 3.1.283) that locks a zoning expander device (see 3.1.287)

Note 1 to entry: See 4.8.6.

#### 3.1.8 address frame segment

SPL packet payload that contains four data dwords of an address frame

#### 3.1.9 affiliation

STP target port (see 3.1.254) state of limiting acceptance of connection requests to those from one or more STP initiator ports (see 3.1.248)

Note 1 to entry: See 6.21.6.

#### 3.1.10 affiliation context

set of registers maintained by an STP target port for an STP initiator port holding an affiliation

Note 1 to entry: See 6.21.6.

**3.1.11 aggregation**

form of association that defines a whole-part relationship between the whole (i.e., aggregate) class and its parts

Note 1 to entry: This definition only applies when used in relation to UML.

Note 2 to entry: See 3.6.

**3.1.12 application client**

object that is the source of SCSI commands and task management function requests (see SAM-5), ATA commands (see ATA8-AAM), or SMP function requests

Note 1 to entry: See 4.1.5.

**3.1.13 association**

relationship between two or more classes that specifies connections among their objects

Note 1 to entry: This definition only applies when used in relation to UML.

Note 2 to entry: An association is a relationship that specifies that objects of one class are connected to objects of another class.

Note 3 to entry: See 3.6.

**3.1.14 AT Attachment (ATA)**

standard for the internal attachment of storage devices to hosts

Note 1 to entry: See ATA8-AAM.

**3.1.15 ATA device**

storage peripheral that processes ATA commands and device management functions

Note 1 to entry: Analogous to a SCSI target device (see 3.1.213).

Note 2 to entry: See ATA8-AAM.

**3.1.16 ATA domain**

I/O system consisting of an ATA host and one or more ATA devices that communicate with one another by means of a service delivery subsystem (see 3.1.221)

Note 1 to entry: Analogous to a SCSI domain (see 3.1.209).

Note 2 to entry: See ATA8-AAM.

**3.1.17 ATA host**

host device that originates requests to be processed by an ATA device

Note 1 to entry: Analogous to a SCSI initiator device (see 3.1.210).

Note 2 to entry: See ATA8-AAM.

**3.1.18 attached**

attribute of a class used when an instance of that class is only accessible over a service delivery subsystem

**3.1.19 attached SAS address**

SAS address (see 3.1.183) of the attached phy or the SAS address of the STP target port in an STP SATA bridge (see 4.5.2)

Note 1 to entry: For example, the SAS address of an attached phy may be received in the incoming IDENTIFY address frame during the initialization sequence (see 4.1.2).



**3.1.20 attribute**

named property of a class that describes a range of values that its objects may hold

Note 1 to entry: This definition only applies when used in relation to UML.

Note 2 to entry: See 3.6.

**3.1.21 big-endian**

format for storage or transmission of binary data in which the most significant byte appears first

Note 1 to entry: In a multi-byte value, the byte containing the most significant bit is stored in the lowest memory address and transmitted first, and the byte containing the least significant bit is stored in the highest memory address and transmitted last (e.g., for the value 0080h, the byte containing 00h is stored in the lowest memory address, and the byte containing 80h is stored in the highest memory address).

**3.1.22 binary primitive**

dword in an SPL packet that contains primitives in which the PRIMITIVE SYNCHRONIZE SELECT field, CONTROL1 field, CONTROL2 field, or CONTROL3 field is set to 01b and the remaining 30 bits contain binary data

**3.1.23 bi-phase mark code (BMC)**

encoding in which a transition occurs at the beginning of each TTIU bit cell (see 3.1.272), a one is represented by a transition in the middle of the TTIU bit cell, and a zero is represented by no transition within the TTIU bit cell

Note 1 to entry: See 5.9.

**3.1.24 broadcast**

information about an event in the SAS domain, communicated between phys with the BROADCAST primitive sequence (see 6.2.6.4) and/or the SMP ZONED BROADCAST function (see 9.4.3.20)

Note 1 to entry: See 4.1.15.

**3.1.25 broadcast propagation processor (BPP)**

object within an expander function (see 3.1.77) that manages Broadcasts (see 3.1.24)

Note 1 to entry: See 4.5.5.

**3.1.26 burst time**

part of an OOB signal where the OOB burst is transmitted

Note 1 to entry: See SAS-4.

**3.1.27 byte**

sequence of eight contiguous bits considered as a unit

**3.1.28 cable assembly**

bulk cable with a separable connector at each end plus any retention, backshell, shielding features, or circuitry used for cable management or signal transmission

Note 1 to entry: See SAS-4.

**3.1.29 character**

sequence of ten contiguous bits considered as a unit

Note 1 to entry: A byte is encoded as a character using 8b10b coding (see 5.2).

**3.1.30 class**

description of a set of objects that share the same attributes, operations relationships, and semantics

Note 1 to entry: This definition only applies when used in relation to UML.

Note 2 to entry: Classes may have attributes and may support operations.

**3.1.31 class diagram**

diagram that shows a collection of classes and their contents and relationships

Note 1 to entry: See 3.6.

**3.1.32 codeword**

message symbols and parity check symbols that are an indivisible group

**3.1.33 command descriptor block (CDB)**

structure used to communicate a command from a SCSI application client to a SCSI device server

Note 1 to entry: See SAM-5.

**3.1.34 command identifier**

numerical identifier of the command

Note 1 to entry: In this standard command identifier is synonymous with initiator port transfer tag.

Note 2 to entry: See SAM-5.

**3.1.35 commonly supported setting**

supported settings bit (see table 70) with a value set to one by both the phy and the attached phy as transmitted during the most recent SNW-3

**3.1.36 compliant jitter tolerance pattern (CJTPAT)**

test pattern for jitter testing

Note 1 to entry: See A.2 and SAS-4.

**3.1.37 configuration subprocess**

subprocess invoked from the discover process (see 3.1.66) to configure an externally configurable expander device (see 3.1.85)

Note 1 to entry: See 4.7.

**3.1.38 confirmation**

information passed from a lower layer state machine to a higher layer state machine, usually in response to a request from another state machine

Note 1 to entry: See 3.7.

**3.1.39 connection**

temporary association between a SAS initiator port and a SAS target port using a pathway (see 3.1.150)

Note 1 to entry: See 4.1.12 and 6.16.

**3.1.40 connection rate**

effective rate of dwords through the pathway (see 3.1.150) between a SAS initiator phy and a SAS target phy, established through the connection request

**3.1.41 connection request**

request to establish a connection originated by a SAS phy (see 3.1.193) using an OPEN address frame (see 6.10.3)

Note 1 to entry: See 6.16.2.1.

**3.1.42 connector**

electro-mechanical components consisting of a receptacle and a plug that provide a separable interface between two transmission segments

Note 1 to entry: See SAS-4.

**3.1.43 constraint**

mechanism for specifying semantics or conditions that are maintained as true between entities

Note 1 to entry: This definition only applies when used in relation to UML.

Note 2 to entry: An example of a constraint is a required condition between associations.

Note 3 to entry: See 3.6.

**3.1.44 control byte**

byte containing control information defined in table 50

Note 1 to entry: See 5.3.7.

**3.1.45 control character (Kxx.y)**

character containing control information defined in table 50

Note 1 to entry: See 5.3.7.

**3.1.46 credit advance**

SSP phy feature that increments a transmit SSP frame credit on receipt of an OPEN address frame without waiting for an RRDY

Note 1 to entry: See 4.1.14.

**3.1.47 cyclic redundancy check (CRC)**

error checking mechanism that checks data integrity by computing a polynomial algorithm based checksum

Note 1 to entry: See 6.7.

**3.1.48 D.C. idle**

differential signal level that is nominally 0 V peak-to-peak

Note 1 to entry: See SAS-4.

**3.1.49 D.C. mode**

mode in which D.C. idle is used, during the idle time and negation time of an OOB signal (see SAS-4), and during the RCDT time of speed negotiation windows (see 5.11.4.2.2)

**3.1.50 data byte**

byte containing data information defined in table 49

Note 1 to entry: See 5.3.6.

**3.1.51 data character (Dxx.y)**

character containing data information defined in table 49 (see 5.2)

Note 1 to entry: See 5.3.6.

**3.1.52 data dword**

dword containing four data bytes or four data characters with correct disparity

**3.1.53 deadlock**

condition in which two or more processes (e.g., connection requests) are waiting on the others to complete, resulting in none of the processes completing

**3.1.54 deletable binary primitive**

binary primitive that may be deleted by a receiver instead of being placed into its elasticity buffer (see 6.4)

Note 1 to entry: See 6.3.4.

**3.1.55 deletable extended binary primitive**

extended binary primitive that may be deleted by a receiver instead of being placed into its elasticity buffer (see 6.4)

Note 1 to entry: See 6.4.1.

**3.1.56 deletable primitive**

primitive which may be deleted by a receiver instead of being placed into its elasticity buffer (see 6.5)

Note 1 to entry: See 6.2.5.

**3.1.57 dependency**

relationship between two classes where a change to one class (i.e., the independent class) may cause a change in the other class (i.e., the dependent class)

Note 1 to entry: This definition only applies when used in relation to UML.

Note 2 to entry: See 3.6.

**3.1.58 device name**

worldwide unique name for a device within a transport protocol

Note 1 to entry: See 4.2.6.

**3.1.59 device server**

object that processes SCSI commands (see SAM-5), ATA commands (see ATA8-AAM), or SMP functions

Note 1 to entry: See 4.1.5.

**3.1.60 device type**

device model implemented by the logical unit and indicated to the application client by the contents of the PERIPHERAL DEVICE TYPE field in the standard INQUIRY data (see SPC-5)

**3.1.61 differential high signal level**

Tx+ signal of a transmitter circuit (see SAS-4) is a higher voltage than the voltage measured on the Tx- signal of a transmitter circuit

**3.1.62 differential low signal level**

Tx+ signal of a transmitter circuit (see SAS-4) is a lower voltage than the voltage measured on the Tx- signal of a transmitter circuit

**3.1.63 direct current (D.C.)**

non-alternating current component (see SAS-4) of a signal

Note 1 to entry: See SAS-4.

**3.1.64 direct routing attribute**

attribute of an expander phy that indicates that the expander phy may be used by the ECM (see 3.1.74) to route a connection request to an end device

Note 1 to entry: See 4.5.7.1.

**3.1.65 direct routing method**

method the ECM (see 3.1.74) uses to route connection requests to an attached end device or an attached expander device

Note 1 to entry: See 4.5.7.1.

**3.1.66 discover process**

process performed by a management application client to discover all the SAS devices and expander devices in the SAS domain

Note 1 to entry: The discover process invokes the configuration subprocess (see 3.1.37) as needed.

Note 2 to entry: See 4.6.

**3.1.67 disparity**

difference between the number of ones and zeros in a character (see 5.2)

**3.1.68 domain**

I/O system consisting of devices that communicate with one another by means of a service delivery subsystem

Note 1 to entry: Examples of domains are a SAS domain (see 3.1.186), a SCSI domain (see 3.1.209), and an ATA domain (see 3.1.16)

Note 2 to entry: See 4.1.9.

**3.1.69 dword**

sequence of four contiguous bytes or four contiguous characters considered as a unit

Note 1 to entry: The meaning depends on the context (e.g., when discussing the bits being transmitted over a physical link, dword represents four characters (i.e., 40 bits) and when discussing the contents of a frame before 8b10b encoding (see 3.1.2) or after 10b8b decoding (see 3.1.3), dword represents four bytes (i.e., 32 bits)).

**3.1.70 dword synchronization**

detection of an incoming stream of dwords from a physical link by a phy

Note 1 to entry: See 5.15.

**3.1.71 enclosure**

box, rack, or set of boxes providing the powering, cooling, mechanical protection, EMI protection, and external electronic interfaces for one or more end devices (see 3.1.72) and/or expander devices (see 3.1.76)

Note 1 to entry: The enclosure provides the outermost electromagnetic boundary and acts as an EMI barrier; an enclosure is not a class in this standard.

**3.1.72 end device**

SAS device (see 3.1.184) or SATA device (see 3.1.198) that is not contained within an expander device (see 3.1.76)

**3.1.73 event notification**

information passed from the SSP transport layer to the SCSI application layer notifying the SCSI application layer that a SCSI event has occurred

Note 1 to entry: See SAM-5.

**3.1.74 expander connection manager (ECM)**

an object within an expander function (see 3.1.77) that manages routing

Note 1 to entry: See 4.5.3.

**3.1.75 expander connection router (ECR)**

portion of an expander function (see 3.1.77) that routes messages between expander phys

Note 1 to entry: See 4.5.4.

**3.1.76 expander device**

device that is part of a service delivery subsystem (see 3.1.221), facilitates communication between SAS devices (see 3.1.184) and SATA devices (see 3.1.198), and is either an externally configurable expander device (see 3.1.85) or a self-configuring expander device (see 3.1.215)

Note 1 to entry: See 4.1.7.

**3.1.77 expander function**

object within an expander device (see 3.1.76) that contains an expander connection manager (see 3.1.74), expander connection router (see 3.1.75), and a BPP (see 3.1.25)

Note 1 to entry: See 4.5.1.

**3.1.78 expander logical phy**

expander phy (see 3.1.79) or a multiplexed portion of an expander phy

Note 1 to entry: See 4.1.2.

**3.1.79 expander phy**

phy in an expander device (see 3.1.76) that interfaces to a service delivery subsystem (see 3.1.221)

**3.1.80 expander port**

expander device object that interfaces to a service delivery subsystem (see 3.1.221) and to SAS ports in other devices

Note 1 to entry: See 4.5.2.

**3.1.81 expander route entry**

SAS address and an enable/disable bit in an expander route table (see 3.1.83)

**3.1.82 expander route index**

value used in combination with a phy identifier to select an expander route entry in an expander route table (see 3.1.83) in an externally configurable expander device (see 3.1.85)

Note 1 to entry: See 4.5.7.4.

**3.1.83 expander route table**

table of expander route entries (see 3.1.81) within an expander device (see 3.1.76)

Note 1 to entry: The table is used by the expander function (see 3.1.77) to resolve connection requests.

Note 2 to entry: See 4.5.7.4.

**3.1.84 extended binary primitive**

SPL packet that contains a primitive in which the PRIMITIVE SYNCHRONIZE SELECT field is set to 10b and the remaining 126 bits contain binary data

**3.1.85 externally configurable expander device**

non-self-configuring expander device (see 3.1.76) containing an expander route table (see 3.1.83)

Note 1 to entry: An externally configurable expander device is configurable with the SMP CONFIGURE ROUTE INFORMATION function (see 9.4.3.27)

Note 2 to entry: See 4.1.7.

**3.1.86 field**

group of one or more contiguous bits

**3.1.87 Final-SNW**

final speed negotiation window for 1.5 Gbit/s or 3 Gbit/s without training (see 5.11.4.2.3.2)

**3.1.88 forward error correction**

transmitted information that is used to check and attempt to correct a received SPL packet

**3.1.89 frame**

sequence of data dwords between a start of frame primitive and an end of frame primitive

Note 1 to entry: Start of frame primitives are SOF, SOAF, and SATA\_SOF.

Note 2 to entry: End of frame primitives are EOF, EOAF, SATA\_EOF, B\_EOF (0), B\_EOF (1), B\_EOF (2), and B\_EOF (3).

**3.1.90 frame information structure (FIS)**

SATA frame format

Note 1 to entry: See SATA.

**3.1.91 generalization**

relationship among classes where one class (i.e., the superclass) shares the attributes and operations of one or more other classes (i.e., the subclasses)

Note 1 to entry: This definition only applies when used in relation to UML.

Note 2 to entry: See 3.6.

**3.1.92 hard reset**

SAS device or expander device action in response to a reset event in which the device performs the operations described in 4.4.2

**3.1.93 hard reset sequence**

sequence that causes a hard reset (see 3.1.92)

Note 1 to entry: See 4.4 and 6.11.

**3.1.94 hardware maximum physical link rate**

maximum physical link rate capability of a phy

**3.1.95 hardware minimum physical link rate**

minimum physical link rate capability of a phy

**3.1.96 hash**

mathematical function that maps values from a larger set of values into a smaller set of values, reducing a long value into a shorter hashed value

**3.1.97 highest priority commonly supported setting**

commonly supported setting based on the priority defined in table 73

**3.1.98 I\_T nexus**

nexus between a SAS initiator port and a SAS target port or a nexus between a SCSI initiator port and a SCSI target port

Note 1 to entry: When referring to SAS ports (see 3.1.194), a nexus between a SAS initiator port and a SAS target port.

Note 2 to entry: When referring to SCSI ports (see 3.1.212), a nexus between a SCSI initiator port and a SCSI target port.

Note 3 to entry: See SAM-5.

**3.1.99 I\_T nexus loss**

condition where a SAS port determines that another SAS port is no longer available or an I\_T nexus loss operation occurs in a SCSI port

Note 1 to entry: When referring to SAS ports (see 3.1.194), a condition where a SAS port determines that another SAS port is no longer available.

Note 2 to entry: When referring to SCSI ports (see 3.1.212) (e.g., SSP ports), a condition resulting from the events defined by SAM-5 in which the SCSI device performs the I\_T nexus loss operations described in SAM-5, SPC-4, and the appropriate command standards.

Note 3 to entry: See 4.4.3 and SAM-5.

**3.1.100 I\_T nexus loss event**

event that results in an I\_T nexus loss condition (see SAM-5)

Note 1 to entry: See 4.4.3 and SAM-5.

**3.1.101 I\_T nexus loss timer event**

event that starts an I\_T nexus loss timer (see 7.2.2.1)

Note 1 to entry: See 4.4.3.

**3.1.102 I\_T\_L nexus**

nexus between a SCSI initiator port, a SCSI target port, and a logical unit

Note 1 to entry: See SAM-5.

**3.1.103 identification sequence**

sequence where phys exchange IDENTIFY address frames

Note 1 to entry: See 4.4 and 6.11.

**3.1.104 idle dword**

data dword that is transmitted outside a frame

Note 1 to entry: See 6.6.

**3.1.105 idle dword segment**

SPL packet payload that contains four idle dwords (see 6.6) and is transmitted outside a frame

**3.1.106 idle time**

part of an OOB signal (see 3.1.145) where OOB idle (see 3.1.143) is being transmitted

Note 1 to entry: See 5.7.

**3.1.107 indication**

information passed from a lower layer state machine to a higher layer state machine, usually responding to a request from that higher layer state machine (e.g., see figure 41)

Note 1 to entry: See 3.7.

**3.1.108 information unit**

portion of an SSP frame that carries command, task management function, data, response, or transfer ready information

Note 1 to entry: See 8.2.2.



**3.1.109 initiator connection tag**

value in the OPEN address frame used for SSP and STP connection requests to provide a SAS initiator port an alternative to using the SAS target port's SAS address for context lookup when the SAS target port originates a connection request

Note 1 to entry: See 6.10.3.

**3.1.110 initiator port transfer tag**

value that allows an SSP initiator port to establish a context for commands and task management functions

Note 1 to entry: See 8.2.1.

**3.1.111 invalid character**

character that is not a control character (see 3.1.45) or a data character (see 3.1.51)

**3.1.112 invalid dword**

dword that is not a data dword, primitive parameter, or a primitive

Note 1 to entry: In the character context, a dword that contains an invalid character, a control character in other than the first character position, a control character other than K28.3 or K28.5 in the first character position, or one or more characters with a running disparity error.

**3.1.113 invalid SPL packet**

SPL packet in which the Reed Solomon code decoding function indicates a decode failure

Note 1 to entry: See 5.5.7.3.

**3.1.114 jitter tolerance pattern (JTPAT)**

test pattern for jitter testing

Note 1 to entry: See SAS-4.

**3.1.115 least significant bit (LSB)**

bit or bit position with the smallest numerical weighting in a group of bits that, when taken as a whole, represent a numerical value

Note 1 to entry: This definition only applies when used in relation to binary codes.

Note 2 to entry: For example in the number 0001b the LSB is the bit that is set to one.

**3.1.116 left-aligned**

type of field containing ASCII data in which unused bytes are placed at the end of the field (highest offset) and are filled with ASCII space (20h) characters

Note 1 to entry: See SPC-4.

**3.1.117 link reset sequence**

phy reset sequence

Note 1 to entry: For SATA, a phy reset sequence (see 3.1.156).

Note 2 to entry: For SAS, a phy reset sequence followed by an identification sequence (see 3.1.103), or a phy reset sequence followed by a hard reset sequence (see 3.1.93), another phy reset sequence, and an identification sequence.

Note 3 to entry: See 4.4 and 6.11.

**3.1.118 little-endian**

format for storage or transmission of binary data in which the least significant byte appears first

Note 1 to entry: In a multi-byte value, the byte containing the least significant bit is stored in the lowest memory address and transmitted first, and the byte containing the most significant bit is stored in the highest

memory address and transmitted last (e.g., for the value 0080h, the byte containing 80h is stored in the lowest memory address, and the byte containing 00h is stored in the highest memory address).

**3.1.119 livelock**

condition where two or more processes (e.g., connection requests) repeatedly change their state in response to changes in other processes, resulting in none of the processes completing

**3.1.120 local**

attribute of a class used when an instance of that class is accessible without having to access any service delivery subsystem

**3.1.121 locked zoning expander device**

zoning expander device (see 3.1.287) that has been locked by a zone manager (see 3.1.283)

Note 1 to entry: See 4.8.6.2.

**3.1.122 logical link**

physical link or a multiplexed portion of a physical link

Note 1 to entry: See 4.1.3.

**3.1.123 logical link rate**

link rate between two logical phys established as a result of speed negotiation and multiplexing negotiation between the physical phys containing those logical phys

**3.1.124 logical phy**

phy (see 3.1.153) or a multiplexed portion of a phy

Note 1 to entry: See 4.1.2.

**3.1.125 logical unit**

object that implements a device model and manages and processes commands sent by a SCSI application client

Note 1 to entry: See SAM-5.

**3.1.126 logical unit number (LUN)**

identifier for a logical unit (see 3.1.125)

Note 1 to entry: See SAM-5.

**3.1.127 low phy power condition**

partial phy power condition or the slumber phy power condition

Note 1 to entry: See 4.10.1.

**3.1.128 media**

material on which data is stored

Note 1 to entry: Media is the plural of medium.

Note 2 to entry: An example of media in which data is stored are magnetic disks.

**3.1.129 message**

information sent between state machines

Note 1 to entry: See 3.7.

**3.1.130 most significant bit (MSB)**

bit or bit position with the largest numerical weighting in a group of bits that, when taken as a whole, represent a numerical value

Note 1 to entry: This definition only applies when used in relation to binary codes.

Note 2 to entry: For example, in the number 1000b, the bit that is set to one.

**3.1.131 multiplexing**

interleaving of dwords

Note 1 to entry: Multiplexing divides a single physical link into two logical links by creating two logical phys from a single physical phy.

Note 2 to entry: See 5.20.

**3.1.132 multiplicity**

indication of the range of allowable instances of an object or an attribute

Note 1 to entry: This definition only applies when used in relation to UML.

Note 2 to entry: See 3.6.

**3.1.133 narrow link**

physical link that attaches a narrow port to another narrow port

Note 1 to entry: See 4.1.4.

**3.1.134 narrow port**

port that contains exactly one phy

Note 1 to entry: See 4.1.4.

**3.1.135 negation time**

part of an OOB signal (see 3.1.145) during which OOB idle (see 3.1.143) is transmitted after the last OOB burst (see 3.1.142)

Note 1 to entry: See 5.7.

**3.1.136 negotiated logical link rate**

current operational logical link rate (see 3.1.123)

**3.1.137 negotiated physical link rate**

current operational physical link rate (see 3.1.158)

**3.1.138 negotiation idle**

transmission of OOB idle (see 3.1.143)

Note 1 to entry: See 5.14.4.2.

**3.1.139 nexus**

relationship between two SAS devices or two SCSI devices

Note 1 to entry: When referring to SAS devices (see 3.1.184), a relationship between two SAS devices, and the SAS initiator port and the SAS target port objects within those SAS devices.

Note 2 to entry: When referring to SCSI devices (see 3.1.208), a relationship between two SCSI devices, and the SCSI initiator port and the SCSI target port objects within those SCSI devices.

Note 3 to entry: See SAM-5.

**3.1.140 object**

entity with a well-defined boundary and identity that encapsulates state and behavior

Note 1 to entry: All objects are instances of classes (i.e., a concrete manifestation of a class is an object).

**3.1.141 object diagram**

diagram that encompasses objects and their relationships at a point in time

Note 1 to entry: See 3.6.

**3.1.142 OOB burst**

transmission of signal transitions for a burst time

Note 1 to entry: See SAS-4.

**3.1.143 OOB idle**

transmission of D.C. idle when D.C. mode (see 3.1.49) is enabled or a defined sequence of dwords when optical mode (see 3.1.147) is enabled

Note 1 to entry: See SAS-4.

**3.1.144 OOB sequence**

sequence where two phys exchange OOB signals (see 3.1.145)

Note 1 to entry: See 5.11.2.1 and 5.11.4.1.

**3.1.145 OOB signal**

pattern of idle time (see 3.1.106), burst time, and negation time (see 3.1.135) used during the link reset sequence

Note 1 to entry: See 5.7.

**3.1.146 operation**

service that may be requested from any object of the class in order to affect behavior

Note 1 to entry: This definition only applies when used in relation to UML.

Note 2 to entry: Operations describe what a class is allowed to do and may be a request or a query.

Note 3 to entry: An operation that is a request may change the state of the object but an operation that is a query should not.

Note 4 to entry: See 3.6.

**3.1.147 optical mode**

mode in which a defined sequence of dwords is used during the idle time and negation time of an OOB signal (see SAS-4) and during the RCDT time of speed negotiation windows (see 5.11.4.2.2)

**3.1.148 partial pathway**

set of logical links participating in a connection request that have not yet conveyed a connection response

Note 1 to entry: See 4.1.11.

**3.1.149 partial phy power condition**

low phy power condition for a SAS phy or expander phy

Note 1 to entry: See 4.10.1.3.

**3.1.150 pathway**

set of logical links between a SAS initiator phy and a SAS target phy being used by a connection (see 3.1.39)

Note 1 to entry: See 4.1.11.

**3.1.151 pathway blocked count**

number of times the port has retried this connection request due to receiving OPEN\_REJECT (PATHWAY BLOCKED), OPEN\_REJECT (RESERVED STOP 0), or OPEN\_REJECT (RESERVED STOP 1)

**3.1.152 persistent connection**

connection that may be extended indefinitely

**3.1.153 phy**

object in a device that is used to interface to other devices (e.g., an expander phy (see 3.1.79) or a SAS phy (see 3.1.193))

Note 1 to entry: See 4.1.2 and SAS-4.

**3.1.154 phy identifier**

value by which a phy is identified within a device

Note 1 to entry: See 4.2.10.

**3.1.155 phy ready state**

condition of a phy when its SP state machine (see 5.14) is in the SP15:SAS\_PHY\_Ready state (see 5.14.4.10)

**3.1.156 phy reset sequence**

OOB sequence (see 3.1.144) followed by a speed negotiation sequence (see 3.1.232)

Note 1 to entry: See 4.4 and 5.11.

**3.1.157 physical link**

two differential signal pairs, one pair in each direction, that connect two physical phys

Note 1 to entry: See 4.1.2 and SAS-4.

**3.1.158 physical link rate**

link rate between two physical phys established as a result of speed negotiation between those phys

**3.1.159 physical phy**

phy that contains a transceiver and electrically interfaces to a physical link to communicate with another physical phy

Note 1 to entry: See 4.1.2 and SAS-4.

**3.1.160 port**

SAS port (see 3.1.194) or an expander port (see 3.1.80)

Note 1 to entry: Each port contains one or more phys (see 3.1.153).

Note 2 to entry: See 4.1.4.

**3.1.161 port identifier**

value by which a port is identified within a domain

Note 1 to entry: See 4.2.9.

**3.1.162 port name**

worldwide unique name for a port within a transport protocol

Note 1 to entry: See 4.2.8.

**3.1.163 potential pathway**

set of logical links between a SAS initiator phy and a SAS target phy

Note 1 to entry: See 4.1.11.

**3.1.164 power on**  
power being applied

**3.1.165 primitive**

8b10b coded dword in which the first byte is a control byte (i.e., 7Ch or BCh) or a control character (i.e., K28.3 or K28.5)

Note 1 to entry: For a phy that supports being attached to SATA phy, an 8b10b coded dword containing a 7Ch or BCh control byte followed by three data bytes, or a K28.3 or K28.5 control character with correct disparity followed by three data characters with correct disparity.

Note 2 to entry: For a phy that does not support being attached to SATA phy, an 8b10b coded dword containing a BCh control byte followed by three data bytes, or a K28.5 control character with correct disparity followed by three data characters with correct disparity.

Note 3 to entry: See 6.2.

**3.1.166 primitive parameter**

one to three dwords containing parameter information associated with a primitive or binary primitive within a primitive segment

**3.1.167 primitive segment**

SPL packet payload that contains primitives, binary primitives, a primitive parameter, or an extended binary primitive

**3.1.168 primitive sequence**

set of primitives treated as a single entity

Note 1 to entry: See 6.2.5.

**3.1.169 programmed maximum physical link rate**

maximum operational physical link rate of a phy

Note 1 to entry: The programmed maximum physical link rate may be programmed via the SMP PHY CONTROL function (see 9.4.3.28) or the Phy Control and Discover mode page (see 9.2.7.5).

**3.1.170 programmed minimum physical link rate**

minimum operational physical link rate of a phy

Note 1 to entry: The programmed minimum physical link rate may be programmed via the SMP PHY CONTROL function (see 9.4.3.28) or the Phy Control and Discover mode page (see 9.2.7.5).

**3.1.171 rate**

data transfer rate of a physical or logical link

Note 1 to entry: Rate examples are 1.5 Gbit/s, 3 Gbit/s, 6 Gbit/s, 12 Gbit/s, or 22.5 Gbit/s.

**3.1.172 read data**

data transferred to the SCSI application client's data-in buffer from the SCSI device server, as requested by the Send Data-In transport protocol service (see 9.2.1.6)

**3.1.173 receiver device (Rx)**

device downstream from a receiver device compliance point containing a portion of the physical link and a receiver circuit

Note 1 to entry: See SAS-4.

**3.1.174 reduced control character**

first character of a primitive that has been reduced from a character to a 2-bit value (see table 64 and table 65)

Note 1 to entry: The first character of a primitive is the control character.

**3.1.175 Reed Solomon code**

set of nonbinary cyclic error correcting codes where the symbol size determines the maximum codeword size

Note 1 to entry: Reed Solomon codes have:

- a) a code length that is at least one less than the number of all possible states that may be assigned to a symbol; and
- b) a smallest possible number of differences between two codewords that is one greater than the number of parity check symbols.

**3.1.176 request**

information passed from a higher layer state machine to a lower layer state machine, usually to initiate some action

Note 1 to entry: See 3.7.

**3.1.177 reset event**

event that triggers a hard reset (see 4.4.2) in a SAS device

**3.1.178 response**

information passed from a higher layer state machine to a lower layer state machine, usually in response to an indication (see 3.1.107)

Note 1 to entry: See 3.7.

**3.1.179 role**

label at the end of an association (see 3.1.13) or aggregation (see 3.1.11) that defines a relationship to the class on the other side of the association or aggregation

Note 1 to entry: This definition only applies when used in relation to UML.

Note 2 to entry: See 3.6.

**3.1.180 route table optimization**

configuration subprocess (see 3.1.37) algorithm that reduces the number of entries required in an expander route table (see 3.1.83) in an externally configurable expander device (see 3.1.85)

Note 1 to entry: See 4.7.3.

**3.1.181 run length**

number of consecutive identical bits in the transmitted signal

Note 1 to entry: For example, the pattern 00 1111010b includes the following run lengths of five 1s, one 0, one 1, and indeterminate run lengths of 0s at the start and end.

**3.1.182 running disparity (RD)**

binary parameter with a negative (-) or positive (+) value indicating the cumulative encoded signal imbalance between the one and zero signal state of all characters since dword synchronization has been achieved

Note 1 to entry: See 5.3.5.

**3.1.183 SAS address**

identifier assigned to a SAS port (see 3.1.194) or expander device (see 3.1.76)

Note 1 to entry: See 4.2.4.

**3.1.184 SAS device**

SAS initiator device and/or a SAS target device

Note 1 to entry: See 4.1.6.

**3.1.185 SAS device type**

end device or expander device in a SAS domain

**3.1.186 SAS domain**

I/O system defined by this standard that may serve as a SCSI domain

Note 1 to entry: See 4.1.9.

**3.1.187 SAS dword mode**

mode with a physical link rate less than or equal to 12 Gbit/s (i.e., G1, G2, G3, or G4)

Note 1 to entry: See 5.4.

Note 2 to entry: SAS dword mode is enabled at the start of an OOB sequence (see 5.14.3.2).

**3.1.188 SAS initiator device**

device containing SSP initiator ports, STP initiator ports, and/or SMP initiator ports in a SAS domain

Note 1 to entry: See 4.1.6.

**3.1.189 SAS initiator phy**

logical phy (see 3.1.124) in a SAS initiator device

**3.1.190 SAS initiator port**

SSP initiator port (see 3.1.240), STP initiator port (see 3.1.248), and/or SMP initiator port (see 3.1.225) in a SAS domain

**3.1.191 SAS logical phy**

SAS phy (see 3.1.193) or a multiplexed portion of a SAS phy

Note 1 to entry: See 4.1.2.

**3.1.192 SAS packet mode**

mode with a physical link rate greater than 12 Gbit/s (i.e., G5)

Note 1 to entry: See 5.4.

**3.1.193 SAS phy**

phy in a SAS device that interfaces to a service delivery subsystem (see 3.1.221)

**3.1.194 SAS port**

SAS initiator port (see 3.1.190) and/or a SAS target port (see 3.1.197)

**3.1.195 SAS target device**

device containing SSP target ports, STP target ports, and/or SMP target ports in a SAS domain

Note 1 to entry: See 4.1.6.

**3.1.196 SAS target phy**

logical phy (see 3.1.124) in a SAS target device

**3.1.197 SAS target port**

SSP target port (see 3.1.244), STP target port (see 3.1.254), and/or SMP target port (see 3.1.229) in a SAS domain

**3.1.198 SATA device**

ATA device that contains a SATA device port in an ATA domain; analogous to a SAS target device (see 3.1.195)



**3.1.199 SATA device port**

ATA device object that interfaces to a service delivery subsystem (see 3.1.221) with SATA

Note 1 to entry: Analogous to a SAS target port (see 3.1.197).

**3.1.200 SATA host**

ATA host that contains a SATA host port in an ATA domain

Note 1 to entry: Analogous to a SAS initiator device (see 3.1.188).

**3.1.201 SATA host port**

ATA host object that interfaces to a service delivery subsystem (see 3.1.221) with SATA

Note 1 to entry: Analogous to a SAS initiator port (see 3.1.190).

**3.1.202 SATA phy**

phy in a SATA device or SATA port selector that interfaces to a service delivery subsystem (see 3.1.221)

Note 1 to entry: Analogous to a SAS phy (see 3.1.193).

**3.1.203 SATA port selector**

device that attaches to two SATA host ports (i.e., two ATA domains) and one SATA device port, and provides the means for one SATA host to access the SATA device at any given time (see SATA)

**3.1.204 SATA spinup hold**

state entered by an expander phy attached to a SATA device in which it halts the automatic phy reset sequence to delay temporary consumption of additional power

Note 1 to entry: An example of when a SATA device consumes additional power is during spin up of rotating media.

Note 2 to entry: See 5.21.

**3.1.205 saturating counter**

counter that remains at its maximum value after reaching its maximum value

**3.1.206 scrambled idle segment**

SPL packet payload that contains four data dwords set to zero

**3.1.207 scrambling**

modifying data by XORing each bit with a pattern generated by a linear feedback shift register to minimize repetitive character patterns

Note 1 to entry: See 6.8.

**3.1.208 SCSI device**

device that contains one or more SCSI ports that are connected to a service delivery subsystem (see 3.1.221) and supports a SCSI application protocol

Note 1 to entry: See SAM-5.

**3.1.209 SCSI domain**

I/O system consisting of a set of SCSI devices that communicate with one another by means of a service delivery subsystem (see 3.1.221)

Note 1 to entry: See SAM-5.

**3.1.210 SCSI initiator device**

SCSI device containing SCSI application clients and SCSI initiator ports that originates device service and task management requests to be processed by a SCSI target device and receives device service and task management responses from SCSI target devices

Note 1 to entry: See SAM-5.

**3.1.211 SCSI initiator port**

SCSI initiator device object that acts as the connection between SCSI application clients and a service delivery subsystem (see 3.1.221) through which requests and confirmations are routed

Note 1 to entry: See SAM-5.

**3.1.212 SCSI port**

SCSI initiator port and/or a SCSI target port

Note 1 to entry: See SAM-5.

**3.1.213 SCSI target device**

SCSI device containing logical units and SCSI target ports that receives device service and task management requests for processing and sends device service and task management responses to SCSI initiator devices

Note 1 to entry: See SAM-5.

**3.1.214 SCSI target port**

SCSI target device object that contains a task router and acts as the connection between SCSI device servers and task managers and a service delivery subsystem (see 3.1.221) through which requests, indications, responses, and confirmations are routed

Note 1 to entry: See SAM-5.

**3.1.215 self-configuring expander device**

expander device (see 3.1.76) containing an SMP initiator port and a management application client to perform the discover process (see 3.1.66) to configure its own expander route table (see 3.1.83)

Note 1 to entry: See 4.6.4.

**3.1.216 Serial ATA (SATA)**

protocol defined by SATA (see clause 2)

**3.1.217 Serial ATA Tunneled Protocol (STP)**

protocol defined in this standard used by STP initiator ports (see 3.1.248) to communicate with STP target ports (see 3.1.254) in a SAS domain

Note 1 to entry: See 6.21 and 8.3.

**3.1.218 Serial Attached SCSI (SAS)**

set of protocols defined by this standard and the interconnects defined in SAS-4

**3.1.219 Serial Management Protocol (SMP)**

protocol defined in this standard used by SMP initiator ports (see 3.1.225) to communicate with SMP target ports (see 3.1.229) in a SAS domain

Note 1 to entry: See 6.22 and 8.4.

**3.1.220 Serial SCSI Protocol (SSP)**

protocol defined in this standard used by SSP initiator ports (see 3.1.240) to communicate with SSP target ports (see 3.1.244) in a SAS domain

Note 1 to entry: See 6.20 and 8.2.

**3.1.221 service delivery subsystem**

part of the domain that transmits information

Note 1 to entry: A service delivery subsystem is the part of:

- a) a SCSI I/O system that transmits information between a SCSI initiator port and a SCSI target port;
- b) an ATA I/O system that transmits information between an ATA host and an ATA device; or
- c) a SAS I/O system that transmits information between a SAS initiator port and a SAS target port.

Note 2 to entry: See 4.1.8.

**3.1.222 slumber phy power condition**

low phy power condition for a SAS phy or expander phy

Note 1 to entry: See 4.10.1.4.

**3.1.223 SMP frame segment**

SPL packet payload that contains four data dwords of an SMP frame (see 8.4.1)

**3.1.224 SMP initiator phy**

SAS initiator phy (see 3.1.189) in an SMP initiator port (see 3.1.225)

**3.1.225 SMP initiator port**

SAS initiator device object in a SAS domain that interfaces to a service delivery subsystem with SMP

**3.1.226 SMP phy**

SAS logical phy (see 3.1.191) in an SMP port

**3.1.227 SMP port**

SMP initiator port (see 3.1.225) and/or an SMP target port (see 3.1.229)

**3.1.228 SMP target phy**

SAS target phy (see 3.1.196) in an SMP target port (see 3.1.229)

**3.1.229 SMP target port**

SAS target device object in a SAS domain that interfaces to a service delivery subsystem (see 3.1.221) with SMP

**3.1.230 SMP zone configuration function**

SMP function that configures zoning expander shadow values (see 3.1.289) and is only accepted by a locked zoning expander device

Note 1 to entry: See 4.8.6.3.

**3.1.231 SNW-3 bit cell time**

time for transmitting a phy capabilities bit during SNW-3 (see 5.11.4.2.3.3)

**3.1.232 speed negotiation sequence**

sequence in which two phys negotiate the operational physical link rate

Note 1 to entry: See 5.11.2.2 and 5.11.4.2.

**3.1.233 speed negotiation window (SNW)**

portion of the SAS speed negotiation sequence

Note 1 to entry: See 5.11.4.2.3.

**3.1.234 SPL frame segment**

SPL packet payload that contains an address frame segment, SSP frame segment, SMP frame segment, or STP frame segment

**3.1.235 SPL packet**

150 bits of which two bits contain header information, 128 bits contain an SPL packet payload, and 20 bits contain forward error correction information

Note 1 to entry: See 5.5.2.

**3.1.236 SPL packet payload**

contains an idle dword segment, primitive segment, scrambled idle segment, or SPL frame segment

Note 1 to entry: See 5.5.2.

**3.1.237 spread spectrum clocking (SSC)**

technique of modulating the operating frequency of a transmitted signal (i.e., the physical link rate) to reduce the measured peak amplitude of radiated emissions

Note 1 to entry: See SAS-4.

**3.1.238 SSP frame segment**

SPL packet payload that contains four data dwords of an SSP frame

**3.1.239 SSP initiator phy**

SAS initiator phy (see 3.1.189) in an SSP initiator port (see 3.1.240)

**3.1.240 SSP initiator port**

SCSI initiator port in a SAS domain that implements SSP

**3.1.241 SSP phy**

SAS logical phy (see 3.1.191) in an SSP port

**3.1.242 SSP port**

SSP initiator port (see 3.1.240) and/or an SSP target port (see 3.1.244)

**3.1.243 SSP target phy**

SAS target phy (see 3.1.196) in an SSP target port (see 3.1.244)

**3.1.244 SSP target port**

SCSI target port in a SAS domain that implements SSP

**3.1.245 state machine variable**

variable that exists within the context of a state machine

Note 1 to entry: A state machine variable may contain status from one state that is used in another state of the same state machine.

Note 2 to entry: The value contained in a state machine variable may affect subsequent state transitions or state machine outputs.

**3.1.246 STP frame segment**

SPL packet payload that contains four data dwords of an STP frame

**3.1.247 STP initiator phy**

SAS initiator phy (see 3.1.189) in an STP initiator port (see 3.1.248)

**3.1.248 STP initiator port**

SAS initiator device object in a SAS domain that interfaces to a service delivery subsystem (see 3.1.221) with STP

**3.1.249 STP phy**

SAS logical phy (see 3.1.191) in an STP port

**3.1.250 STP port**

STP initiator port (see 3.1.248) and/or an STP target port (see 3.1.254)

**3.1.251 STP primitive** (see 6.2.2)

primitive used only inside STP connections and on SATA physical links

**3.1.252 STP SATA bridge**

expander device object containing an STP target port, a SATA host port, and the functions required to forward information between the STP target port and SATA host port to enable STP initiator ports in a SAS domain to communicate with SATA devices in an ATA domain

**3.1.253 STP target phy**

SAS target phy (see 3.1.196) in an STP target port (see 3.1.254)

**3.1.254 STP target port**

SAS target device object in a SAS domain that interfaces to a service delivery subsystem (see 3.1.221) with STP

**3.1.255 subtractive routing attribute**

attribute of an expander phy that indicates that the expander phy may be used by the ECM (see 3.1.74) to route connection requests to an attached expander device that were not resolved using the direct routing method or table routing method

Note 1 to entry: See 4.5.7.1.

**3.1.256 subtractive routing method**

method the ECM (see 3.1.74) uses to route connection requests not resolved using the direct routing method or table routing method to an attached expander device (see 3.1.76)

Note 1 to entry: See 4.5.7.1.

**3.1.257 symbol**

computational unit of a codeword

Note 1 to entry: This definition only applies when referring to forward error correction (see 5.5.7).

**3.1.258 table routing attribute**

attribute of an expander phy that indicates that the expander phy may be used by the ECM (see 3.1.74) to route connection requests using an expander route table (see 3.1.83)

Note 1 to entry: See 4.5.7.1.

**3.1.259 table routing method**

method the ECM (see 3.1.74) uses to route connection requests to an attached expander device (see 3.1.76) using an expander route table (see 3.1.83)

Note 1 to entry: See 4.5.7.1.

**3.1.260 target port transfer tag**

optional value that allows an SSP target port to establish the write data context when receiving a write DATA frame

Note 1 to entry: See 8.2.1.

**3.1.261 task management function**

task manager service capable of being requested by a SCSI application client to affect the processing of one or more commands

Note 1 to entry: See SAM-5.

**3.1.262 task manager**

object that controls the sequencing of SCSI commands and processes SCSI task management functions

Note 1 to entry: See SAM-5.

**3.1.263 Train\_Rx-SNW**

speed negotiation window with receiver training

Note 1 to entry: See 5.11.4.2.3.5.

**3.1.264 Train\_Rx-SNW window time**

actual duration of Train\_Rx-SNW

**3.1.265 Train\_Tx-SNW**

speed negotiation window with transmitter training

Note 1 to entry: See 5.11.4.2.3.4.

**3.1.266 transmitter device (Tx)**

device upstream from a transmitter device compliance point containing a portion of the physical link and a transmitter circuit (see SAS-4)

**3.1.267 transmitter training information unit (TTIU)**

32 TTIU bit cells that follow a pattern marker transmitted during Train\_Tx-SNW (see 5.11.4.2.3.4)

Note 1 to entry: See 5.11.4.2.3.5.

**3.1.268 transport protocol service confirmation**

information passed from the SSP transport layer to the SCSI application layer (e.g., from the SSP initiator port to the SCSI application client) that notifies the SCSI application layer that a SCSI transport protocol service has completed

**3.1.269 transport protocol service indication**

information passed from the SSP transport layer to the SCSI application layer notifying the SCSI application layer (e.g., from the SSP target port to the SCSI device server) to begin a SCSI transport protocol service

**3.1.270 transport protocol service request**

information passed from the SCSI application layer to the SSP transport layer (e.g., from the SCSI application client to the SCSI initiator port) to begin a SCSI transport protocol service

**3.1.271 transport protocol service response**

information passed from the SCSI application layer to the SSP transport layer (e.g., from the SCSI device server to the SSP target port) that completes the SCSI transport protocol service

**3.1.272 TTIU bit cell**

group of ten UIs that encode a single bit of BMC (see 3.1.23) information

**3.1.273 unit interval (UI)**

time period that has the normalized, dimensionless, nominal duration of a symbol (see SAS-4) (e.g.,  $666.\overline{6}$  ps at 1.5 Gbit/s,  $333.\overline{3}$  ps at 3 Gbit/s,  $166.\overline{6}$  ps at 6 Gbit/s,  $83.\overline{3}$  ps at 12 Gbit/s, and  $44.\overline{4}$  ps at 22.5 Gbit/s)

Note 1 to entry: See SAS-4.

**3.1.274 valid character**

character that is a control character (see 3.1.45) or a data character (see 3.1.51)

**3.1.275 valid dword**

dword that is a data dword (see 3.1.52) or a primitive (see 3.1.165)

**3.1.276 virtual phy**

phy (see 3.1.153) that interfaces with a vendor specific interface to another virtual phy inside the same device

Note 1 to entry: See 4.1.2.

**3.1.277 wide link**

group of physical links that attaches a wide port to another wide port

Note 1 to entry: See 4.1.4.

**3.1.278 wide port**

port that contains more than one phy

Note 1 to entry: See 4.1.4.

**3.1.279 word**

sequence of 16 contiguous bits considered as a unit

**3.1.280 wrapping counter**

counter that wraps back to zero after reaching its maximum value

**3.1.281 write data**

data transferred from the SCSI application client's data-out buffer to the SCSI device server, as requested by the Request Data-Out transport protocol service (see 9.2.1.8)

**3.1.282 zone group**

set of phys in a ZPSDS (see 3.1.285) that all have the same access permission

Note 1 to entry: See 4.8.

**3.1.283 zone manager**

entity responsible for configuring a ZPSDS (see 3.1.285)

Note 1 to entry: See 4.8.1.

**3.1.284 zone permission table**

table that defines access permission between the zone group of a source phy and the zone group of the destination phy

Note 1 to entry: See 4.8.3.3.

**3.1.285 zoned portion of a service delivery subsystem (ZPSDS)**

group of zoning expander devices (see 3.1.287) that cooperate to control access between phys; the ZPSDS may include all or part of a service delivery subsystem (see 3.1.221)

Note 1 to entry: See 4.8.

**3.1.286 zoning expander current values**

current zone permission table (see 3.1.284) and zone phy information in a zoning expander device (see 3.1.287)

Note 1 to entry: See 4.8.6.

**3.1.287 zoning expander device**

expander device (see 3.1.76) that supports zoning

Note 1 to entry: See 4.8.

**3.1.288 zoning expander phy**

expander phy in a zoning expander device (see 3.1.287)

**3.1.289 zoning expander shadow values**

shadow zone permission table (see 3.1.284) and zone phy information in a zoning expander device, which are changed by SMP zone configuration functions (see 3.1.230) but do not become active until the activate step (see 4.8.6.4) is performed

Note 1 to entry: See 4.8.6.

**3.2 Symbols and abbreviations****3.2.1 Abbreviations**

See Annex M for abbreviations of standards bodies (e.g., ISO).

Abbreviations used in this standard:

<b>Abbreviation</b>	<b>Meaning</b>
ACA	auto contingent allegiance (see SAM-5)
ACK	acknowledge (see 6.2.7.1)
AIP	arbitration in progress (see 6.2.6.1)
APTA	active phy transmitter adjustment
ATA	AT attachment (see 3.1.14)
ATAPI	AT attachment packet interface
ATA8-AAM	AT Attachment - 8 ATA/ATAPI Architecture Model standard (see clause 2)
ACS-4	ATA Command Set-4 standard (see clause 2)
AWT	arbitration wait time
BCH	Bose, Chaudhuri and Hocquenghem code (see 4.2.5)
BIST	built in self test
BMC	bi-phase mark code (see 3.1.23)
BPP	Broadcast propagation processor (see 3.1.25)
C1	coefficient 1 (see SAS-4)
C2	coefficient 2 (see SAS-4)
C3	coefficient 3 (see SAS-4)
CDB	command descriptor block (see 3.1.33)
CJTPAT	compliant jitter tolerance pattern (see 3.1.36)
CRC	cyclic redundancy check (see 3.1.47)
CRN	command reference number (see SAM-5)
D.C.	direct current (see 3.1.63)
ECM	expander connection manager (see 3.1.74)
ECR	expander connection router (see 3.1.75)
EMI	electromagnetic interference
EOAF	end of address frame (see 6.2.6.6)
EOF	end of frame (see 6.2.7.4)
FIS	frame information structure (see 3.1.90)
G1	generation 1 physical link rate (i.e., 1.5 Gbit/s)
G2	generation 2 physical link rate (i.e., 3 Gbit/s)



<b>Abbreviation</b>	<b>Meaning</b>
G3	generation 3 physical link rate (i.e., 6 Gbit/s)
G4	generation 4 physical link rate (i.e., 12 Gbit/s)
G5	generation 5 physical link rate (i.e., 22.5 Gbit/s)
ID	identifier
JTPAT	jitter tolerance pattern (see 3.1.114)
LED	light-emitting diode
LSB	least significant bit (see 3.1.115)
LUN	logical unit number (see 3.1.126)
MA	management application layer (see 9.4)
MRTT	maximum receiver training time (see table 87)
MSB	most significant bit (see 3.1.130)
MT	SMP transport layer state machines (see 8.4.5)
MTTT	maximum transmitter training time (see table 87)
MTWT	maximum Train_Rx-SNW window time (see table 87)
MTXT	maximum Train_Tx-SNW window time (see table 87)
N/A	not applicable
NAA	network address authority (see 4.2)
NAK	negative acknowledge (see 6.2.7.6)
OOB	out-of-band
OOBI	out-of-band interval
OUI	organizationally unique identifier (i.e., company identifier)
PL	port layer state machines (see 7.2)
PL_OC	port layer overall control (see 7.2.2)
PL_PM	port layer phy manager (see 7.2.3)
PTT	phy layer transmitter training state machines (see 5.18)
RCDT	rate change delay time (see table 87)
RD	running disparity (see 3.1.182)
RRDY	receiver ready (see 6.2.7.7)
Rx	receiver device (see 3.1.173)
SA	SCSI application (see 9.2)
SAM-5	SCSI Architecture Model - 5 standard (see clause 2)
SA_PC	SCSI application layer power condition state machine (see 9.2.10.2)
SAS	Serial Attached SCSI (see 3.1.218)
SATA	Serial ATA (see 3.1.216) or the Serial ATA 3.3 specification (see clause 2)
SBC-3	SCSI Block Commands - 3 standard (see clause 2)
SCSI	Small Computer System Interface family of standards
SGPIO	Serial GPIO (see clause 2)
SL	link layer for SSP phys state machines (see 6.18)
SL_IR	link layer identification and hard reset state machines (see 6.12)
SL_CC	link layer connection control state machine (see 6.18.4)
SL_P	SL_P_C or SL_P_S

<b>Abbreviation</b>	<b>Meaning</b>
SL_P_C	link layer power consumer device state machine (see 6.14.5)
SL_P_S	link layer power source device state machine (see 6.14.4)
SL_RA	link layer receive OPEN address frame state machine (see 6.18.3)
SMP	Serial Management Protocol (see 3.1.219) or link layer for SMP phys state machines (see 6.22.6)
SNLT	speed negotiation lock time (see table 87)
SNTT	speed negotiation transmit time (see table 87)
SNW	speed negotiation window (see 3.1.233)
SNW-1	speed negotiation window for 1.5 Gbit/s without training (see 5.11.4.2.3.2)
SNW-2	speed negotiation window for 3 Gbit/s without training (see 5.11.4.2.3.2)
SNW-3	speed negotiation window negotiating physical link rates with training (see 5.11.4.2.3.3)
SNWT	speed negotiation window time (see table 87)
SOAF	start of address frame (see 6.2.6.18)
SOF	start of frame (see 6.2.7.8)
SP	phy layer state machine (see 5.14)
SP_DWS	phy layer dword synchronization state machine (see 5.15)
SPC-4	SCSI Primary Commands - 4 standard (see clause 2)
SPL	SAS protocol layer
SSP	Serial SCSI Protocol (see 3.1.220) or link layer for SSP phys state machines (see 6.20.9)
ST	transport layer for SSP ports state machines (see 8.2.6)
ST_I	transport layer for SSP initiator ports state machines (see 8.2.6.2)
ST_IFR	transport layer for SSP initiator ports initiator frame router state machine (see 8.2.6.2.2)
ST_ITS	transport layer for SSP initiator ports initiator transport server state machine (see 8.2.6.2.3)
STP	Serial ATA Tunneled Protocol (see 3.1.217) or link layer for STP phys state machines (see 6.21.10)
ST_T	transport layer for SSP target ports state machines (see 8.2.6.3)
ST_TFR	transport layer for SSP target ports target frame router state machine (see 8.2.6.3.2)
ST_TTS	transport layer for SSP target ports target transport server state machine (see 8.2.6.3.3)
TLT	training lock time (see table 87)
TT	STP transport layer state machines (see 8.3.3)
TTIU	transmitter training information unit (see 3.1.272)
Tx	transmitter device (see 3.1.266)
UI	unit interval (see 3.1.273)
UML	Unified Modeling Language
VPD	vital product data (see 9.2.11)
XL	link layer for expander phys state machine (see 6.19)
ZPSDS	zoned portion of a service delivery subsystem (see 3.1.285)

### 3.2.2 Units

Units used in this standard:

Units	Meaning
Gbit/s	gigabits per second (i.e., $10^9$ bits per second)
$\mu$ s	microsecond (i.e., $10^{-6}$ seconds)
m	meter
ms	millisecond (i.e., $10^{-3}$ seconds)
ns	nanosecond (i.e., $10^{-9}$ seconds)
ps	picosecond (i.e., $10^{-12}$ seconds)
s	second (unit of time)
V	volt

### 3.2.3 Symbols

Symbols used in this standard:

Symbols	Meaning
Kxx.y	control character (see 3.1.45)
Dxx.y	data character (see 3.1.51)
ZP[s, d]	zone permission bit for a source zone group (i.e., s) and a destination zone group (i.e., d) in the zone permission table (see 4.8.3.3)
®	registered trademark

### 3.2.4 Mathematical operators

Mathematical operators used in this standard:

Mathematical Operators	Meaning
NE	not equal
XOR	exclusive logical OR
$\wedge$	exclusive logical OR
<	less than
$\leq$	less than or equal to
>	greater than
$\geq$	greater than or equal to
+	plus
-	minus
$\pm$	plus or minus
$\times$	multiplied by
/	divided by
$\sim$	approximately equal to
$\in$	set membership

### 3.3 Keywords

#### 3.3.1 invalid

keyword used to describe an illegal or unsupported bit, byte, word, field or code value

Note 1 to entry: Receipt of an invalid bit, byte, word, field or code value shall be reported as an error.

#### 3.3.2 mandatory

keyword indicating an item that is required to be implemented as defined in this standard

#### 3.3.3 may

keyword that indicates flexibility of choice with no implied preference (equivalent to “may or may not”)

#### 3.3.4 may not

keywords that indicate flexibility of choice with no implied preference (equivalent to “may or may not”)

#### 3.3.5 obsolete

keyword indicating that an item was defined in prior standards but has been removed from this standard

#### 3.3.6 optional

keyword that describes features that are not required to be implemented by this standard

Note 1 to entry: If any optional feature defined in this standard is implemented, then it shall be implemented as defined in this standard.

#### 3.3.7 reserved

keyword referring to bits, bytes, words, fields, and code values that are set aside for future standardization

Note 1 to entry: A reserved bit, byte, word or field shall be set to zero, or in accordance with a future extension to this standard.

Note 2 to entry: Recipients are not required to check reserved bits, bytes, words or fields for zero values; receipt of reserved code values in defined fields shall be reported as an error.

#### 3.3.8 restricted

keyword referring to bits, bytes, words, and fields that are set aside for other identified standardization purposes

Note 1 to entry: A restricted bit, byte, word, or field shall be treated as a reserved bit, byte, word or field in the context where the restricted designation appears.

#### 3.3.9 shall

keyword indicating a mandatory requirement

Note 1 to entry: Designers are required to implement all such mandatory requirements to ensure interoperability with other products that conform to this standard.

#### 3.3.10 should

keyword indicating flexibility of choice with a strongly preferred alternative (equivalent to “is strongly recommended”)

#### 3.3.11 vendor specific

something (e.g., a bit, field, or code value) that is not defined by this standard

Note 1 to entry: Specification of the referenced item is determined by the SCSI device vendor and may be used differently in various implementations.

### 3.4 Editorial conventions

Certain words and terms used in this standard have a specific meaning beyond the normal English meaning. These words and terms are defined either in clause 3 or in the text where they first appear.

Names of signals, address frames, primitives and primitive sequences, SMP functions, state machines, SCSI and ATA commands, SCSI statuses, SCSI sense keys, and SCSI additional sense codes are in all uppercase (e.g., REQUEST SENSE command).

Names of messages, arguments, requests, confirmations, indications, responses, event notifications, timers, SCSI diagnostic pages, SCSI mode pages, and SCSI log pages are in mixed case (e.g., Disconnect-Reconnect mode page).

Names of fields are in small uppercase (e.g., DESTINATION SAS ADDRESS). Normal case is used when the contents of a field are being discussed. Fields containing only one bit are usually referred to as the NAME bit instead of the NAME field.

Names of procedure calls are identified by a name in bold type (e.g., **Execute Command**). For more information on procedure calls see 3.9.

Lists sequenced by lowercase or uppercase letters show no ordering relationship between the listed items.

EXAMPLE 1 -The following list shows no relationship between the named items:

- a) red, specifically one of the following colors:
  - A) crimson; or
  - B) amber;
- b) blue; or
- c) green.

Lists sequenced by numbers show an ordering relationship between the listed items.

EXAMPLE 2 -The following list shows an ordered relationship between the named items:

- 1) top;
- 2) middle; and
- 3) bottom.

Lists are associated with an introductory paragraph or phrase, and are numbered relative to that paragraph or phrase (i.e., all lists begin with an a) or 1) entry).

If a conflict arises between text, tables, or figures, the order of precedence to resolve the conflicts is text, then tables, and finally figures. Not all tables or figures are fully described in the text. Tables show data format and values.

Notes and example do not constitute any requirements for implementers and notes are numbered consecutively throughout this standard.

### 3.5 Numeric and character conventions

#### 3.5.1 Numeric conventions

A binary number is represented in this standard by any sequence of digits consisting of only the Arabic numerals 0 and 1 immediately followed by a lower-case b (e.g., 0101b). Underscores or spaces may be included between characters in binary number representations to increase readability or delineate field boundaries (e.g., 00010101 11001110b, 00010101\_11001110b, 0 0101 1010b, or 0\_0101\_1010b).

A hexadecimal number is represented in this standard by any sequence of digits consisting of only the Arabic numerals 0 to 9 and/or the upper-case English letters A to F immediately followed by a lower-case h (e.g., FA23h). Underscores or spaces may be included between characters in hexadecimal number representations to increase readability or delineate field boundaries (e.g., B FD8C FA23h or B\_FD8C\_FA23h).

A decimal number is represented in this standard by any sequence of digits consisting of only the Arabic numerals 0 to 9 not immediately followed by a lower-case b or lower-case h (e.g., 25).

A range of numeric values is represented in this standard in the form “a to z”, where a is the first value included in the range, all values between a and z are included in the range, and z is the last value included in the range (e.g., the representation “0h to 3h” includes the values 0h, 1h, 2h, and 3h).

Variables (i.e., alphanumeric names that represent values in computations and other statements) are represented in the same San-serif font as other information in this standard.

This standard uses the following conventions for representing decimal numbers:

- a) the decimal separator (i.e., separating the integer and fractional portions of the number) is a period;
- b) the thousands separator (i.e., separating groups of three digits in a portion of the number) is a space;
- c) the thousands separator is used in both the integer portion and the fraction portion of a number; and
- d) the decimal representation for a year is 1999 not 1 999.

Table 1 shows some examples of decimal numbers using various numbering conventions.

**Table 1 – Numbering conventions**

French	English	This standard
0,6	0.6	0.6
3,141 592 65	3.14159265	3.141 592 65
1 000	1,000	1 000
1 323 462,95	1,323,462.95	1 323 462.95

A decimal number represented in this standard with an overline over one or more digits following the decimal point is a number where the overlined digits are infinitely repeating (e.g.,  $666.\overline{6}$  means 666.666 666... or  $666 \frac{2}{3}$ , and  $12.142 \overline{857}$  means 12.142 857 142 857... or  $12 \frac{1}{7}$ ).

### 3.5.2 Units of measure

This standard represents values using both decimal units of measure and binary units of measure. Values are represented by the following formats:

- a) for values based on decimal units of measure:
  - 1) numerical value (e.g., 100);
  - 2) space; and
  - 3) prefix symbol and unit:
    - 1) decimal prefix symbol (e.g., M) (see table 2); and
    - 2) unit abbreviation;

and
- b) for values based on binary units of measure:
  - 1) numerical value (e.g., 1 024);
  - 2) space; and
  - 3) prefix symbol and unit:
    - 1) binary prefix symbol (e.g., Gi) (see table 2); and
    - 2) unit abbreviation.

Table 2 compares the prefix, symbols, and power of the binary and decimal units.

**Table 2 – Comparison of decimal prefixes and binary prefixes**

Decimal			Binary		
Prefix name	Prefix symbol	Power (base-10)	Prefix name	Prefix symbol	Power (base-2)
kilo	k	$10^3$	kibi	Ki	$2^{10}$
mega	M	$10^6$	mebi	Mi	$2^{20}$
giga	G	$10^9$	gibi	Gi	$2^{30}$
tera	T	$10^{12}$	tebi	Ti	$2^{40}$
peta	P	$10^{15}$	pebi	Pi	$2^{50}$
exa	E	$10^{18}$	exbi	Ei	$2^{60}$
zetta	Z	$10^{21}$	zebi	Zi	$2^{70}$
yotta	Y	$10^{24}$	yobi	Yi	$2^{80}$

### 3.5.3 Byte encoded character strings conventions

When this standard requires one or more bytes to contain specific encoded characters, the specific characters are enclosed in single quotation marks. The single quotation marks identify the start and end of the characters that are required to be encoded but are not themselves to be encoded. The characters that are to be encoded are shown in the case that is to be encoded.

An ASCII space character (i.e., 20h) may be represented in a string by the character '¬' (e.g., 'SCSI¬device').

The encoded characters and the single quotation marks that enclose them are preceded by text that specifies the character encoding methodology and the number of characters required to be encoded.

EXAMPLE - Using the notation described in this subclause, stating that eleven ASCII characters 'SCSI device' are to be encoded is the same as writing out the following sequence of byte values: 53h 43h 53h 49h 20h 64h 65h 76h 69h 63h 65h.

## 3.6 UML notation conventions

### 3.6.1 Notation conventions overview

This standard uses class diagrams and object diagrams with notation that is based on the UML.

See 3.6.3 for the conventions used for class diagrams.

See 3.6.4 for the conventions used for object diagrams.

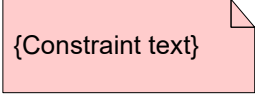

### 3.6.2 Constraint and note conventions

Class diagrams and object diagrams may include:

- a) constraints, which specify requirements; and
- b) notes, which are informative.

Table 3 shows the notation used for constraints and notes.

**Table 3 – Constraint and note notation**

Notation	Description
	The presence of the curly brackets (i.e., {}) defines constraint that is a normative requirement. An example of a constraint is shown in figure 4.
	The absence of curly brackets defines a note that is informative. An example of a note is shown in figure 5.



### 3.6.3 Class diagram conventions

Table 4 shows the notation used for classes in class diagrams.

**Table 4 – Class diagram notation for classes**

Notation	Description
<div style="display: flex; justify-content: space-around; align-items: flex-start;"> <div style="border: 1px solid black; padding: 2px; text-align: center; width: 150px;"><b>Class Name</b></div> <div style="border: 1px solid black; padding: 2px; text-align: center; width: 150px;"><b>Class Name</b></div> <div style="border: 1px solid black; padding: 2px; text-align: center; width: 150px;"><b>Class Name</b></div> </div>	A class that may or may not have attributes or operations
<div style="display: flex; justify-content: space-around; align-items: flex-start;"> <div style="border: 1px solid black; padding: 2px; text-align: center; width: 150px;"><b>Class Name</b></div> <div style="border: 1px solid black; padding: 2px; text-align: center; width: 150px;"><b>Class Name</b></div> </div> <div style="display: flex; justify-content: space-around; align-items: flex-start;"> <div style="border: 1px solid black; padding: 2px; width: 150px;">Attribute01[1] Attribute02[1]</div> <div style="border: 1px solid black; padding: 2px; width: 150px;">Attribute01[1] Attribute02[1]</div> </div>	A class that has attributes and may or may not have operations
<div style="border: 1px solid black; padding: 2px; text-align: center; width: 150px;"><b>Class Name</b></div> <div style="border: 1px solid black; padding: 2px; width: 150px;">Operation01() Operation02()</div>	A class that has operations and may or may not have attributes
<div style="border: 1px solid black; padding: 2px; text-align: center; width: 150px;"><b>Class Name</b></div> <div style="border: 1px solid black; padding: 2px; width: 150px;">Attribute01[1] Attribute02[1] Operation01() Operation02()</div>	A class with attributes and operations
<div style="border: 1px solid black; padding: 2px; text-align: center; width: 150px;"><b>Class Name</b></div> <div style="border: 1px solid black; padding: 2px; width: 150px;">Attribute01[1..*] Attribute02[1] Operation01() Operation02()</div>	A class with attributes that have a specified multiplicity (see table 5) and operations

Table 5 shows the notation used to indicate multiplicity of classes and attributes in class diagrams.

**Table 5 – Multiplicity notation**

Notation <sup>a</sup>	Description
not specified	The number of instances of a class or attribute is not specified.
1	One instance of the class or attribute exists.
0..*	Zero or more instances of the class or attribute exist.
1..*	One or more instances of the class or attribute exist.
0..1	Zero or one instance of the class or attribute exists.
n..m	n to m instances of the class or attribute exist (e.g., 2..8).
x, n..m	Multiple disjoint instances of the class or attribute exist (e.g., 2, 8..15).
<sup>a</sup> See figure 3 and figure 4 for examples of multiplicity notation.	

Table 6 shows the notation used to denote association (i.e., “knows about”) relationships between classes.

Unless the two classes in an association relationship also have an aggregation relationship, association relationships have a multiplicity notation (see table 5) at each end of the relationship line.

**Table 6 – Class diagram notation for associations**

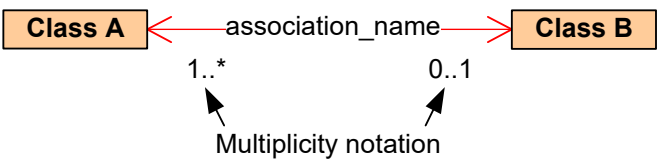
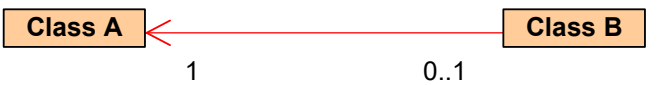
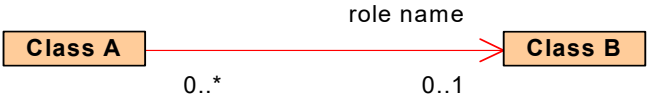
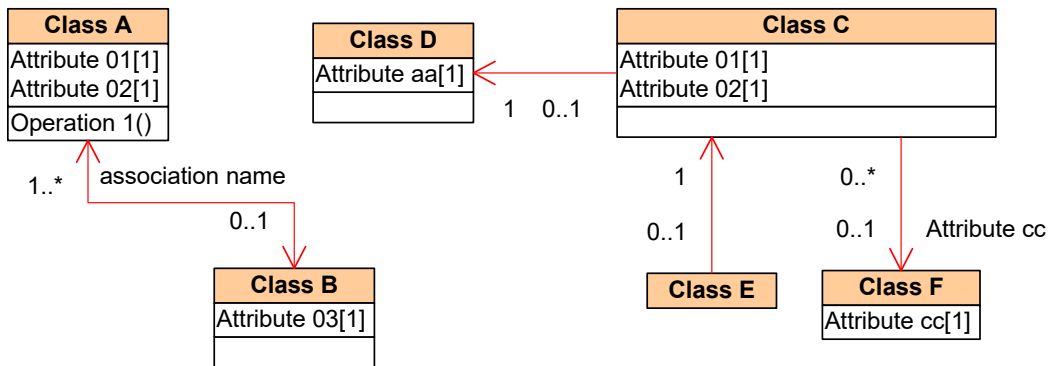
Notation	Description
	Class A knows about Class B (i.e., read as “Class A association_name Class B”) and Class B knows about Class A (i.e., read as “Class B association_name Class A”).
	Class B knows about Class A (i.e., read as “Class B knows about Class A”) but Class A does not know about Class B.
	Class A knows about Class B (i.e., read as “Class A uses the role name attribute of Class B”) but Class B does not know about Class A.
Note - The use of role names and association names are optional.	

Figure 3 shows examples of association relationships between classes.



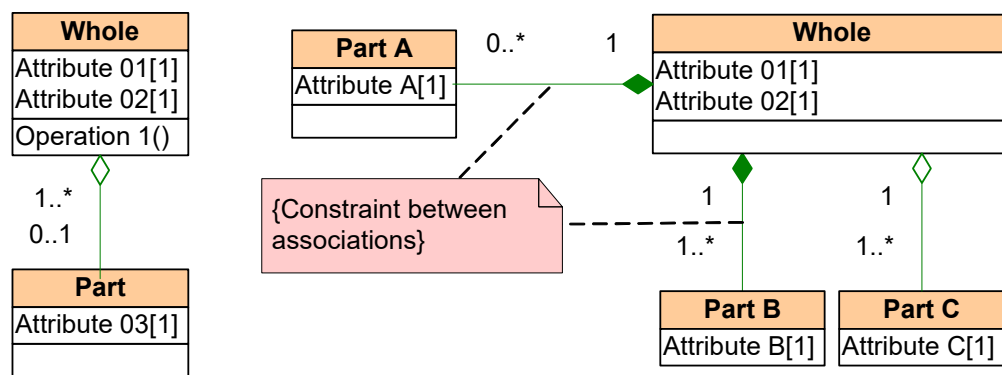
**Figure 3 – Examples of association relationships in class diagrams**

Table 7 shows the notation used to denote aggregation (i.e., “is a part of” or “contains”) relationships between classes. The aggregation relationship is a specific type of association (see table 6) and always include multiplicity notation (see table 5) at each end of the relationship line.

**Table 7 – Class diagram notation for aggregations**

Notation	Description
	The Part class is part of the Whole class (i.e., read as “the whole contains the part”) and may continue to exist even if the Whole class is removed.
	The Part class is part of the Whole class, shall only belong to one Whole class (i.e., read as “the whole contains the part”), and shall not continue to exist if the Whole class is removed.

Figure 4 shows examples of aggregation relationships between classes.



**Figure 4 – Examples of aggregation relationships in class diagrams**

Table 8 shows the notation used to denote generalization (i.e., “is a kind of”) relationships between classes.

**Table 8 – Class diagram notation for generalizations**


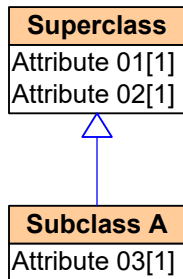
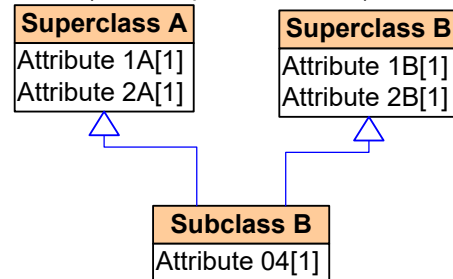
Notation	Description
	Subclass is a kind of superclass. A subclass shares all the attributes and operations of the superclass (i.e., the subclass inherits from the superclass). Inherited attributes are not duplicated in UML drawings.

Figure 5 shows examples of generalization relationships between classes.

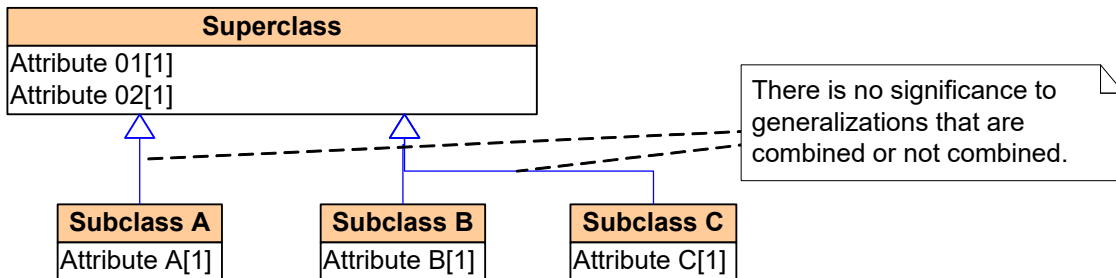
Single superclass/single subclass



Multiple superclasses/single subclass  
(i.e., multiple inheritance)



Single superclass/multiple subclasses



**Figure 5 – Example of generalization relationships in class diagrams**

Table 9 shows the notation used to denote dependency (i.e., “depends on”) relationships between classes.

**Table 9 – Class diagram notation for dependency**

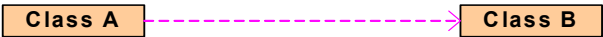
Notation	Description
	Class A depends on class B. A change in class B may cause a change in class A.

Figure 6 shows an example of a dependency relationship between classes.



**Figure 6 – Example of a dependency relationship in class diagrams**

### 3.6.4 Object diagram conventions

Table 10 shows the notation used for objects in object diagrams.

**Table 10 – Object diagram notation for objects**

Notation	Description
<code>label : Class Name</code>	Notation for a named object that may or may not have attributes
<code>label : Class Name</code> Attribute01 = x Attribute02 = y	Notation for a named object with attributes
<code>: Class Name</code>	Notation for an anonymous object that may or may not have attributes
<code>: Class Name</code> Attribute01 = x Attribute02 = y	Notation for a anonymous object with attributes

Table 11 shows the notation used to denote link relationships between objects.

**Table 11 – Object diagram notation for link**

Notation	Description
<code>: Object A</code> — <code>: Object B</code>	An instance of an association between object A and object B.

Figure 7 shows examples of a link relationships between objects.

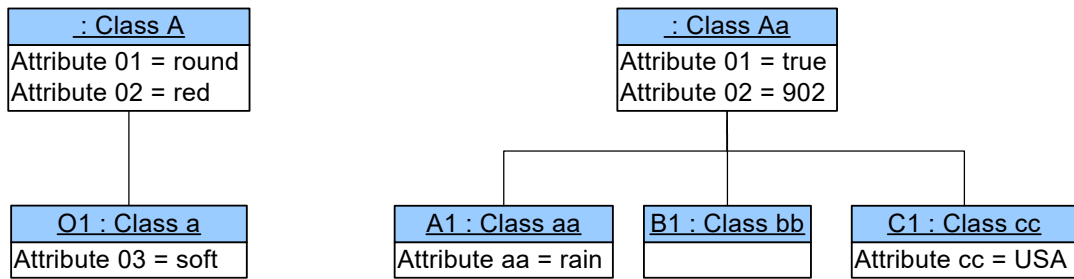
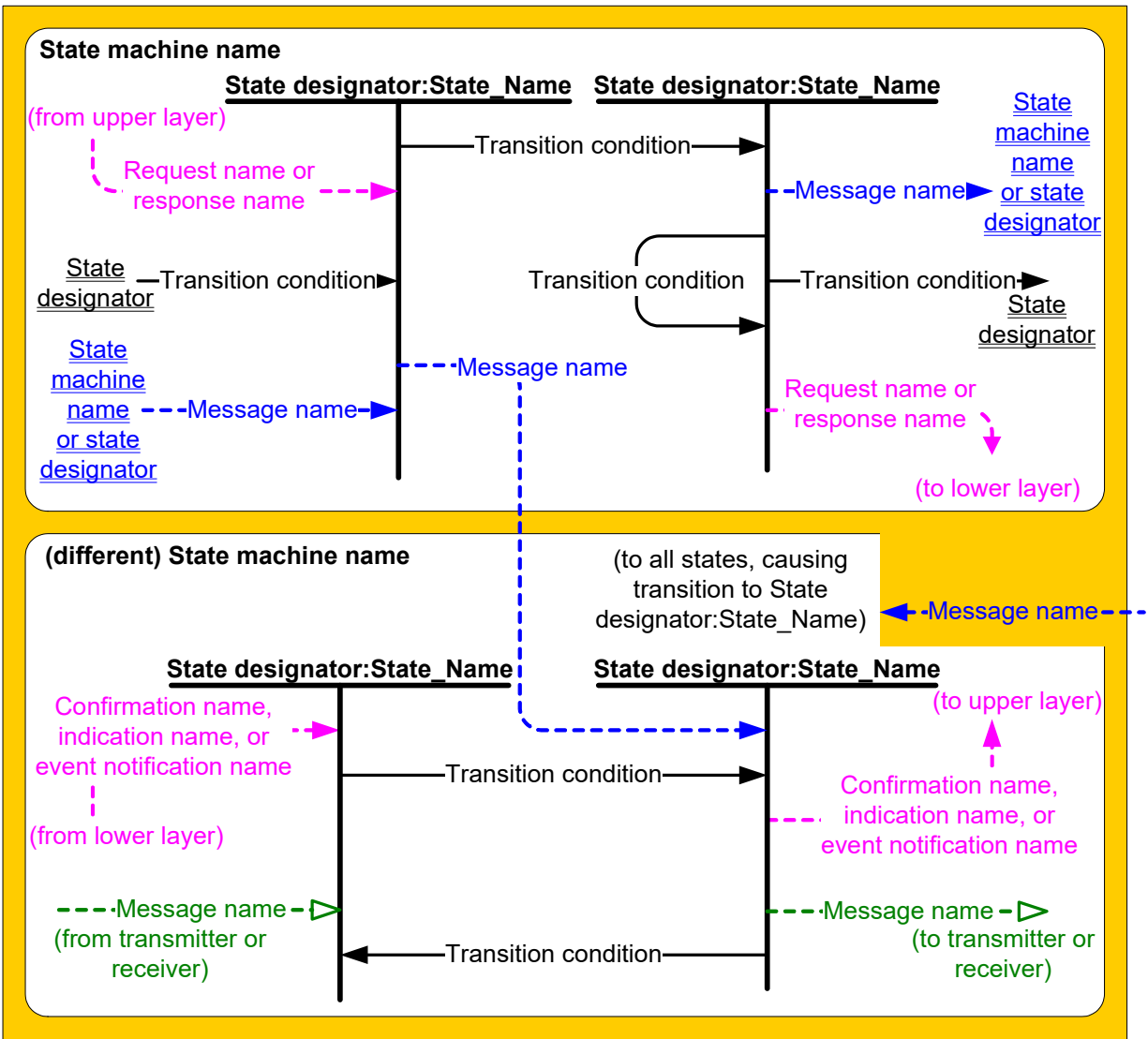


Figure 7 – Examples of link relationships for object diagrams

### 3.7 State machine conventions

#### 3.7.1 State machine conventions overview

Figure 8 shows how state machines are described. See 4.3 for a summary of the state machines in this standard.



**Figure 8 – State machine conventions**

When multiple state machines are present in a figure, they are enclosed in boxes with rounded corners.

Each state machine is identified by a state machine name. In state machines with one state, the state machine is identified by a state designer. In state machines with multiple states, each state is identified by a state designer and a state name. The state designer (e.g., SL1) is unique among all state machines in this standard. The state name (e.g., Idle) is a brief description of the primary action taken during the state and the same state name may be used by other state machines. Actions taken while in each state are described in the state description text.

The definition of the state machine includes an introduction that:

- a) summarizes the states in the state machine;

- b) defines the initial state of the state machine after power on; and
- c) summarizes the state machine counters, timers, and variables (see 3.7.4), if any, used by the state machine.

For each state machine, an overview describes the operating environment (e.g., relationships with other state machines, if the state machine operates in an identified object (see SAM-5) such as the device server).

### 3.7.2 Transitions

Transitions between states are shown with solid lines, with an arrow pointing to the destination state. A transition may be labeled with a transition condition label (i.e., a brief description of the event or condition that causes the transition to occur).

If the state transition in one figure goes to or comes from a state machine or state in a different figure, then the transition is shown going to or coming from a state machine name or a state designator label with double underlines.

The conditions and actions are described fully in the transition description text. In case of a conflict between a figure and the text, the text shall take precedence.

Upon entry into a state, all actions to be processed in that state are processed. If a state is re-entered from itself, then all actions to be processed in the state are processed again. A state may be entered and exited in zero time if the conditions for exiting the state are valid upon entry into the state. Transitions between states are instantaneous.

### 3.7.3 Messages, requests, indications, confirmations, responses, and event notifications

Messages passed between state machines are shown with dashed lines labeled with a message name. When messages are passed between state machines within the same layer of the protocol, they are identified by either:

- a) a dashed line to or from a state machine name label with double underlines and/or state name label with double underlines, if the destination is in a different figure from the source;
- b) a dashed line to or from a state in another state machine in the same figure; or
- c) a dashed line from a state machine name label with double underlines to a "(to all states)" label, if the destination is every state in the state machine.

The meaning of each message is described in the state description text.

Requests, indications, confirmations, responses, and event notifications are shown with curved dashed lines originating from or going toward the top or bottom of the figure. Each request, indication, confirmation, response, and event notification is labeled. The meaning of each request, indication, confirmation, response, and event notification is described in the state description text.

Messages with unfilled arrowheads are passed to or from the state machine's transmitter or receiver, not shown in the state machine figures, and are directly related to data being transmitted on or received from the physical link.

The state machine description text for each state wholly defines the messages sent while the state machine is in that state. If a state is repeatedly sending a message transitions to another state, then that state stops sending that message before making the transition.

### 3.7.4 State machine counters, timers, and variables

State machines may contain counters, timers, and variables that affect the operation of the state machine. The following properties apply to counters, timers, and variables:

- a) their scope is the state machine itself;
  - b) they are created and deleted within the state machines with which they are associated;
  - c) their initialization and modification is specified in the state descriptions and the transition descriptions;
- and



- d) their current values may be used to determine the behavior of a state and select the transition out of a state.

State machine timers may continue to run while a state machine is in a given state and a timer may cause a state transition upon reaching a defined threshold value (e.g., zero for a timer that counts down).

### 3.7.5 State machine arguments

State machines may contain one or more arguments received in a message or confirmation as state machine arguments. The following properties apply to state machine arguments:

- a) the state machine that sends an argument owns that argument's value;
- b) the state machine that receives an argument shall not modify that argument's value;
- c) the state machine that sends an argument may resend that argument with a different value;
- d) the scope of a state machine argument is the state machine itself; and
- e) state machine argument usage is described in the state descriptions and the transition descriptions.

## 3.8 Bit and byte ordering

In a field in a table consisting of more than one bit that contains a single value (e.g., a number), the least significant bit (LSB) is shown on the right and the most significant bit (MSB) is shown on the left (e.g., in a byte, bit 7 is the MSB and is shown on the left and bit 0 is the LSB and is shown on the right). The MSB and LSB are not labeled if the field consists of 8 or fewer bits.

In a field in a table consisting of more than one byte that contains a single value (e.g., a number), the byte containing the MSB is stored at the lowest address and the byte containing the LSB is stored at the highest address (i.e., big-endian byte ordering). The MSB and LSB are labeled.

NOTE 3 - SATA numbers bits within fields the same as this standard, but uses little-endian byte ordering.

In a field in a table consisting of more than one byte that contains multiple fields each with their own values (e.g., a descriptor), there is no MSB and LSB of the field itself and thus there are no MSB and LSB labels. Each individual field has an MSB and LSB, but they are not labeled.

In a field containing a text string (e.g., ASCII or UTF-8), the MSB label is the MSB of the first character, and the LSB label is the LSB of the last character.

Multiple byte fields are represented by only two rows, where a non-sequentially increasing byte number indicates the presence of additional bytes.

A data dword consists of 32 bits. Table 12 shows a data dword containing a single value, where the MSB is on the left in bit 31, and the LSB is on the right in bit 0.

**Table 12 – Data dword containing a value**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
MSB																Value																LSB

Table 13 shows a data dword containing four one-byte fields, where byte 0 (the first byte) is on the left and

byte 3 (the fourth byte) is on the right. Each byte has an MSB on the left and an LSB on the right.

**Table 13 – Data dword containing four one-byte fields**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSB      Byte 0 (First byte)      LSB								MSB      Byte 1 (Second byte)      LSB								MSB      Byte 2 (Third byte)      LSB								MSB      Byte 3 (Fourth byte)      LSB							

### 3.9 Notation for procedures and functions

In this standard, the model for functional interfaces between objects is the callable procedure. Such interfaces are specified using the following notation:

**[Result =] Procedure Name (IN ([input-1] [,input-2] ...), OUT ([output-1] [,output-2] ...))**

where:

Result	A single value representing the outcome of the procedure or function.
Procedure Name	A descriptive name for the function to be performed.
IN (Input-1, Input-2, ...)	A comma-separated list of names identifying caller-supplied input data objects.
OUT (Output-1, Output-2, ...)	A comma-separated list of names identifying output data objects to be returned by the procedure.
[ ... ]	Brackets enclose optional or conditional parameters and arguments.

This notation allows data objects to be specified as inputs and outputs. An interface between entities may require only inputs. If a procedure call has no output arguments, then the word OUT, preceding comma, and associated pair of balanced parentheses are omitted.

## 4 General

### 4.1 Architecture

#### 4.1.1 Architecture overview

A SAS domain (see 4.1.9) contains one or more SAS devices and a service delivery subsystem. A SAS domain may be a SCSI domain (see SAM-5).

A SAS device (see 4.1.6) contains one or more SAS ports (see 4.1.4). A SAS device may be a SCSI device (see SAM-5).

A SAS port (see 4.1.4) contains one or more phys (see 4.1.2). A SAS port may be a SCSI port (see SAM-5).

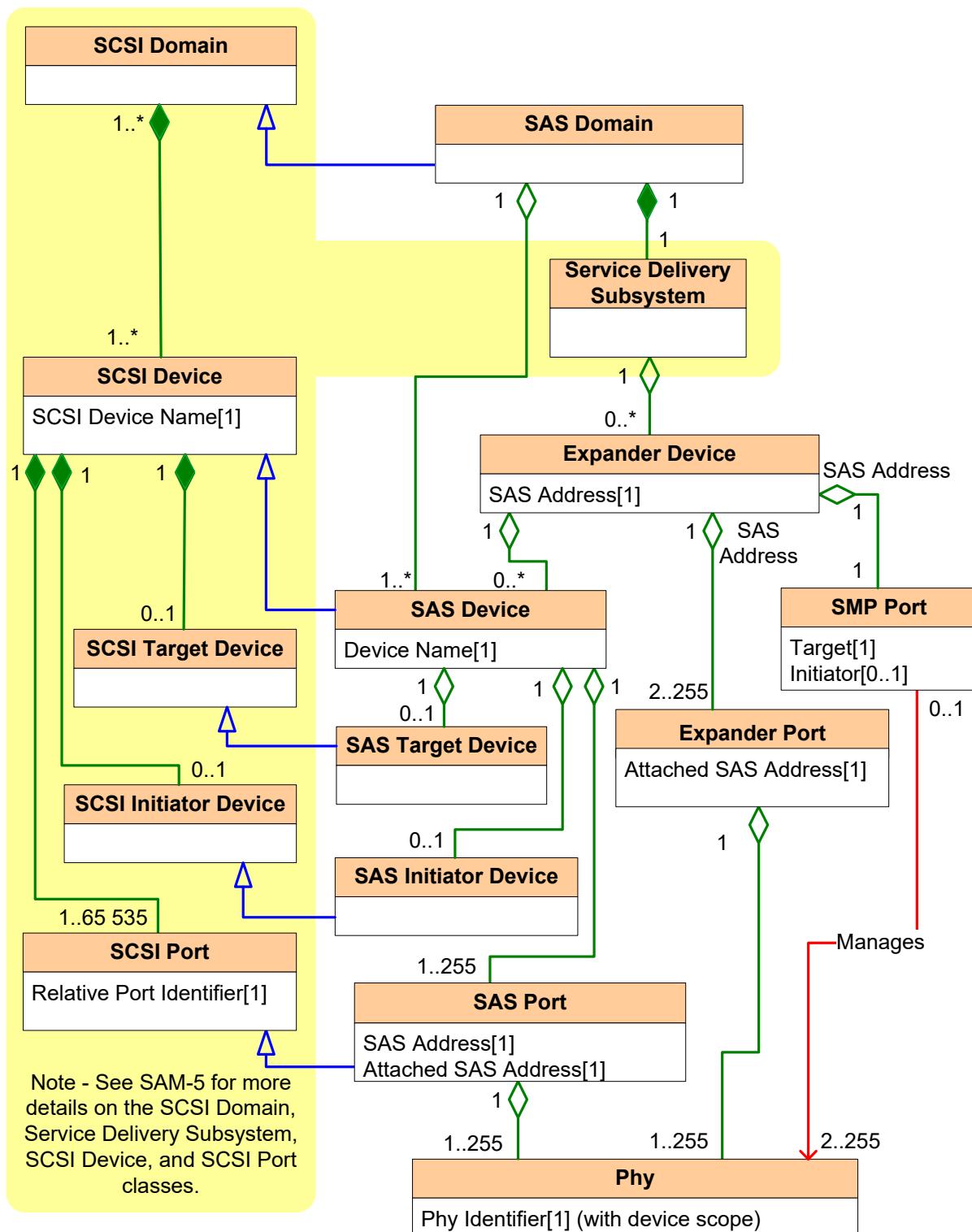
The service delivery subsystem (see 4.1.8) in a SAS domain may contain expander devices (see 4.1.7).

Expander devices contain two or more expander ports (see 4.1.4) and one SMP port.

An expander port contains one or more phys (see 4.1.2).

An expander device shares its phys with the SAS devices contained within the expander device.

Figure 9 shows the class diagram for a SAS domain, showing the relationships between SAS Domain, SCSI Domain, Service Delivery Subsystem, Expander Device, Expander Port, SAS Device, SAS Target Device, SAS Initiator Device, SCSI Device, SCSI Target Device, SCSI Initiator Device, SAS Port, SMP Port, SCSI Port, and Phy classes. Not all attributes are shown.



**Figure 9 – SAS Domain class diagram**

#### 4.1.2 Physical links and phys

For the electrical and mechanical specifications of the physical interconnect (i.e., physical link, physical layer, and physical phy), see SAS-4.

A device (i.e., a SAS device (see 4.1.6) or expander device (see 4.1.7)) contains one or more phys.

Each phy has:

- a) a SAS address (see 4.2.4), inherited from the SAS port (see 4.1.4) or expander device;
- b) a phy identifier (see 4.2.10) that is unique within the device;
- c) optionally, support for being an SSP initiator phy;
- d) optionally, support for being an STP initiator phy;
- e) optionally, support for being an SMP initiator phy;
- f) optionally, support for being an SSP target phy;
- g) optionally, support for being an STP target phy; and
- h) optionally, support for being an SMP target phy.

A phy may be used as one or two logical phys based on multiplexing (see 5.20).

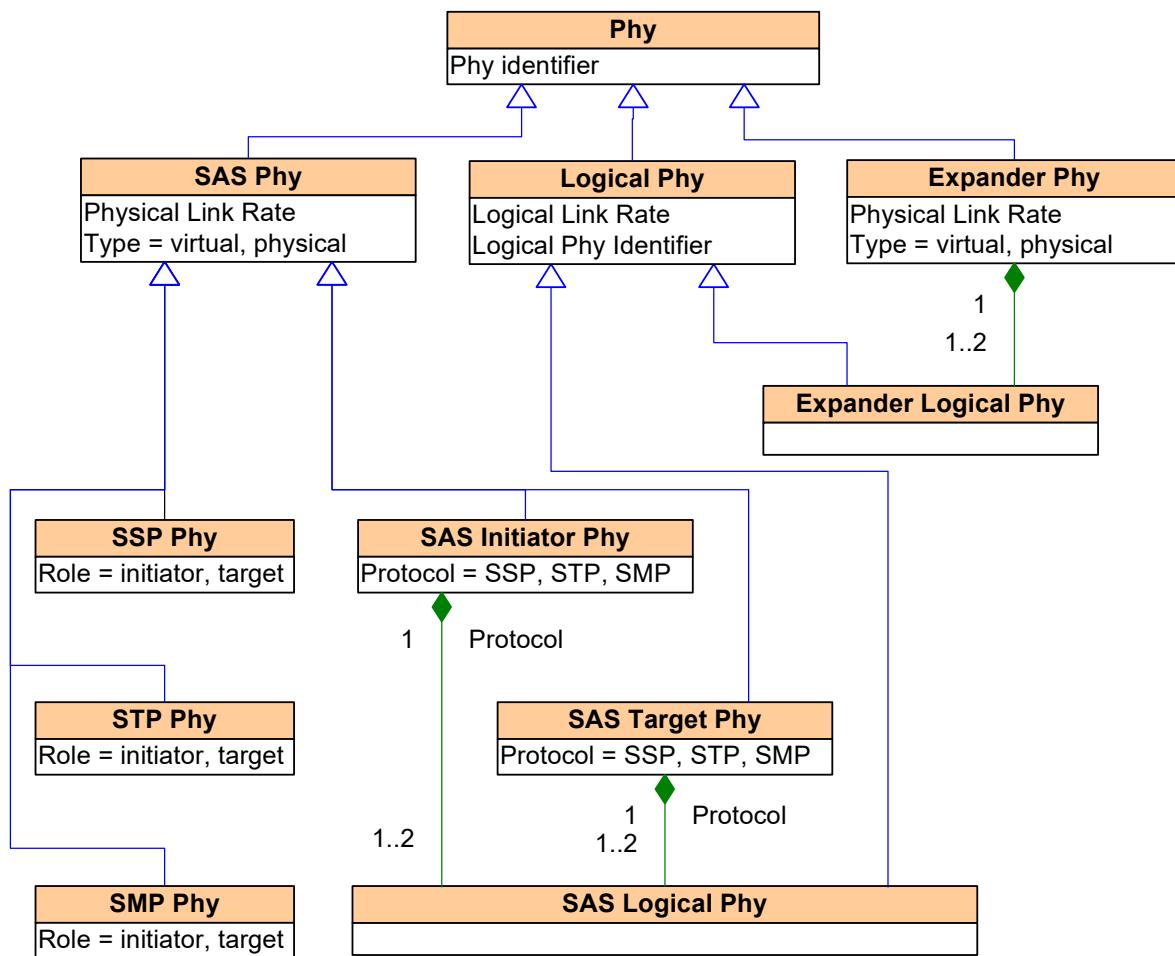
During the identification sequence (see 6.11), a logical phy:

- a) transmits an IDENTIFY address frame including the SAS device type (i.e., end device or expander device) of the device containing the phy, the SAS address of the SAS port or expander device containing the logical phy, the device name, and other information; and
- b) receives an IDENTIFY address frame containing the same set of information from the attached logical phy, including the attached SAS device type, the attached SAS address, the attached device name, and other attached information.

Figure 10 defines the Phy classes, showing the relationships between the following classes:

- a) Phy;
- b) Logical Phy;
- c) SAS Phy;
- d) SAS Logical Phy;
- e) Expander Phy;
- f) Expander Logical Phy;
- g) SAS Initiator Phy;
- h) SAS Target Phy;
- i) SSP Phy;
- j) STP Phy; and
- k) SMP Phy.

SATA phys are also referenced in this standard but are defined by SATA.



**Figure 10 – Phy class diagram**

Figure 11 shows example objects instantiated from the SAS Phy class, including:

- SSP initiator phy;
- SSP target phy;
- virtual SMP initiator phy;
- virtual SMP target phy; and
- SAS logical phy.

A SAS phy is represented by one of these objects during each connection. A SAS phy may be represented by different phy objects in different connections.

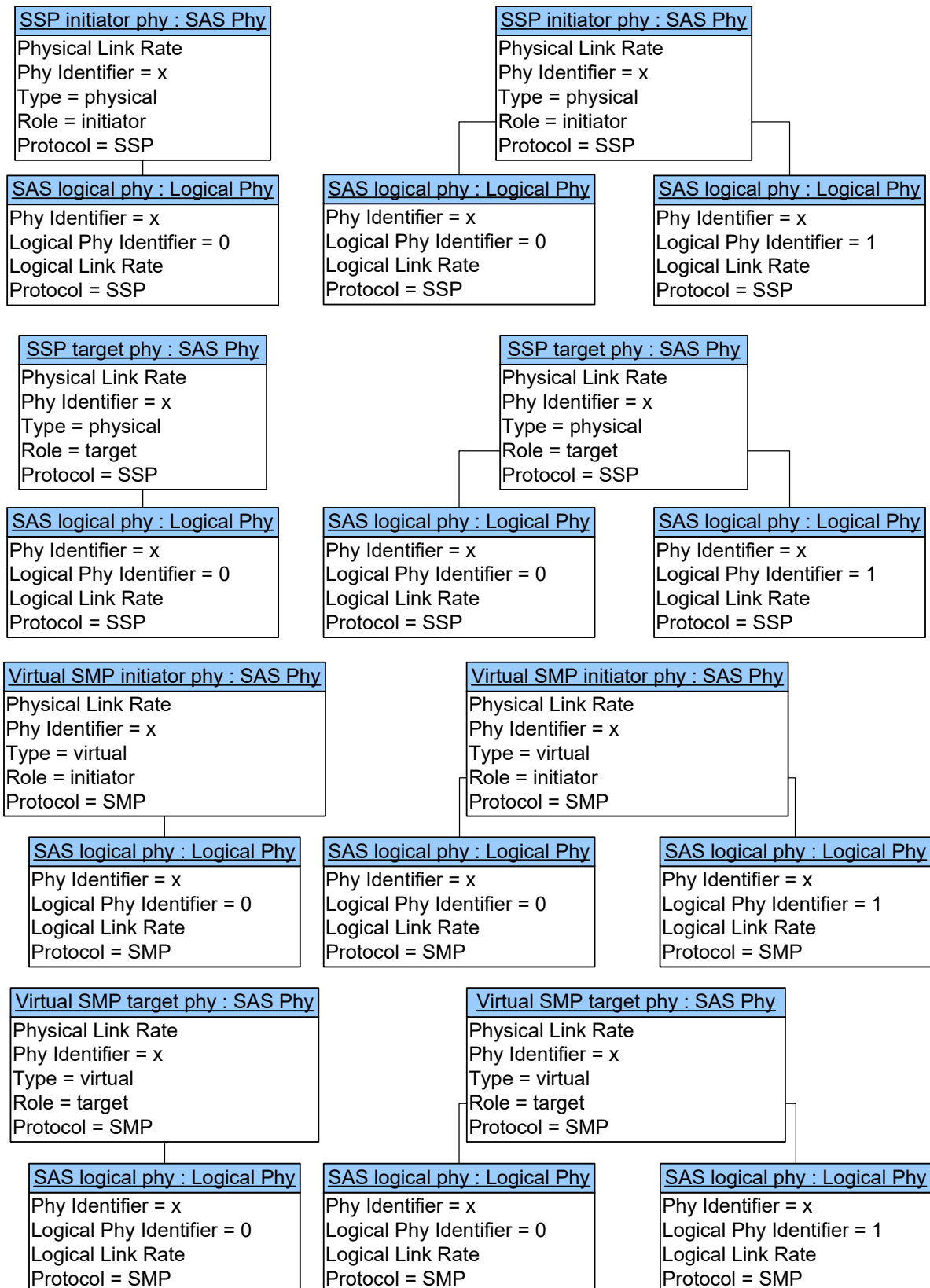


Figure 11 – SAS phy object diagram

Figure 12 shows the objects instantiated from the Expander Phy class, including:

- a) expander phy;
- b) virtual expander phy; and
- c) expander logical phy.

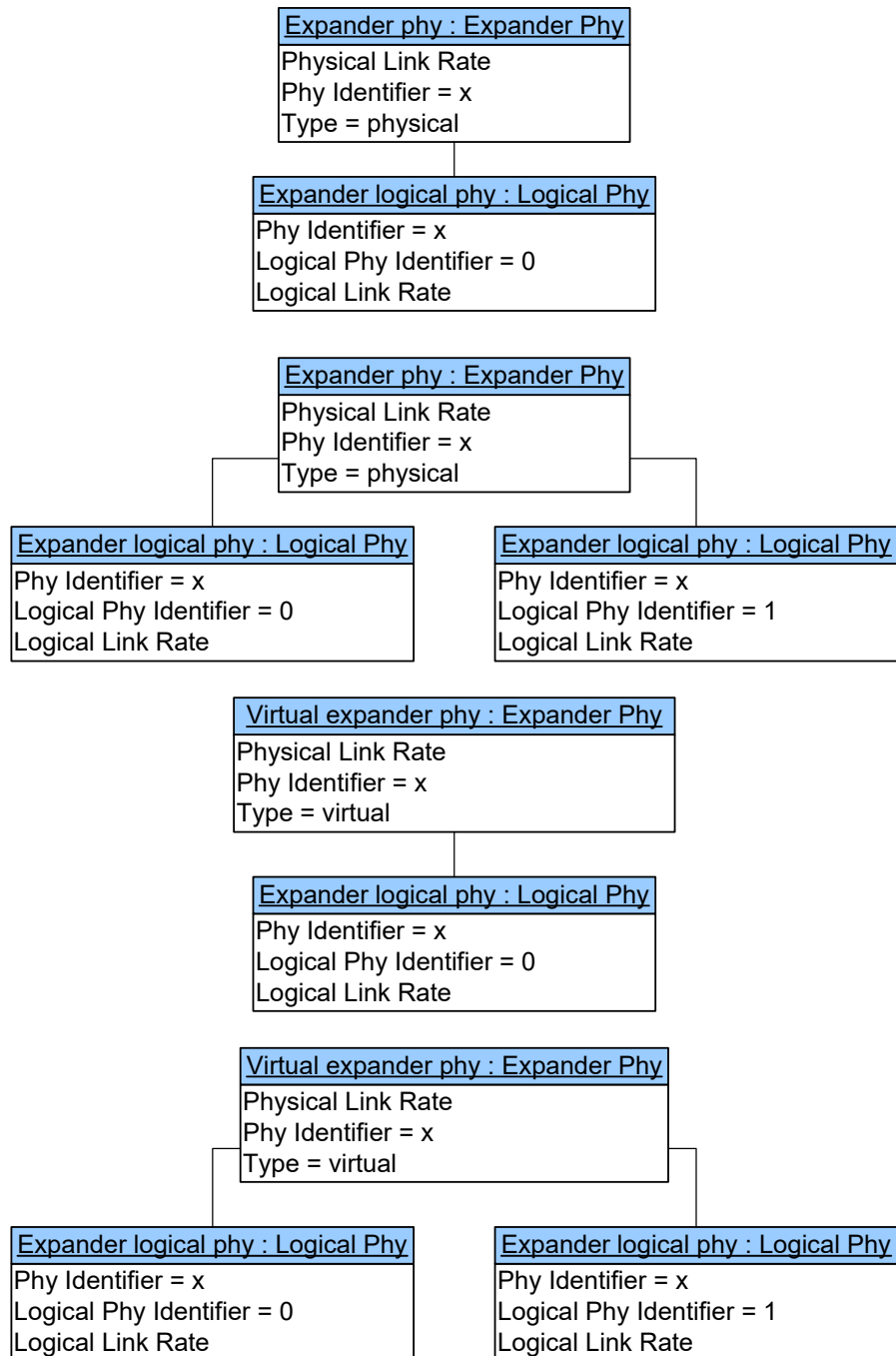


Figure 12 – Expander phy object diagram



### 4.1.3 Logical links

A physical link with a physical link rate greater than 1.5 Gbit/s and less than 12 Gbit/s may be multiplexed into two logical links as defined in table 14.

**Table 14 – Logical links**

Physical link rate	Logical links
6 Gbit/s	One 6 Gbit/s logical link
	Two 3 Gbit/s logical links
3 Gbit/s	One 3 Gbit/s logical link
	Two 1.5 Gbit/s logical links
1.5 Gbit/s	One 1.5 Gbit/s logical link

Multiplexing is defined in 5.20.

### 4.1.4 Narrow ports and wide ports

A port contains one or more phys. Ports in a device are associated with physical phys based on the identification sequence (see 6.11). Ports are associated with virtual phys based on the design of the device.

A port is a narrow port if there is only one phy in the port.

A port is a wide port if there is more than one phy in the port.

A narrow port is created after transmitting and receiving SAS addresses, unless a wide port is created.

A wide port is created from two or more physical phys if, during the identification sequence (see 6.11), the phys:

- a) transmitted the same SAS address (see 4.2.4) that the other physical phys in that port also transmitted in their outgoing IDENTIFY address frames (i.e., the SAS address is the same); and
- b) received the same SAS address that the other physical phys in that port also received in their incoming IDENTIFY address frames (i.e., the attached SAS address is the same).

A narrow link is the physical link that attaches a narrow port to another narrow port. A wide link is the set of physical links that attach a wide port to another wide port.

Attaching a phy within a wide port to another phy in the same port (i.e., the SAS address transmitted in the outgoing IDENTIFY address frame is the same as the SAS address received in the incoming IDENTIFY address frame) is outside the scope of this standard.

Phys that are able to become part of the same wide port shall set the following bits and fields in the IDENTIFY address frame (see 6.10.2) transmitted during the identification sequence to the same set of values on each phy:

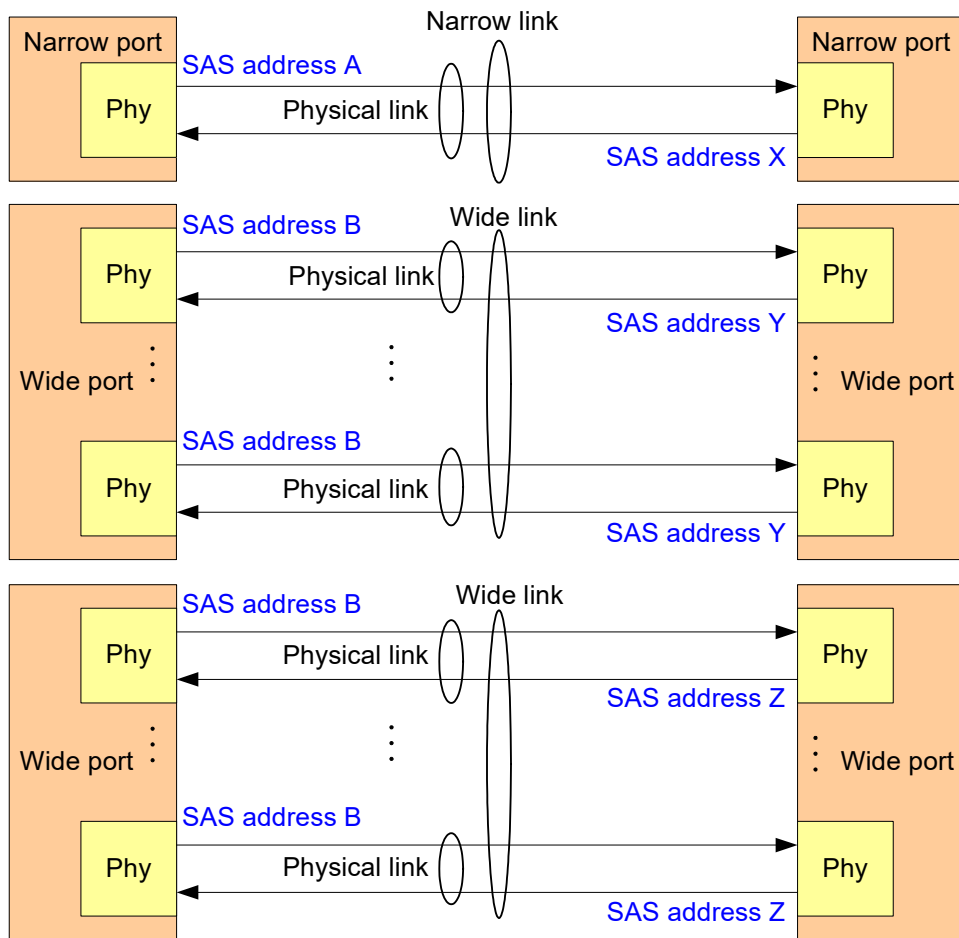
- a) SAS DEVICE TYPE field;
- b) BREAK\_REPLY CAPABLE bit;
- c) SSP INITIATOR PORT bit;
- d) STP INITIATOR PORT bit;
- e) SMP INITIATOR PORT bit;
- f) SSP TARGET PORT bit;
- g) STP TARGET PORT bit;
- h) SMP TARGET PORT bit;
- i) SAS ADDRESS field;

- j) INSIDE ZPSDS PERSISTENT bit;
- k) REQUESTED INSIDE ZPSDS bit;
- l) SLUMBER CAPABLE bit;
- m) PARTIAL CAPABLE bit;
- n) POWER CAPABLE bit;
- o) PERSISTENT CAPABLE bit;
- p) PWR\_DIS CAPABLE bit; and
- q) SMP PRIORITY CAPABLE bit.

Recipient wide ports are not required to check the consistency of the IDENTIFY address frames fields across the phys within a recipients wide port.

Each phy in a port may be in a different phy power condition (see 4.10).

Figure 13 shows examples of narrow ports and wide ports, with a representation of the SAS address transmitted during the identification sequence. Although several phys on the left transmit SAS addresses of B, only phys attached to the same SAS addresses become part of the same ports. The set of phys with SAS address B attached to the set of phys with SAS address Y become one port, while the set of phys with SAS address B attached to the set of phys with SAS address Z become another port.



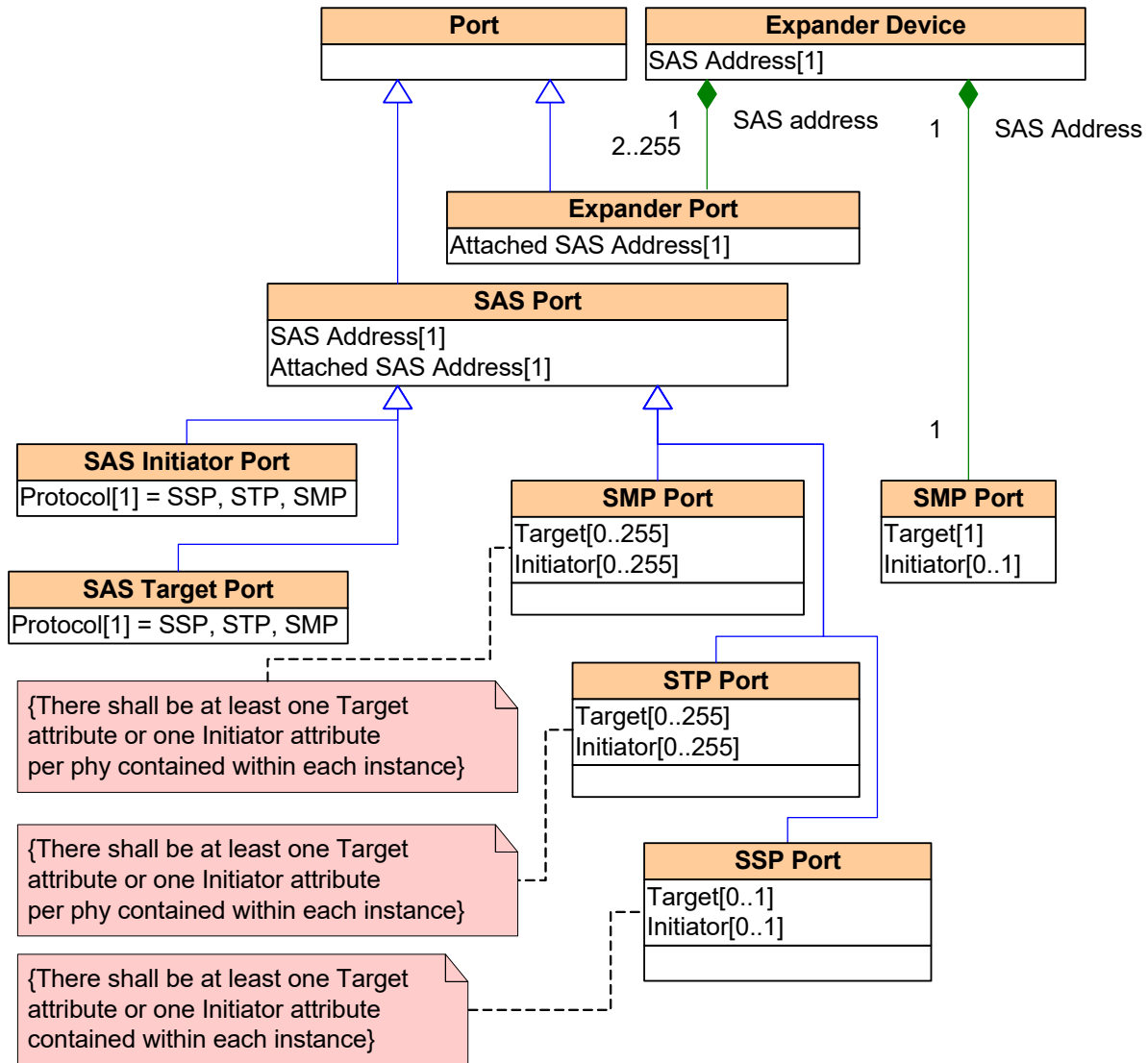
Each horizontal line represents a differential signal pair

**Figure 13 – Ports (narrow ports and wide ports)**

Figures in this standard that show ports but not phys, the phy level of detail is not shown, however, each port always contains one or more phys.

Figure 14 defines the Port classes, showing the relationships between the following classes:

- a) Port;
- b) Expander Device;
- c) Expander Port;
- d) SAS Port;
- e) SAS Initiator Port;
- f) SAS Target Port;
- g) SSP Port;
- h) STP Port; and
- i) SMP Port.



**Figure 14 – Port class diagram**

Figure 15 shows the objects instantiated from the Port classes:

- a) SAS Target Port class (i.e., SSP target port, STP target port, SMP target port);
- b) SAS Initiator Port class (i.e., SSP initiator port, STP initiator port, SMP initiator port);
- c) STP Port class (i.e., STP initiator port, STP target port, STP port);
- d) SMP Port class (i.e., SMP initiator port, SMP target port, SMP port);
- e) SSP Port class (i.e., SSP initiator port, SSP target port, SSP port);

- f) Expander Device SMP Port class (i.e., SMP target port, SMP port); and
- g) Expander Port class (i.e., expander port).

Port objects remain instantiated even while no connection is open on any of the phys within the port.

Valid objects for the SAS Initiator Port class	Valid objects for the SAS Target Port class	Valid objects for the Expander Device SMP Port class
<u>SSP initiator port : SAS Initiator Port</u> SAS Address Attached SAS Address Protocol = SSP	<u>SSP target port : SAS Target Port</u> SAS Address Attached SAS Address Protocol = SSP	<u>SMP target port : SMP port</u> SAS Address Target
<u>STP initiator port : SAS Initiator Port</u> SAS Address Attached SAS Address Protocol = STP	<u>STP target port : SAS Target Port</u> SAS Address Attached SAS Address Protocol = STP	<u>SMP port : SMP Port</u> SAS Address Target Initiator
<u>SMP initiator port : SAS Initiator Port</u> SAS Address Attached SAS Address Protocol = SMP	<u>SMP target port : SAS Target Port</u> SAS Address Attached SAS Address Protocol = SMP	Valid object for the Expander Port class <u>Expander port : Expander Port</u> SAS Address Attached SAS Address

Valid objects for the SSP Port class	Examples of valid objects for the STP Port class	Examples of valid objects for the SMP Port class
<u>SSP target port : SSP Port</u> SAS Address Attached SAS Address Target	<u>STP target port : STP Port</u> SAS Address Attached SAS Address Target01 Target02	<u>SMP target port : SMP Port</u> SAS Address Attached SAS Address Target
<u>SSP initiator port : SSP Port</u> SAS Address Attached SAS Address Initiator	<u>STP initiator port : STP Port</u> SAS Address Attached SAS Address Initiator	<u>SMP initiator port : SMP Port</u> SAS Address Attached SAS Address Initiator01 Initiator02
<u>SSP port : SSP Port</u> SAS Address Attached SAS Address Initiator Target	<u>STP port : STP Port</u> SAS Address Attached SAS Address Target01 Target02 Initiator	<u>SMP port : SMP port</u> Attached SAS Address Target01 Target02 Initiator01 Initiator02

Figure 15 – Port object diagram

#### 4.1.5 Application clients and device servers

This standard defines the following application clients:

- a) a SCSI application client (see SAM-5) is the source of SCSI commands and task management function requests. A SCSI application client uses an SSP initiator port to interface to a service delivery subsystem;
- b) an ATA application client (see ATA8-AAM) is the source of ATA commands and device management operation requests. An ATA application client uses an STP initiator port to interface to a service delivery subsystem; and
- c) a management application client is the source of SMP function requests. A management application client uses an SMP initiator port to interface to a service delivery subsystem.

This standard defines the following device servers:

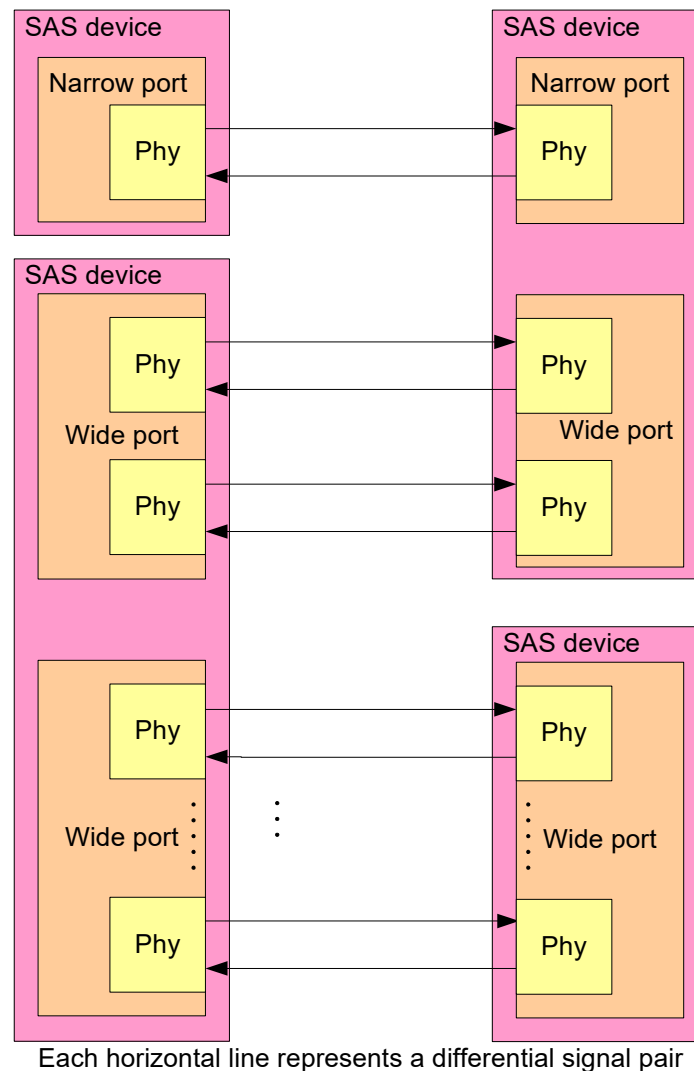
- a) a SCSI device server (see SAM-5) processes SCSI commands. A SCSI device server uses an SSP target port to interface to a service delivery subsystem;
- b) an ATA device server (see ATA8-AAM) processes ATA commands and device management functions. An ATA device server uses an STP target port to interface to a service delivery subsystem; and
- c) a management device server processes SMP functions. A management device server uses an SMP target port to interface to a service delivery subsystem.

A SCSI to ATA translation layer (see SAT-4) may be implemented to enable SCSI application clients to communicate with ATA devices.

#### 4.1.6 SAS devices

A SAS device contains one or more SAS ports, each containing one or more phys (i.e., a SAS port may be a narrow port or a wide port).

Figure 16 shows examples of SAS devices with different port and phy configurations.



**Figure 16 – SAS devices**

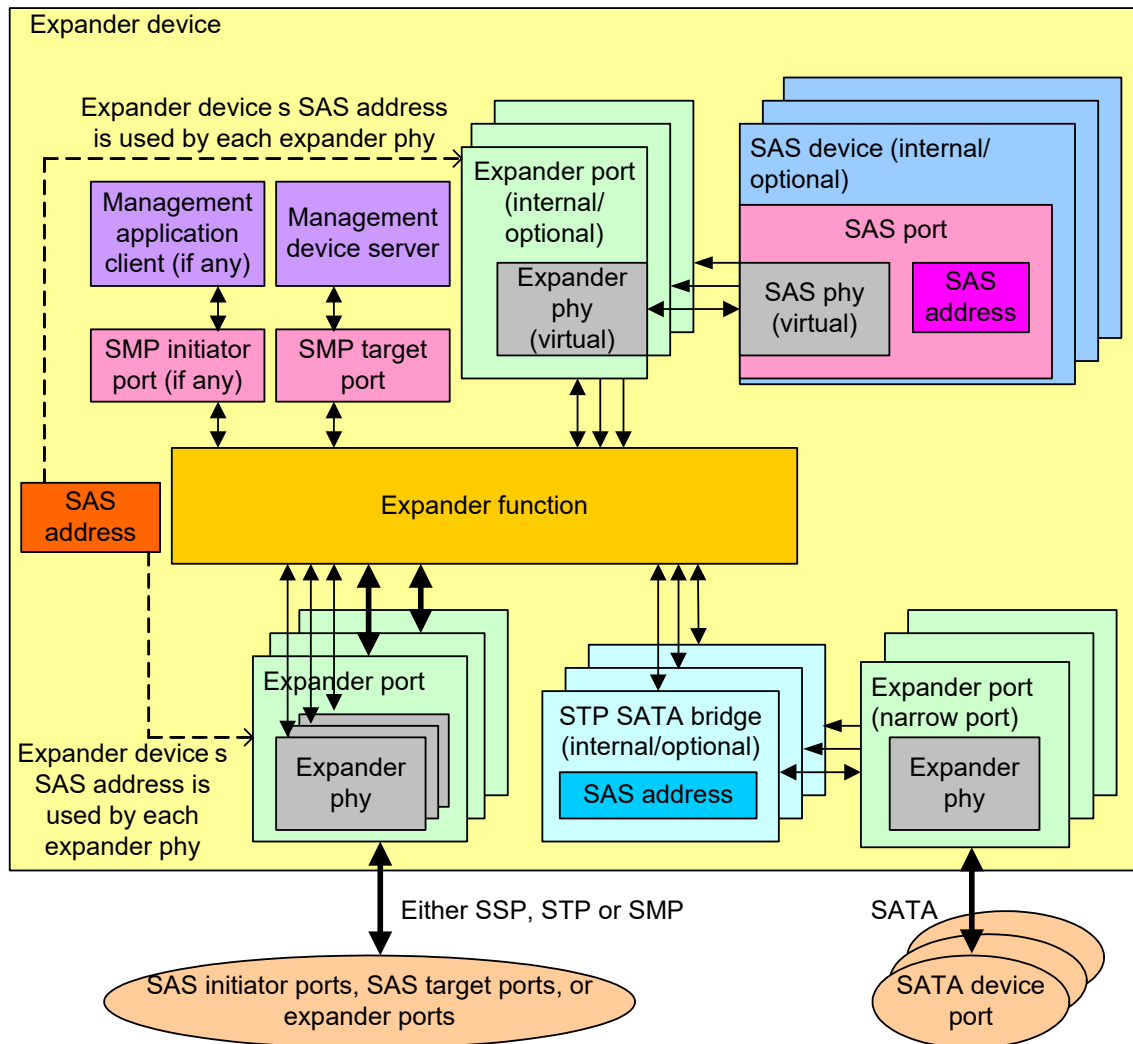
An end device is a SAS device that is not contained in an expander device (see 4.1.7).

#### 4.1.7 Expander devices

Expander devices are part of a service delivery subsystem and facilitate communication between multiple SAS devices. Expander devices contain two or more external expander ports. Each expander device:

- contains one SMP target port and one management device server;
- contains one SMP initiator port and one management application client, if the expander device is self-configuring;
- may contain one SMP initiator port and one management application client, if the expander device is not self-configuring; and
- may contain SAS devices (e.g., an expander device may include an SSP target port for access to a logical unit with a device type of 0Dh (i.e., enclosure services device) (see SPC-4 and SES-3)).

Figure 17 shows an expander device.



**Figure 17 – Expander device**

See 4.5 for a detailed model of an expander device.

Each expander phy has one of the following routing attributes (see 4.5.7.1):

- a) direct routing attribute;
- b) table routing attribute; or
- c) subtractive routing attribute.

Expander devices containing expander phys with the table routing attribute also contain an expander route table (see 4.5.7.4). An externally configurable expander device depends on a management application client within the SAS domain to use the discover process (see 4.6) and the configuration subprocess (see 4.7) to configure the expander route table.

A self-configuring expander device contains

- a) a management application client that performs the discover process (see 4.6) and configures the expander device's own expander route table; and
- b) an SMP initiator port.

#### 4.1.8 Service delivery subsystem

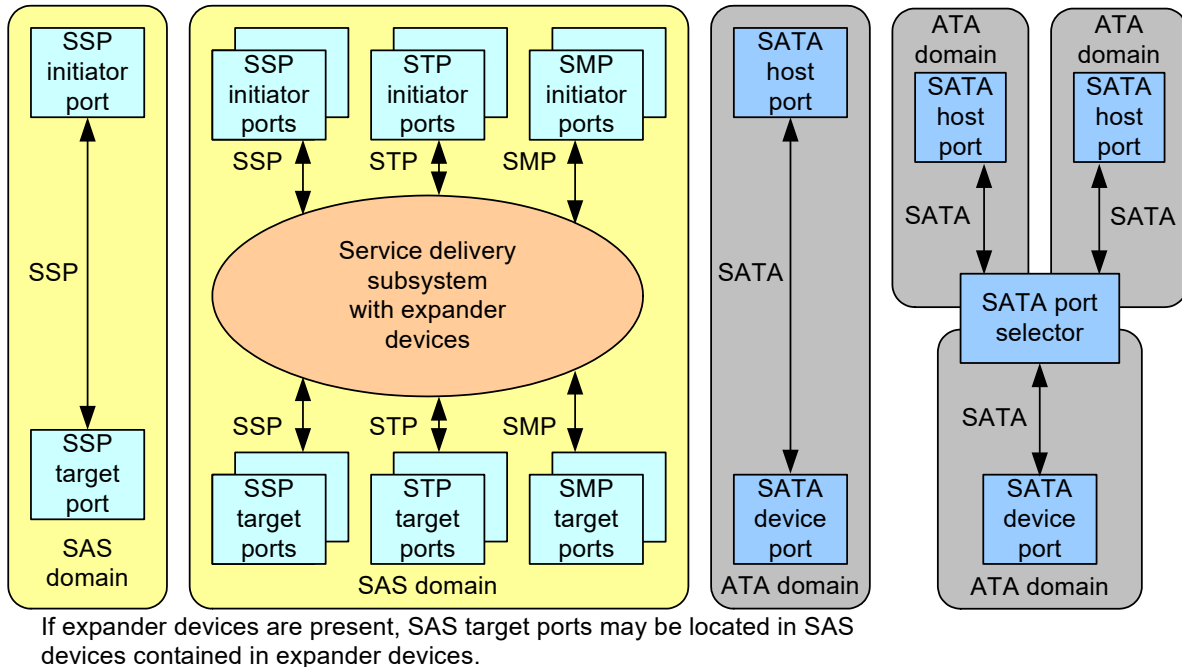
A service delivery subsystem is either:

- a) a set of physical links between a SAS initiator port and a SAS target port; or
- b) a set of physical links and expander devices, supporting more than two SAS ports.

See 4.1.10 for rules on constructing service delivery subsystems from multiple expander devices.

#### 4.1.9 Domains

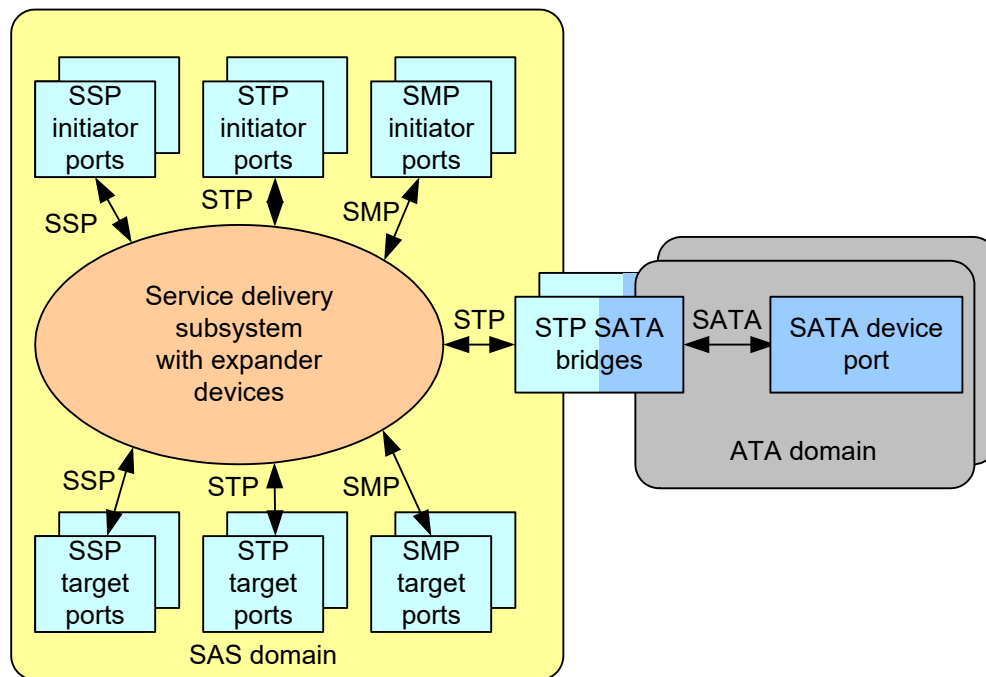
Figure 18 shows examples of SAS domains and ATA domains.



**Figure 18 – Domains**



Figure 19 shows a SAS domain bridging to one or more ATA domains.



**Figure 19 – SAS domain bridging to ATA domains**

Figure 20 shows two SAS domains bridging to one or more ATA domains containing SATA devices with SATA port selectors.

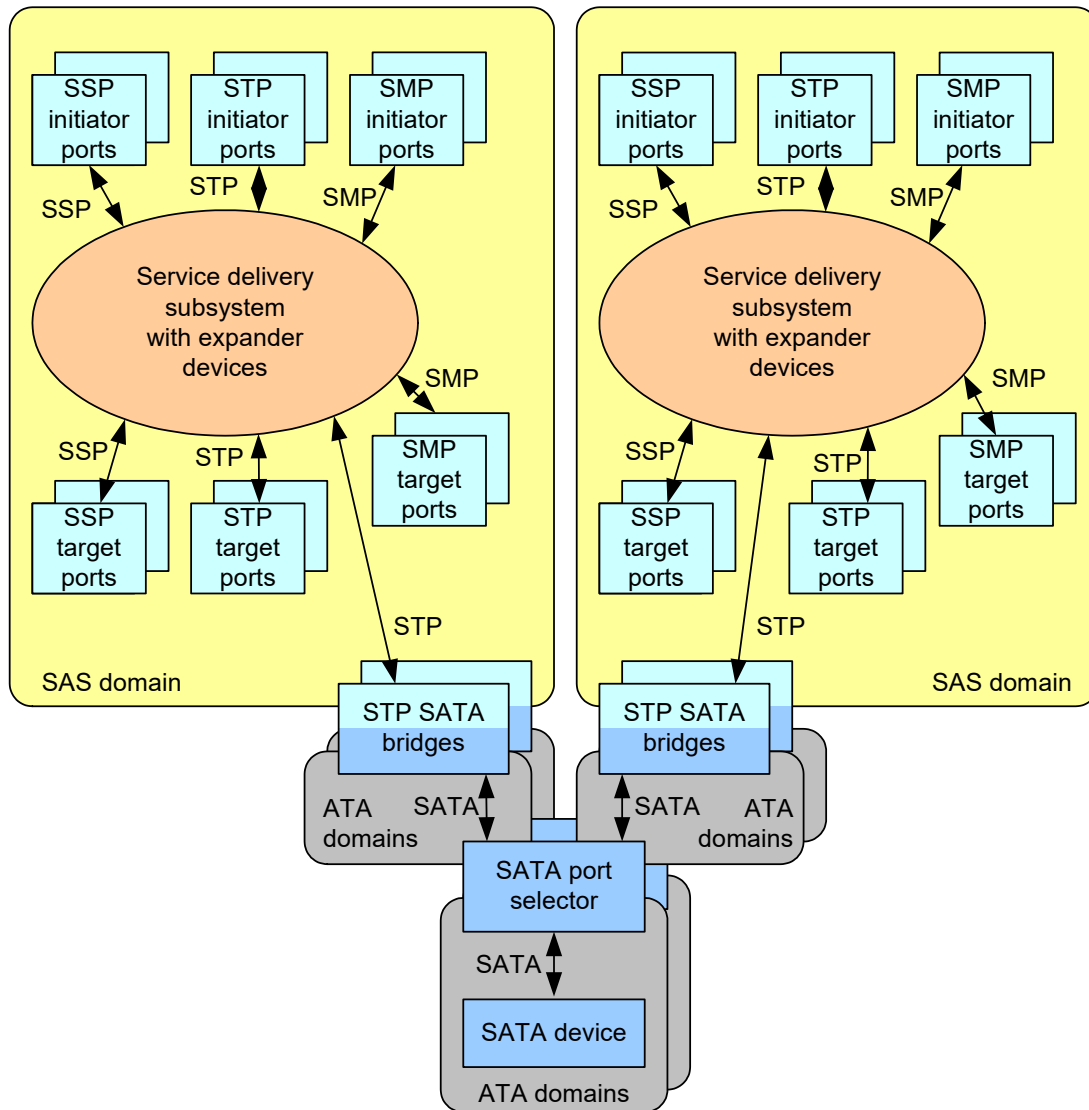
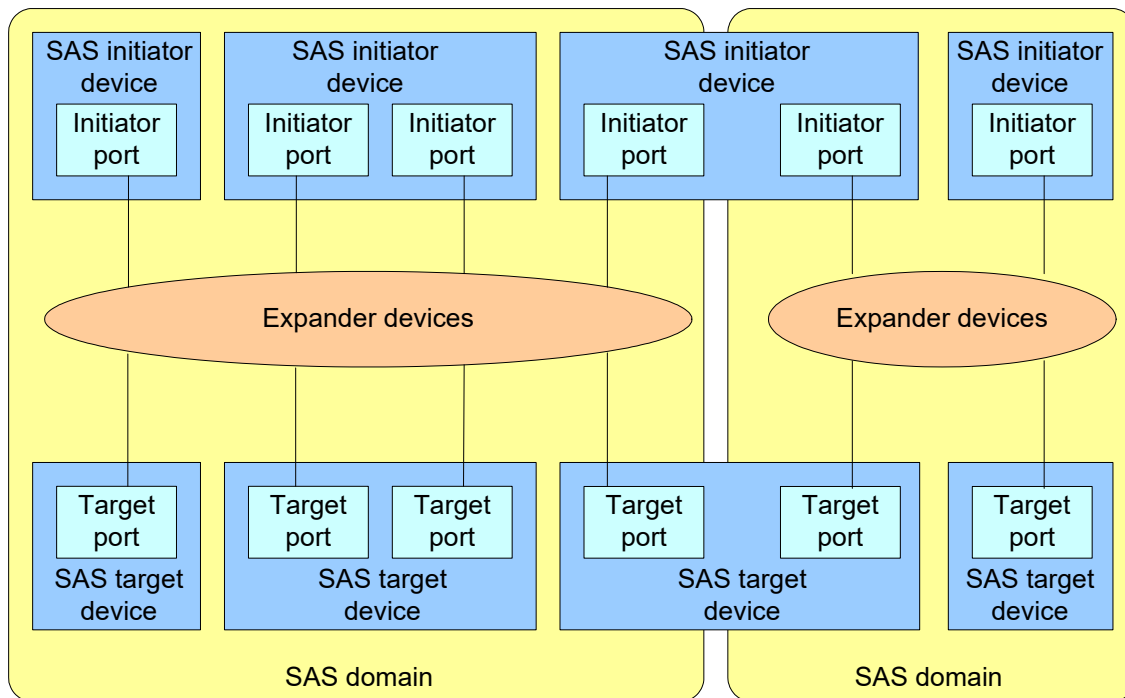


Figure 20 – SAS domains bridging to ATA domains with SATA port selectors

Figure 21 shows SAS initiator devices and SAS target devices with SAS ports in the same SAS domains and in different SAS domains. If a SAS device has ports in the same SAS domain, then the ports shall have different SAS addresses. If a SAS device has ports in different SAS domains, then the ports may have the same SAS address (see 4.2.4).



**Figure 21 – Devices spanning SAS domains**

#### 4.1.10 Expander device topologies

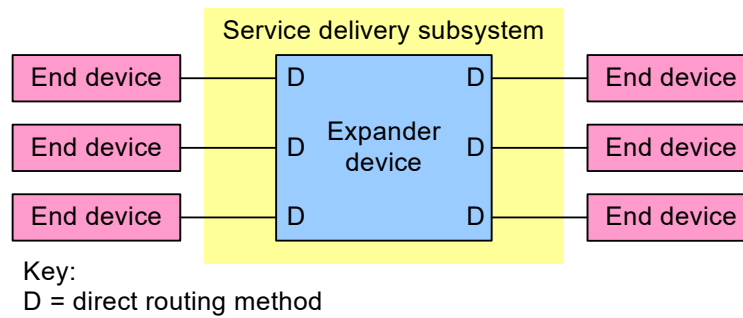
##### 4.1.10.1 Expander device topology overview

More than one expander device may be part of a service delivery subsystem.

To avoid an overflow of an expander route index during the configuration subprocess (see 4.7), a SAS domain containing an externally configurable expander device shall be constructed such that the number of expander route indexes available for each expander phy with the table routing attribute is greater than or equal to the number of SAS addresses addressable through that expander phy.

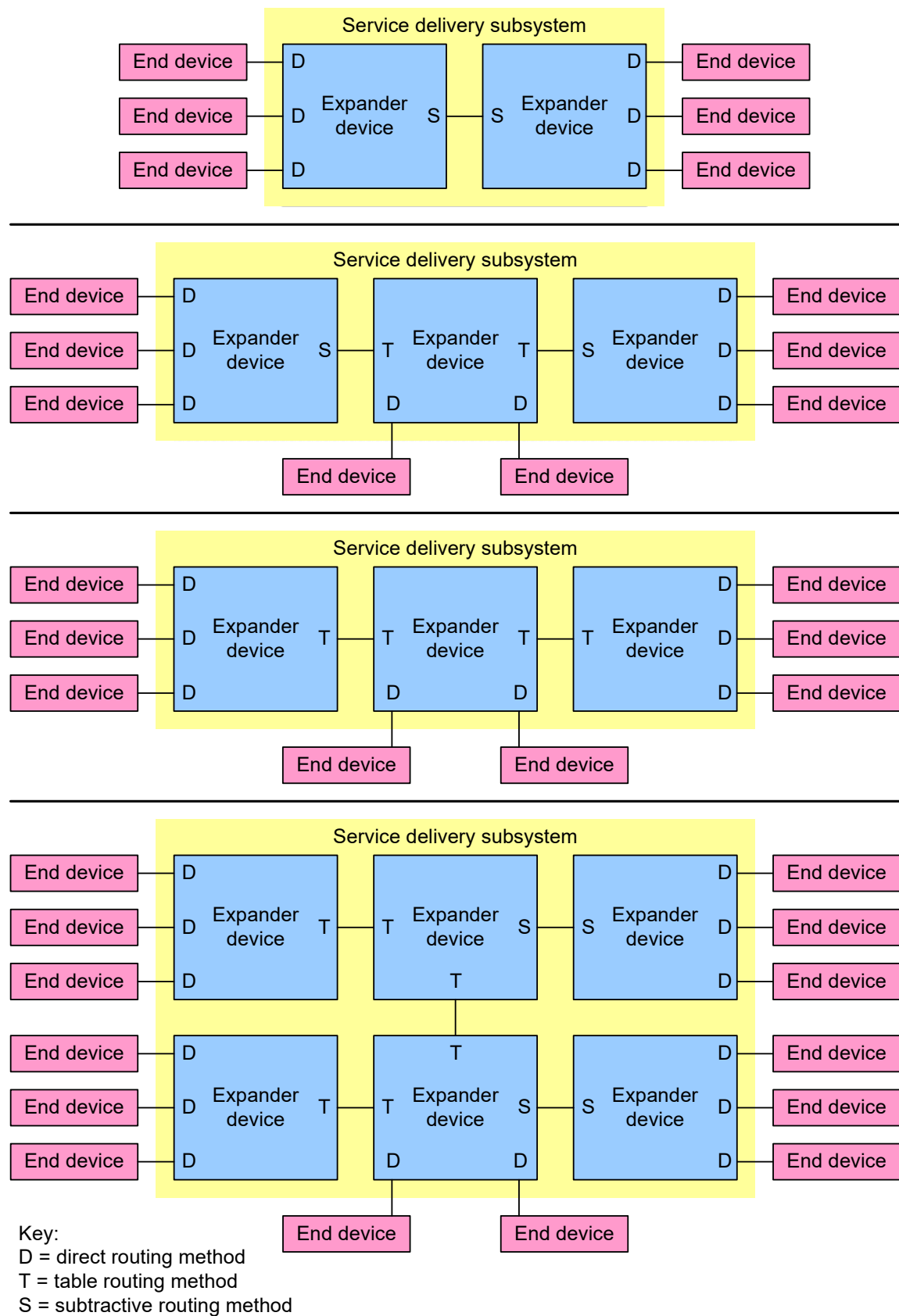
#### 4.1.10.2 Expander device topologies

Figure 22 shows an example of an expander topology with one expander device.



**Figure 22 – Single expander device topology example**

Figure 23 shows examples of expander topologies with multiple expander devices.

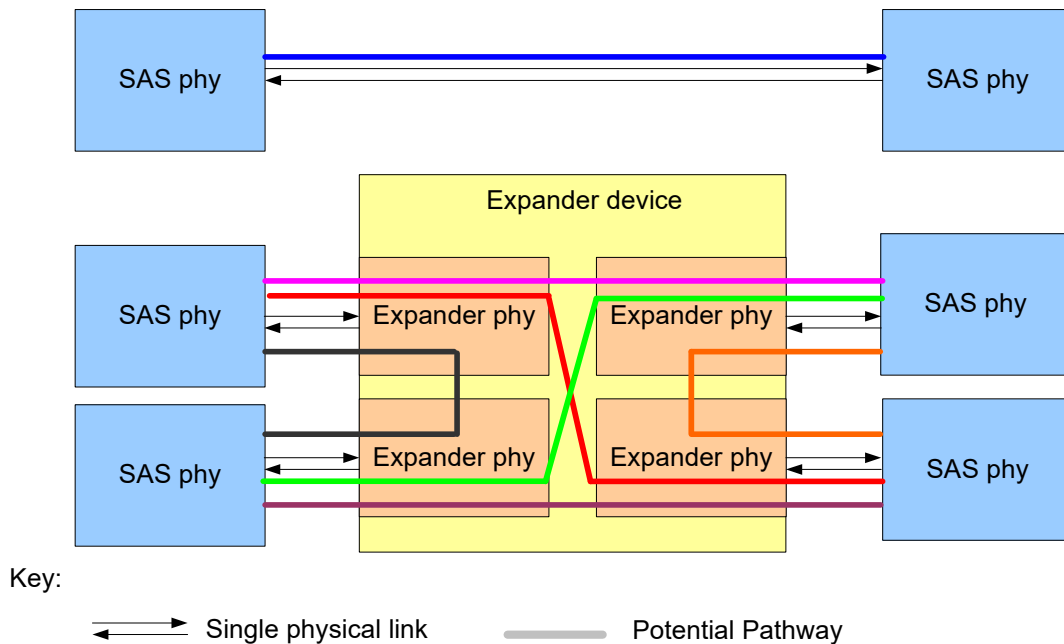


**Figure 23 – Multiple expander device topologies and routing methods**

#### 4.1.11 Pathways

A potential pathway is a set of logical links between a SAS initiator phy and a SAS target phy. If a SAS initiator phy is directly attached to a SAS target phy with a non-multiplexed physical link, then there is one potential pathway. If the physical link is multiplexed or there are expander devices between a SAS initiator phy and a SAS target phy, then it is possible that there is more than one potential pathway, each consisting of a set of logical links between the SAS initiator phy and the SAS target phy. The physical links may or may not be using the same physical link rate.

Figure 24 shows examples of potential pathways.



**Figure 24 – Potential pathways**

A pathway is a set of logical links between a SAS initiator phy and a SAS target phy being used by a connection (see 4.1.12).

A partial pathway is the set of logical links participating in a connection request that have not yet conveyed a connection response (see 6.16).

A partial pathway is blocked while path resources that it requires are held by another partial pathway (see 6.16).

#### 4.1.12 Connections

A connection is a temporary association between a SAS initiator phy and a SAS target phy. During a connection:

- all dwords and SPL packets from the SAS initiator phy that are not deletable primitives, deletable binary primitives, or deletable extended binary primitives are forwarded to the SAS target phy via a pathway; and
- all dwords and SPL packets from the SAS target phy that are not deletable primitives, deletable binary primitives, or deletable extended binary primitives are forwarded to the SAS initiator phy via the same pathway.

A source phy transmits an OPEN address frame (see 6.10.3) specifying the SAS address of a destination phy to attempt to establish a connection.

A connection is pending when an OPEN address frame has been delivered along a completed pathway to the destination phy but the destination phy has not yet responded to the connection request. A connection is established when the source phy receives an OPEN\_ACCEPT (see 6.16) from the destination phy.

A connection enables communication for one protocol:

- a) SSP;
- b) STP; or
- c) SMP.

For SSP and STP, connections may be opened and closed multiple times during the processing of a command (see 6.16).

The connection rate is the effective rate of dwords through the pathway between a SAS initiator phy and a SAS target phy, established through the connection request. Every logical phy shall support a 1.5 Gbit/s connection rate regardless of its logical link rate.

No more than one connection is active on a logical link at a time. If the connection is an SSP or SMP connection:

- a) SAS dword mode is enabled (see 5.8) and there are no dwords to transmit associated with that connection, then idle dwords are transmitted; or
- b) SAS packet mode is enabled (see 5.8) and there are no SPL packets to transmit associated with that connection, then SPL packet payloads containing idle dword segments (see 5.5.6) are transmitted.

If the connection is an STP connection and there are no dwords to transmit associated with that connection, then SATA\_SYNCs, SATA\_CONTs, or vendor specific scrambled data dwords are transmitted as defined in SATA.

If there is no connection on a logical link and:

- a) SAS dword mode is enabled, then idle dwords are transmitted; or
- b) SAS packet mode is enabled, then SPL packet payloads containing idle dword segments are transmitted.

The number of connections established by a SAS port shall not exceed the number of SAS logical phys within the SAS port (i.e., only one connection per SAS logical phy is allowed). There shall be a separate connection on each logical link.

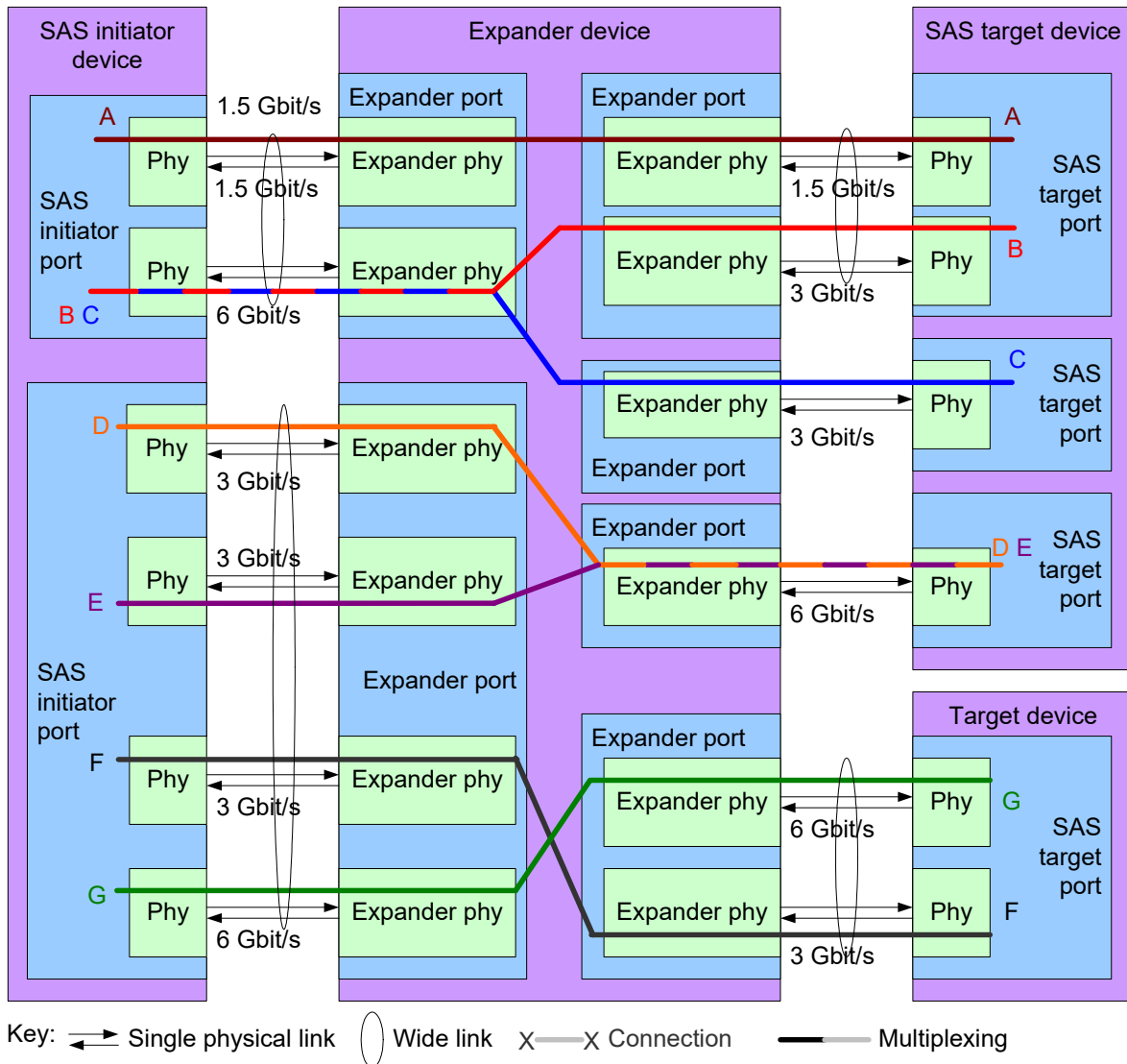
If multiple potential pathways exist between the SAS initiator ports and the SAS target ports, then multiple connections may be established by a SAS port between the following:

- a) one SAS initiator port to multiple SAS target ports;
- b) one SAS target port to multiple SAS initiator ports; or
- c) one SAS initiator port to one SAS target port.

Once a connection is established, the pathway used for that connection shall not be changed (i.e., all the logical links that make up the pathway remain dedicated to the connection until the connection is closed).

Figure 25 shows examples of connections between wide and narrow ports. All the connections shown may occur simultaneously. For the connections shown in figure 25:

- a) those labeled A, B, and C are an example of one SAS initiator port with connections to multiple SAS target ports;
- b) those labeled E and F are an example of multiple connections between one SAS initiator port and multiple SAS target ports; and
- c) those labeled C, D, E, and F are an example of one SAS initiator port with connections to multiple SAS target ports with one of those SAS target ports having multiple connections with that SAS initiator port.



Note - The expander device, each SAS initiator port, and each SAS target port has a unique SAS address. Connections D and E represent a wide SAS initiator port with two simultaneous connections to a narrow SAS target port supporting multiplexing. Connections F and G represent a wide SAS initiator port with two simultaneous connections to a wide SAS target port.

**Figure 25 – Multiple connections on wide ports**

#### 4.1.13 Persistent connections

##### 4.1.13.1 Persistent connection operation

A persistent connection is an SSP connection (see 4.1.12) that:

- is established after an EXTEND\_CONNECTION (NORMAL) (see 6.2.7.5) has been transmitted and received inside the SSP connection;
- persists as long as the connected SAS initiator phy and SAS target phy:
  - transmit SSP frames; or
  - periodically exchange EXTEND\_CONNECTION (NORMAL)s (see 6.20.9.12);
- causes the PL\_PM state machine to ignore the Bus Inactivity Time Limit timer, the Maximum Connect Time Limit timer (see 7.2.3.4.1), and the MAXIMUM BURST SIZE field (see 9.2.7.2.4); and
- ends:



- A) after a DONE is received;
- B) if a Phy Layer Not Ready confirmation from the phy layer occurs during the SSP connection; or
- C) if an abort connection occurs (see 6.16.6).

#### 4.1.13.2 Persistent connection support

If an end device's SSP phy supports persistent connections (e.g., the SSP PERSISTENT CAPABLE bit is set to one in the Protocol Specific Port Information VPD page (see 9.2.11.4) for the SSP phy), then that SSP phy sets the PERSISTENT CAPABLE bit to one in the IDENTIFY address frame (see 6.10.2).

Support for persistent connections on attached phys is reported using the SMP DISCOVER function (see 9.4.3.10).

#### 4.1.14 Advancing credit

If an SSP phy has receive resources available, then it may advance credit. To advance credit an SSP phy shall:

- 1) set the CREDIT ADVANCE bit to one in the OPEN address frame; and
- 2) request that the SSP transmitter send an RRDY after an OPEN\_ACCEPT is received.

If an SSP phy that implements credit advance receives an OPEN address frame with the CREDIT ADVANCE bit set to one, then the SSP phy:

- 1) increments the transmit SSP frame credit by one (see 6.20.9.1); and
- 2) ignores the next RRDY.

A destination SSP phy that does not implement the CREDIT ADVANCE bit (see 6.10.3) does not advance credit.

#### 4.1.15 Broadcasts

Broadcasts are used to notify SAS ports and expander devices in the SAS domain about certain events. Broadcasts are transmitted using BROADCAST (see 6.2.6.4) and/or the SMP ZONED BROADCAST function (see 9.4.3.20).

Table 15 defines the Broadcast types.

**Table 15 – Broadcast types** (part 1 of 2)

Broadcast	Primitive <sup>a</sup>	Description
Broadcast (Change)	yes	<p>Originated by an expander device to notify SAS initiator ports that a SAS domain change has occurred (see 6.15). May also be originated by SAS initiator ports.</p> <p>SAS target ports shall ignore this Broadcast.</p> <p>See 4.6.2 for management application client handling of Broadcast (Change).</p>
Broadcast (Reserved Change 0)	yes	Reserved. SAS ports (i.e., SAS initiator ports and SAS target ports) shall process this Broadcast the same as Broadcast (Change).
Broadcast (Reserved Change 1)	yes	Reserved. SAS ports shall process this Broadcast the same as Broadcast (Change).
Broadcast (SES)	yes	<p>Originated by a logical unit with a device type of 0Dh (i.e., enclosure services device) (see SPC-4 and SES-3) accessible through a SAS target port in the SAS domain to notify SAS initiator ports of an asynchronous event.</p> <p>A SCSI application client should poll all the logical units in the SAS domain with device types of 0Dh to determine the source.</p> <p>SAS target ports shall ignore this Broadcast.</p>
Broadcast (Expander)	yes	<p>Originated by an expander device to notify SAS initiator ports that an expander event has occurred, including:</p> <ul style="list-style-type: none"> <li>a) the expander device is going to have reduced functionality for a period of time (see 4.5.8);</li> <li>b) a phy event peak value detector has reached its threshold value; or</li> <li>c) a phy event peak value detector has been cleared by an SMP CONFIGURE PHY EVENT function (see 9.4.3.30).</li> </ul> <p>Expander events do not include SAS domain changes, which are communicated with Broadcast (Change).</p> <p>SAS target ports shall ignore this Broadcast.</p> <p>See 4.5.9 for management application client handling of Broadcast (Expander).</p>
<sup>a</sup> All Broadcasts are supported by the SMP ZONED BROADCAST function (see 9.4.3.20), which defines additional reserved Broadcast types. Broadcasts labeled “yes” are also transmitted via BROADCAST primitive sequences (see 6.2.6.4).		

**Table 15 – Broadcast types** (part 2 of 2)

Broadcast	Primitive <sup>a</sup>	Description
Broadcast (Asynchronous Event)	yes	<p>Originated by an SSP target port when an event occurs (e.g., a hard reset) that causes one or more unit attention conditions to be established for one or more logical units accessible through the SSP target port.</p> <p>An SSP target port shall only originate one Broadcast (Asynchronous Event) for each event that affects multiple logical units accessible through the SSP target port (e.g., only one Broadcast (Asynchronous Event) is originated when a hard reset occurs).</p> <p>SAS ports other than SSP initiator ports shall ignore this Broadcast.</p>
Broadcast (Reserved 3)	yes	Reserved. SAS ports shall ignore this Broadcast.
Broadcast (Reserved 4)	yes	Reserved. SAS ports shall ignore this Broadcast.
Broadcast (Zone Activate)	no	<p>Initiates the zone activate step (see 4.8.6.4).</p> <p>Devices that are not locked zoning expander devices shall ignore this Broadcast.</p>
<sup>a</sup> All Broadcasts are supported by the SMP ZONED BROADCAST function (see 9.4.3.20), which defines additional reserved Broadcast types. Broadcasts labeled “yes” are also transmitted via BROADCAST primitive sequences (see 6.2.6.4).		

When an expander port receives a Broadcast, the BPP (see 4.5.5) shall forward the Broadcast on at least one phy in each other expander port if zoning is disabled or forward the Broadcast as described in 4.8.5 if zoning is enabled.

An expander device is not required to queue multiple identical Broadcasts for the same expander port. If a second identical Broadcast is requested before the first Broadcast has been transmitted, then the second Broadcast may be ignored.

A SAS device or expander device may implement counters for Broadcasts it originates and report them in the REPORT BROADCAST response (see 9.4.3.9). If counters are supported, then the SAS device or expander device shall, for each combination of Broadcast type and Broadcast reason that the SAS device or expander device supports:

- a) if the Broadcast is related to a phy, then maintain a separate Broadcast counter for each phy; or
- b) if the Broadcast is not related to a phy, then maintain one originated Broadcast counter.

Broadcast (Change)s originated by an expander device are counted and reported in the REPORT GENERAL response (see 9.4.3.4) and other SMP response frames containing an EXPANDER CHANGE COUNT field.

An expander device may implement counters for Broadcasts received from attached end devices and report them in the REPORT BROADCAST response (see 9.4.3.9).

A SAS device or an expander device is not required to maintain originated Broadcast count information in non-volatile storage or across reset events.

See 4.12 for details on phy events.

## 4.2 Names and identifiers

### 4.2.1 Names and identifiers overview

Device names are worldwide unique names for devices within a transport protocol.

Port names are worldwide unique names for ports within a transport protocol.

Port identifiers are the values by which ports are identified within a domain. Phy identifiers are the values by which phys are identified within a device.

Table 16 describes the definitions of names and identifiers for SAS.

**Table 16 – Names and identifiers**

Attribute	Format	SAS usage	References
Device name	SAS address (see 4.2.4) for SAS devices and expander devices.  NAA IEEE Registered format (see 4.2.2) for SATA devices with worldwide names. <sup>a</sup>	Reported in: a) the IDENTIFY address frame (see 6.10.2) DEVICE NAME field; b) the Device Identification VPD page (see 9.2.11.2); and c) the DISCOVER response (see 9.4.3.10) ATTACHED DEVICE NAME field.	4.2.6 and 4.2.7
Port name	Not defined		4.2.8
Port identifier	SAS address (see 4.2.4)	Reported in: a) the IDENTIFY address frame (see 6.10.2) for SAS ports; and b) the Device Identification VPD page (see 9.2.11.2).	4.2.9
Phy identifier	8-bit value	Phy identifier	4.2.10
<sup>a</sup> For SATA devices without worldwide names, a device name may be set in the PHY CONTROL request (see 9.4.3.28) ATTACHED DEVICE NAME field.			

Table 17 describes the identifier attributes and name attributes for SCSI architecture model objects (see SAM-5) using SAS SSP.

**Table 17 – SCSI architecture model object attribute mapping**

SCSI architecture model object attribute	SAS SSP implementation
Initiator port identifier	SAS address of an SSP initiator port

**Table 17 – SCSI architecture model object attribute mapping**

SCSI architecture model object attribute	SAS SSP implementation
Initiator port name	Not defined
Target port identifier	SAS address of an SSP target port
Target port name	Not defined
SCSI device name	Device name of SAS device containing an SSP port

#### 4.2.2 NAA IEEE Registered format identifier

Table 18 defines the NAA IEEE Registered format identifier used by device names and port identifiers. This format is the same as that defined in SPC-4.

**Table 18 – NAA IEEE Registered format**

Byte\Bit	7	6	5	4	3	2	1	0				
0	NAA (5h)				(MSB)							
1	IEEE COMPANY ID											
2												
3	(LSB)											
4												
5												
6	VENDOR SPECIFIC IDENTIFIER											
7												

The NAA field shall be set as shown in table 18 for the NAA IEEE Registered format.

The IEEE COMPANY ID field contains a 24-bit canonical form company identifier (i.e., OUI) assigned by the IEEE.

NOTE 4 - Information about IEEE company identifiers is obtained from the IEEE Registration Authority web site (see <http://standards.ieee.org/regauth/oui>).

The VENDOR SPECIFIC IDENTIFIER field contains a 36-bit value that is assigned by the organization associated with the company identifier in the IEEE COMPANY ID field. The VENDOR SPECIFIC IDENTIFIER field shall be assigned so the NAA IEEE Registered format identifier is worldwide unique.

### 4.2.3 NAA Locally Assigned format identifier

Table 19 defines the NAA Locally Assigned format identifier used by device names and port identifiers. This format is the same as that defined in SPC-4.

**Table 19 – NAA Locally Assigned format**

Byte\Bit	7	6	5	4	3	2	1	0
0	NAA (3h)							
1								
2								
3								
4								
5								
6								
7								

The NAA field shall be set as shown in table 19 for the NAA Locally Assigned format.

The LOCALLY ADMINISTERED VALUE field contains a 60-bit value that is assigned by an administrator to be unique within the set of SCSI domains that are accessible by a common instance of an administration tool or tools.

### 4.2.4 SAS address

A SAS address is an identifier using either:

- a) the NAA IEEE Registered format (see 4.2.2); or
- b) the NAA Locally Assigned format (see 4.2.3).

A SAS address should use the NAA IEEE Registered format (see 4.2.2).

A SAS address of 00000000 00000000h indicates an invalid identifier.

### 4.2.5 Hashed SAS addresses

SSP frames include hashed versions of SAS addresses of SAS ports to provide an additional level of verification of proper frame routing.

The code used for the hashing algorithm is a cyclic binary Bose, Chaudhuri, and Hocquenghem (BCH) (63, 39, 9) code. Table 20 lists the parameters for the code.

**Table 20 – Hashed SAS address code parameters**

Parameter	Value
Number of bits per codeword	63
Number of data bits	39
Number of redundant bits	24
Minimum distance of the code	9

The generator polynomial for this code is:

$$G(x) = (x^6 + x + 1) (x^6 + x^4 + x^2 + x + 1) (x^6 + x^5 + x^2 + x + 1) (x^6 + x^3 + 1)$$

After multiplication of the factors, the generator polynomial is:

$$G(x) = x^{24} + x^{23} + x^{22} + x^{20} + x^{19} + x^{17} + x^{16} + x^{13} + x^{10} + x^9 + x^8 + x^6 + x^5 + x^4 + x^2 + x + 1$$

Annex E contains additional information on SAS address hashing.

#### 4.2.6 Device names and expander device SAS addresses

Each expander device and SAS device shall include a SAS address (see 4.2.4) as its device name.

A SAS address used as a device name shall not be used as any other name or identifier (e.g., a device name, port name, port identifier, or logical unit name (see SAM-5)), except the SAS address of an expander device is the same as the SAS address of the SMP port in that expander device.

SAS devices and expander devices report their device names in the IDENTIFY address frame (see 6.10.2).

Logical units accessed through SSP target ports report SAS target device names through SCSI vital product data (see 9.2.11).

See Annex J for more information on SAS addresses.

#### 4.2.7 Device names for SATA devices with world wide names

Table 21 defines the NAA IEEE Registered format identifier (see 4.2.2) used by device names for SATA devices that provide world wide names in their IDENTIFY DEVICE data (see ACS-4).

**Table 21 – Device name created from the IDENTIFY DEVICE world wide name**

Subformat field name <sup>a</sup>	Specific bits in	Contents <sup>b</sup>
NAA	Byte 0 bits 7:4	IDENTIFY (PACKET) DEVICE data word 108 bits 15:12 <sup>c</sup>
IEEE COMPANY ID	Byte 0 bits 3:0	IDENTIFY (PACKET) DEVICE data word 108 bits 11:8
	Byte 1	IDENTIFY (PACKET) DEVICE data word 108 bits 7:0
	Byte 2	IDENTIFY (PACKET) DEVICE data word 109 bits 15:8
	Byte 3 bits 7:4	IDENTIFY (PACKET) DEVICE data word 109 bits 7:4
VENDOR SPECIFIC IDENTIFIER	Byte 3 bits 3:0	IDENTIFY (PACKET) DEVICE data word 109 bits 3:0
	Byte 4	IDENTIFY (PACKET) DEVICE data word 110 bits 15:8
	Byte 5	IDENTIFY (PACKET) DEVICE data word 110 bits 7:0
	Byte 6	IDENTIFY (PACKET) DEVICE data word 111 bits 15:8
	Byte 7	IDENTIFY (PACKET) DEVICE data word 111 bits 7:0

<sup>a</sup> See table 18.  
<sup>b</sup> IDENTIFY (PACKET) DEVICE data words 108 to 111 contain the world wide name field (see ACS-4).  
<sup>c</sup> This 4-bit field is required to be set to 5h (i.e., IEEE Registered) by ACS-4.

#### 4.2.8 Port names

Port names are not defined in SAS.

See Annex J for more information on SAS addresses.

#### 4.2.9 Port identifiers and SAS port SAS addresses

Each SAS port (e.g., including the STP target port in each STP SATA bridge) shall include a SAS address (see 4.2.4) as its port identifier.

A SAS address used as a port identifier shall not be used as any other name or identifier (e.g., a device name, port name, or logical unit name (see SAM-5)) except:

- a) a SAS address may be used as a port identifier in one or more other SAS domains (see 4.1.4); and
- b) the SAS address of an SMP port in an expander device is the same as the SAS address of the expander device containing that SMP port.

Expander ports do not have port identifiers.

SAS ports in end devices report their port identifiers in the IDENTIFY address frame (see 6.10.2). Expander devices containing SAS ports (e.g., SAS ports attached to virtual phys or STP target ports in STP SATA bridges) report the port identifiers of those SAS ports in the SMP DISCOVER response (see 9.4.3.10).

Port identifiers are used as source and destination SAS addresses in OPEN address frames (see 6.10.3).

Logical units accessed through SSP target ports report SAS target port identifiers through SCSI vital product data (see 9.2.11).

See Annex J for more information on SAS addresses.



#### 4.2.10 Phy identifiers

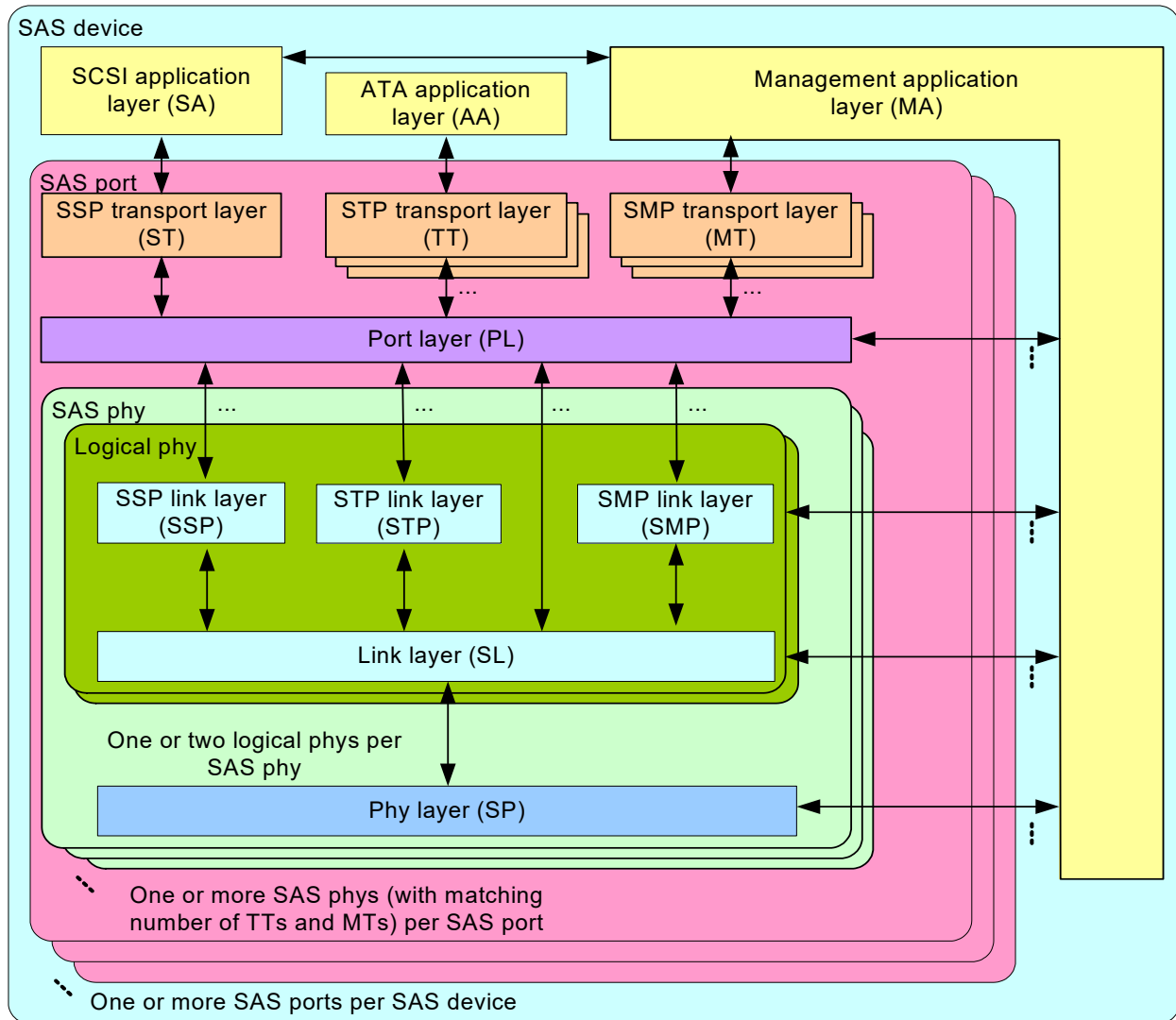
Each SAS phy and expander phy shall be assigned an identifier called a phy identifier that is unique within the SAS device and/or expander device. Each SAS logical phy within a SAS phy shall use the same phy identifier. Each expander logical phy within an expander phy shall use the same phy identifier. The phy identifier is used for SMP functions (see 9.4).

Phy identifiers shall be greater than or equal to 00h and less than or equal to FEh (i.e., 254) 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 response (see 9.4.3.4). 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 Phy Control And Discover mode page (see 9.2.7.5).

## 4.3 State machines

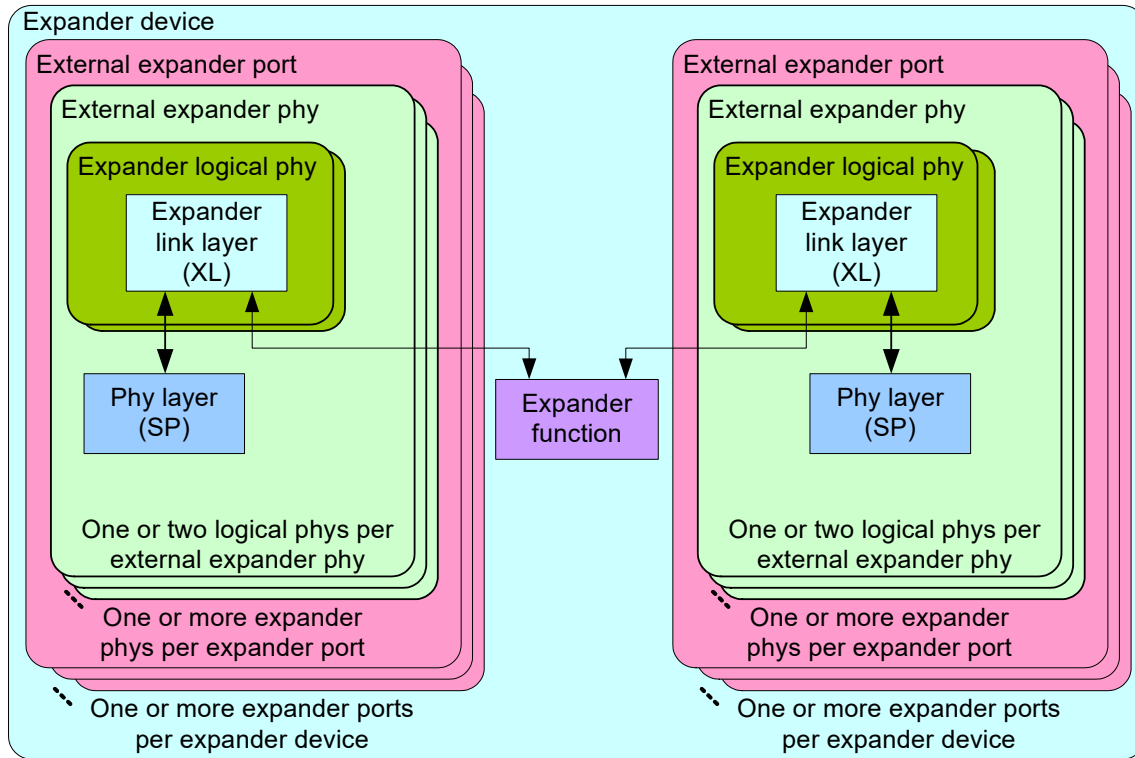
### 4.3.1 State machine overview

Figure 26 shows the state machines for SAS devices and their relationships to each other and to the SAS Device, SAS Port, and SAS Phy classes.



**Figure 26 – State machines for SAS devices**

Figure 27 shows the state machines for expander devices and their relationships to each other and to the Expander Device, Expander Port, and Expander Phy classes. Expander function state machines are not defined in this standard, but the interface to the expander function is defined in 4.5.6.



Note - The expander link layer includes the SL\_IR state machines.

**Figure 27 – State machines for expander devices**

#### 4.3.2 Transmit data path

Figure 28 shows the transmit data path in a SAS phy, showing the relationship between:

- the SP state machine (see 5.14), the PTT state machines (see 5.18), and the SP transmitter (see 5.14.2 and 5.18.2);
- multiplexing (see 5.20);
- the SL\_IR state machines (see 6.12) and the SL\_IR transmitter (see 6.12.2);
- physical link rate tolerance management (see 6.5);
- the SP\_P state machines (see 6.14) and the SL\_P\_S transmitter (see 6.14.4.2);
- the SL state machines (see 6.18) and the SL transmitter (see 6.18.2);
- rate matching (see 6.17); and
- the SSP transmit data path (see figure 29), SMP transmit data path (see figure 30), and STP transmit data path (see figure 31).

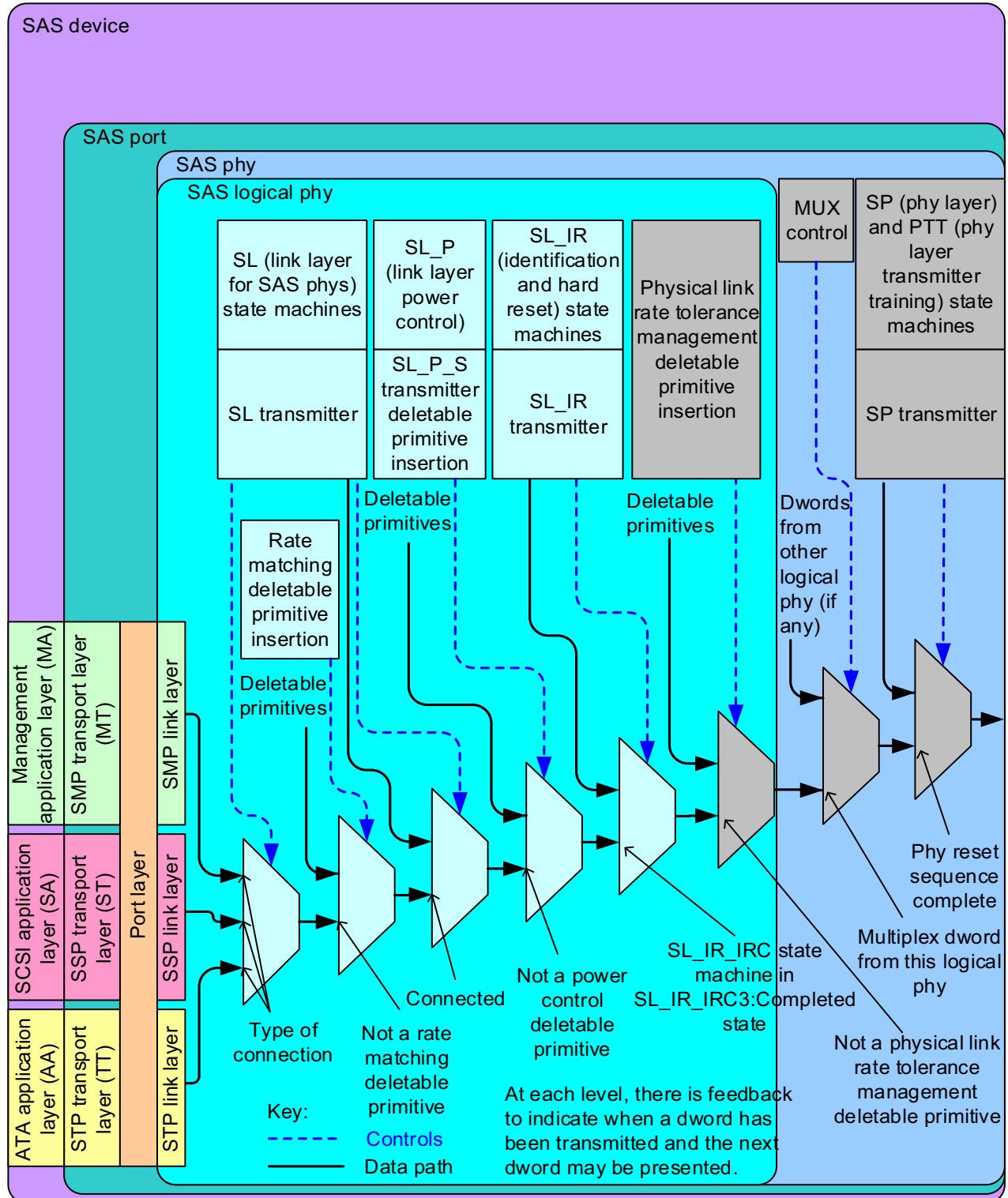


Figure 28 – Transmit data path in a SAS phy

Figure 29 shows the transmit data path for the SSP link layer, including:

- the SSP state machines and the SSP transmitter (see 6.20.9 and 6.20.9.2); and
- the communication to the port layer, SSP transport layer, and SCSI application layer.

Only the SSP link layer (i.e., not the port layer, SSP transport layer, or SCSI application layer) transmits dwords.

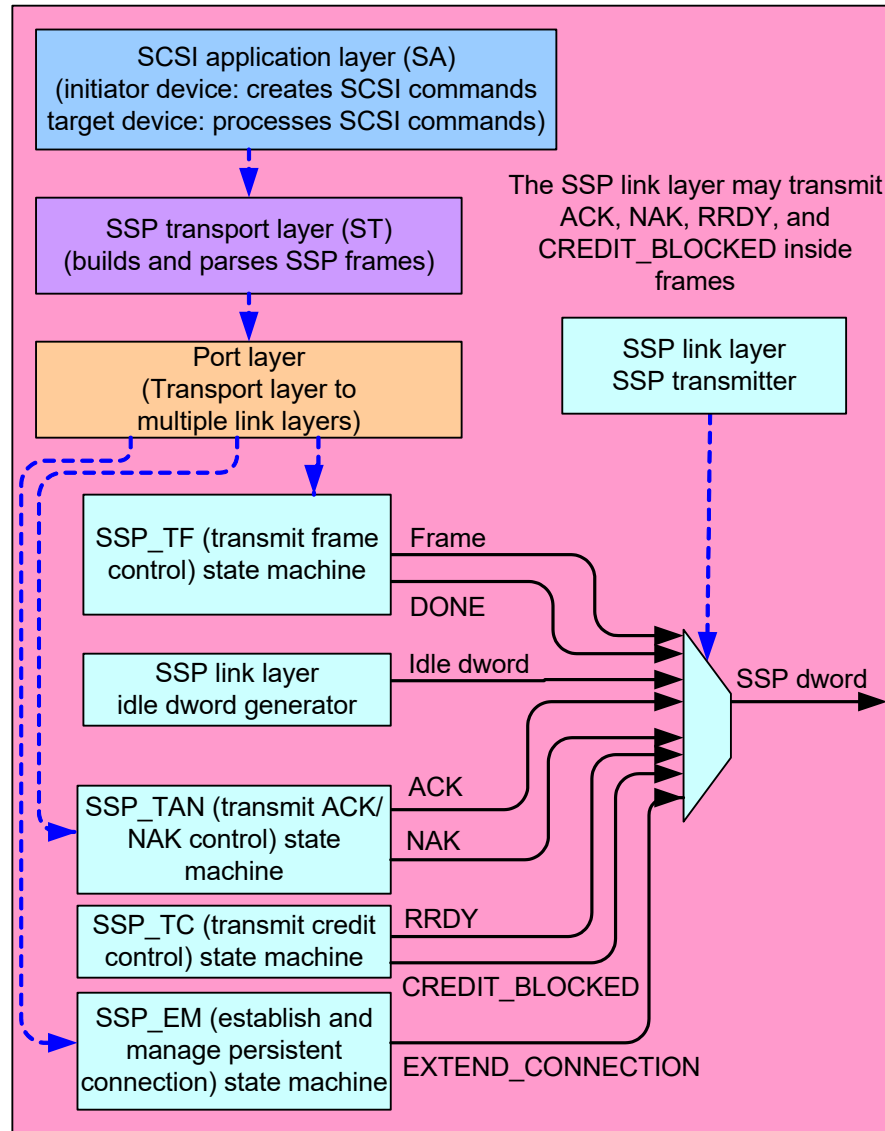
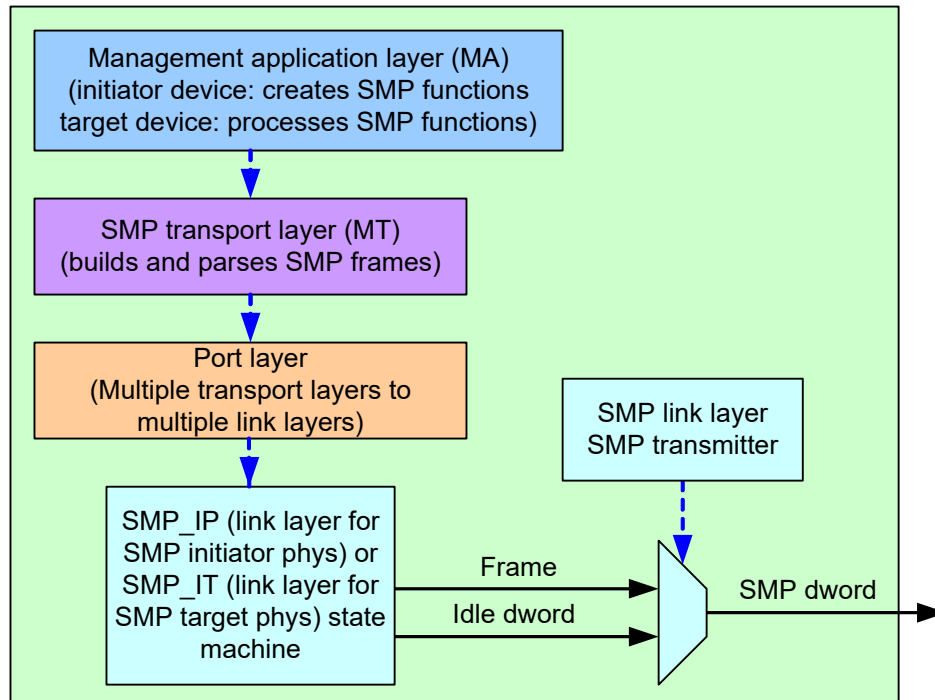


Figure 29 – SSP link, port, SSP transport, and SCSI application layer state machines

Figure 30 shows the transmit data path for the SMP link layer, including:

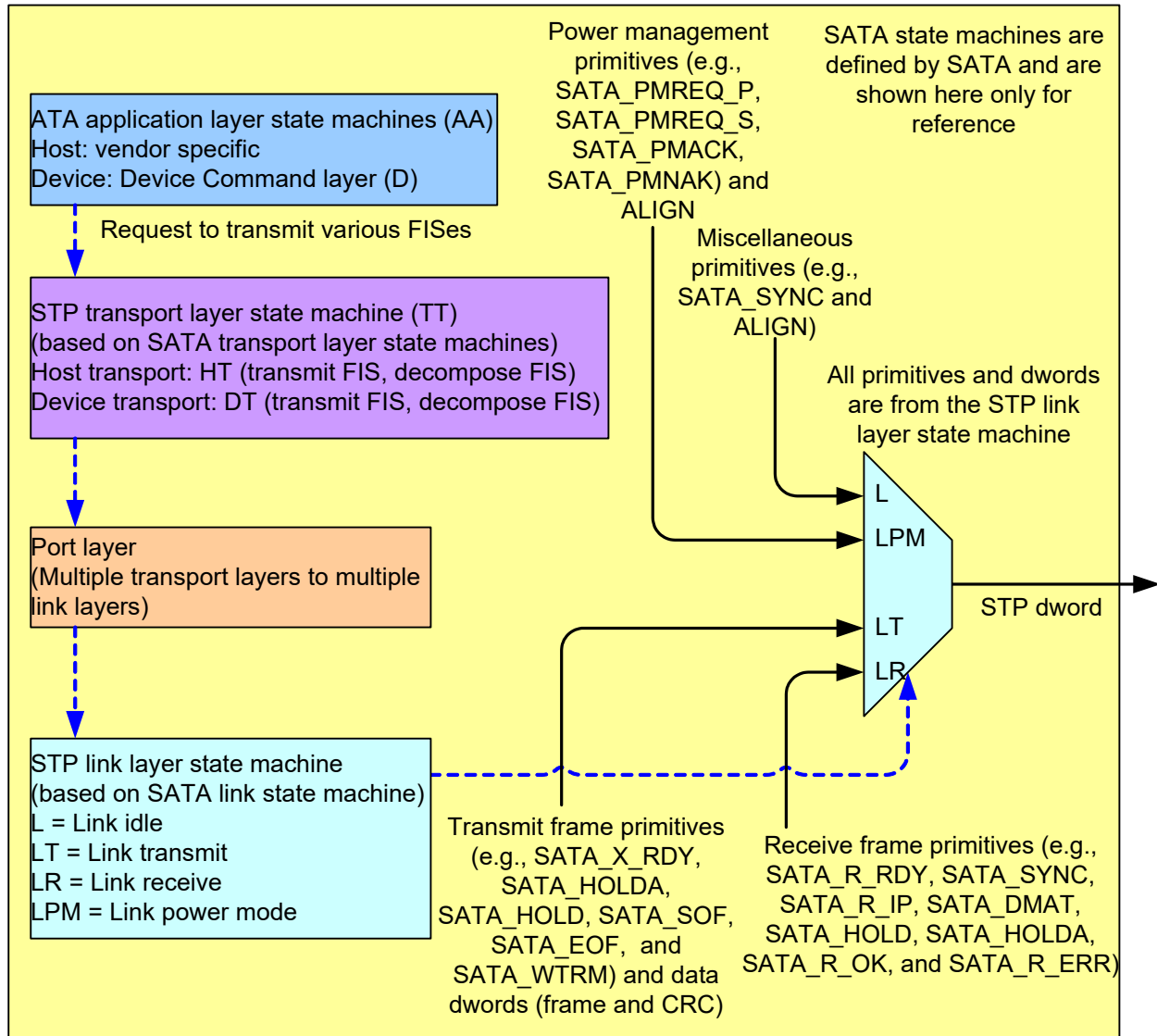
- a) the SMP state machines and the SMP transmitter (see 6.22.6 and 6.22.6.2); and
- b) the communication to the port layer, SMP transport layer, and management application layer.

Only the SMP link layer (i.e., not the port layer, SSP transport layer, or SCSI application layer) transmits dwords.



**Figure 30 – SMP link, port, SMP transport, and management application layer state machines**

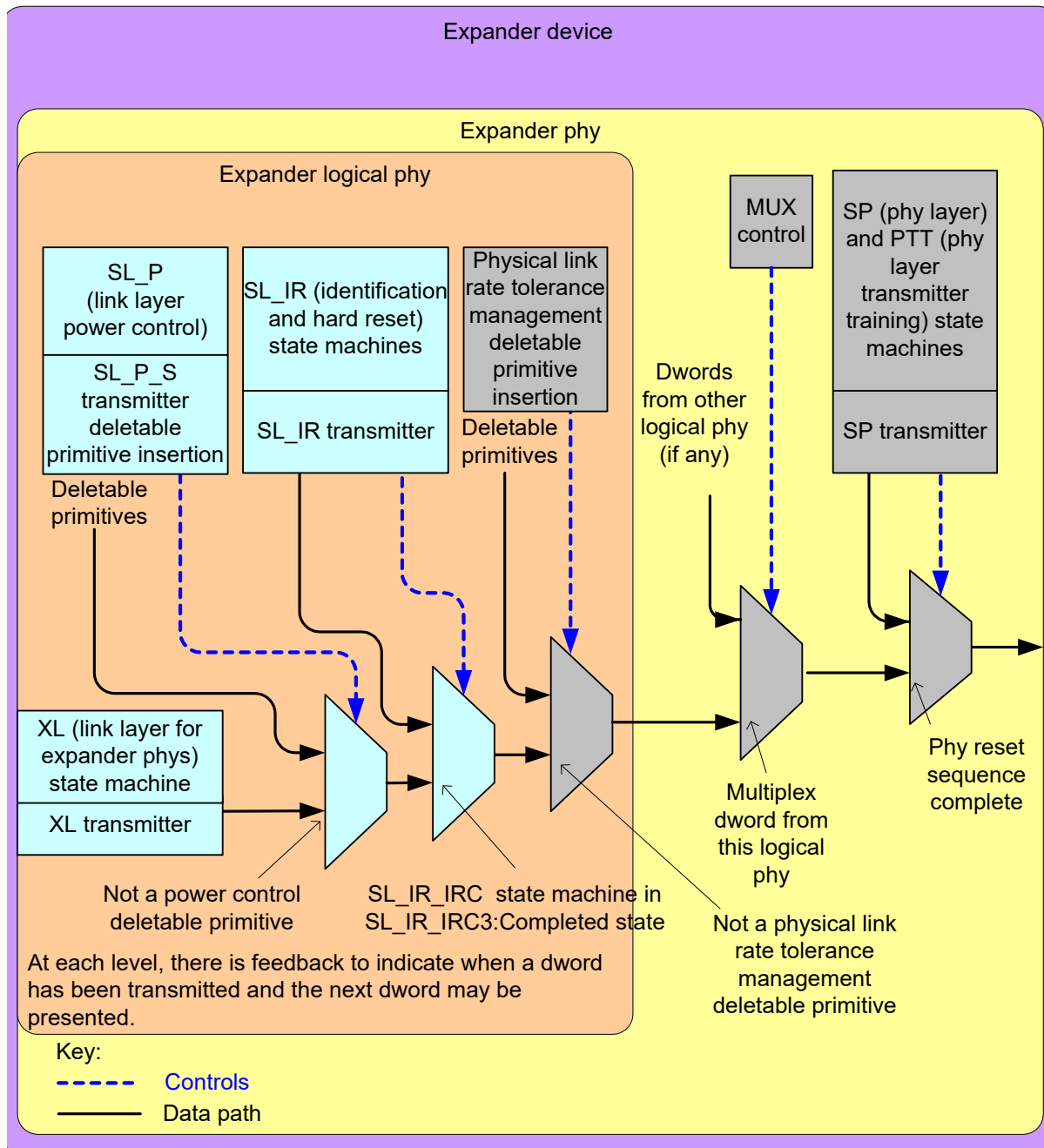
Figure 31 shows the transmit data path for the STP link layer, including the STP state machines (see 6.21.10), and communication to the port layer, STP transport layer, and ATA application layer. Only the STP link layer (i.e., not the port layer, STP transport layer, or ATA application layer) transmits dwords.



**Figure 31 – STP link, port, STP transport, and ATA application layer state machines**

Figure 32 shows the transmit data path in an expander phy, showing the relationship between:

- the SP state machine (see 5.14), the PTT state machines (see 5.18), and the SP transmitter (see 5.14.2);
- multiplexing (see 5.20);
- the SL\_IR state machines (see 6.12) and the SL\_IR transmitter (see 6.12.2);
- physical link rate tolerance management (see 6.5);
- power control (see 6.14) and the SL\_P\_S transmitter (see 6.14.4.2); and
- the XL state machines (see 6.19) and the XL transmitter (see 6.19.2).



**Figure 32 – Transmit data path and state machines in an expander phy**

### 4.3.3 Receive data path

#### 4.3.3.1 Receive data path while in the SAS dword mode

If SAS dword mode is enabled, then the SP\_DWS receiver (see 5.15.2) establishes dword synchronization and sends dwords to:

- a) the SP\_DWS state machine (see 5.15); and
- b) the link layer state machine receivers.

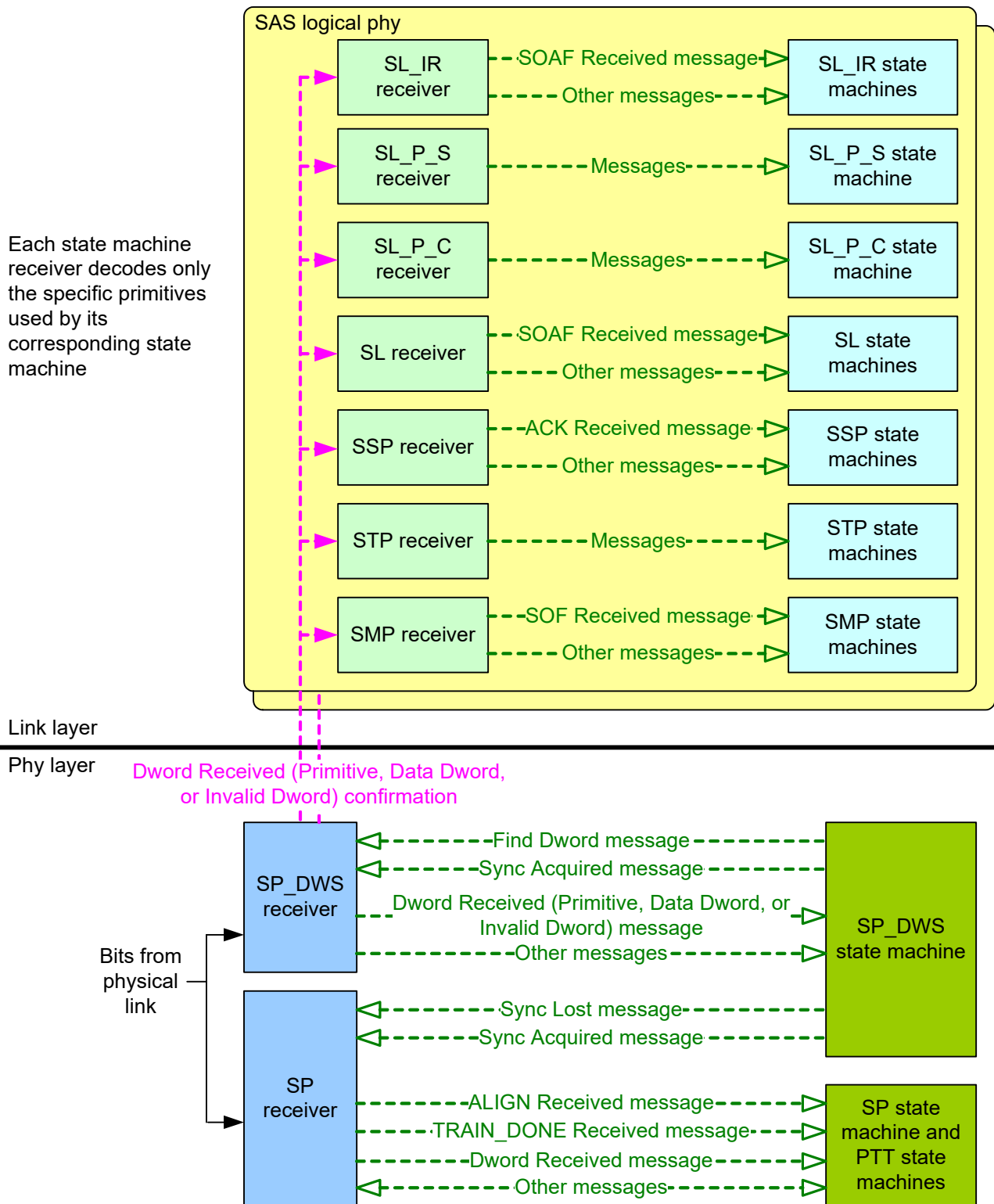
If multiplexing (see 5.20) is enabled (see table 72), then the SP\_DWS receiver uses incoming MUXs to determine the logical link numbers and route the dwords to the appropriate link layer receivers.



Figure 33 shows the receive data path in a SAS phy, showing the relationship between:

- a) the SP state machine (see 5.14) and the SP receiver (see 5.14.2);
- b) the SP\_DWS state machine (see 5.15) and the SP\_DWS receiver (see 5.15.2);
- c) the SL\_IR state machines (see 6.12) and the SL\_IR receiver (see 6.12.2);
- d) the SL\_P\_S state machine (see 6.14.4) and the SL\_P\_S receiver (see 6.14.4.2);
- e) the SL\_P\_C state machine (see 6.14.5) and the SL\_P\_C receiver (see 6.14.5.2);
- f) the SL state machines (see 6.18) and the SL receiver (see 6.18.2);
- g) the SSP state machines (see 6.20.9) and the SSP receiver (see 6.20.9.2);
- h) the SMP state machines (see 6.22.6) and the SMP receiver (see 6.22.6.2); and
- i) the STP state machines and the STP receiver (see 6.21.10).

See figure 134 for more information about the elasticity buffer, which is not shown in figure 33.



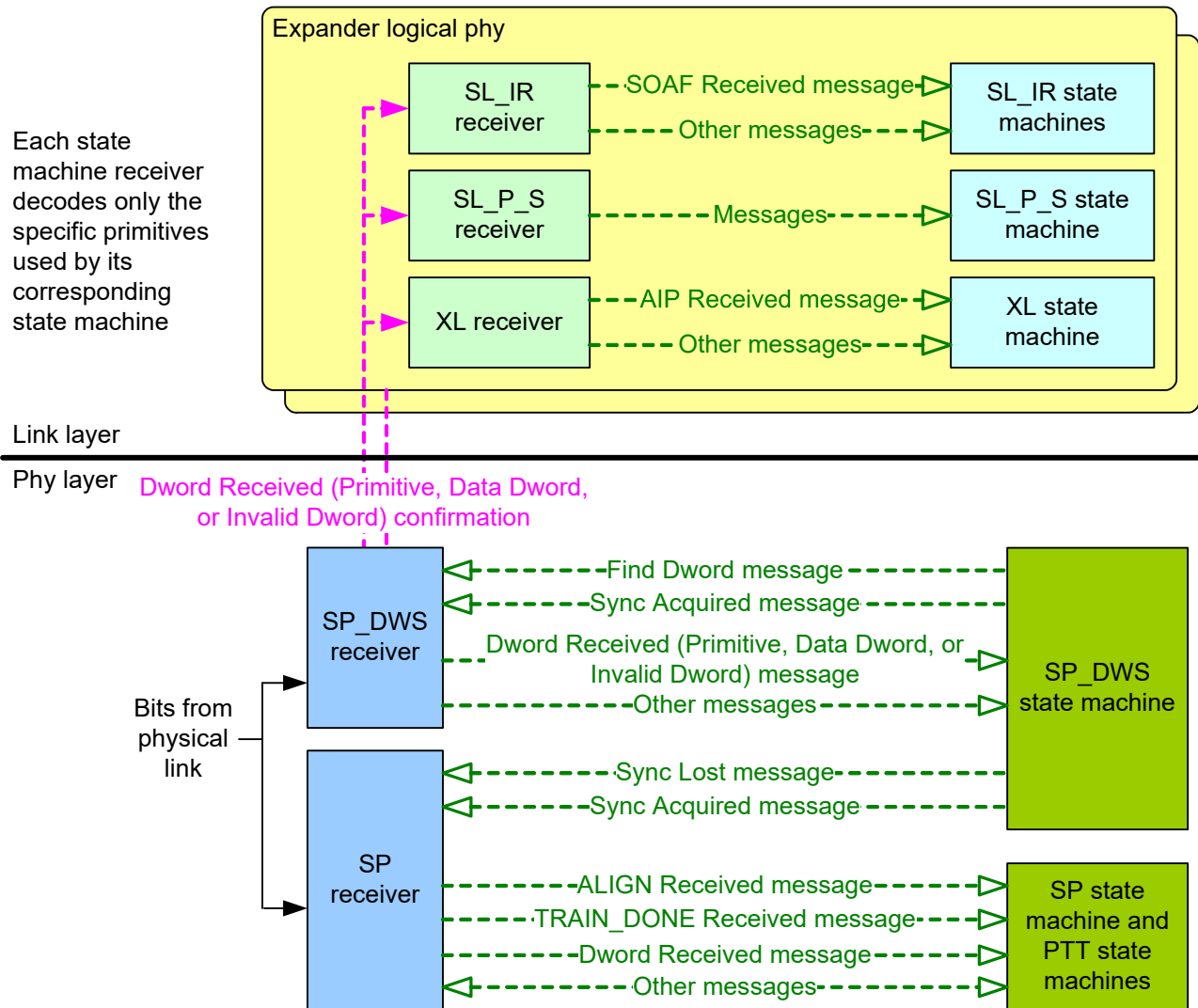
**Figure 33 – Receive data path in a SAS phy while in the SAS dword mode**

Figure 34 shows the receive data path in an expander phy showing the relationship between:

- the SP state machine (see 5.14) and the SP receiver (see 5.14.2);

- b) the SP\_DWS state machine (see 5.15) and the SP\_DWS receiver (see 5.15.2);
- c) the SL\_IR state machines (see 6.12) and the SL\_IR receiver (see 6.12.2);
- d) the SL\_P\_S state machine (see 6.14.4) and the SL\_P\_S receiver (see 6.14.4.2); and
- e) the XL state machine (see 6.19) and the XL receiver (see 6.19.2).

See figure 134 for more information about the elasticity buffer, which is not shown in figure 34.



**Figure 34 – Receive data path in an expander phy while in the SAS dword mode**

#### 4.3.3.2 Receive data path while in the SAS packet mode

If SAS packet mode is enabled, then the SP\_PS receiver (see 5.16.2) establishes SPL packet synchronization and sends SPL packets to:

- a) the SP\_PS state machine (see 5.16); and
- b) the link layer state machine receivers.

Figure 35 shows the receive data path in a SAS phy, showing the relationship between:

- a) the SP state machine (see 5.14) and the SP receiver (see 5.14.2);
- b) the SP\_PS state machine (see 5.16) and the SP\_PS receiver (see 5.16.2);
- c) the SP\_ReSync state machine (see 5.17) and the SP\_PS receiver (see 5.16.2);
- d) the SL\_IR state machines (see 6.12) and the SL\_IR receiver (see 6.12.2);

- e) the SL\_P\_S state machine (see 6.14.4) and the SL\_P\_S receiver (see 6.14.4.2);
- f) the SL\_P\_C state machine (see 6.14.5) and the SL\_P\_C receiver (see 6.14.5.2);
- g) the SL state machines (see 6.18) and the SL receiver (see 6.18.2);
- h) the SSP state machines (see 6.20.9) and the SSP receiver (see 6.20.9.2);
- i) the SMP state machines (see 6.22.6) and the SMP receiver (see 6.22.6.2); and
- j) the STP state machines and the STP receiver (see 6.21.10).

See figure 134 for more information about the elasticity buffer, which is not shown in figure 35.

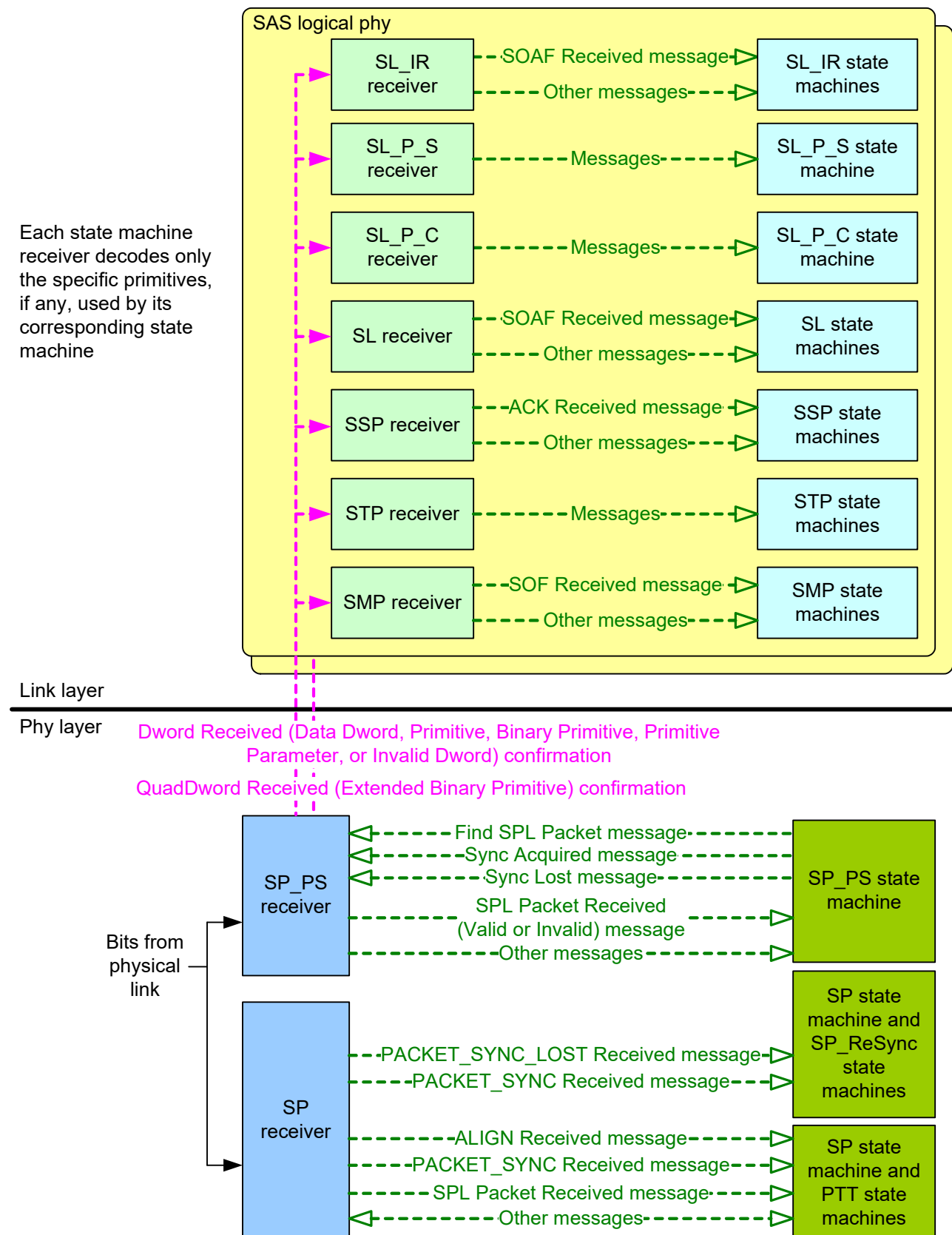


Figure 35 – Receive data path in a SAS phy while in the SAS packet mode

Figure 36 shows the receive data path in an expander phy showing the relationship between:

- the SP state machine (see 5.14) and the SP receiver (see 5.14.2);
- the SP\_PS state machine (see 5.16) and the SP\_PS receiver (see 5.16.2);
- the SP\_ReSync state machine (see 5.17) and the SP\_PS receiver (see 5.16.2);
- the SL\_IR state machines (see 6.12) and the SL\_IR receiver (see 6.12.2);
- the SL\_P\_S state machine (see 6.14.4) and the SL\_P\_S receiver (see 6.14.4.2); and
- the XL state machine (see 6.19) and the XL receiver (see 6.19.2).

See figure 134 for more information about the elasticity buffer, which is not shown in figure 36.

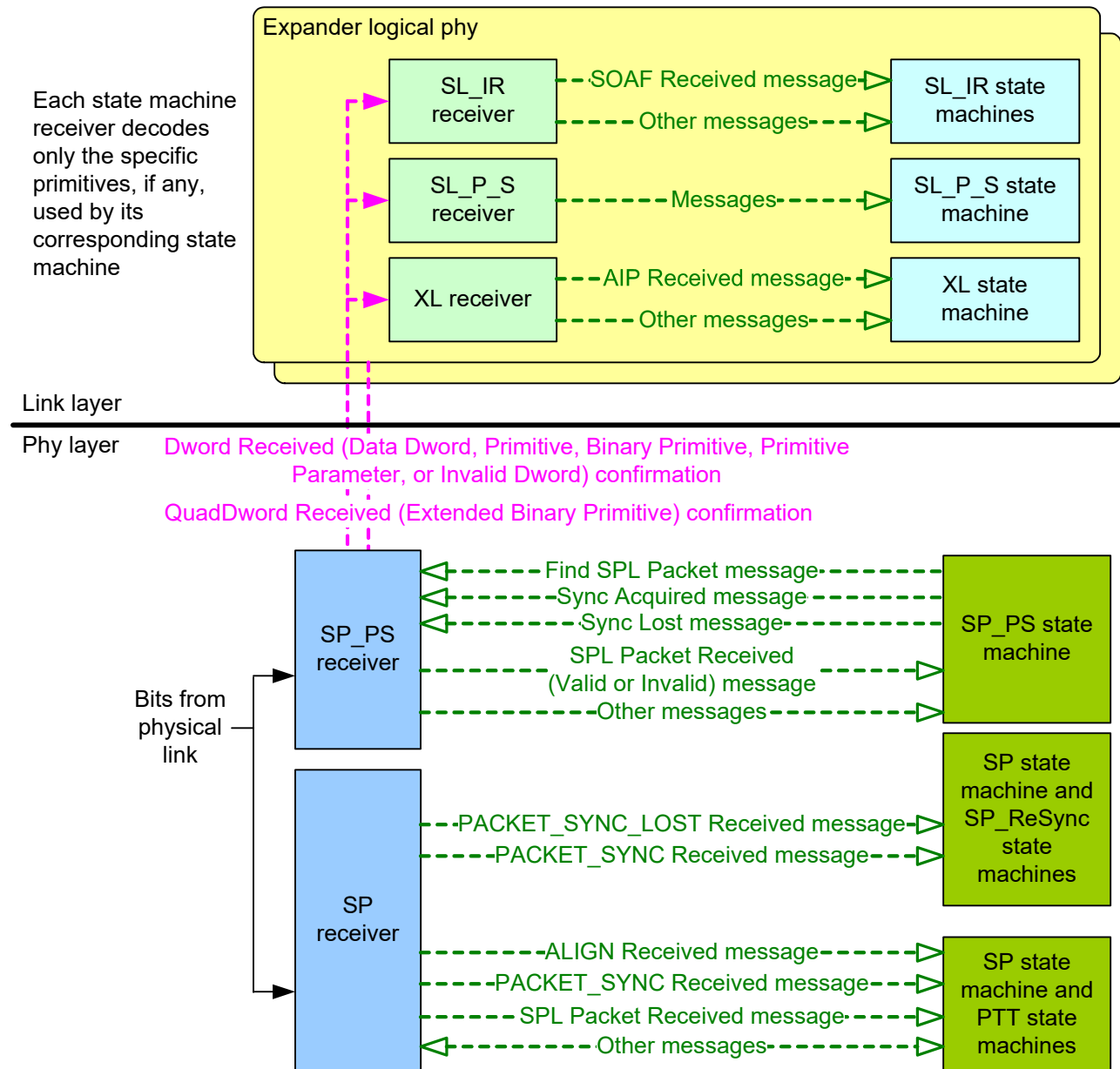


Figure 36 – Receive data path in an expander phy while in the SAS packet mode

#### 4.3.4 State machines and SAS Device, SAS Port, SAS Phy, and SAS Logical Phy classes

Figure 37 shows which state machines are contained within the SAS Device, SAS Port, SAS Phy, and SAS Logical Phy classes.

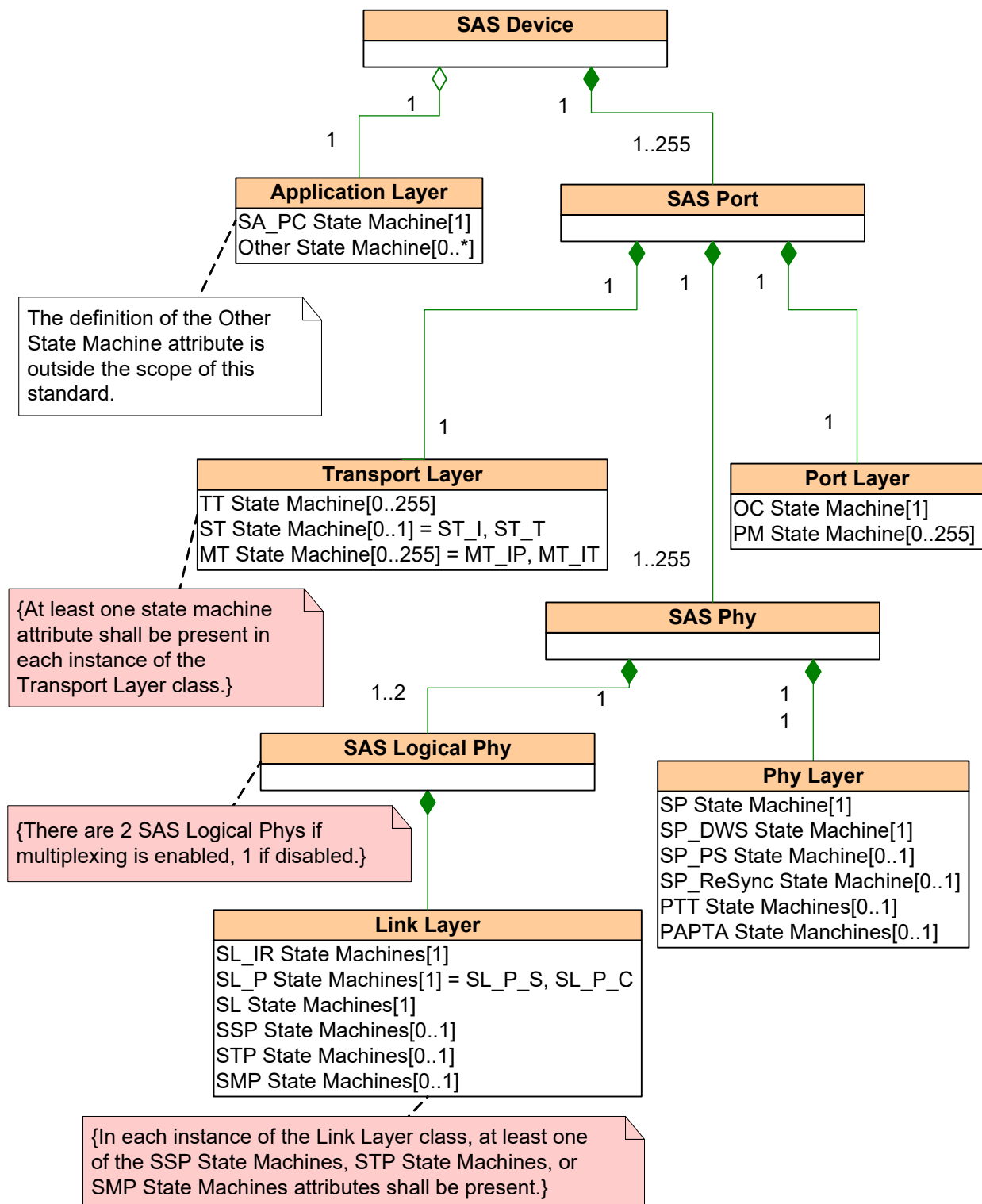


Figure 37 – State machines and SAS Device classes

Figure 38 shows which state machines are contained within the Expander Device, Expander Port, Expander Phy, Expander Logical Phy, SMP Port, SMP Phy, and SMP Logical Phy classes.

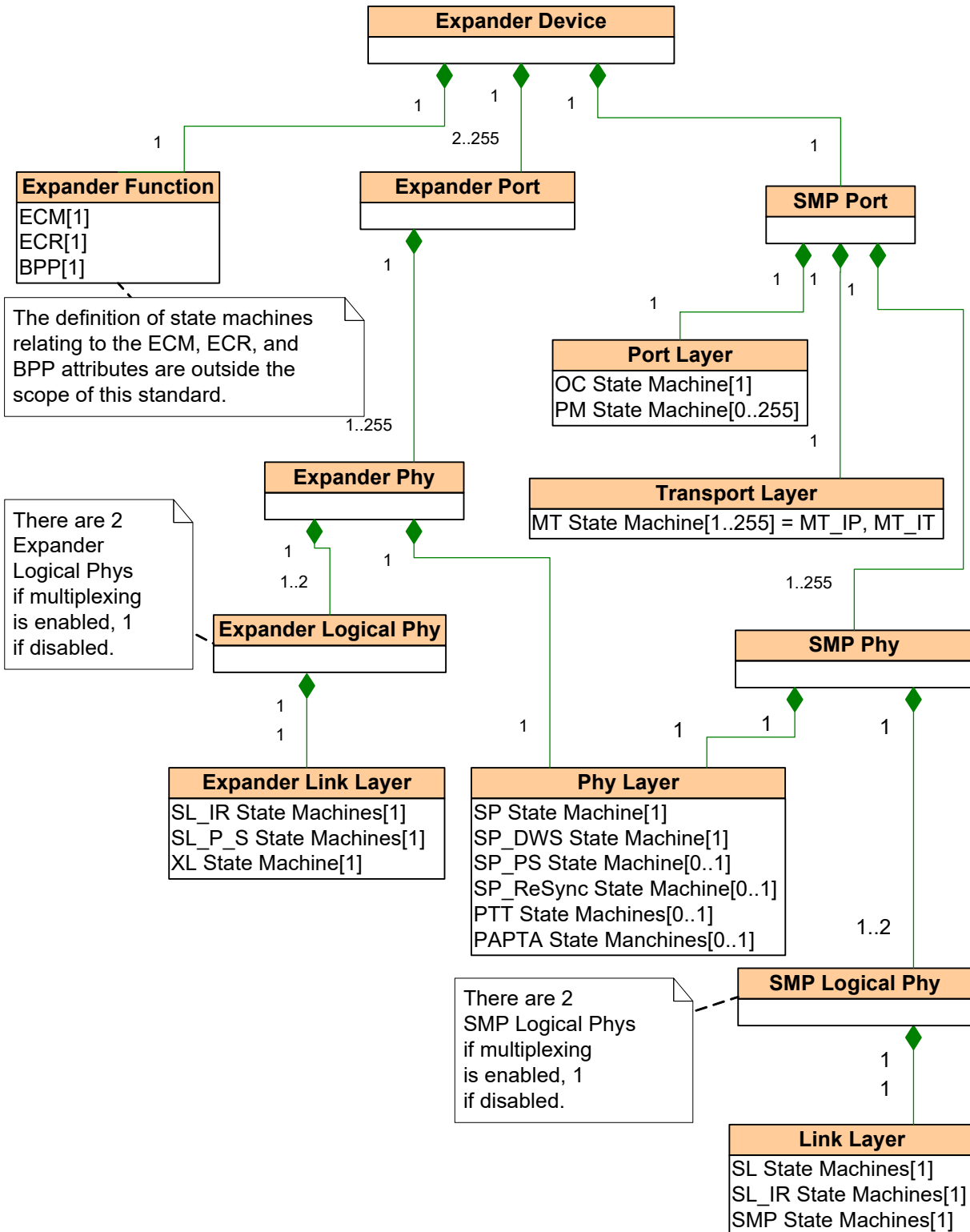


Figure 38 – State machines and Expander Device classes



## 4.4 Events

### 4.4.1 Reset sequences

Figure 39 shows the reset terminology used in this standard:

- a) link reset sequence (see 6.11);
- b) phy reset sequence (see 5.11);
- c) SATA OOB sequence (see 5.11.2.1);
- d) SATA speed negotiation sequence (see 5.11.2.2);
- e) SAS OOB sequence (see 5.11.4.1);
- f) SAS speed negotiation sequence (see 5.11.4.2);
- g) identification sequence (see 6.11); and
- h) hard reset sequence (see 6.11).

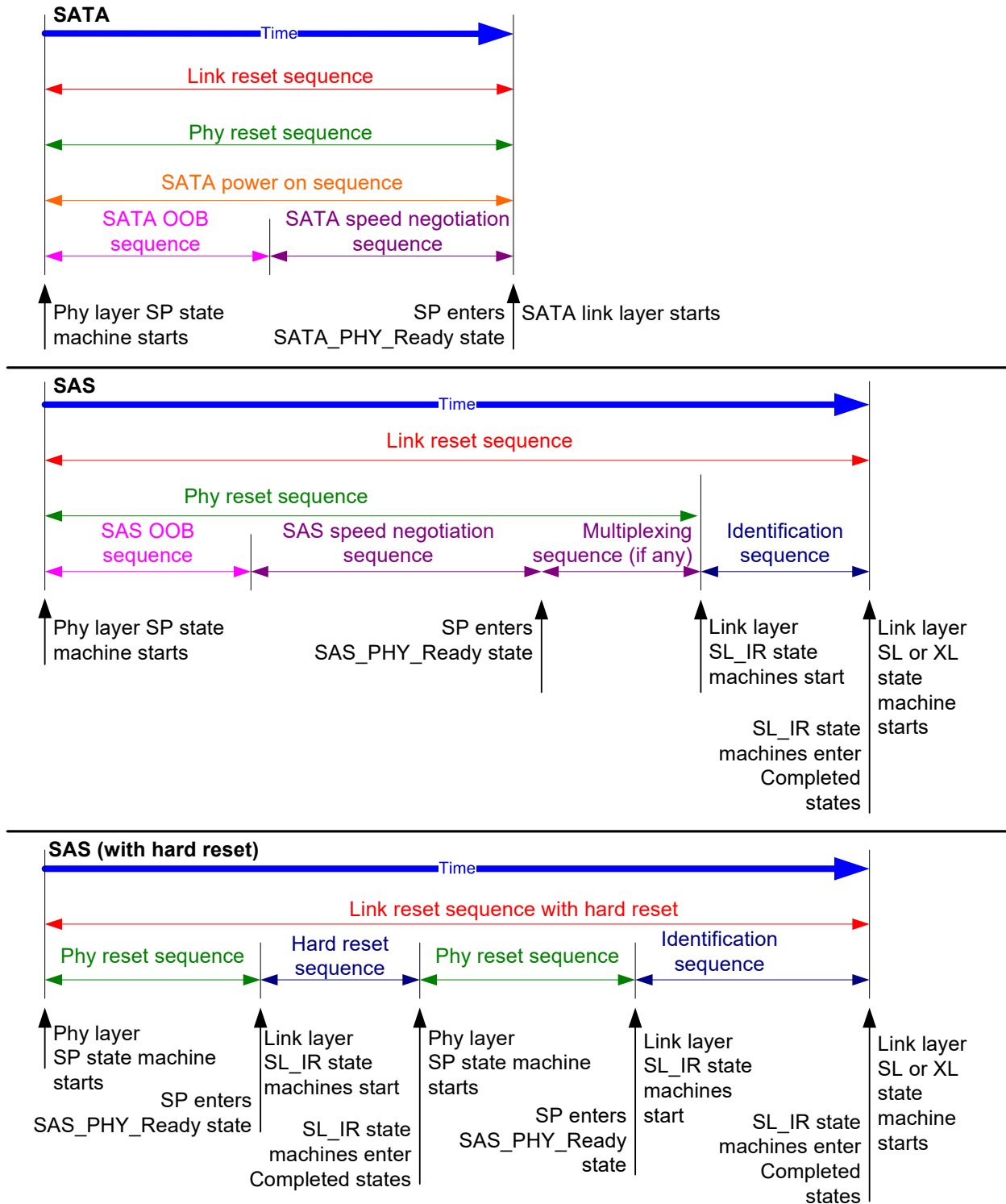


Figure 39 – Reset terminology

The phy reset sequences, including the OOB sequence, the speed negotiation sequence, and the multiplexing sequence, if any, are implemented by the SP state machine and the SP\_DWS state machine and are described in 5.11, 5.14, and 5.15. The identification sequence and hard reset sequence are implemented by the SL\_IR state machines and are described in 6.11 and 6.12.

The link reset sequence has no effect on the transport layer and application layer unless the link reset sequence disrupts frame transmission. A hard reset sequence replaces the identification sequence to initiate a hard reset. The link reset sequence serves as a hard reset for SATA devices (see SATA).

#### 4.4.2 Hard reset

##### 4.4.2.1 Hard reset overview

If, after the phy reset sequence, a phy receives a HARD\_RESET primitive sequence before an IDENTIFY address frame, then the phy shall consider this to be a reset event and the port containing the phy shall process a hard reset.

When a port processes a hard reset, the port shall stop transmitting valid dwords on each of the phys contained in that port. Each phy may then participate in new phy reset sequences (e.g., respond to incoming COMINITs (see SATA)) and shall originate a new link reset sequence if one is not detected. The hard reset shall not affect any other ports in the device.

If a SAS device is contained in an expander device, then the SSP ports, STP ports, and/or SATA ports in the expander device shall initiate a hard reset when an SMP PHY CONTROL function with a phy operation of HARD RESET and phy identifier specifying a virtual expander phy attached to such a SAS port is processed (see 9.4.3.28).

##### 4.4.2.2 Additional hard reset processing by SAS ports

If the port processing the hard reset is an SSP port, then the hard reset causes a Transport Reset event notification to be sent to the SCSI application layer (see 9.2.5) and the SCSI device shall perform the actions defined for hard reset in SAM-5. After processing a hard reset, each logical unit to which the SSP target port has access shall establish a unit attention condition for all SSP initiator ports with the additional sense code set to SCSI BUS RESET OCCURRED (see SAM-5 and SPC-4).

If the port processing the hard reset is an STP port in an STP SATA bridge, then the SATA host port shall originate a link reset sequence.

If the port processing the hard reset is an STP port that is not in an STP SATA bridge, then the STP target device shall perform the actions defined for power on or hardware reset in ATA8-AAM.

##### 4.4.2.3 Additional hard reset processing by expander ports

If the port processing a hard reset is an expander port, then the expander device shall not originate a hard reset sequence on any of its other phys.

If the port processing a hard reset is an expander port, then the expander function and other expander ports in the expander device shall not be affected by hard reset. SAS devices contained in the expander device shall not be affected by hard resets received by external expander ports in the expander device.

#### 4.4.3 I\_T nexus loss

A SAS port that maintains an I\_T nexus loss timer (see 7.2.2.1) for a destination port uses the I\_T nexus loss timer to delay detection of an I\_T nexus loss (e.g., for cases where a phy on the pathway between the SAS initiator port and SAS target port loses dword synchronization and performs a new link reset sequence).

The SAS port establishes an I\_T nexus loss timer event for a destination port (see 7.2.2.3.3) if:

- a) a connection request to the destination port results in an OPEN\_REJECT (NO DESTINATION), OPEN\_REJECT (RESERVED INITIALIZE 0), or OPEN\_REJECT (RESERVED INITIALIZE 1) (see table 137);
- b) a connection request to the destination port results in the Open Timeout timer expiring (see 6.16.2);
- c) a connection request to the logical phy on the destination port receives a BREAK (see 7.2.2.3.3);
- d) a management application client completing the discover process (see 4.6) detects that the destination port is no longer in the SAS domain; or

- e) there are no phys in the SAS port (see 7.2.2.2).

The SAS port initializes and starts an I\_T nexus loss timer for a destination port after establishing an I\_T nexus loss timer event for the destination port. The SAS port stops the I\_T nexus loss timer if:

- a) a connection request to the destination port results in an OPEN\_REJECT (RESERVED CONTINUE 0), OPEN\_REJECT (RESERVED CONTINUE 1), or OPEN\_REJECT (RETRY) (see table 137);
- b) a connection to the destination port is established; or
- c) a management application client completing the discover process (see 4.6) detects that the destination port is in the SAS domain.

After the SAS port establishes an I\_T nexus loss timer event for a destination port, the SAS port retries any connection request to that destination port until:

- a) a connection to that destination port is established; or
- b) an I\_T nexus loss event occurs.

The SAS port establishes an I\_T nexus loss event for a destination port if:

- a) the I\_T nexus loss timer expires (see 7.2.2.1, 9.2.7.4, and 9.4.3.18);
- b) a connection request to the destination port results in an abandon-class OPEN\_REJECT (see table 136); or
- c) the SAS port receives a vendor specific indication (e.g., SGPIO presence detection) that the destination port is no longer in the SAS domain.

An I\_T nexus loss occurs for a destination port if an I\_T nexus loss event occurs (see SAM-5).

I\_T nexus loss is handled by the port layer state machines (see 7.2.2.3). In some cases, the I\_T nexus loss timer is overridden for connection requests through self-configuring expander devices as described in 4.6.1.

If an I\_T nexus loss occurs in an SSP port, then the SSP port sends a Nexus Loss event notification to the SCSI application layer (see 9.2.5) and the SCSI device shall perform the actions defined for I\_T nexus loss in SAM-5. If an I\_T nexus loss occurs in an SSP initiator port, then a SCSI application client should send an I\_T NEXUS RESET task management function to the SSP target port during the next connection between that SSP initiator port and that SSP target port.

If an I\_T nexus loss occurs in an STP initiator port, then the STP initiator port shall send a Transport Event Notification (Nexus Loss, [Device]) indication to the ATA application client (i.e., create a nexus loss event) (see ATA8-AAM). The ATA application client shall consider any commands for the lost STP target port to be completed with an error.

If an I\_T nexus loss occurs in an STP target port, then the ATA device server shall abort all outstanding commands for the lost STP initiator port. If the STP target port is in an STP SATA bridge, then the STP SATA bridge shall originate a link reset sequence to the SATA device so the ATA device server in the SATA device aborts all outstanding commands.

If an I\_T nexus loss occurs in an SMP initiator port, then the SMP initiator port shall stop attempting to establish connections to the lost SMP target port.

If an I\_T nexus loss occurs in an initiator port due to I\_T nexus loss timer expiration, then a management application client should cause a link reset sequence on the phys attached to the lost target port (e.g, if directly attached, then the phys in the initiator port, if attached via expander devices, then the phys in the expander device attached to the target port).

#### 4.4.4 Power loss expected

If, after the phy reset sequence, a phy receives a NOTIFY (POWER LOSS EXPECTED), then the phy shall consider this to be a power loss expected event and the port containing the phy shall process a power loss expected (see 6.2.5.3.3).

## 4.5 Expander device model

### 4.5.1 Expander device model overview

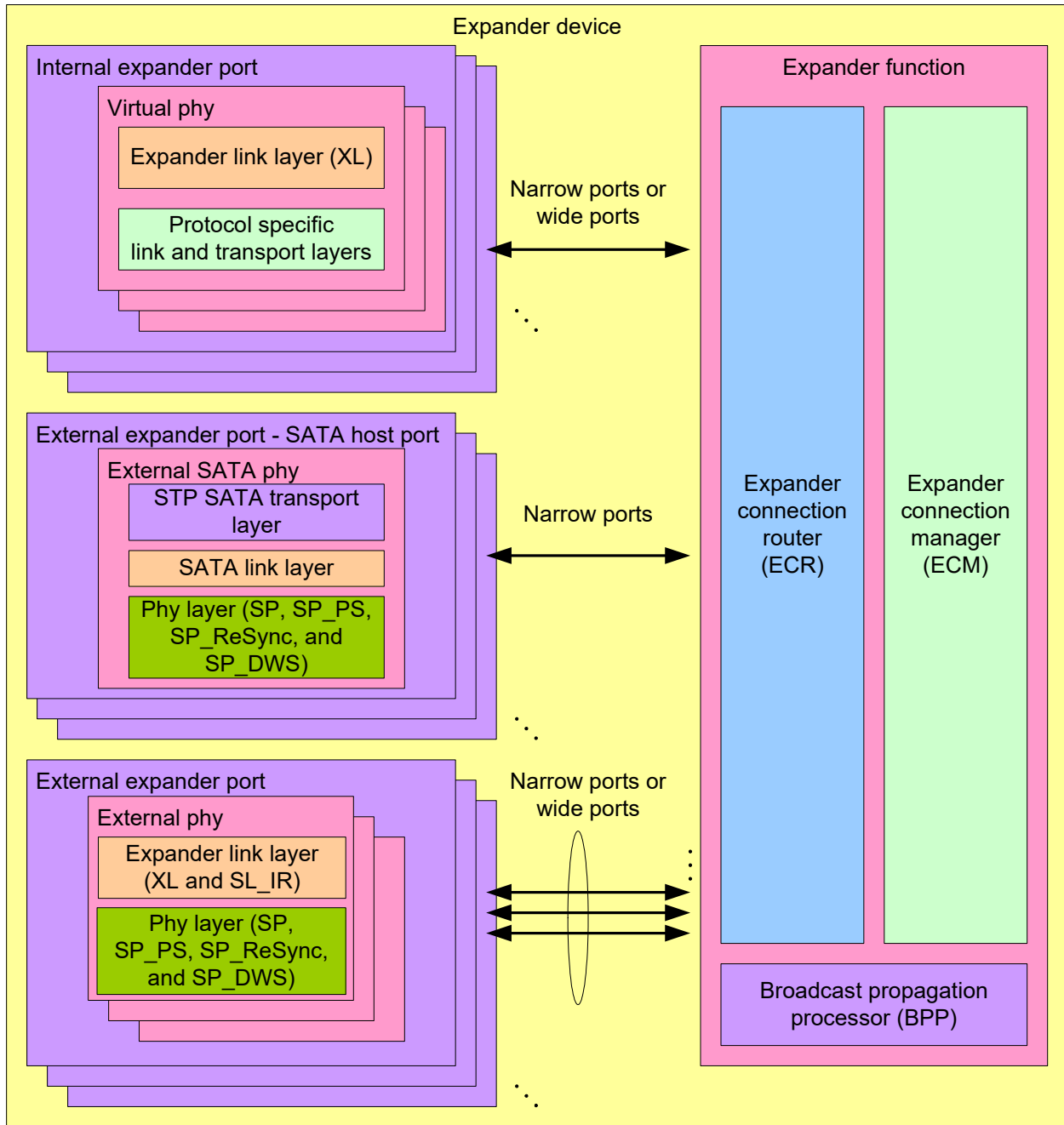
An expander device shall contain the following:

- a) an expander function containing:
  - A) an ECM (see 4.5.3);
  - B) an ECR (see 4.5.4); and
  - C) a BPP (see 4.5.5);
- b) two or more physical expander phys;
- c) an expander port available per phy; and
- d) an SMP target port and a management device server.

An expander device may contain any of the following:

- a) an SMP initiator port and a management application client; or
- b) SAS devices with SSP ports, STP ports, and/or SMP ports and their associated device servers and/or application clients.

Figure 40 shows a model of an expander device showing the state machines in each expander port. The internal SMP ports are not shown.



**Figure 40 – Expander device model**

#### 4.5.2 Expander ports

An external expander port contains one or more physical phys (see 4.1.2). Since each phy in the expander device has the same SAS address, expander ports are created based on the attached SAS addresses (see 4.1.4).

Each phy in an expander port shall have the same routing attribute (see 4.5.7.1). The management device server shall return the same value in the ROUTING ATTRIBUTE field for each phy in an expander port in the SMP DISCOVER response (see 9.4.3.10).

Each phy in an expander port containing phys with table routing attributes in an externally configurable expander device shall have the same number of routing table entries (see 4.5.7.4).

A set of expander phys with table routing attributes in an expander device not supporting table-to-table attachments using the same external connector is called an enclosure out port (see SAS-4). A set of expander phys with subtractive routing attributes using the same external connector is called an enclosure in port (see SAS-4). A set of expander phys with table routing attributes in an expander device supporting table-to-table attachments using the same external connector is called an enclosure universal port (see SAS-4).

Each phy in an expander port shall have the same zone phy information (see 4.8.3.1). The zone phy information associated with each of the phys in an expander port is treated as the zoning properties of the expander port.

Each expander logical phy contains an expander link layer with an XL state machine (see 6.19) and one set of SL\_IR state machines (see 6.12). The XL state machine in each expander logical phy within an expander port processes connection requests independently of the XL state machines in other expander logical phys.

An internal expander port contains a virtual phy with an expander link layer and a protocol specific transport layer (e.g., to provide access as an SSP target port to a logical unit with a device type of 0Dh (i.e., enclosure services device) (see SPC-4 and SES-3)).

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. This SAS address is reported in the ATTACHED SAS ADDRESS field in the SMP DISCOVER response (see 9.4.3.10) for the expander phy containing the STP SATA bridge (i.e., the expander phy attached to the SATA device or SATA port selector).

#### **4.5.3 Expander connection manager (ECM)**

The ECM performs the following functions:

- a) mapping a destination SAS address in a connection request to a destination phy using direct, subtractive, or table routing methods;
- b) arbitrating and assigning or denying path resources for connection requests following SAS arbitration and pathway recovery rules;
- c) configuring the ECR;
- d) managing phy power conditions (see 4.10); and
- e) managing APTA (see 4.14).

#### **4.5.4 Expander connection router (ECR)**

The ECR routes messages between pairs of expander logical phys as configured by the ECM. Enough routing resources shall be provided to support at least one connection.

While forwarding dwords during a connection from a phy with a higher logical link rate to a phy with a lower logical link rate, rate matching (see 6.17) ensures the dwords are at a connection rate equal to or less than the lower logical link rate. However, the ECR may be requested to forward more dwords than the receiving phy is able to accept if:

- a) an invalid dword occurs during a deletable primitive;
- b) an invalid dword occurs during a CLOSE primitive sequence; or
- c) multiple invalid dwords occur during a BREAK primitive sequence.

If an elasticity buffer (see 6.5) overflow occurs, then the ECR may discard dwords and count that event as an elasticity buffer overflow (see 4.12).

While forwarding dwords from a SATA physical link with a higher physical link rate to a SAS logical link with a lower logical link rate, the SATA host port in the STP SATA bridge shall throttle incoming FISes with SATA\_HOLD (see 6.21.4).

NOTE 5 - SATA allows the receiver of a SATA\_HOLD to transmit up to 20 data dwords after detection of SATA\_HOLD. Therefore, the transmitter of SATA\_HOLD receives up to 21+n data dwords for Gen1 or Gen2, or 25+n data dwords for Gen3 (see 6.21.4). SATA\_HOLD does not affect primitives (see SATA). The STP SATA bridge expands or contracts repeated and continued primitives without changing their functional meaning.

When forwarding dwords from a SAS logical link with a lower logical link rate to a SATA physical link with a higher physical link rate, the SATA host port in the STP SATA bridge shall perform a process similar to rate matching (see 6.17) by inserting ALIGN (0) and/or SATA\_HOLD on the SATA physical link whenever it underflows.

NOTE 6 - SATA requires that ALIGN (0) be transmitted in pairs (see SATA).

#### **4.5.5 Broadcast propagation processor (BPP)**

The BPP receives Broadcasts from each expander logical phy or from the management device server on behalf of an expander logical phy and requests transmission of those Broadcasts on all expander ports except the expander port from which the Broadcast was received.

In a zoning expander device with zoning enabled (see 4.8.2), Broadcasts are forwarded as described in 4.8.5.

#### **4.5.6 Expander device interfaces**

##### **4.5.6.1 Expander device interface overview**

The expander device arbitrates and routes between expander logical phys. All routing occurs between expander logical phys, not expander ports. The interaction between an XL state machine and the expander function consists of requests, confirmations, indications, and responses. This interaction is called the expander device interface.



Figure 41 describes the interfaces present within an expander device.

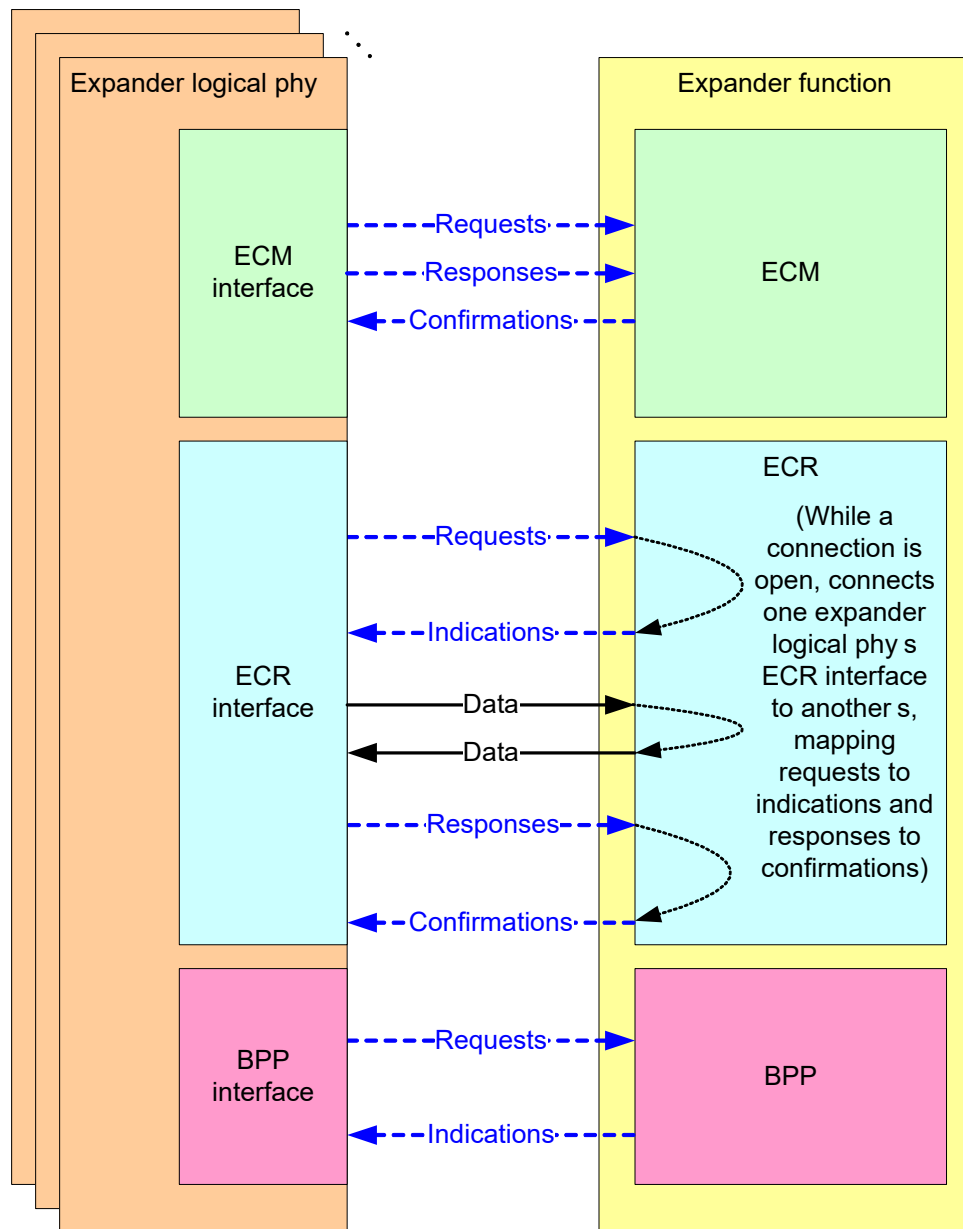


Figure 41 – Expander device interfaces

#### 4.5.6.2 Expander device interfaces detail

Figure 42 shows the interface requests, confirmations, indications, and responses used by an expander device to manage connections.

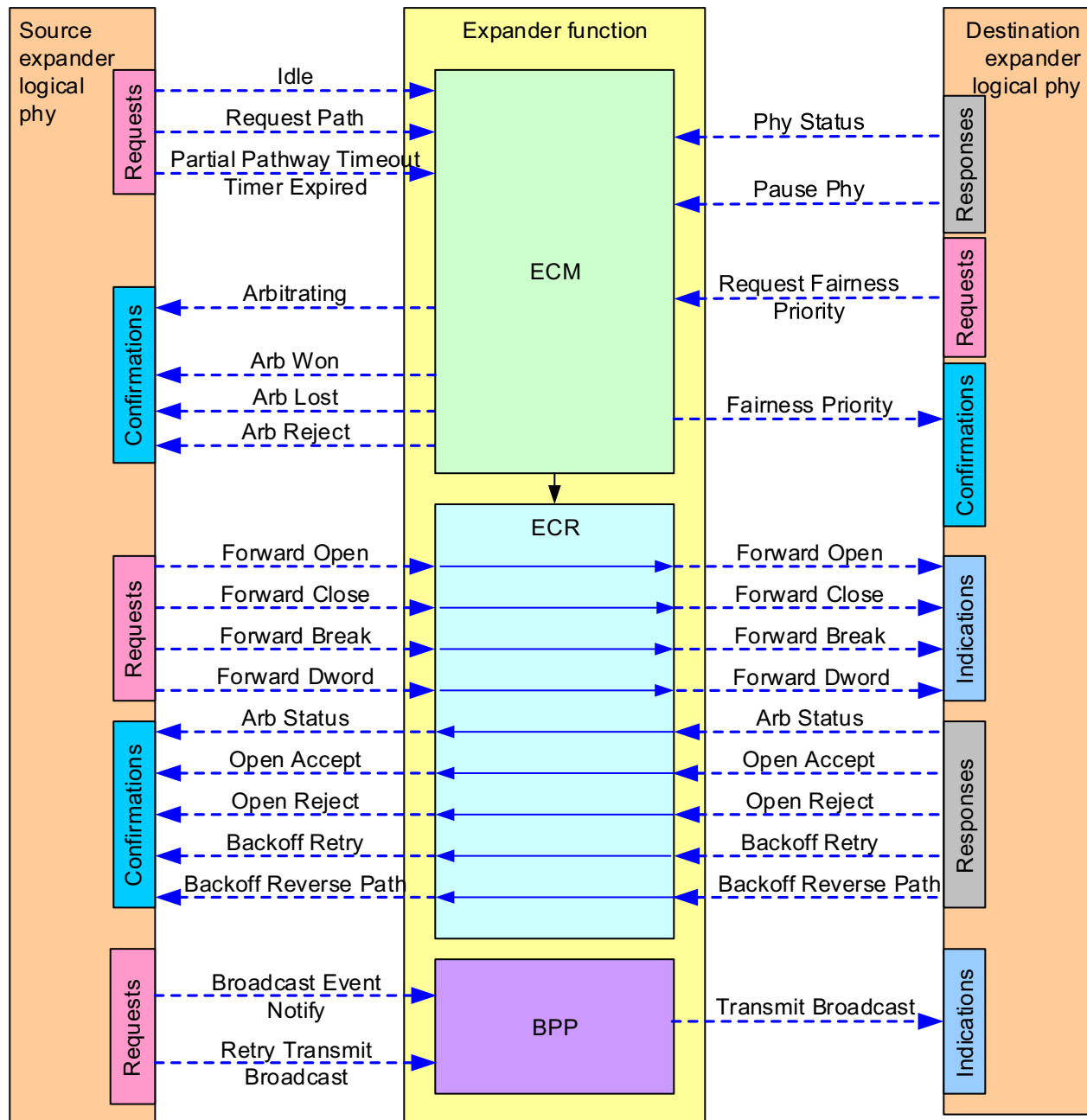


Figure 42 – Expander device interface detail

#### 4.5.6.3 ECM interface

Table 22 describes the requests from an expander logical phy to the ECM. The XL state machine (see 6.19) defines when each request is sent.

**Table 22 – Expander logical phy to ECM requests**

Request	Description
Idle	The XL state machine is in the XL0:Idle state (e.g., after receiving an Enable Disable SAS Link (Disable) message).
Request Path (arguments)	Request for a connection.
Partial Pathway Timeout Timer Expired	The Partial Pathway Timeout Timer expired.
Request Fairness Priority	Request for information on the highest priority OPEN address frame for the expander logical phy specified by a Phy Identifier argument.

Table 23 describes the responses from an expander logical phy to the ECM. The XL state machine (see 6.19) defines when each response is sent.

**Table 23 – Expander logical phy to ECM responses**

Response	Description
Phy Status (Partial Pathway)	Response meaning that an expander logical phy: a) is being used for an unblocked partial pathway; or b) is waiting on another expander logical phy being used for a partial pathway.
Phy Status (Blocked Partial Pathway)	Response meaning that an expander logical phy: a) is being used for a blocked partial pathway; or b) is waiting on another expander logical phy being used for a blocked partial pathway.
Phy Status (Connection)	Response meaning that an expander logical phy: a) is being used for a connection; or b) is waiting on another expander logical phy being used for a connection.
Phy Status (Breaking Connection)	Response meaning that an expander logical phy is breaking a connection.
Pause Phy	Response meaning that the ECM does not arbitrate or assign path resources for Request Path requests for the requesting expander logical phy except if the arbitrate or assign path resource request is from the expander logical phy specified by a Phy Identifier argument.

Table 24 describes the confirmations from the ECM to an expander logical phy. These confirmations are sent in confirmation of a Request Path request. See 6.16.5 for specific definitions for when each confirmation is sent.

**Table 24 – ECM to expander logical phy confirmations**

Confirmation	Description
Arbitrating (Normal)	Confirmation that the ECM has received the Request Path request.
Arbitrating (Waiting On Partial)	Confirmation that the ECM is waiting on a partial pathway (see 4.1.11).
Arbitrating (Blocked On Partial)	Confirmation that the ECM is waiting on a blocked partial pathway (see 4.1.11).
Arbitrating (Waiting On Connection)	Confirmation that the ECM is waiting for a connection to complete (see 4.1.12).
Arb Won	Confirmation that an expander logical phy has won path arbitration.
Arb Lost	Confirmation that an expander logical phy has lost path arbitration.
Arb Reject (No Destination)	Confirmation that the request is rejected because the expander device is not configuring (see 4.6.4) and there is no path to the destination.
Arb Reject (Bad Destination)	Confirmation that the request is rejected because the path to the destination maps back to the requesting expander port.
Arb Reject (Connection Rate Not Supported)	Confirmation that the request is rejected because there is a destination port capable of routing to the requested destination SAS address but no phys within the destination port are configured to support the requested connection rate.
Arb Reject (Zone Violation)	Confirmation that the request is rejected because the expander device is not locked and there is a zoning violation (see 4.8.3).
Arb Reject (Pathway Blocked)	Confirmation that the request is rejected because SAS pathway recovery rules require the requesting expander logical phy to back off (see 6.16.5.2.5).
Arb Reject (Retry)	Confirmation that the request is rejected because: <ul style="list-style-type: none"> <li>a) the expander device is configuring (see 4.6.4) and the ECM detects a connection that results in an Arb Reject (No Destination) if the condition is not resolved;</li> <li>b) the expander device is locked (see 4.8.6.2) and the ECM detects a connection that results in an Arb Reject (Zone Violation) if the condition is not resolved; or</li> <li>c) the expander device has reduced functionality (see 4.5.8 and 6.16.5.2.5).</li> </ul>
Fairness Priority (arguments)	Confirmation that includes the following arguments that specify information on the highest priority OPEN address frame requesting access to the logical phy specified by the Phy Identifier argument in a Request Fairness Priority request: <ul style="list-style-type: none"> <li>a) High Priority;</li> <li>b) SMP Open Priority;</li> <li>c) Arbitration Wait Time;</li> <li>d) Connection Rate; and</li> <li>e) Open Destination SAS Address.</li> </ul>

#### 4.5.6.4 ECR interface

Table 25 describes the requests from an expander logical phy to the ECR and the corresponding indications from the ECR to another expander logical phy. The XL state machine (see 6.19) defines when each request is sent.

**Table 25 – Expander logical phy to ECR to expander logical phy requests and indications**

Request/indication	Description
Forward Open (arguments)	Request/indication to forward an OPEN address frame.
Forward Close (arguments)	Request/indication to forward a CLOSE with type (e.g., Normal or Clear Affiliation), primitive parameters, if any, and the phy identifier, if any, of the expander logical phy that received the CLOSE.
Forward Break	Request/indication to forward a BREAK.
Forward Dword	Request/indication to forward a dword.

Table 26 describes the responses from an expander logical phy to the ECR and the corresponding confirmations from the ECR to another expander logical phy. These responses are sent in response to a Forward Open indication. The XL state machine (see 6.19) defines when each response is sent.

**Table 26 – Expander logical phy to ECR to expander logical phy responses and confirmations**

Response/confirmation	Description
Arb Status (Normal)	Confirmation/response that AIP (NORMAL) has been received.
Arb Status (Waiting On Partial)	Confirmation/response that AIP (WAITING ON PARTIAL) has been received.
Arb Status (Waiting On Connection)	Confirmation/response that AIP (WAITING ON CONNECTION) has been received.
Arb Status (Waiting On Device)	Confirmation/response that either: a) AIP (WAITING ON DEVICE) has been received; or b) the expander logical phy has completed the forwarding of an OPEN address frame and has entered the XL6:Open_Response_Wait state.
Open Accept	Confirmation/response that OPEN_ACCEPT has been received.
Open Reject	Confirmation/response that either: a) OPEN_REJECT has been received; or b) that the phy is in the slumber phy power condition (see 4.10.1.4).
Backoff Retry	Confirmation/response that the phy is in the partial phy power condition (see 4.10.1.3) or: a) a higher priority OPEN address frame has been received (see 6.16.4); and b) the source SAS address and connection rate of the received OPEN address frame are not equal to the destination SAS address and connection rate of the transmitted OPEN address frame.
Backoff Reverse Path	Confirmation/response that: a) a higher priority OPEN address frame has been received (see 6.16.4); and b) the source SAS address and connection rate of the received OPEN address frame are equal to the destination SAS address and connection rate of the transmitted OPEN address frame.

#### 4.5.6.5 BPP interface

Table 27 describes the requests from an expander logical phy to the BPP. Requests from the management device server about SMP ZONED BROADCAST requests (see table 362) received from the SMP target port in zoning expander devices with zoning enabled are not described by this standard. See 4.8.5 for more information on how zoning expander devices with zoning enabled handle Broadcasts.

**Table 27 – Expander logical phy to BPP requests**

Request	Description
Broadcast Event Notify (Phy Not Ready)	Request to originate a Broadcast (Change) because the expander logical phy's SP state machine transitioned from the SP15:SAS_PHY_Ready state, SP22:SATA_PHY_Ready state, SP31:SAS_PS_Low_Phy_Power state, SP32:SAS_PS_ALIGN0 state, or SP33:SAS_PS_ALIGN1 state to the SP0:OOB_COMINIT state (see 5.14).
Broadcast Event Notify (SATA Spinup Hold)	Request to originate a Broadcast (Change) because the SATA spinup hold state has been reached (see 5.14 and 5.21) by the expander phy.
Broadcast Event Notify (Identification Sequence Complete)	Request to originate a Broadcast (Change) because the expander logical phy has completed the identification sequence (see 6.11).
Broadcast Event Notify (SATA Port Selector Change)	Request to originate a Broadcast (Change) because the expander phy detected that a SATA port selector appeared or disappeared.
Broadcast Event Notify (Change Received)	Request to forward a Broadcast (Change) because the expander logical phy received a Broadcast (Change). See 6.15 and 6.19.
Broadcast Event Notify (Reserved Change 0 Received)	Request to forward a Broadcast (Reserved Change 0) because the expander logical phy received a Broadcast (Reserved Change 0). See 6.15 and 6.19.
Broadcast Event Notify (Reserved Change 1 Received)	Request to forward a Broadcast (Reserved Change 1) because the expander logical phy received a Broadcast (Reserved Change 1). See 6.15 and 6.19.
Broadcast Event Notify (SES Received)	Request to forward a Broadcast (SES) because the expander logical phy received a Broadcast (SES). See 6.19.
Broadcast Event Notify (Expander Received)	Request to forward a Broadcast (Expander) because the expander logical phy received a Broadcast (Expander). See 6.19.
Broadcast Event Notify (Asynchronous Event Received)	Request to forward a Broadcast (Asynchronous Event) because the expander logical phy received a Broadcast (Asynchronous Event). See 6.19.
Broadcast Event Notify (Reserved 3 Received)	Request to forward a Broadcast (Reserved 3) because the expander logical phy received a Broadcast (Reserved 3). See 6.19.
Broadcast Event Notify (Reserved 4 Received)	Request to forward a Broadcast (Reserved 4) because the expander logical phy received a Broadcast (Reserved 4). See 6.19.

Table 28 describes the indications from the BPP to an expander logical phy. Indications to the management application client to generate SMP ZONED BROADCAST functions from the SMP initiator port in a zoning expander device with zoning enabled are not described. See 4.8.5 for more information on how zoning expander devices with zoning enabled handle Broadcasts.

**Table 28 – BPP to expander logical phy indications**

Message	Description
Transmit Broadcast (type)	Indication to transmit a BROADCAST primitive sequence with the specified type.

#### 4.5.7 Expander device routing

##### 4.5.7.1 Routing attributes and routing methods

Each expander phy in an expander device shall support one of the following routing attributes:

- a) a direct routing attribute;
- b) a table routing attribute; or
- c) a subtractive routing attribute.

The routing attributes allow the ECM to determine which routing method to use when routing connection requests to the expander logical phys in the expander phy:

- a) the table routing method routes connection requests to attached expander devices using an expander route table;
- b) the subtractive routing method routes unresolved connection requests to an attached expander device; or
- c) the direct routing method routes connection requests to attached end devices, the SMP port of an attached expander device, or SAS devices contained in the expander device.

Table 29 describes the routing methods that the ECM uses based on the routing attributes of an expander phy.

**Table 29 – Routing attributes and routing methods**

Routing attribute of an expander phy	Routing method used by ECM for the expander phy	
	If attached to an end device	If attached to an expander device
Direct	Direct <sup>a</sup>	
Table	Direct	Direct for the SAS address of the expander device. Table for SAS addresses beyond the expander device.
Subtractive	Direct	Subtractive
<sup>a</sup> If attached to an expander device, then the ECM is only able to route to the expander device itself through a phy with the direct routing attribute.		



An expander device may have zero or more phys with the table routing attribute. A self-configuring expander device may support table-to-table attachment (i.e., having its table routing phys attached to the table routing phys of other expander devices). An externally configurable expander device shall not support table-to-table attachment.

An expander device shall have at most one defined port containing phys with the subtractive routing attribute.

The SMP REPORT GENERAL function (see 9.4.3.4) reports whether or not the expander device is self-configuring and supports table-to-table attachment. The SMP DISCOVER function (see 9.4.3.10) reports the routing attribute of each expander phy (see 9.4.3.4).

#### **4.5.7.2 Expander device topology routing attribute restrictions**

If an expander device does not support table-to-table attachment, then its table-routing phys shall not be attached to table routing phys in other expander devices (e.g., they may be attached to subtractive routing phys).

If multiple phys within an expander device have subtractive routing attributes and are attached to expander devices, then they shall be attached to phys with identical SAS addresses (i.e., the same expander device).

If multiple phys within an expander device have subtractive routing attributes and are attached to expander devices that do not have identical SAS addresses, then the management application client that is performing the discover process (see 4.6) shall report an error in a vendor specific manner.

#### **4.5.7.3 Connection request routing**

The ECM shall determine how to route a connection request from a source expander logical phy to a destination expander logical phy in a different expander port if the destination expander logical phy is enabled, operating at a valid logical link rate (see 9.4.3.10), and not excluded because of zoning (see 4.8.2) using the following precedence:

- 1) route to an expander logical phy with the direct routing attribute or table routing attribute when the destination SAS address matches the attached SAS address;
- 2) route to an expander logical phy with the table routing attribute when the destination SAS address matches an enabled SAS address in the expander route table;
- 3) route to an expander logical phy with the subtractive routing attribute; and
- 4) return an Arb Reject confirmation (see 4.5.6.3) to the source expander logical phy.

If the destination expander logical phy only matches an expander logical phy in the same expander port from which the connection request originated, then the ECM shall return an Arb Reject confirmation (see 4.5.6.3).

If the destination SAS address of a connection request matches a disabled SAS address in an expander route table, then the ECM shall ignore the match.

If low phy power conditions (see 4.10.1) are enabled in the expander device, then the ECM should use the following precedence in selecting the destination expander logical phy when the expander port contains multiple expander logical phys:

- 1) expander logical phys in the active phy power condition (see 4.10.1.2);
- 2) expander logical phys in the partial phy power condition (see 4.10.1.3); and
- 3) expander logical phys in the slumber phy power condition (see 4.10.1.4).

#### **4.5.7.4 Expander route table**

##### **4.5.7.4.1 Expander route table overview**

An expander device that supports the table routing method shall contain an expander route table. The expander route table is a structure that provides an association between destination SAS addresses (i.e., routed SAS addresses) and the expander phys to which connection requests to those destination SAS addresses are forwarded.

Zoning expander devices include additional fields in their expander route tables (see 4.8.3.4).

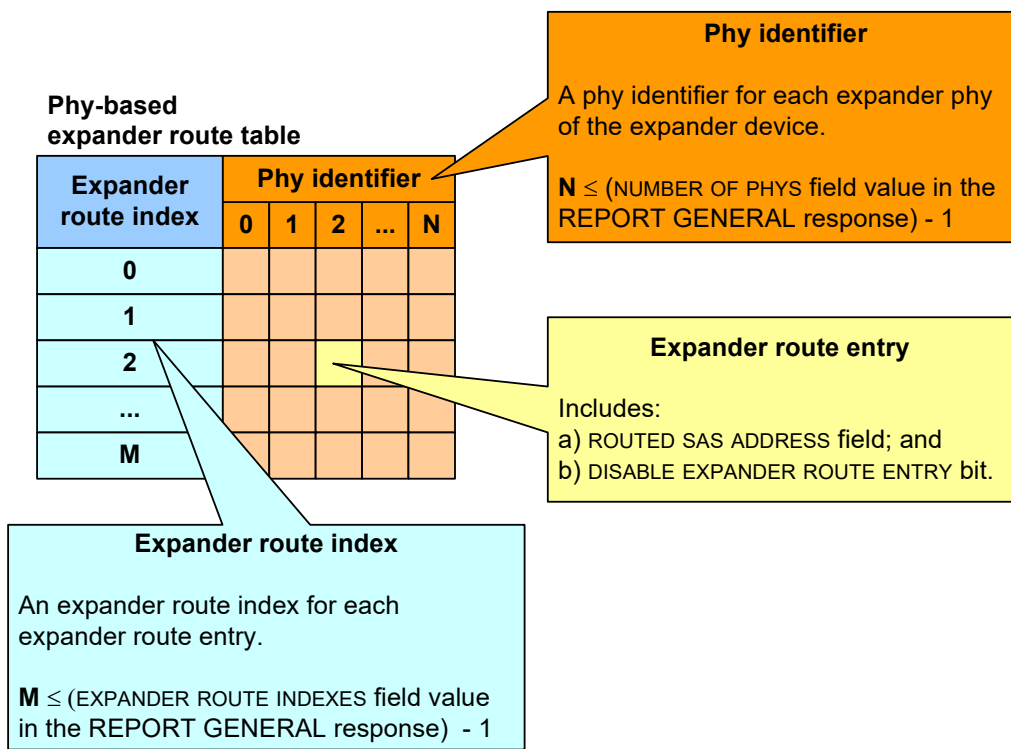
Table 30 defines the types of expander route tables.

**Table 30 – Expander route table types**

Type	SMP functions to access	Reference
Phy-based	REPORT ROUTE INFORMATION (see 9.4.3.13) and CONFIGURE ROUTE INFORMATION (see 9.4.3.27)	4.5.7.4.2
Expander-based	REPORT EXPANDER ROUTE TABLE LIST (see 9.4.3.17)	4.5.7.4.3

#### 4.5.7.4.2 Phy-based expander route table

Figure 43 shows a representation of a phy-based expander route table.



**Figure 43 – Phy-based expander route table**

For each expander route index and phy identifier combination, the phy-based expander route table contains an expander route entry containing a ROUTED SAS ADDRESS field and a DISABLE EXPANDER ROUTE ENTRY bit.

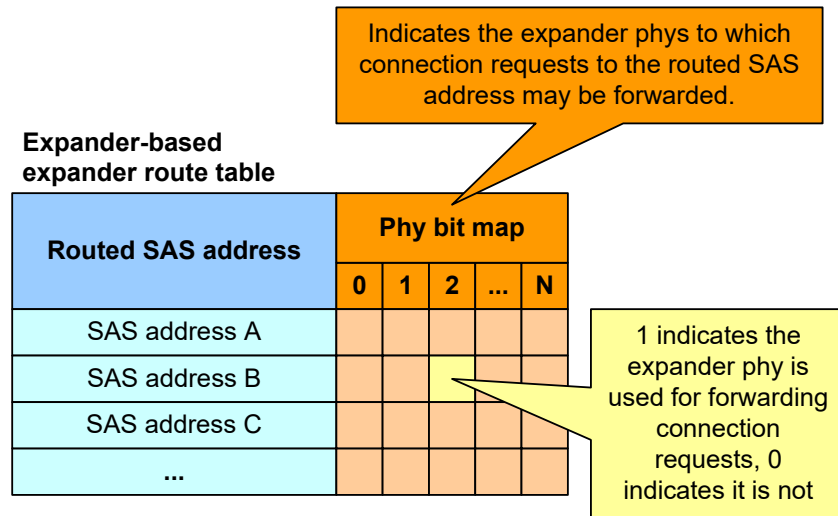
A management application client may access a specific expander route entry within a phy-based expander route table with the SMP REPORT ROUTE INFORMATION function (see 9.4.3.13) and the SMP CONFIGURE ROUTE INFORMATION function (see 9.4.3.27).

An expander device reports the maximum expander route index in the EXPANDER ROUTE INDEXES field and indicates if the phy-based expander route table is configurable in the EXTERNALLY CONFIGURABLE ROUTE TABLE bit in the SMP REPORT GENERAL response (see 9.4.3.4).

Each expander route entry shall be disabled after power on.

#### 4.5.7.4.3 Expander-based expander route table

Figure 44 shows a representation of an expander-based expander route table.



**Figure 44 – Expander-based expander route table**

Routed SAS addresses are not required to be sorted in any particular order.

For each routed SAS address, the expander-based expander route table contains a phy bit map.

A management application client may access an expander-based expander route table with the SMP REPORT EXPANDER ROUTE TABLE LIST function (see 9.4.3.17).

An expander device reports the size of its expander-based expander route table in the MAXIMUM NUMBER OF ROUTED SAS ADDRESSES field in the SMP REPORT GENERAL response (see 9.4.3.4).

#### 4.5.8 Expander device reduced functionality

An expander device shall originate a Broadcast (Expander) to indicate that the expander device is going to have reduced functionality for a period of time (e.g., if, during a microcode update, the expander device disables the ECM and ECR access to its SMP target port or to one or more expander phys, or if the expander device experiences reduced performance). The maximum period of time that the expander device is going to have reduced functionality is indicated:

- in the REPORT SUPPORTED OPERATION CODES command parameter data (see SPC-4) for the WRITE BUFFER command reported by a logical unit with a device type of 0Dh (i.e., enclosure services device) accessed via an SSP target port contained in the expander device; and
- from the contents of the MAXIMUM REDUCED FUNCTIONALITY TIME field in the REPORT GENERAL response (see 9.4.3.4).

After the expander device originates a Broadcast (Expander) to indicate that it is going to have reduced functionality for a period of time, the expander device shall:

- set the REDUCED FUNCTIONALITY bit to one in the REPORT GENERAL response (see 9.4.3.4);
- initialize the reduced functionality delay timer to the value indicated by the INITIAL REDUCED FUNCTIONALITY DELAY field in the REPORT GENERAL response (see 9.4.3.4) and start the reduced functionality delay timer;
- wait for the reduced functionality delay timer to expire before reducing any expander functionality; and
- not stop or restart the reduced functionality delay timer until after the expander device enters the reduced functionality condition.

If the expander device receives a connection request that maps to an expander phy or its SMP target port that is not accessible because of the reduced functionality, then the expander device shall respond with an OPEN REJECT (RETRY) until the operation that caused the expander device to have reduced functionality is complete.

After the operation that caused the expander device to have reduced functionality is complete, the expander device shall:

- 1) set the REDUCED FUNCTIONALITY bit to zero in the REPORT GENERAL response (see 9.4.3.4); and
- 2) originate a Broadcast (Change) or a link reset sequence on each expander phy.

#### 4.5.9 Broadcast (Expander) handling

After receiving a Broadcast (Expander), a management application client behind an SMP initiator port should issue a REPORT GENERAL function (see 9.4.3.4) to all expander devices to determine:

- a) the expander devices, if any, that are reducing their functionality (i.e., the REDUCED FUNCTIONALITY bit is set to one in the REPORT GENERAL response) (see 4.5.8); and
- b) the amount of time remaining until the reduced functionality occurs (i.e., the contents of the TIME TO REDUCED FUNCTIONALITY field in the REPORT GENERAL response).

If an application client determines that an expander device is reducing its functionality, then that application client should:

- a) terminate any outstanding command whose associated I\_T\_L nexus connects through that expander device; and
- b) not create any new commands whose I\_T\_L nexus requires a connection through that expander device.

## 4.6 Discover process

### 4.6.1 Discover process overview

Management application clients direct an SMP initiator port to request SMP functions from an SMP target port. Management application clients are located in every SAS initiator device and every self-configuring expander device. A management application client performs a discover process to discover all the SAS devices and expander devices in the SAS domain (i.e., determining their SAS device types, SAS addresses, and supported protocols). A SAS initiator device uses this information to determine SAS addresses with which it is able to establish connections (i.e., establish I\_T nexuses) and to select connection rates for connection requests (see 6.10.3). A self-configuring expander device uses this information to fill in its expander route table.

### 4.6.2 Starting the discover process (Broadcast (Change) handling)

In a SAS initiator device, a management application client behind an SMP initiator port should perform a discover process after a link reset sequence or after receiving a Broadcast (Change).

In a self-configuring expander device, the management application client behind an SMP initiator port in a self-configuring expander device shall perform a discover process after a link reset sequence or after receiving a Broadcast (Change).

When a discover process is performed after a link reset sequence, the management application client discovers all of the devices in the SAS domain. When a discover process is performed after a Broadcast (Change), the management application client determines which devices have been added to or removed from the SAS domain.

During a discover process a management application client in a SAS initiator device:

- a) may request that the SAS port establish an I\_T nexus loss timer event (see 7.2.2) for each device that has been removed from the SAS domain; and

- b) shall request that the SAS port stop the I\_T nexus loss timer (see 7.2.2), if any, for each device that has been added to the SAS domain.

#### 4.6.3 Discover process traversal

A management application client performing the discover process shall perform a level-order (i.e., breadth-first) traversal of the SAS domain. The management application client shall discover devices in the following order:

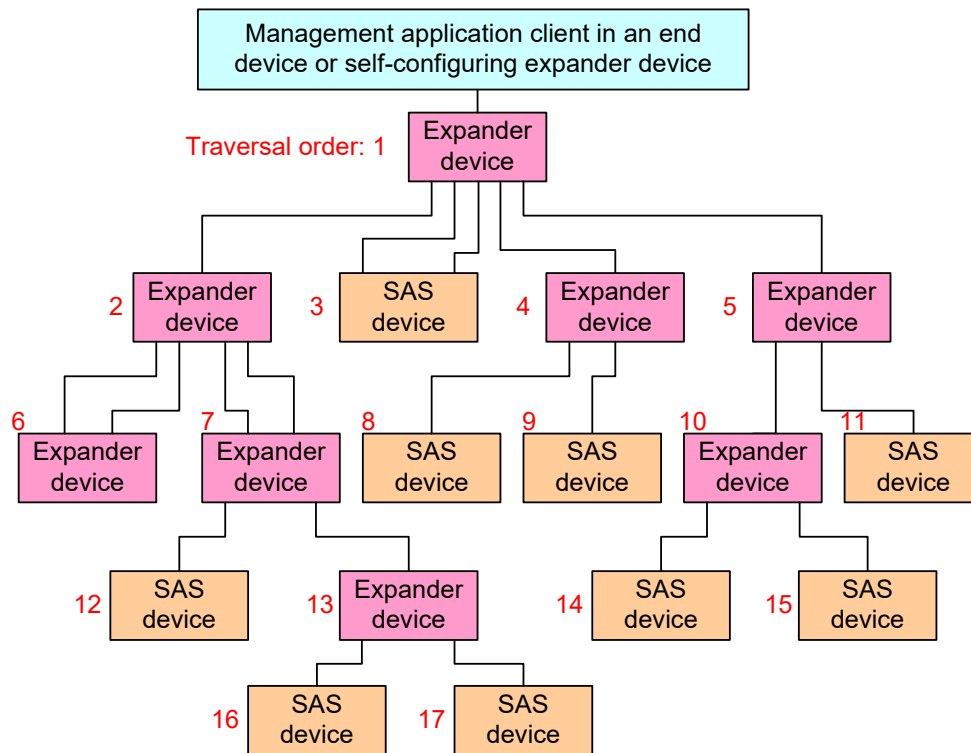
- 1) the devices to which the device containing the management application client is attached;
- 2) for each expander phy with the subtractive routing attribute or the table routing attribute, if the attached device is an expander device, then every device attached to that expander device; and
- 3) repeat step 2) for each additional expander device found attached to that expander device.

The discover process completes when all expander devices have been traversed. If the management application client discovers an externally configurable expander device that is not located beyond a self-configuring expander device with the CONFIGURES OTHERS bit set to one in the REPORT GENERAL response (see 9.4.3.4), then the management application client shall perform the configuration subprocess (see 4.7) to configure the expander route table before attempting to establish connections with devices attached two or more levels (see 4.7.4) beyond that externally configurable expander device.

If a management application client is in an end device or a self-configuring expander device that is directly attached to a self-configuring expander device with the CONFIGURES OTHERS bit set to one in the REPORT GENERAL response (see 9.4.3.4), then the management application client is not required to perform the configuration subprocess. If all the expander devices in the SAS domain are self-configuring expander devices, then management application clients in end devices are not required to perform the configuration subprocess.

If the management application client is inside a self-configuring expander device, then the discover process shall be repeated on each expander port.

Figure 45 shows an example of level-order traversal.



Note - Assume that the phy with the lowest phy identifier in each expander device is on the top right and the remaining phys have increasing phy identifiers assigned in a counter-clockwise direction

**Figure 45 – Level-order traversal example**

The management application client determines whether an expander device or SAS device is attached at each point in the traversal. For the first device (i.e., the device that is directly attached), this is determined from the SAS DEVICE TYPE field in the IDENTIFY address frame (see 6.10.2) information received by the phy that the management application client is using. For other devices (i.e., devices that are not directly attached), this is determined from ATTACHED SAS DEVICE TYPE field the SMP DISCOVER response (see 9.4.3.10).

If an expander device is attached, then the management application client shall use the SMP REPORT GENERAL function (see 9.4.3.4) to determine how many phys are in the expander device and then use the SMP DISCOVER function (see 9.4.3.10) and/or the SMP DISCOVER LIST function (see 9.4.3.15) to determine what is attached to each expander phy (e.g., the SAS device type, SAS address, and supported protocols).

NOTE 7 - Expander devices compliant with SAS-1.1 do not implement the SMP DISCOVER LIST function.

If the expander device's EXTERNALLY CONFIGURABLE ROUTE TABLE bit is set to zero in the SMP REPORT GENERAL response, then its own management application client shall configure its own expander route table as described in 4.7.

If a self-configuring expander device's SELF CONFIGURING bit is set to one in the SMP REPORT GENERAL response, then any connection request in which there is no path to the requested destination returns OPEN\_REJECT (RETRY) instead of OPEN\_REJECT (NO DESTINATION) (see 4.5.6.3, 4.6.4, and 4.8.6.3).

If a SAS device is attached, then the discover process is not required to obtain any more information about the SAS device. Additional discovery software may access that SAS device, however, as follows:

- a) if the SAS device supports an SMP target port, then the management application client may use SMP functions (e.g., REPORT GENERAL and REPORT MANUFACTURER INFORMATION) to determine additional information about the SAS target device;
- b) if the SAS device supports an SSP target port, then a SCSI application client may transmit SCSI commands (e.g., INQUIRY and REPORT LUNS) to determine additional information about the SAS target device; and
- c) if the end device supports an STP target port, then an ATA application client may transmit ATA commands (e.g., IDENTIFY DEVICE and IDENTIFY PACKET DEVICE (see ACS-4)) to determine additional information about the ATA device.

The result of the discover process is that the management application client has the necessary information (e.g., the SAS device type, SAS address, and supported protocols) to communicate with each SAS device and expander device in the SAS domain and each externally configurable expander device is configured with the necessary expander route entries to allow routing of connection requests through the SAS domain.

The discover process may be aborted prior to completion and restarted if there is an indication that it may be using incorrect information (e.g., reception of a Broadcast (Change) or a change in the EXPANDER CHANGE COUNT field returned in an SMP response frame).

#### 4.6.4 Discover process in a self-configuring expander device

The management application client of a self-configuring expander device shall configure:

- a) the expander routing table in that expander device; and
- b) the expander routing table in each externally configurable expander device in the SAS domain that is not located behind another self-configuring expander device with the CONFIGURES OTHERS bit set to one in the REPORT GENERAL response (see 9.4.3.4).

When a self-configuring expander device receives a Broadcast (Change) the management application client shall start the discover process using the expander port that received the Broadcast (Change). If a change to the expander route table is identified, then the management device server shall set its SELF CONFIGURING bit to one in the SMP REPORT GENERAL response (see 9.4.3.4).

If zoning is enabled, then the management application client in a self-configuring expander device shall use the SMP DISCOVER response (see 9.4.3.10) or SMP DISCOVER LIST response (see 9.4.3.15) values to set the zone group values in the zoning expander route table for all SAS addresses in the zoning expander route table (see 4.8.3.4).

The management device server shall set the SELF CONFIGURING bit to zero when the discover process is complete. When the SELF CONFIGURING bit changes from one to zero:

- a) a zoning expander device with zoning enabled shall originate a Broadcast (Change) on each expander port that has access to the expander port through which the discover process was performed based on the zone permission table; and
- b) an expander device with zoning disabled shall originate a Broadcast (Change) on each expander port other than the one through which the discover process was performed.

After receiving a Broadcast (Change), a self-configuring expander device shall continue to route connection requests for each previously valid SAS address until the expander device determines that the SAS address is no longer valid. After determining that a SAS address is no longer valid, the self-configuring expander device shall continue to route connection requests for other SAS addresses.

If the SELF CONFIGURING bit is set to one and there is a connection request in which there is no path to the requested destination, then the expander device shall return OPEN\_REJECT (RETRY) instead of OPEN\_REJECT (NO DESTINATION) (see 4.5.6.3).

The management application client in a self-configuring expander device shall maintain self-configuration status for the last vendor specific number of errors encountered during self-configuration and should maintain at least one self-configuration status per phy. The management device server shall assign descriptors to the statuses sequentially starting at 0001h and shall return the descriptors in the SMP REPORT SELF-CONFIGURATION STATUS response (see 9.4.3.6). The management device server shall return the index of the last self-configuration status descriptor in the SMP REPORT GENERAL response (see 9.4.3.4), the SMP REPORT SELF-CONFIGURATION STATUS response (see 9.4.3.6), and the SMP DISCOVER LIST response (see 9.4.3.15). The management device server shall wrap the index to 0001h when the highest supported descriptor index has been used.

The management device server shall support self-configuration status descriptor indexes from 0001h to FFFFh. The actual number of self-configuration status descriptors that the management device server maintains for retrieval with the REPORT SELF-CONFIGURATION STATUS request is vendor specific and is indicated by the MAXIMUM NUMBER OF STORED SELF-CONFIGURATION STATUS DESCRIPTORS field defined in the REPORT GENERAL response (see 9.4.3.4). The volatility of these stored descriptors is vendor specific. The management device server shall replace the oldest self-configuration status descriptor with a new one once the number of recorded descriptors exceeds the value indicated by the MAXIMUM NUMBER OF STORED SELF-CONFIGURATION STATUS DESCRIPTORS field.

#### 4.6.5 Enabling multiplexing

For physical links with a physical link rate greater than 1.5 Gbit/s and less than 12 Gbit/s a management application client may configure multiplexing (see 5.20) in expander devices. Self-configuring expander devices may configure multiplexing for their own phys. The choice of whether or not to enable multiplexing on a physical link is vendor specific.

If the SAS domain contains 6 Gbit/s and 3 Gbit/s SAS phys, then the management application clients should:

- a) multiplex each 6 Gbit/s physical link into two 3 Gbit/s logical links; and
- b) not multiplex 3 Gbit/s physical links.

If the SAS domain contains 6 Gbit/s, 3 Gbit/s, and 1.5 Gbit/s SAS phys, then the management application client should:

- a) multiplex each 6 Gbit/s physical link into two 3 Gbit/s logical links; and
- b) multiplex each 3 Gbit/s physical link into two 1.5 Gbit/s logical links.

NOTE 8 - Rate matching is used for 1.5 Gbit/s connections carried on 3 Gbit/s logical links.

### 4.7 Configuration subprocess

#### 4.7.1 Configuration subprocess overview

As part of the discover process (see 4.6), if the management application client discovers an externally configurable expander device, then the management application client performs the configuration subprocess to configure the expander routing table in that externally configurable expander device with SAS addresses discovered two or more levels beyond each table routing phy in that externally configurable expander device. A single discover process performs the configuration subprocess at least once per externally configurable expander device.

Configuring the routing table in an expander device is required before connections are able to be established with devices attached two or more levels beyond that expander device.

#### 4.7.2 Allowed expander device topologies

If the management application client detects:

- a) an expander phy with the table routing attribute in an externally configurable expander device; or



- b) an expander phy with the table routing attribute in a self-configuring expander device with the TABLE TO TABLE bit set to zero in the SMP REPORT GENERAL response,

attached to:

- a) an expander phy with either the direct routing attribute or the table routing attribute in either an externally configurable expander device or a self-configuring expander device,

then the management application client shall report an error in a vendor specific manner.

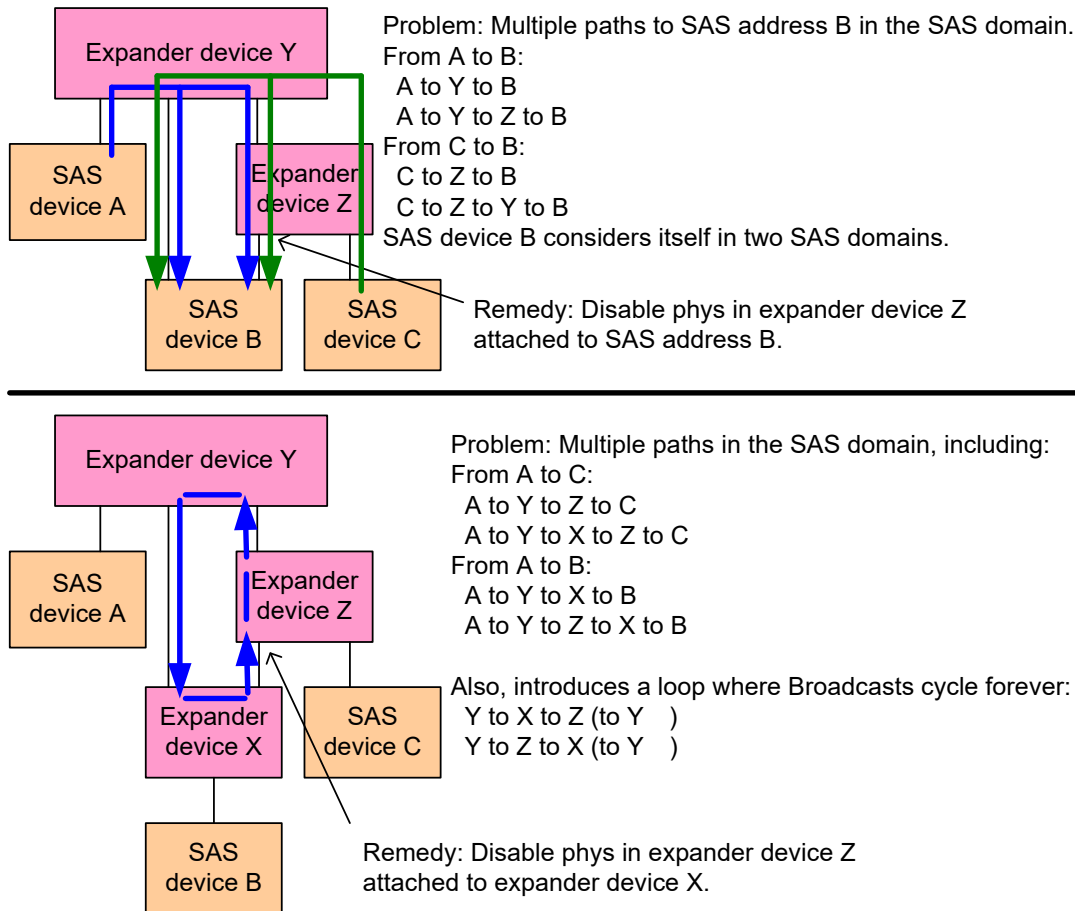
If the management application client detects an overflow of an expander route index, then it shall report an error in a vendor specific manner.

If the route table optimization (see 4.7.3) is disabled and the management application client detects an expander route entry that references the SAS address of the expander device itself (i.e., self-reference), then the management application client shall disable the expander route entry by setting the DISABLE EXPANDER ROUTE ENTRY bit to one in the SMP CONFIGURE ROUTE INFORMATION request (see 9.4.3.27). The management application client shall disable each expander route entry in the expander route table by setting the DISABLE EXPANDER ROUTE ENTRY bit to one in the SMP CONFIGURE ROUTE INFORMATION request (see 9.4.3.27) for each expander phy that has its attached SAS device type of 000b (i.e., no device attached).

If the management application client detects a port that:

- a) the configuration subprocess has not already configured with a SAS address; and
- b) it has already found attached to another expander device,

then the management application client should use the SMP PHY CONTROL function (see 9.4.3.28) to disable all the expander phys attached to that SAS address except for phys in the expander device with the lowest SAS address. Figure 46 shows some invalid topologies.



**Figure 46 – Examples of invalid topologies**

#### 4.7.3 Externally configurable expander device route table optimization

The management application client shall support a route table optimization that reduces the number of entries required in an expander route table in an externally configurable expander device. The method used to enable and disable the route table optimization is vendor specific.

If the route table optimization is enabled, then the management application client shall exclude discovered SAS addresses from the expander route table when any of the following conditions are met:

- a) in the SMP DISCOVER response (see 9.4.3.10) for the discovered phy:
  - A) the FUNCTION RESULT field is set to a non-zero value (i.e., not SMP FUNCTION ACCEPTED);
- b) in the SMP DISCOVER response for the discovered phy:
  - A) the FUNCTION RESULT field is set to zero (i.e., SMP FUNCTION ACCEPTED);
  - B) the ROUTING ATTRIBUTE field is set to 1h (i.e., subtractive) or 2h (i.e., table); and
  - C) the ATTACHED SAS DEVICE TYPE field is set to zero (i.e., no device attached);
- c) in the SMP DISCOVER response for the discovered phy:
  - A) the FUNCTION RESULT field is set to zero (i.e., SMP FUNCTION ACCEPTED);

- B) the ROUTING ATTRIBUTE field is set to 1h (i.e., subtractive) or 2h (i.e., table);
  - C) the ATTACHED SAS DEVICE TYPE field is set to a non-zero value (e.g., end device or expander device); and
  - D) the ATTACHED SAS ADDRESS field contains the SAS address of the expander device being configured (i.e., a self-referencing address);
- d) in the SMP DISCOVER response for the discovered phy:
- A) the FUNCTION RESULT field is set to zero (i.e., SMP FUNCTION ACCEPTED);
  - B) the ROUTING ATTRIBUTE field is set to 1h (i.e., subtractive) or 2h (i.e., table);
  - C) the ATTACHED SAS DEVICE TYPE field is set to a non-zero value (e.g., end device or expander device); and
  - D) the ATTACHED SAS ADDRESS field contains the SAS address of a device directly attached to the expander device being configured;
- or
- e) in the SMP DISCOVER response for the discovered phy:
- A) the FUNCTION RESULT field is set to zero (i.e., SMP FUNCTION ACCEPTED);
  - B) the ROUTING ATTRIBUTE field is set to 1h (i.e., subtractive) or 2h (i.e., table);
  - C) the ATTACHED SAS DEVICE TYPE field is set to a non-zero value (e.g., end device or expander device); and
  - D) the ATTACHED SAS ADDRESS field contains a SAS address that already exists in the expander route table.

If the discovered SAS address being included in the expander route table is for a device that is not attached (i.e., the ATTACHED SAS DEVICE TYPE field is set to zero (i.e., no device attached) and the ROUTE ATTRIBUTE field is set to 0h (i.e., direct)), then the entry shall be inserted with the ROUTED SAS ADDRESS field set to 00000000 00000000h and the DISABLE EXPANDER ROUTE ENTRY bit set to one (see 9.4.3.27).

If route table optimization is disabled, then all SAS addresses shall be qualified for insertion in the expander route table.

If the management application client supports route table optimization, then the management application client should provide a vendor specific method for initiating a check of the resulting expander route tables. The check should be performed under the following situations:

- a) when an I\_T nexus loss occurs for a destination port that is expected to be present;
- b) when a discover process has been completed;
- c) when another SMP initiator port is discovered in the SAS domain; or
- d) when a self-configuring expander device is discovered in the SAS domain.

If the management application client detects an inconsistency in the expander route tables when route table optimization is enabled (e.g., detects entries that appear to have been filled in by a discover process with route table optimization disabled), then the management application client shall report an error in a vendor specific manner and shall disable route table optimization. The management application client should then reinitiate a discover process with route table optimization disabled.

#### 4.7.4 Externally configurable expander device expander route index order

The expander route table in an externally configurable expander device shall be configured for each expander phy that has a table routing attribute.

If the expander phy is not attached to an expander device, then every expander route entry for that expander phy shall be disabled (i.e., the ROUTED SAS ADDRESS field shall be set to 00000000 00000000h and the DISABLE EXPANDER ROUTE ENTRY bit shall be set to one).

If the expander phy is attached to an expander device, then the expander route table shall be configured for that expander phy as follows. For purposes of configuring the expander route table for that phy, the expander devices attached to the expander phy are assigned levels:

- 1) the expander device in which the expander route table is being configured is level 0;
- 2) the attached expander device is considered level 1;

- 3) devices attached to the level 1 expander device, except for the level 0 expander device, are considered level 2;
- 4) devices attached to level 2 expander devices, except for level 1 expander devices, are considered level 3; and
- 5) for each  $n$  greater than 3, devices attached to level  $n-1$  expander devices, except for level  $n-2$  expander devices, are considered level  $n$ .

The expander route table for each expander phy shall be configured starting from expander route index 0 by level (e.g., if there are three levels, then all level 1 entries first, then all level 2 entries, then all level 3 entries) up to the value of the EXPANDER ROUTE INDEXES field reported by the SMP REPORT GENERAL function (see 9.4.3.4).

If the level 1 expander device has expander phys attached to  $N$  phys with qualified SAS addresses (see 4.7.3), then the first  $N$  entries shall be used for those SAS addresses in expander phy order (i.e., the addresses attached to lower expander phy numbers first).

For each of the level 2 devices that:

- a) is an expander device attached to  $M$  phys with qualified SAS addresses; and
- b) is attached to an expander phy in the level 1 expander device with the table routing attribute,

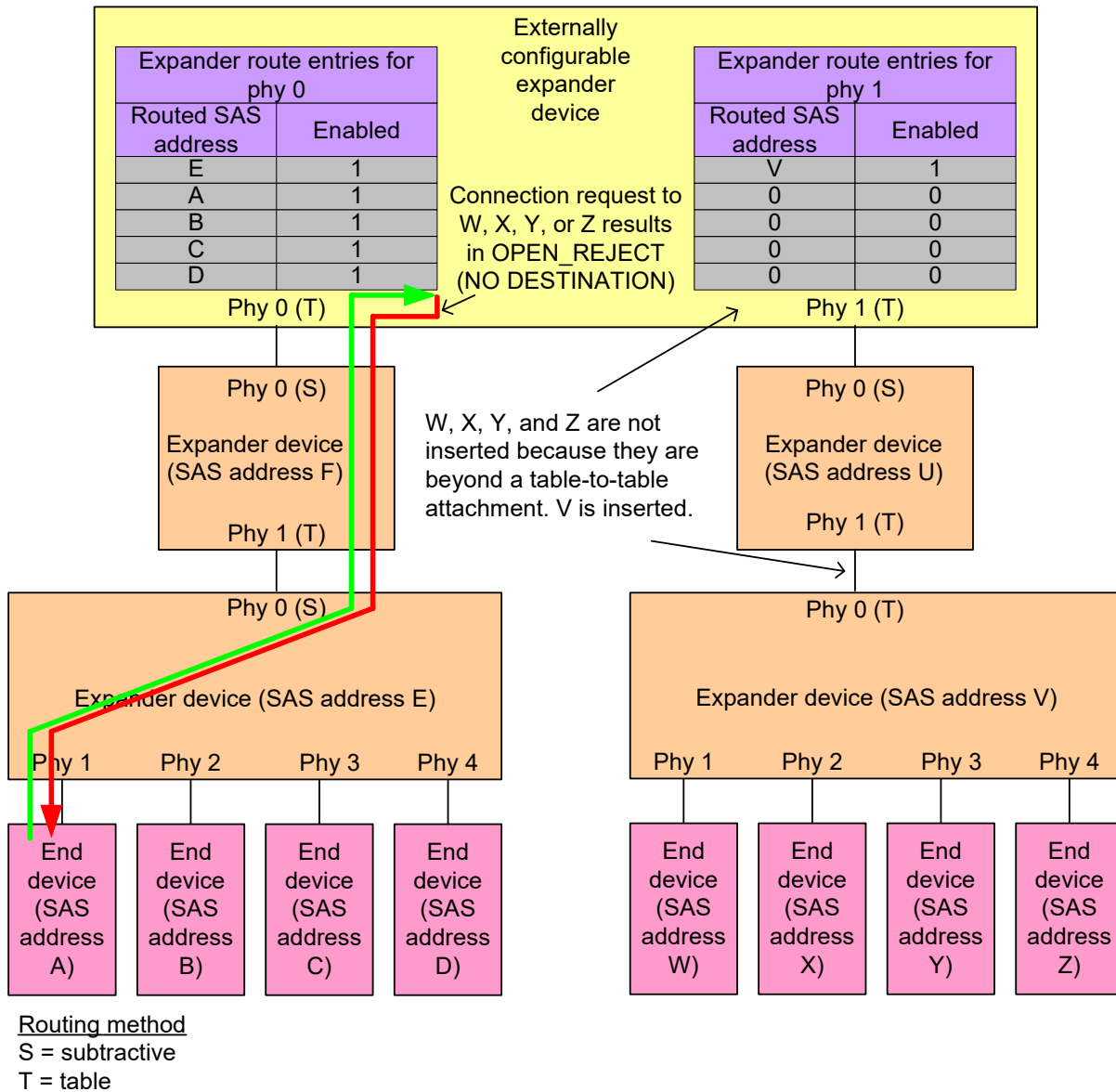
the next  $M$  entries shall be used for the level 2 expander device's qualified SAS addresses in expander phy order (i.e., lower phy numbers first).

This process shall be repeated for all levels of expander devices.

SAS addresses of devices attached beyond expander phys that are attached table-to-table shall not be included in the expander route table. The SAS address of the first expander device that is attached table-to-table shall be included and the SAS address of every device attached beyond that expander device shall not be included. As a result, end devices in SAS domains containing externally configurable expander devices and table-to-table attachments may not be able to establish connections to each other.

NOTE 9 - Not including those SAS addresses provides compatibility with management application clients compliant with SAS-1.1. End devices in SAS domains containing only self-configuring expander devices supporting table-to-table attachments are able to establish connections to any other end device.

Figure 47 shows an example of a route table that does not include SAS addresses beyond a table-to-table attachment.



**Figure 47 – Externally configurable expander device and table-to-table attachment**

After the expander route table has been configured with entries for all levels of expander devices, all remaining expander route entries, if any, shall be disabled (i.e., the ROUTED SAS ADDRESS field shall be set to 00000000 00000000h and the DISABLE EXPANDER ROUTE ENTRY bit shall be set to one). The management application client is not required to disable entries if the topology of expander devices has not changed.

Figure 48 shows a portion of a SAS domain, where phy A in expander device R is being configured.

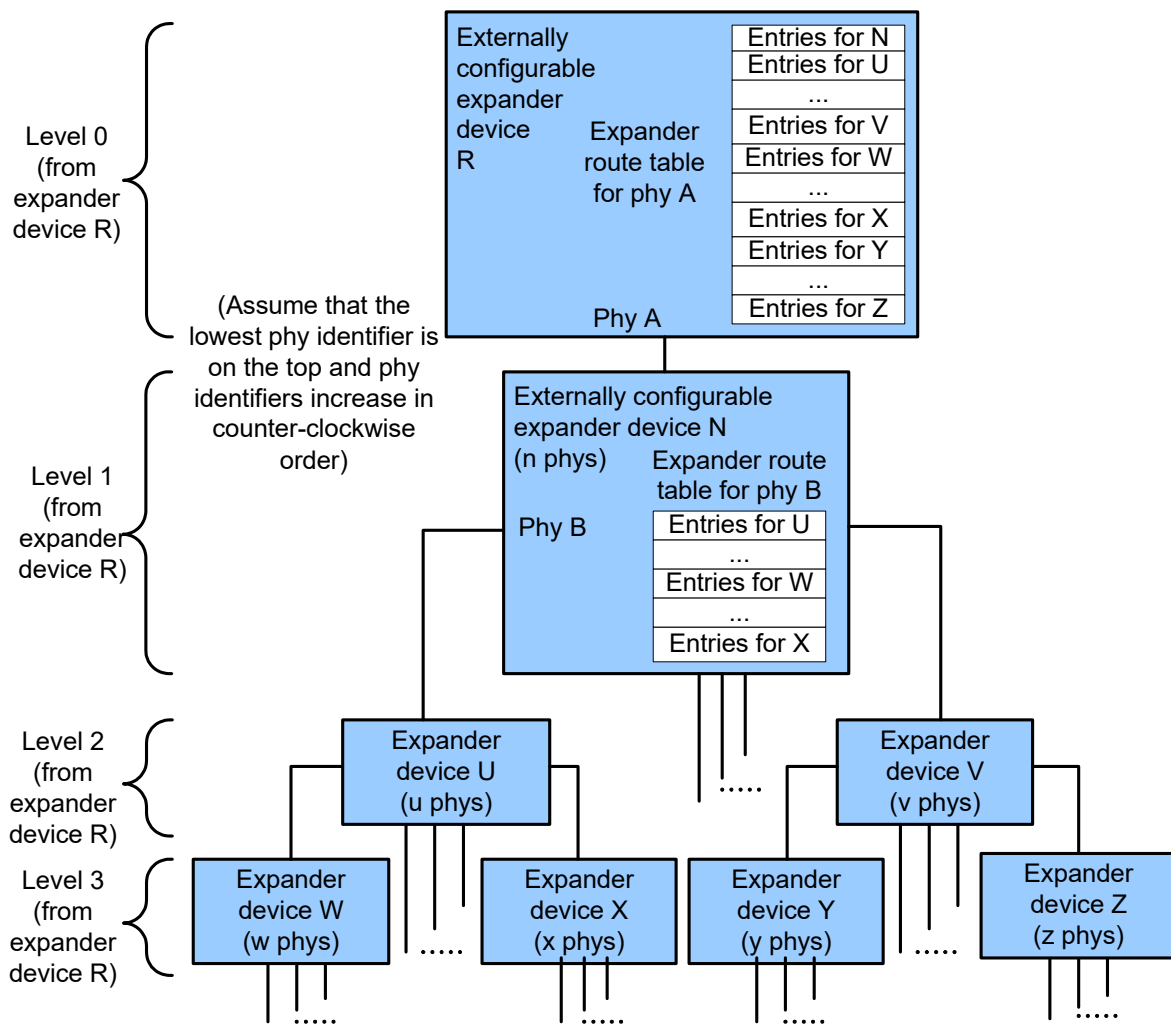


Figure 48 – Expander route index levels example

Table 31 shows how the expander route table is configured for externally configurable expander device R phy A in figure 48.

**Table 31 – Expander route table levels for externally configurable expander device R phy A**

Expander route index	Expander route entry contents
Level 1 (from expanded device R) entries	
0 to ( $\leq n$ entries)	Qualified SAS addresses attached to expander device N
Level 2 (from expanded device R) entries	
( $\leq u$ entries)	Qualified SAS addresses attached to expander device U
...	...additional qualified SAS addresses for expander devices at level 2...
( $\leq v$ entries)	Qualified SAS addresses attached to expander device V
Level 3 (from expanded device R) entries	
( $\leq w$ entries)	Qualified SAS addresses attached to expander device W
...	...additional qualified SAS addresses for expander devices at level 3...
( $\leq x$ entries)	Qualified SAS addresses attached to expander device X
( $\leq y$ entries)	Qualified SAS addresses attached to expander device Y
...	...additional qualified SAS addresses for expander devices at level 3...
( $\leq z$ entries)	Qualified SAS addresses attached to expander device Z
Entries for additional levels	
...	...
Disabled entries	
...	...

Table 32 shows how the expander route table is configured for externally configurable expander device N phy B in figure 48.

**Table 32 – Expander route table levels for externally configurable expander device N**

Expander route index	Expander route entry contents
Level 1 (from expanded device N) entries	
0 .. ( $\leq u$ entries)	Qualified SAS addresses attached to expander device U
Level 2 (from expanded device N) entries	
( $\leq w$ entries)	Qualified SAS addresses attached to expander device W
...	...additional qualified SAS addresses for expander devices at level 2...
( $\leq x$ entries)	Qualified SAS addresses attached to expander device X
Entries for additional levels	
...	...
Disabled entries	
...	...



Figure 49 shows an example topology.

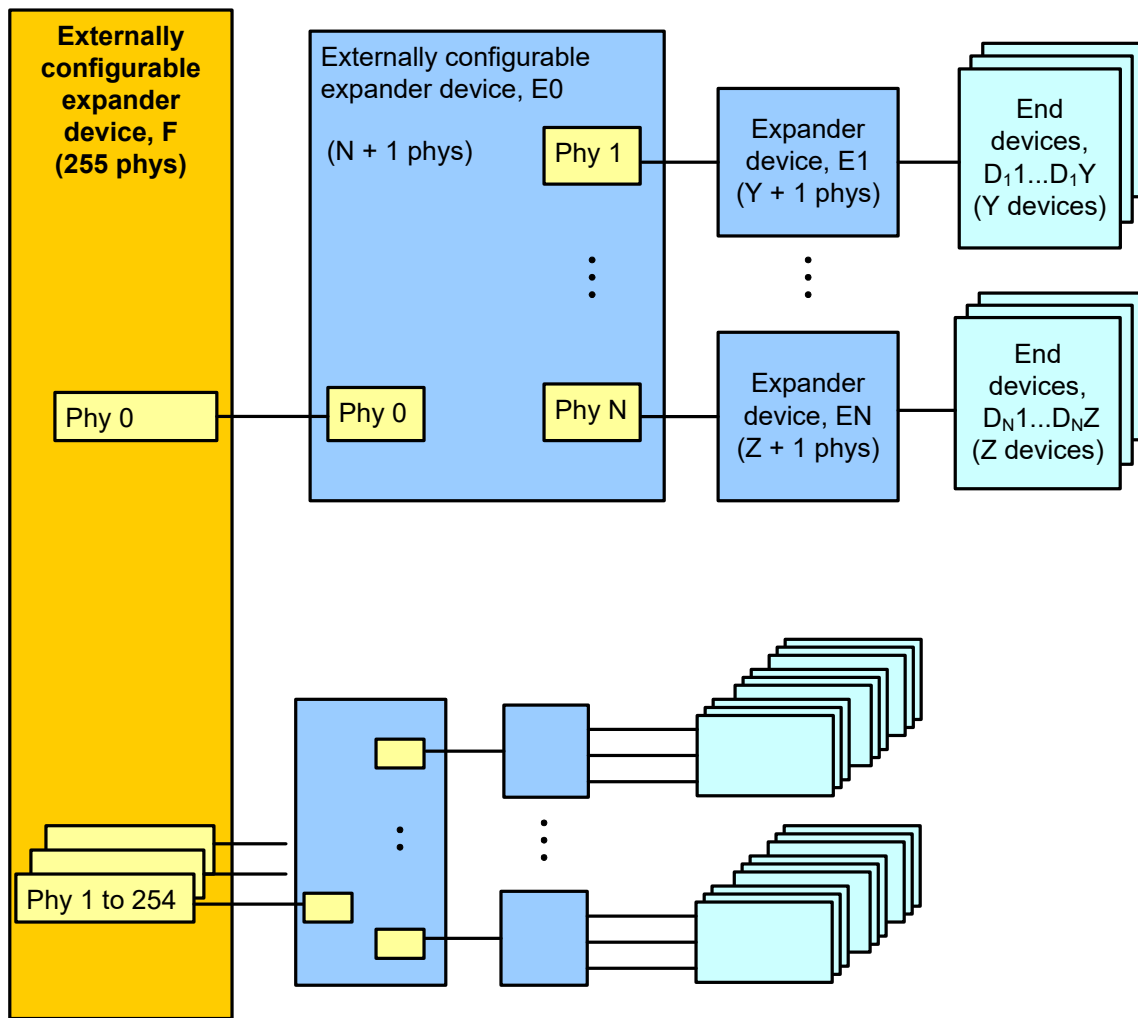


Figure 49 – Expander route index order example

Table 33 shows the expander route index order for externally configurable expander device E0 phy 1 in figure 49, assuming that all phys are present and not subject to exclusion by route table optimization (see 4.7.3).

**Table 33 – Expander route entries for externally configurable expander device E0 phy 1**

Expander route index	Expander route entry contents
Level 1 entries	
0	SAS address (e.g., D <sub>1</sub> 1) of the device attached to phy 1 of expander device E1
1	SAS address (e.g., D <sub>1</sub> 2) of the device attached to phy 2 of expander device E1
...	...
Y - 1	SAS address (e.g., D <sub>1</sub> Y) of the device attached to phy Y of expander device E1
Level 2 and beyond	
	No entries
Disabled entries	
	Any remaining entries are disabled

Table 34 shows the expander route index order for externally configurable expander device F phy 0 in figure 49, assuming that all phys are present and not subject to exclusion by route table optimization (see 4.7.3).

**Table 34 – Expander route entries for externally configurable expander device F phy 0**

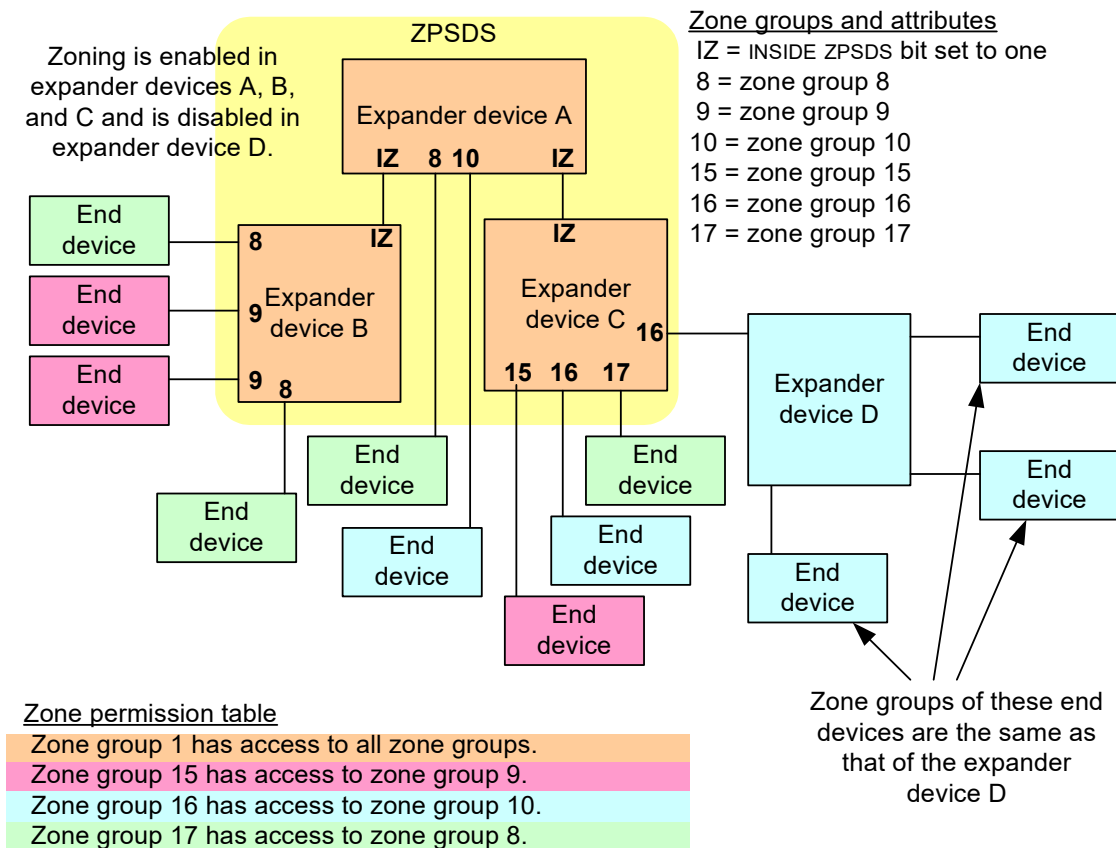
Expander route index	Expander route entry contents
Level 1 entries	
0	SAS address (e.g., E1) of the device attached to phy 1 of expander device E0
...	...additional qualified SAS addresses for expander device E0...
N - 1	SAS address (e.g., EN) of the device attached to phy N of expander device E0
Level 2 entries	
N	SAS address (e.g., D <sub>1</sub> 1) of the device attached to phy 1 of expander device E1
...	...additional qualified SAS addresses for expander device E1...
	SAS address (e.g., D <sub>1</sub> Y) of the device attached to phy Y of expander device E1
...	...additional qualified SAS addresses for expander devices E2 to EN-1...
	SAS address (e.g., D <sub>N</sub> 1) of the device attached to phy 1 of expander device EN
...	...additional qualified SAS addresses for expander device EN...
	SAS address (e.g., D <sub>N</sub> Z) of the device attached to phy Z of expander device EN
Level 3 and beyond	
	No entries since all devices attached to E1 to EN, except for E0, are end devices
Disabled entries	
	Any remaining entries are disabled

## 4.8 Zoning

### 4.8.1 Zoning overview

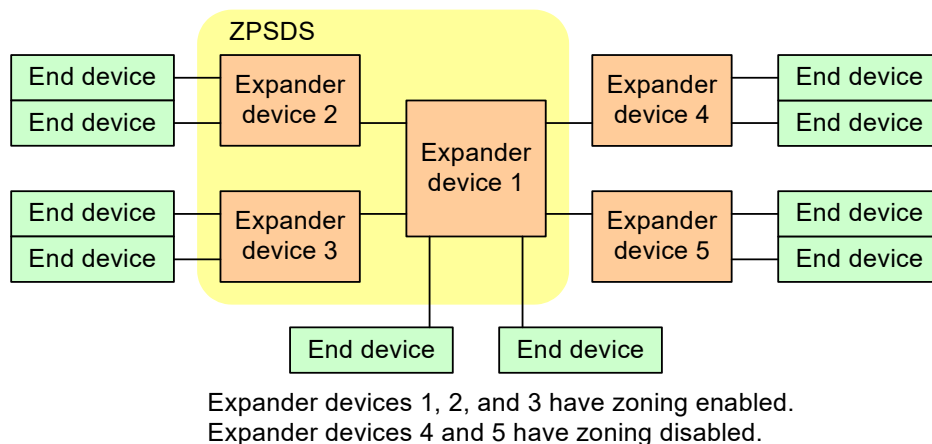
SAS zoning is implemented by a set of zoning expander devices with zoning enabled that define a zoned portion of a service delivery subsystem (ZPSDS). The zoning expander devices control whether a phy is permitted to participate in a connection to another phy.

Figure 50 shows an example of zoning.



**Figure 50 – Zoning example**

Figure 51 shows an example of one ZPSDS in a SAS domain.



**Figure 51 – One ZPSDS example**

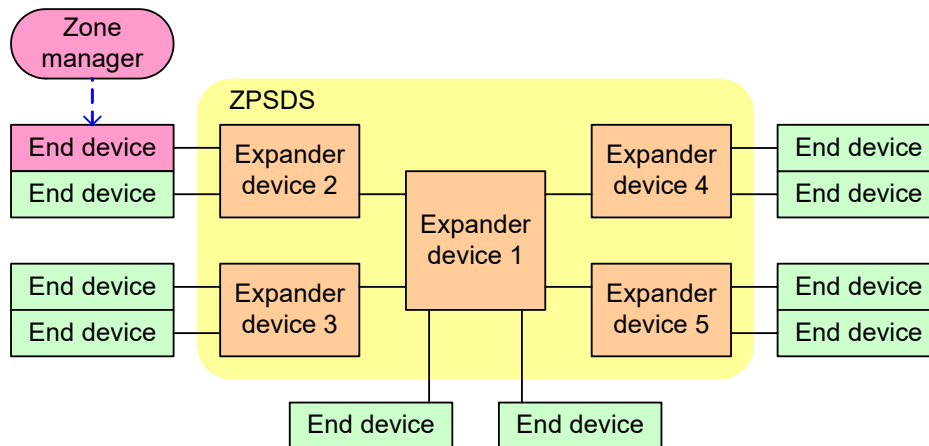
A ZPSDS has a zone manager responsible for its configuration. The zone manager may have access to:

- a) an end device with a SAS port whose zone group (see 4.8.3.2) has access to zone group 2; or

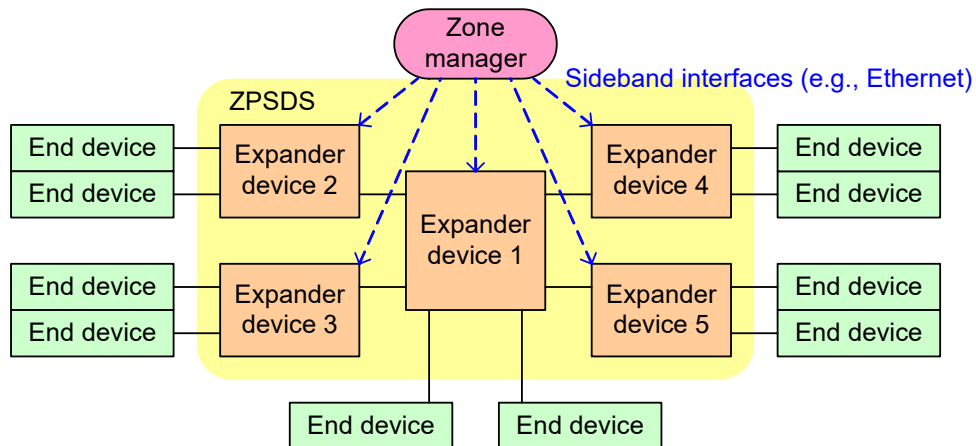
- b) one or more zoning expander devices through a sideband interface (e.g., Ethernet) outside the scope of this standard. The SAS address reported for a sideband zone manager is 00000000 00000000h.

Figure 52 shows examples of zone manager locations in a SAS domain.

Zone manager attached to an end device with a SAS port whose zone group has access to zone group 2



Zone manager attached directly to the expander devices in the ZPSDS



Zone manager attached directly to one expander device in the ZPSDS

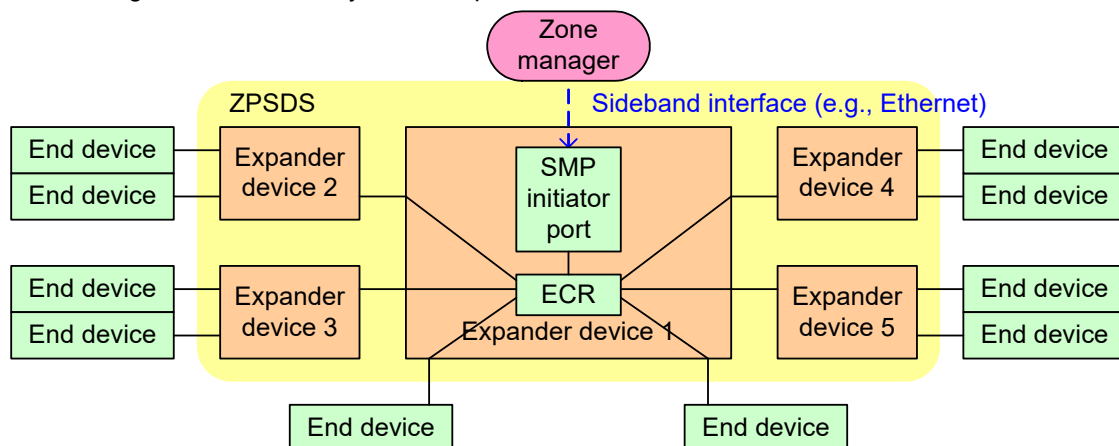
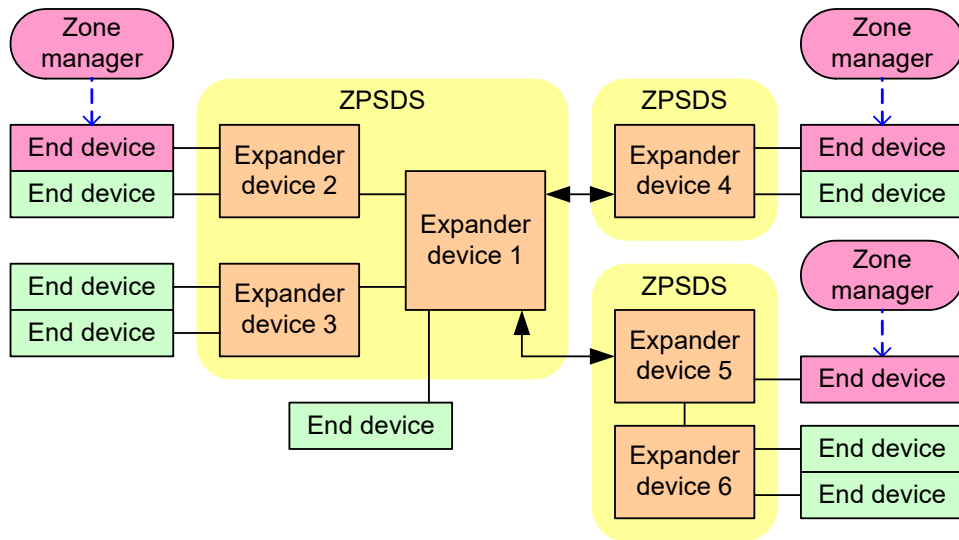


Figure 52 – Zone manager location examples

There may be any number of non-overlapping ZPSDSes in a service delivery subsystem, particularly as a SAS domain is being reconfigured (e.g., as a user is attaching enclosures together). A SAS domain with more than one ZPSDS should be transitory. A ZPSDS may encompass some or all of a service delivery subsystem.

Figure 53 shows an example of three ZPSDSes in a SAS domain.



**Figure 53 – Three ZPSDSes example**

The zone manager assigns zone groups (see 4.8.3.2) to all zoning expander phys inside the ZPSDS. There are 128 or 256 zone groups numbered 0 to 127 or 0 to 255. All phys in a wide port shall be assigned to the same zone group. Zone groups are assigned to zoning expander phys as part of the zone phy information (see 4.8.3.1) and are stored along with SAS addresses in the zoning expander route table (see 4.8.3.4). The zone groups assigned in one ZPSDS have no relationship to the zone groups assigned in another ZPSDS.

The zone manager shall assign each zoning expander phy attached to another zoning expander phy inside a ZPSDS to zone group 1. The zone manager shall assign each zoning expander phy on the boundary of the ZPSDS (i.e., with the INSIDE ZPSDS bit set to zero) to a zone group. All phys in the SAS domain beyond that boundary zoning expander phy are considered to be in the same zone group as that zoning expander phy.

Each zoning expander device contains a zone permission table (see 4.8.3.3) that controls whether a connection is allowed between phys based on their zone groups. As defined in 4.8.3.5, a requested connection shall only be established if the zone permission table indicates that access between the zone group of the source port and the zone group of the destination port is allowed.

The zoning expander route table (see 4.8.3.4) is an extended version of the expander route table (see 4.5.7.4) that also includes the zone group of each SAS address.

Physical presence detection is a mechanism used to allow management access. The definition of physical presence detection is vendor specific (e.g., a user pressing a button or inserting a key).

The zone manager password is a value used to allow management access. The zone manager password is 32 bytes long and is specified in table 35.

**Table 35 – Zone manager password**

Code	Name	Description
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000h	ZERO	Well-known value that provides access to any zone manager that presents it.
FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFFh	DISABLED	Well-known value that does not provide access to any zone manager (e.g., zone manager password usage is disabled).
All others		Random value, providing access only to a zone manager that presents the correct value.

The expander device:

- a) shall maintain a current value;
- b) shall maintain a shadow value;
- c) may maintain a saved value; and
- d) shall have a default value,

for each of the following settings:

- a) zoning enabled;
- b) the zone permission table; and
- c) zone phy information.

The expander device:

- a) shall maintain a current value;
- b) may maintain a saved value; and
- c) shall have a default value,

for the zone manager password, if any.

Support or lack of support for saved values for one setting does not imply support or lack of support for saved values for any other setting (e.g., the expander device may maintain a saved value for zoning enabled but not for the zone permission table).

For each setting, after power on or expander device reduced functionality, the expander device shall set the current value to the saved value, if any, or the default value, if there is no saved value.

#### 4.8.2 Zoning expander device requirements

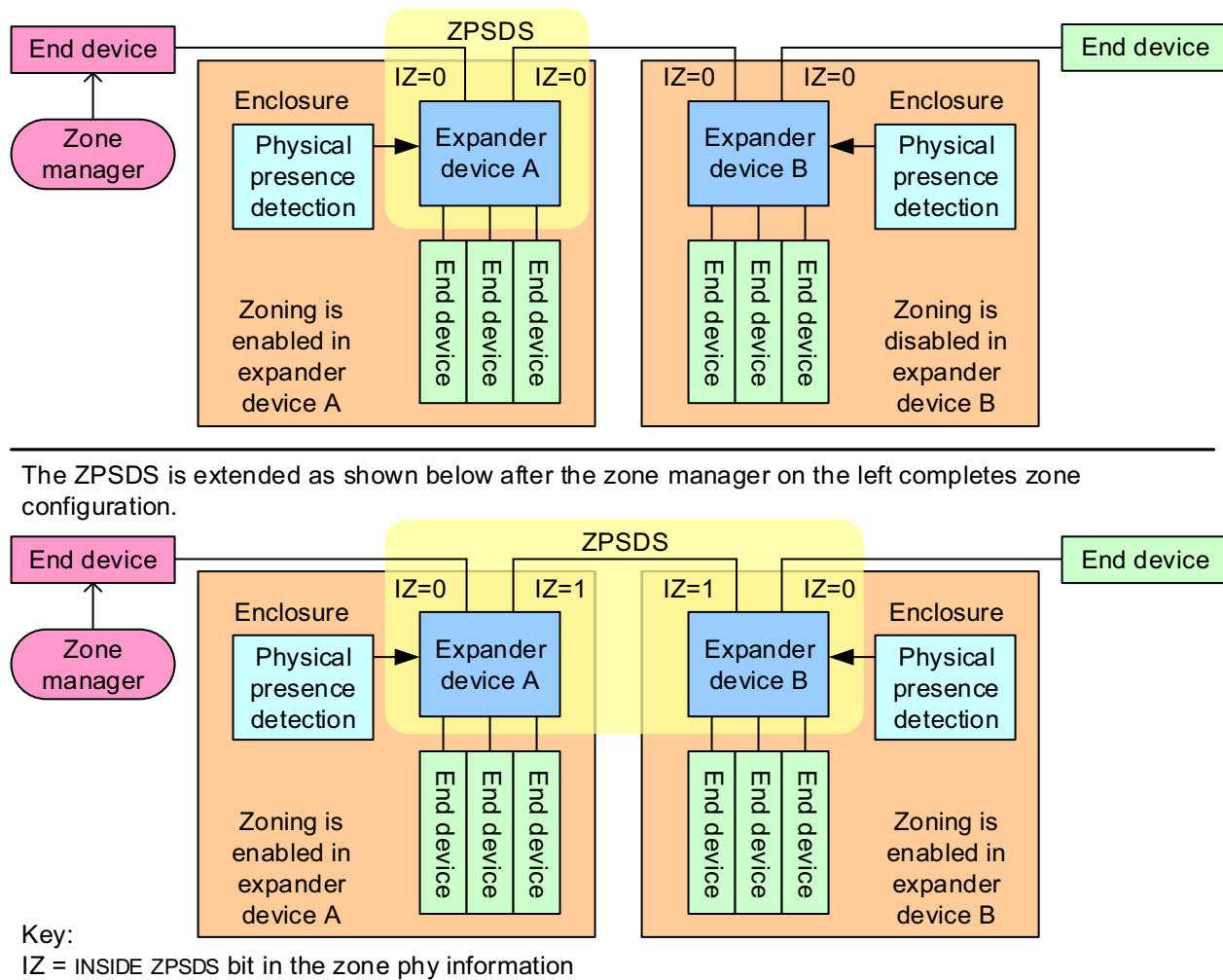
In addition to the requirements for expander devices described in 4.5, a zoning expander device shall:

- a) contain a zoning expander route table (see 4.8.3.4);
- b) contain current and shadow zone permission tables that supports 128 or 256 zone groups (see 4.8.3.3);
- c) contain current and shadow zone phy information for each phy;
- d) if zoning is enabled, then allow or deny connection requests based on the current zone permission table (see 4.8.3.5);
- e) support fields related to zoning in its SMP REPORT GENERAL response (see 9.4.3.4);
- f) support the zone lock inactivity timer;

- g) be self-configuring;
- h) contain an SMP initiator port (see 4.5.1); and
- i) support zoning-related SMP functions.

A zoning expander device may support physical presence detection and/or a zone manager password to allow management access.

Figure 54 shows an example of two enclosures with physical presence detection where zoning is enabled in the expander device in the left enclosure, but is not enabled in the expander devices in the right enclosure. The zone manager is able to configure zoning in zoning expander device A because the zone group of its SMP initiator port has access to zone group 2. However, the zone manager is not able to enable or configure zoning in expander device B unless physical presence is asserted or the zone manager presents the correct zone manager password for that expander device.



**Figure 54 – Extending a ZPSDS example**



Figure 55 shows an example of two enclosures with physical presence detection where zoning is enabled in both expander devices. The zone manager is able to configure zoning in zoning expander device A because the zone group of its SMP initiator port has access to zone group 2. However, the zone manager is not able to configure zoning in expander device B unless physical presence is asserted or the zone manager presents the correct zone manager password for that expander device.

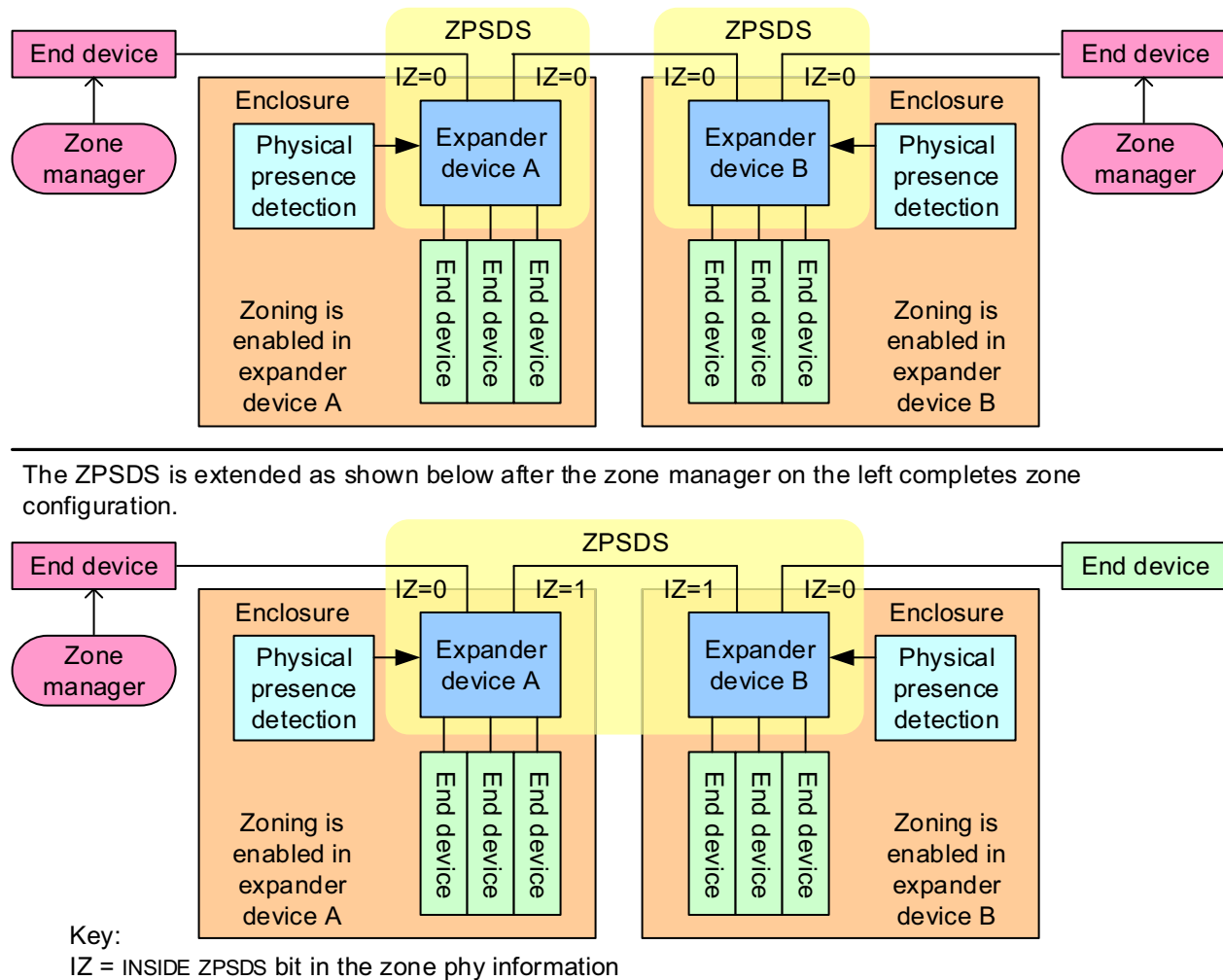


Figure 55 – Overtaking a ZPSDS example

### 4.8.3 Zoning operation

#### 4.8.3.1 Zone phy information

Each phy of a zoning expander device shall support the zone phy information fields defined in table 36.

**Table 36 – Zone phy information**

Field	Description	Recommended default
INSIDE ZPSDS bit	Indicates if the phy is inside or on the boundary of a ZPSDS	N/A <sup>a</sup>
REQUESTED INSIDE ZPSDS bit	Used to establish the boundary of the ZPSDS	0
INSIDE ZPSDS PERSISTENT bit	Used to determine the value of the INSIDE ZPSDS bit after a link reset sequence	0
ZONE GROUP PERSISTENT bit	Used to determine the zone group of the phy after a link reset sequence if the INSIDE ZPSDS bit is set to zero	0
ZONE GROUP field	The zone group to which the phy belongs	00h
<sup>a</sup> The INSIDE ZPSDS bit is determined from the values exchanged during the link reset sequence.		

Table 37 lists the usage of the current values of the zone phy information fields.

**Table 37 – Zone phy information usage**

Field	Transmitted in IDENTIFY address frame <sup>a</sup>	Indicated in DISCOVER function and DISCOVER LIST function <sup>b</sup>	Attached value indicated in DISCOVER function <sup>c</sup>	Programmable with the CONFIGURE ZONE PHY INFORMATION function <sup>d</sup>	Changeable by the expander device after a link reset sequence <sup>e</sup>
INSIDE ZPSDS bit	No	Yes	No	No	Yes
REQUESTED INSIDE ZPSDS bit	Yes	Yes	Yes	Yes	Yes
INSIDE ZPSDS PERSISTENT bit	Yes	Yes	Yes	Yes	No
ZONE GROUP PERSISTENT bit	No	Yes	No	Yes	No
ZONE GROUP field	No	Yes	No	Yes	Yes <sup>f</sup>
<sup>a</sup> Defined in the IDENTIFY address frame (see 6.10.2). <sup>b</sup> Defined in the DISCOVER response (see 9.4.3.10) and the DISCOVER LIST response SHORT FORMAT descriptor (see 9.4.3.15.4). <sup>c</sup> Defined in the DISCOVER response (see 9.4.3.10). <sup>d</sup> Defined in the zone phy configuration descriptor (see 9.4.3.25.3). Current values are not updated until the activate step (see 4.8.6.4). The saved values are also programmable with this function. <sup>e</sup> See 4.8.4. <sup>f</sup> Only changes to 00h after a link reset sequence. See 4.8.4.					

All phys in an expander port shall have the same zone phy information.

The expander device shall preserve the zone phy information while:

- a) zoning is disabled;
- b) no power loss occurs; and
- c) there is no expander device reduced functionality (see 4.5.8).

The INSIDE ZPSDS bit indicates if the phy is inside or on the boundary of a ZPSDS. An INSIDE ZPSDS bit set to zero indicates that:

- a) zoning is disabled;
- b) the phy is attached to an end device;
- c) the phy is attached to an expander device that does not support zoning;
- d) the phy is attached to an expander device that supports zoning, but zoning is disabled; or
- e) the phy is attached to an expander device that supports zoning, zoning is enabled, but the phy is outside the ZPSDS (i.e., is in another ZPSDS).

An INSIDE ZPSDS bit set to one indicates that the phy is attached to a zoning expander device with zoning enabled and is thus inside a ZPSDS. The INSIDE ZPSDS bit only changes following a link reset sequence (see 4.8.4), based on:

- a) the REQUESTED INSIDE ZPSDS bit;
- b) the REQUESTED INSIDE ZPSDS bit received in the incoming IDENTIFY address frame (see 6.10.2);
- c) the INSIDE ZPSDS PERSISTENT bit; and
- d) the INSIDE ZPSDS PERSISTENT bit received in the incoming IDENTIFY address frame.

The REQUESTED INSIDE ZPSDS bit is used to establish the boundary of the ZPSDS. The REQUESTED INSIDE ZPSDS bit is used to indicate the values of other zone phy information fields after a link reset sequence (see 4.8.4).

The INSIDE ZPSDS PERSISTENT bit is used to indicate the value of the INSIDE ZPSDS bit after a link reset sequence (see 4.8.4).

The ZONE GROUP field contains the zone group to which the phy belongs (see 4.8.3.2). The zone group of the SMP initiator port and SMP target port in a zoning expander device shall be 01h.

The ZONE GROUP PERSISTENT bit is used to indicate the method of determining the zone group of the phy after a link reset sequence if the INSIDE ZPSDS bit is set to zero (see 4.8.4).

#### 4.8.3.2 Zone groups

The zone groups are defined in table 38.

**Table 38 – Zone groups**

Zone group	Configurable in the zone permission table <sup>a</sup>	Description
0	No	Phys in zone group 0 have access to phys in zone group 1 and do not have access to phys in other zone groups.
1	No	Phys in zone group 1 have access to phys in all zone groups.
2	Yes	<p>Phys in zone group 2 have access to phys in the zone groups indicated by the zone permission table.</p> <p>A management device server in a zoning expander device with zoning enabled only allows management application clients using phys in zone groups with access to zone group 2 to perform the following SMP functions:</p> <ul style="list-style-type: none"> <li>a) CONFIGURE GENERAL (see 9.4.3.18);</li> <li>b) ZONE LOCK (see 9.4.3.21); and</li> <li>c) SMP zone configuration functions (see 4.8.6.1) performed while the zoning expander device is locked.</li> </ul> <p>A management device server in a zoning expander device with zoning enabled only allows management application clients to perform certain SMP phy-based control and configuration functions (e.g., PHY CONTROL, PHY TEST FUNCTION, and CONFIGURE PHY EVENT) if the zone group of the management application client's phy has access to zone group 2 or the zone group of the specified phy.</p>
3	Yes	<p>Phys in zone group 3 have access to phys in the zone groups indicated by the zone permission table.</p> <p>A management device server in a zoning expander device with zoning enabled only allows management application clients using a phy in a zone group with access to zone group 3 to perform certain SMP zoning-related functions (i.e., ZONED BROADCAST (see 9.4.3.20)).</p>
4 to 7	Reserved	
8 to 255	Yes	Phys in zone groups 8 to 255 have access to phys in the zone groups indicated by the zone permission table.
<sup>a</sup> A zone group defined as configurable is able to be changed with the SMP CONFIGURE ZONE PERMISSION TABLE function (see 9.4.3.26).		

#### 4.8.3.3 Zone permission table

The zone permission table specifies access permission between zone groups. If a bit in the zone permission table is set to one, then connection requests shall be permitted between phys in the zone groups. If a bit in the zone permission table is set to zero, then connection requests between phys in the zone groups shall be rejected with OPEN\_REJECT (ZONE VIOLATION) or OPEN\_REJECT (RETRY) as described in 4.8.3.5.

The zone permission table structure is shown in table 39.

**Table 39 – Zone permission table**

Destination zone group (i.e., d)	Source zone group (i.e., s) <sup>a b</sup>				
	0	1	2 to 3	4 to 7	8 to (z-1) <sup>c</sup>
0	0	1	0	0	0
1	1	1	1	1	1
2 to 3	0	1	ZP[s = 2 to 3, d = 2 to 3]	Reserved	ZP[s = 8 to (z-1), d = 2 to 3]
4 to 7	0	1	Reserved	Reserved	Reserved
8 to (z-1) <sup>c</sup>	0	1	ZP[s = 2 to 3, d = 8 to (z-1)]	Reserved	ZP[s = 8 to (z-1), d = 8 to (z-1)]
<sup>a</sup> Shading identifies configurable zone groups. <sup>b</sup> All reserved ZP bits shall be set to zero (e.g., bits ZP[4 to 7, 4 to (z-1)] are set to zero). <sup>c</sup> The number of zone groups (i.e., z) is reported in NUMBER OF ZONE GROUPS field in the REPORT GENERAL response (see 9.4.3.4).					

A ZP[s, d] bit set to one specifies that the source zone group (i.e., s) has permission to access the destination zone group (i.e., d). A ZP[s, d] bit set to zero specifies that the source zone group (i.e., s) does not have permission to access the destination zone group (i.e., d).

ZP[s, d] shall be set to the same value as ZP[d, s].

The zoning expander device:

- a) shall preserve the zone permission table while zoning is disabled; and
- b) may or may not preserve the zone permission table through power loss and expander device reduced functionality.

If the zoning expander device preserves whether or not zoning is enabled and does not preserve the zone permission table, then the zoning expander device shall set the current zone permission table to grant minimal permissions after power on or expander device reduced functionality as specified in table 40.

**Table 40 – Zone permission table granting minimal permissions**

Destination zone group (i.e., d)	Source zone group (i.e., s) <sup>a b</sup>				
	0	1	2 to 3	4 to 7	8 to (z-1) <sup>c</sup>
0	0	1	0	0	0
1	1	1	1	1	1
2 to 3	0	1	0	Reserved	0
4 to 7	0	1	Reserved	Reserved	Reserved
8 to (z-1) <sup>c</sup>	0	1	0	Reserved	0

<sup>a</sup> Shading identifies configurable zone groups.  
<sup>b</sup> All reserved ZP bits shall be set to zero (e.g., bits ZP[4 to 7, 4 to (z-1)] are set to zero).  
<sup>c</sup> The number of zone groups (i.e., z) is reported in NUMBER OF ZONE GROUPS field in the REPORT GENERAL response (see 9.4.3.4).

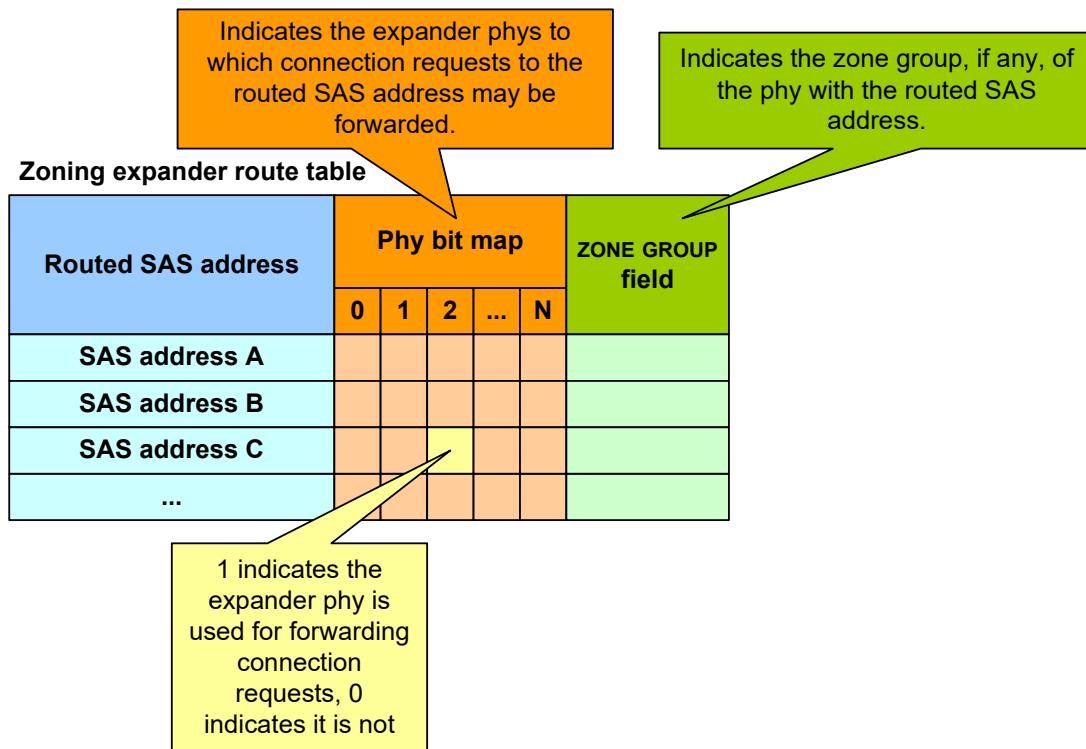
If a zone manager enables zoning on zoning capable expander devices that report different values in the NUMBER OF ZONE GROUPS field in the REPORT GENERAL response (see 9.4.3.4) (e.g., some support 128 and others support 256), then the zone manager shall:

- configure all zoning enabled expander devices contained within the ZPSDS to use the highest number of zone groups supported by all of the zoning enabled expander devices in the ZPSDS (e.g., 128);
- configure the zone phy information in all the zoning expander devices to set each phy to a zone group less than the highest number of zone groups supported by all of the zoning enabled expander devices in the ZPSD; and
- configure the zone permission table in all the zoning expander devices to set each entry to zero that is higher than the highest number of zone groups supported by all of the zoning enabled expander devices in the ZPSDS.

#### 4.8.3.4 Zoning expander route table

A zoning expander route table is an expander-based expander route table (see 4.5.7.4) that is able to hold the zone group of each routed SAS address.

Figure 56 shows a representation of the zoning expander route table.



**Figure 56 – Zoning expander route table**

The zoning expander route table:

- shall include discovered SAS addresses discovered behind each expander phy with the ROUTING ATTRIBUTE field set to 2h (i.e., table) in the DISCOVER response; and
- may include discovered SAS addresses discovered behind expander phys with the ROUTING ATTRIBUTE field set to 1h (i.e., subtractive) in the DISCOVER response as long as they do not prevent inclusion of SAS addresses for expander phys with the ROUTING ATTRIBUTE field set to 2h (i.e., table). The determination of which such SAS addresses to include is vendor specific.

The total number of routed SAS addresses shall not exceed the value indicated in the MAXIMUM NUMBER OF ROUTED SAS ADDRESSES field in the REPORT GENERAL response.

#### 4.8.3.5 Source zone group and destination zone group determination

When a zoning expander device with zoning enabled receives an OPEN address frame (see 6.10.3):

- the zone group of the source port (i.e., s) is identified as defined in table 41; and
- the zone group of the destination port (i.e., d) is identified as defined in table 42.

If the ZP[s, d] bit is set to one, then:

- access between the source and destination phys shall be permitted; and
- the zoning expander device shall perform the ECM arbitration procedure.

If the ZP[s, d] bit is set to zero, then access between the source and destination phys is not permitted and the zoning expander device shall transmit an OPEN\_REJECT in response to the connection request as follows:

- OPEN\_REJECT (RETRY) if the zoning expander device is locked; or
- OPEN\_REJECT (ZONE VIOLATION) if the zoning expander device is unlocked.



Zoning expander devices with zoning enabled shall follow the rules in table 41 to determine the zone group of the source port.

**Table 41 – Source zone group determination**

INSIDE ZPSDS bit of the expander phy that received the OPEN address frame	Source zone group
0	Zone group of the receiving expander phy
1	Source zone group specified by the SOURCE ZONE GROUP field in the received OPEN address frame

Zoning expander devices with zoning enabled shall follow the rules in table 42 to determine the zone group of the destination port.

**Table 42 – Destination zone group determination**

Routing method of the destination expander phy	Destination zone group
Direct	Zone group of the destination expander phy
Subtractive	<p>If the destination SAS address is in the zoning expander route table, then the zone group stored in the zoning expander route table for the destination SAS address.</p> <p>If the destination SAS address is not in the zoning expander route table, then the zone group of the destination expander phy (i.e., the subtractive routing phy).</p>
Table	The zone group stored in the zoning expander route table for the destination SAS address

#### 4.8.4 Zone phy information and link reset sequences

At the completion of a link reset sequence (see 4.4), if a SATA device is attached to an expander phy, then the zoning expander device with zoning enabled shall set the INSIDE ZPSDS bit to zero for that expander phy.

At the completion of a link reset sequence, if a SATA device is not attached to an expander phy, then the zoning expander device with zoning enabled shall update the current REQUESTED INSIDE ZPSDS bit and INSIDE ZPSDS PERSISTENT bit as defined in table 43 based on:

- a) the REQUESTED INSIDE ZPSDS bit and the INSIDE ZPSDS PERSISTENT bit in the zone phy information (i.e., the bits transmitted in the outgoing IDENTIFY address frame (see 6.10.2)); and

- b) the REQUESTED INSIDE ZPSDS bit and INSIDE ZPSDS PERSISTENT bit received in the incoming IDENTIFY address frame.

**Table 43 – REQUESTED INSIDE ZPSDS bit and INSIDE ZPSDS PERSISTENT bit changes after a link reset sequence**

REQUESTED INSIDE ZPSDS bit		INSIDE ZPSDS PERSISTENT bit		Zone phy information field changes
Transmitted	Received	Transmitted	Received	
0	0 or 1	0 or 1	0 or 1	The zoning expander device shall set the INSIDE ZPSDS bit to zero.
1	0			
1	1	0	0	If the SAS ADDRESS field received in the IDENTIFY address frame during the identification sequence is different from the SAS ADDRESS field received prior to the completion of the link reset sequence, then the zoning expander device shall: a) set the REQUESTED INSIDE ZPSDS bit to zero; and b) set the INSIDE ZPSDS bit to zero.
		0	1	
		1	0	If the SAS ADDRESS field received in the IDENTIFY address frame during the identification sequence is the same as the SAS ADDRESS field received prior to the completion of the link reset sequence, then the zoning expander device shall set the INSIDE ZPSDS bit to one.
		1	1	The zoning expander device shall set the INSIDE ZPSDS bit to one.

If the ZONE GROUP PERSISTENT bit is set to one, then the zone group of an expander shall be set as shown in table 44. If the ZONE GROUP PERSISTENT bit is set to zero, then table 45 specifies events based on the initial condition of an expander phy that shall cause a zoning expander device with zoning enabled to change the ZONE GROUP field of the expander phy to its reset value (i.e., the saved value, if any, or the default value (e.g., 00h) if there is no saved value).

**Table 44 – ZONE GROUP field values if the ZONE GROUP PERSISTENT bit is set to one**

Current INSIDE ZPSDS Bit <sup>a</sup>	New INSIDE ZPSDS Bit <sup>b</sup>	ZONE GROUP field change
0	0	No change.
0	1	The zoning expander device shall set the ZONE GROUP field to 01h.
1	0	The zoning expander device shall set the ZONE GROUP field to its reset value (i.e., the saved value, if any, or the default value if there is no saved value).
1	1	The zoning expander device shall set the ZONE GROUP field to 01h.
<sup>a</sup> Current INSIDE ZPSDS bit is the value before the link reset sequence. <sup>b</sup> New INSIDE ZPSDS bit is the computed value based upon table 43.		

**Table 45 – Conditions that cause the ZONE GROUP field to be updated if the ZONE GROUP PERSISTENT bit is set to zero**

Initial condition	Event after the initial condition is established
Completed link reset sequence with a SAS device attached	<p>A subsequent link reset sequence completes and:</p> <ul style="list-style-type: none"> <li>a) any of the following fields received in the IDENTIFY address frame (see 6.10.2) during the identification sequence are different from their values prior to the completion of the link reset sequence: <ul style="list-style-type: none"> <li>A) SAS ADDRESS field; or</li> <li>B) REQUESTED INSIDE ZPSDS field;</li> </ul> </li> <li>or</li> <li>b) a SATA device is attached.</li> </ul>
Completed link reset sequence with a SATA device attached	<p>Either:</p> <ul style="list-style-type: none"> <li>a) a subsequent link reset sequence completes and: <ul style="list-style-type: none"> <li>A) a hot-plug timeout (see 5.11.5) occurred between the time of the initial condition and the time the link reset sequence completed;</li> <li>B) the zoning expander device has detected the possibility that a new SATA device has been inserted. The method of detection is outside the scope of this standard (e.g., an enclosure services process reports a change in the ELEMENT STATUS CODE field in the Device or Array Device element (see SES-3), or a change in the WORLD WIDE NAME field in the attached SATA device's IDENTIFY DEVICE data (see ACS-4)); or</li> <li>C) a SAS phy or expander phy is attached;</li> </ul> </li> <li>or</li> <li>b) the expander phy is disabled with the SMP PHY CONTROL function (see 9.4.3.28) DISABLE phy operation.</li> </ul>

#### 4.8.5 Broadcast processing in a zoning expander device with zoning enabled

The BPP determines the source zone groups of the Broadcast as follows:

- a) if the BPP receives a Broadcast Event Notify request from an expander logical phy (i.e., a zoning expander logical phy received a BROADCAST primitive sequence), then the Broadcast has a single source zone group set to the zone group of that expander phy; or
- b) if the BPP receives a message from the management device server indicating that the management device server received an SMP ZONED BROADCAST request (see table 362) from an SMP initiator port that has access to zone group 3, then the Broadcast has each of the source zone groups specified in the SMP ZONED BROADCAST request.

The BPP forwards the Broadcast to each expander port other than the one on which the Broadcast was received (i.e., the expander port that received the BROADCAST primitive sequence or SMP ZONED BROADCAST request) if:

- a) the Broadcast is not a Broadcast (Zone Activate) and any of the source zone groups have access to the zone group of the expander port;
- b) the Broadcast is a Broadcast (Zone Activate), the BPP is in a locked zoning expander device, the INSIDE ZPSDS bit is set to one, and the source zone group has access to zone group 2; or
- c) the Broadcast is a Broadcast (Zone Activate), the BPP is not in a locked zoning expander device, and any of the source zone groups have access to the zone group of the expander port.

To forward a Broadcast to an expander port:

- a) if the expander port's INSIDE ZPSDS bit is set to one, then the BPP shall request that the SMP initiator port establish a connection on at least one phy in the expander port to the SMP target port of the attached expander device and transmit an SMP ZONED BROADCAST request specifying the source zone groups; or
- b) if the expander port's INSIDE ZPSDS bit is set to zero, then the BPP shall send a Transmit Broadcast message to at least one phy in the expander port, causing it to transmit a BROADCAST primitive sequence.

#### 4.8.6 Zone configuration

##### 4.8.6.1 Zone configuration overview

Zoning expander devices implement a lock to coordinate zoning configuration by zone managers.

There are four steps in the zone configuration process:

- 1) lock (see 4.8.6.2);
- 2) load (see 4.8.6.3);
- 3) activate (see 4.8.6.4); and
- 4) unlock (see 4.8.6.5).

The management device server in a zoning expander device only accepts SMP zone configuration function requests, SMP ZONE ACTIVATE requests, and SMP ZONE UNLOCK requests while it is locked, and only accepts SMP zone configuration function requests from the zone manager that locked the zoning expander device (i.e., the active zone manager). SMP zone configuration functions change zoning expander shadow values. When changes are complete, the zone manager activates the changes and the zoning expander device sets the zoning expander current values equal to the zoning expander shadow values. The zone manager then unlocks the zoning expander devices.

A ZPSDS only functions correctly while all zoning expander devices within the ZPSDS have identical values in their zone permission tables. To change zone permission tables, a zone manager device locks all zoning expander devices in a ZPSDS.

To change zone phy information, a zone manager locks only the zoning expander devices containing the phys to be changed.

When a zoning expander device with zoning disabled is being added to a ZPSDS (see figure 54 in 4.8.1) or two or more ZPSDSes are being merged (see figure 55 in 4.8.1), the zone manager locks all of the zoning expander devices that are to be included in the final ZPSDS. The zone manager configures the zone phy information in each zoning expander device (e.g., sets the REQUESTED INSIDE ZPSDS bit to one for phys inside the final ZPSDS) and configures all of the zone permission tables to be identical.

If the zone lock inactivity timer expires, then the zoning expander device performs the unlock step. The zoning expander device is unlocked and the zoning expander shadow values are not activated.

##### 4.8.6.2 Lock step

The lock step ensures that the same zone manager locks each zoning expander device. A zone manager sends the SMP ZONE LOCK request (see 9.4.3.21) to lock a zoning expander device. A zoning expander device is locked while the ZONE LOCKED bit is set to one in the SMP REPORT GENERAL response and after the SAS address of the zone manager has been stored. The management device server in a locked zoning expander device processes SMP zone configuration function requests, SMP ZONE ACTIVATE requests, and SMP ZONE UNLOCK requests.

If more than one zone manager attempts to lock a group of zoning expander devices, then the following rules ensure that any concurrent requests are resolved:

- a) if the first SMP ZONE LOCK response received by a zone manager has the FUNCTION RESULT field set to ZONE LOCK VIOLATION (see 9.4.3.3), then the group of zoning expander devices is locked by

another zone manager and the zone manager should originate no further requests until it receives a Broadcast (Change);

- b) if at least one SMP ZONE LOCK request is successful and at least one other response has:
  - A) the FUNCTION RESULT field set to ZONE LOCK VIOLATION (see 9.4.3.3); and
  - B) the ZONE CONFIGURING bit set to one (see 4.8.6.3),

then at least one zoning expander device is locked and being configured by another zone manager. The zone manager that failed to lock the zoning expander devices should unlock all zoning expander devices that it has locked. When a Broadcast (Change) is received, then the zone manager should retry the lock step; and

- c) if at least one SMP ZONE LOCK request is successful and at least one other response has:
  - A) the FUNCTION RESULT field set to ZONE LOCK VIOLATION (see 9.4.3.3); and
  - B) the ZONE CONFIGURING bit set to zero,

then another zone manager has locked at least one zoning expander device in the group of zoning expander devices and the zone manager shall evaluate the ACTIVE ZONE MANAGER SAS ADDRESS field in the SMP ZONE LOCK response as follows:

- a) if the returned SAS address has a lower numeric value than the SMP port SAS address of the zone manager, then the zone manager with the higher numeric value SAS address shall repeat the SMP ZONE LOCK request to all zoning expander devices that it has not already locked until all required zoning expander devices are locked or until a Broadcast (Change) is received; or
- b) if the returned ACTIVE ZONE MANAGER SAS ADDRESS field has a higher numeric value than the SMP port SAS address of the zone manager, then the zone manager with the lower numeric value SAS address shall originate an SMP ZONE UNLOCK request to unlock all zoning expander devices that it locked.

The lock step is complete after a zone manager receives a successful SMP ZONE LOCK response from all required zoning expander devices.

#### 4.8.6.3 Load step

The load step stores SMP zone configuration information as zoning expander shadow values. A zoning expander device only processes SMP zone configuration function requests originated by the active zone manager while the zoning expander device is locked.

The SMP zone configuration functions are:

- a) SMP CONFIGURE ZONE PHY INFORMATION (see 9.4.3.25);
- b) SMP CONFIGURE ZONE PERMISSION TABLE (see 9.4.3.26); and
- c) SMP ENABLE DISABLE ZONING (see 9.4.3.19).

After a locked zoning expander device processes any SMP zone configuration function request, it sets the ZONE CONFIGURING bit to one in the SMP REPORT GENERAL response (see 9.4.3.4).

If the ZONE CONFIGURING bit is set to one and a zoning violation occurs on a connection request, then the expander device shall return OPEN\_REJECT (RETRY) instead of OPEN\_REJECT (ZONE VIOLATION) (see 4.5.6.3).

SMP zone configuration functions change the zoning expander shadow values and do not affect the zoning expander current values. The zoning expander shadow values become zoning expander current values during the activate step (see 4.8.6.4).

If the active zone manager receives a response to an SMP zone configuration function with the FUNCTION RESULT field set to ZONE LOCK VIOLATION (see 9.4.3.3), then it should unlock all locked zoning expander devices.

The load step may be skipped when a locked zoning expander device is unlocked:

- a) by a zone manager with a higher SAS address during the lock step (see 4.8.6.2); or
- b) because the zone lock inactivity timer expires.

#### 4.8.6.4 Activate step

The activate step:

- a) saves the zoning expander shadow values, if saving was requested;
- b) copies the zoning expander shadow zone permission table to the current zone permission table;
- c) for each phy with the INSIDE ZPSDS bit set to zero, copies the shadow zone phy information to the current zone phy information; and
- d) for each phy with the INSIDE ZPSDS bit set to one:
  - A) sets the current zone group to 01h; and
  - B) for the remaining fields in the zone phy information, copies the shadow zone phy information to the current zone phy information.

The active zone manager originates one of the following:

- a) a Broadcast (Zone Activate) (see 4.1.15); or
- b) an SMP ZONE ACTIVATE request (see 9.4.3.22) to all locked zoning expander devices.

After a locked zoning expander device receives a Broadcast (Zone Activate) or processes an SMP ZONE ACTIVATE request the zoning expander device shall:

- a) copy the zoning expander shadow zone permission table to the current zone permission table;
- b) for each phy with the INSIDE ZPSDS bit set to zero, copy the shadow zone phy information to the current zone phy information; and
- c) for each phy with the INSIDE ZPSDS bit set to one:
  - A) set the current zone group to 01h; and
  - B) for the remaining fields in the zone phy information, copy the shadow zone phy information to the current zone phy information.

If the active zone manager receives an SMP ZONE ACTIVATE response with the FUNCTION RESULT field set to ZONE LOCK VIOLATION (see 9.4.3.3), then it should unlock all locked zoning expander devices.

The activate step may be skipped when a locked zoning expander device is unlocked:

- a) by a zone manager with a higher SAS address during the lock step (see 4.8.6.2); or
- b) because the zone lock inactivity timer expires.

#### 4.8.6.5 Unlock step

The unlock step ensures that:

- a) the active zone manager unlocks the locked zoning expander devices; or
- b) if the zone manager fails, then the zone lock inactivity timer expires and the zoning expander devices unlock.

If the active zone manager originated Broadcast (Zone Activate), then it sends an SMP ZONE UNLOCK request (see 9.4.3.23) with the ACTIVATE REQUIRED bit set to one to each of the locked zoning expander devices. This ensures that the activate step precedes the unlock step in each zoning expander device. If the active zone manager receives an SMP ZONE UNLOCK response with the FUNCTION RESULT field set to NOT ACTIVATED (see 9.4.3.3), then it retries the SMP ZONE UNLOCK request a vendor specific number of times, then originates an SMP ZONE ACTIVATE request to each locked zoning expander device.

If the active zone manager originated SMP ZONE ACTIVATE requests, then after all the SMP ZONE ACTIVATE functions have completed without error, it sends an SMP ZONE UNLOCK request with the ACTIVATE REQUIRED bit set to zero to each of the locked zoning expander devices. If the active zone manager receives an SMP ZONE UNLOCK response with the FUNCTION RESULT field set to BUSY (see 9.4.3.3), then it retries the SMP ZONE UNLOCK request.

When the SMP ZONE UNLOCK request is successful or the zone lock inactivity timer expires, then the zoning expander device is unlocked and shall:

- a) set the ZONE LOCKED bit to zero in the SMP REPORT GENERAL response (see 9.4.3.4);
- b) set the ZONE CONFIGURING bit to zero in the SMP REPORT GENERAL response;

- c) if the zone lock timer expired, then originate a Broadcast (Change) from zone group 1; and
- d) if the management device server processed an SMP ZONE UNLOCK request, then originate a Broadcast (Change) (see 6.15) from either:
  - A) each zone group whose zone permission table entries or zone phy information has changed; or
  - B) zone group 1.

When all SMP ZONE UNLOCK requests are successful, the zone configuration process is complete.

#### 4.8.6.6 Zone lock inactivity timer

The zone lock inactivity timer ensures that if the zone manager disappears without performing the unlock step that all locked zoning expander devices are unlocked.

When a zoning expander device processes an SMP ZONE LOCK request (see 9.4.3.21), then the zone lock inactivity timer default value is set to the value of the ZONE LOCK INACTIVITY TIME LIMIT field.

The zone lock inactivity timer is initialized and started if the default value is non-zero and:

- a) the zoning expander device completes processing of any SMP zone configuration function request or SMP ZONE ACTIVATE request while the ZONE LOCKED bit is set to one in the SMP REPORT GENERAL response (see 9.4.3.4); or
- b) the zoning expander device completes processing of a successful SMP ZONE LOCK request.

The zone lock inactivity timer is stopped if:

- a) the ZONE LOCK INACTIVITY TIME LIMIT field is set to zero in an SMP ZONE LOCK request; or
- b) the ZONE LOCKED bit is set to zero in the SMP REPORT GENERAL response (e.g., an SMP ZONE UNLOCK request (see 9.4.3.23) is processed or the zone lock inactivity timer expires).

If the zone lock inactivity timer expires, then the zoning expander device:

- a) sets the ZONE LOCKED bit to zero in the SMP REPORT GENERAL response;
- b) sets the ZONE CONFIGURING bit to zero in the SMP REPORT GENERAL response; and
- c) sends Broadcast (Change) on all ports.

If the zone lock inactivity timer expires while the zoning expander device is processing an SMP configuration function, then the zoning expander device may complete the request without error or return a function result of ZONE LOCK VIOLATION.

#### 4.8.6.7 Enable a zoning expander device

If a zoning expander device has the ZONING SUPPORTED bit set to one and the ZONING ENABLED bit set to zero in the REPORT GENERAL response (see 9.4.3.4), then a zone manager configures the zoning expander device using the zone configuration process. This ensures that the zone permission table is the same in all zoning expander devices inside the ZPSDS.

Changes made by the SMP ENABLE DISABLE ZONING function sent by the active zone manager become active during the activate step (see 4.8.6.4).

### 4.9 SAS device and expander device power conditions

SCSI idle and standby power conditions, implemented with the START STOP UNIT command (see SBC-3) and the Power Condition mode page (see SPC-4), may be supported by SSP initiator ports and SSP target ports as described in 9.2.10.

Except for Sleep mode (e.g., requested with the ATA SLEEP command), the ATA Power Management feature set, Extended Power Management feature set, or Advanced Power Management feature set (see ACS-4) may be supported by an ATA application client using an STP initiator port.



## 4.10 Phy power conditions

### 4.10.1 Low phy power conditions

#### 4.10.1.1 Low phy power conditions overview

Low phy power conditions are phy conditions where the phy is in a reduced power state (e.g., has disabled circuitry in order to reduce power). This standard defines low phy power conditions that are differentiated by time to return to the active phy power condition (see 4.10.1.2 and table 85) and the amount of power consumed in that low phy power condition. The low phy power conditions include the partial phy power condition (see 4.10.1.3) and the slumber phy power condition (see 4.10.1.4).

A phy in a low phy power condition shall not change to a different low phy power condition without first making a change to the active phy power condition (see 5.13.2).

Low phy power conditions shall only be enabled on a phy if multiplexing (see 5.20) and optical mode are disabled.

If the partial phy power condition is enabled and the received IDENTIFY address frame has the PARTIAL CAPABLE bit set to one (see 6.10.2), then the phy may generate PS\_REQ (PARTIAL) primitive sequences. If the slumber phy power condition is enabled and the received IDENTIFY address frame has the SLUMBER CAPABLE bit set to one (see 6.10.2), then the phy may generate PS\_REQ (SLUMBER) primitive sequences. If low phy power conditions are enabled, then the phy may reply with a PS\_ACK primitive sequence to accept a low phy power condition request. If low phy power conditions are supported and disabled, then the phy shall reject a low phy power condition request by replying with a PS\_NAK primitive sequence.

If a SAS phy or expander phy is in a low phy power condition and that phy is requested to transmit a NOTIFY, then that phy shall not transmit the NOTIFY and shall remain in the same low phy power condition.

#### 4.10.1.2 Active phy power condition

While in the active phy power condition:

- a) the phy is capable of transmitting information and responding to received information; and
- b) the phy may consume more power than while the phy is in a low phy power condition.

#### 4.10.1.3 Partial phy power condition

While in the partial phy power condition:

- a) the phy is only capable of processing a COMINIT or COMWAKE (see SATA);
- b) the phy may take less time to return to the active phy power condition (see table 85) than while in the slumber phy power condition; and
- c) the power consumed by the phy should be less than the power consumed while the phy is in the active phy power condition and may be greater than the power consumed while the phy is in the slumber phy power condition.

#### 4.10.1.4 Slumber phy power condition

While in the slumber phy power condition:

- a) the phy is only capable of processing a COMINIT or COMWAKE;
- b) the phy may take more time to return to the active phy power condition (see table 85) than while in the partial phy power condition; and
- c) the power consumed by the phy should be less than the power consumed while the phy is in the active phy power condition and while the phy is in the partial phy power condition.

#### 4.10.1.5 End device low phy power conditions

Support for low phy power conditions is reported in SAS target devices using the Phy Control And Discover mode page (see 9.2.7.5).

Partial phy power condition may be enabled and disabled in SAS target devices using the Enhanced Phy Control mode page (see 9.2.7.7).

Slumber phy power condition may be enabled and disabled in SAS target devices using the Enhanced Phy Control mode page (see 9.2.7.7).

The management application layer shall only:

- a) enable a low phy power condition (i.e., send a Manage Power Conditions (Accept Partial) request or Manage Power Conditions (Accept Slumber) request); and
- b) request a phy enter a low phy condition (i.e., send a Change Phy Power Condition request),

after receiving a Phy Power Condition Status (Enable Low Phy Power Conditions) message from the SA\_PC state machine (see 9.2.10.2).

The management application layer:

- a) shall disable a low phy power condition (i.e., send a Manage Power Conditions (Reject Partial) request and Manage Power Conditions (Reject Slumber) request); and
- b) shall not request that a phy enter a low phy power condition (i.e., send a Change Phy Power Condition request),

after receiving a Phy Power Condition Status (Disable Low Phy Power Conditions) message from the SA\_PC state machine (see 9.2.10.2).

If a SAS phy is in a low phy power condition, then to originate a Broadcast the management application layer:

- 1) shall initiate the exit power condition procedure (see 5.13.2) on that SAS phy;
- 2) shall originate the Broadcast; and
- 3) may initiate the procedure to return that SAS phy to a low phy power condition.

#### 4.10.1.6 Expander device low phy power conditions

Support for low phy power conditions is reported in expander devices using the SMP DISCOVER function (see 9.4.3.10).

Partial phy power conditions may be enabled and disabled in expander devices using the SMP PHY CONTROL function (see 9.4.3.28).

If an expander phy is in the partial phy power condition and the ECM receives a connection request routed to that expander phy, then the expander device initiates the exit power condition procedure (see 5.13.2) on that expander phy and responds with AIP (NORMAL) until the OPEN address frame is forwarded to that expander phy.

Slumber phy power conditions may be enabled and disabled in expander devices using the SMP PHY CONTROL function (see 9.4.3.28).

If an expander phy is in slumber phy power condition and the ECM receives a connection request routed to that expander phy, then the expander device initiates the exit power condition procedure (see 5.13.2) on that expander phy and responds with OPEN\_REJECT (RETRY) until a phy ready state (see 5.14.4.10) is established for that expander phy.

If an expander phy is in a low phy power condition, then to originate or forward a Broadcast the BPP:

- 1) shall initiate the exit power condition procedure (see 5.13.2) on that expander phy;
- 2) shall originate or forward the Broadcast; and
- 3) may initiate the procedure to return that expander phy to a low phy power condition.

#### 4.10.2 SATA phy power conditions

STP initiator ports shall not generate SATA\_PMREQ\_P, SATA\_PMREQ\_S, or SATA\_PMACK. If an STP initiator port receives SATA\_PMREQ\_P or SATA\_PMREQ\_S, then the STP initiator port shall reply with:

- a) SATA\_X\_RDY if the STP initiator port has a FIS ready to transmit to the STP target port; or
- b) a CLOSE primitive sequence.

NOTE 10 - SAS-2 required the STP initiator port to transmit SATA\_PMNAK.

SATA interface power management sequences (see SATA) may be enabled in an expander phy using the SMP PHY CONTROL function (see 9.4.3.28).

If an expander device receives SATA\_PMREQ\_P or SATA\_PMREQ\_S from a SATA device while an STP connection is not open, then it shall not forward the primitive to any STP initiator port and shall reply with SATA\_PMNAK or SATA\_PMACK as defined by SATA. If SATA interface power management sequences are not enabled, then the expander device shall reply with SATA\_PMNAK.

If an expander device receives SATA\_PMREQ\_P or SATA\_PMREQ\_S while an STP connection is open, then the expander device may or may not forward the primitive to the STP initiator port. If the expander device forwards a SATA\_PMREQ\_P or SATA\_PMREQ\_S to the STP initiator port during an STP connection, then the expander device shall not reply with SATA\_PMACK or SATA\_PMNAK within that connection.

### 4.11 Phy test functions

#### 4.11.1 Phy test functions overview

Phy test functions (e.g., transmission of test patterns) are used for phy and SAS interconnect characterization and diagnosis. The phy may be attached to test equipment while performing a phy test function. The following optional mechanisms are defined for invoking phy test functions:

- a) the Protocol Specific diagnostic page for SAS (see 9.2.9.2) invokes a phy test function in a selected phy in a SAS target device with an SSP target port. The SEND DIAGNOSTIC command (see SPC-4) may be sent through any SSP target port to any logical unit in the SAS target device that contains the phy that is to perform the phy test function. The phy test function starts some time after the SSP target port receives an ACK for the RESPONSE frame transmitted in response to the SEND DIAGNOSTIC command; and
- b) the SMP PHY TEST FUNCTION function (see 9.4.3.29) invokes a phy test function in a phy controlled by a management device server other than the phy that receives the function. The phy test function starts some time after the SMP target port transmits the SMP response frame.

Each phy test function is optional.

If the phy test function requires a specific phy test pattern and/or phy test function physical link rate, then the mechanism for invoking the phy test function specifies the phy test pattern and phy test function physical link rate.

The phy test function on one phy may affect the negotiated settings on other phys (e.g., in a device with a common SSC clock, the SSC modulation type may change from none to down-spreading even on phys that negotiated no SSC).

While a phy is performing a phy test function, the link layer receivers (i.e., the SL\_IR receiver, SL receiver, SL\_P\_S receiver, SL\_P\_C receiver, SSP receiver, STP receiver, and SMP receiver) shall ignore all incoming dwords and the OOB signal detector shall detect COMINIT (see SATA). The phy shall ignore any other OOB signals (i.e., COMSAS and COMWAKE).

A phy stops performing a phy test function:

- a) after the SCSI device server, if any, processes a Protocol Specific diagnostic page specifying the phy and specifying a phy test function of 00h (i.e., STOP);

- b) after the management device server, if any, processes an SMP PHY TEST FUNCTION request specifying the phy and specifying a phy test function of 00h (i.e., STOP);
- c) after the phy receives COMINIT; or
- d) upon power off.

The time it takes for a phy to stop performing the phy test function is vendor specific. After a phy stops performing a phy test function, the phy performs a link reset sequence.

#### 4.11.2 Transmit pattern phy test function

While a phy is performing the transmit pattern phy test function, the test equipment attached to that phy:

- a) shall not transmit COMSAS or COMWAKE (see SATA); and
- b) shall not transmit COMINIT (see SATA) except to stop the phy test function.

While performing the transmit pattern phy test function, a phy:

- a) shall ignore all dwords received; and
- b) shall repeatedly transmit the specified pattern at the specified physical link rate.

### 4.12 Phy events

Phys shall count the following events using saturating counters and report them in the Protocol Specific Port log page (see 9.2.8.1) and/or the SMP REPORT PHY ERROR LOG function (see 9.4.3.11):

- a) invalid dwords received;
- b) dwords received with running disparity errors;
- c) loss of dword synchronization; and
- d) phy reset problems.

The saturating counters are each up to 4 bytes wide.

Phys may count those events and certain other events (e.g., elasticity buffer overflows) using wrapping counters and record peak values for certain events (e.g., the longest connection time) using peak value detectors, reporting them in the Protocol Specific Port log page (see 9.2.8.1), SMP REPORT PHY EVENT function (see 9.4.3.14), and/or the SMP REPORT PHY EVENT LIST function (see 9.4.3.16). The wrapping counters and peak value detectors are each 4 bytes wide. Peak value detectors trigger Broadcast (Expander) under certain circumstances (see 6.2.6.4).

The number of additional events monitored is vendor specific. For phys not controlled by SMP target ports, the events that are monitored are vendor specific. For phys controlled by SMP target ports, the SMP CONFIGURE PHY EVENT function (see 9.4.3.30) allows for specification of the events to monitor.

The management device server shall maintain phy events for the last vendor specific number of events and should maintain at least one phy event per phy. The management device server shall assign descriptors to the events sequentially starting at 0001h and shall return the descriptors in the SMP REPORT PHY EVENT LIST response (see 9.4.3.16). The management device server shall return the index of the descriptor for the last phy event in the SMP REPORT GENERAL response (see 9.4.3.4), the SMP REPORT PHY EVENT LIST response (see 9.4.3.16), and the SMP DISCOVER LIST response (see 9.4.3.15). The management device server shall wrap the index to 0001h when the highest supported descriptor index has been used.

The management device server shall support phy event list descriptor (see 9.4.3.16.4) indexes from 0001h to FFFFh. The actual number of phy event list descriptors that the management device server maintains for retrieval with the REPORT PHY EVENT LIST request is vendor specific and is indicated by the MAXIMUM NUMBER OF STORED PHY EVENT LIST DESCRIPTORS field defined in the REPORT GENERAL response (see 9.4.3.4). The volatility of these stored descriptors is vendor specific. The management device server shall replace the oldest phy event list descriptor with a new one once the number of recorded descriptors exceeds the value indicated by the MAXIMUM NUMBER OF STORED PHY EVENT LIST DESCRIPTORS field.

The PHY EVENT SOURCE field, defined in table 46, is used in the Protocol Specific Port log page (see 9.2.8.1), the REPORT PHY EVENT function (see 9.4.3.14), the REPORT PHY EVENT LIST function (see 9.4.3.16), and the CONFIGURE PHY EVENT function (see 9.4.3.30) and indicates or specifies the type of phy event in the accompanying PHY EVENT field.

**Table 46 – PHY EVENT SOURCE field (part 1 of 5)**

Code	Name	Type <sup>a</sup>	Description
00h	No event	N/A	No event. The PHY EVENT field is not valid.
Phy layer-based phy events (01h to 1Fh)			
01h	Invalid dword count <sup>b</sup>	WC	Number of invalid dwords that have been received outside of phy reset sequences (i.e., between when the SP state machine (see 5.14) sends a Phy Layer Ready (SAS) confirmation or Phy Layer Ready (SATA) confirmation and when it sends a Phy Layer Not Ready confirmation to the link layer).
02h	Running disparity error count <sup>b</sup>	WC	Number of dwords containing running disparity errors (see 5.3.5) that have been received outside of phy reset sequences.
03h	Loss of dword synchronization count <sup>b</sup>	WC	Number of times the phy has restarted the link reset sequence because it lost dword synchronization while in SAS dword mode (i.e., the SP state machine transitioned from SP15:SAS_PHY_Ready or SP22:SATA_PHY_Ready to SP0:OOB_COMINIT (see 5.14)).
04h	Phy reset problem count <sup>b</sup>	WC	Number of times a phy reset problem has occurred (see 5.11.4.2.4).
05h	Elasticity buffer overflow count	WC	Number of times the phy's elasticity buffer (see 6.5) has overflowed outside of phy reset sequences (e.g., because it did not receive a sufficient number of deletable primitives).
06h	Received ERROR count	WC	Number of times the phy received an ERROR.
07h	Invalid SPL packet count	WC	Number of invalid SPL packets that have been received outside of phy reset sequences while in SAS packet mode (i.e., between when the SP state machine (see 5.14) sends a Phy Layer Ready (SAS) confirmation or Phy Layer Ready (SATA) confirmation and when it sends a Phy Layer Not Ready confirmation to the link layer).
<sup>a</sup> The Type column indicates the source type: a) WC = wrapping counter; b) PVD = peak value detector; and c) N/A = not applicable. <sup>b</sup> This standard also defines a saturating counter that counts this event (see 9.2.8.1 and 9.4.3.11).			

**Table 46 – PHY EVENT SOURCE field** (part 2 of 5)

Code	Name	Type <sup>a</sup>	Description
08h	Loss of SPL packet synchronization count	WC	Number of times the phy has restarted the link reset sequence because it lost SPL packet synchronization while in SAS packet mode (i.e., the SP state machine transitioned from SP15:SAS_PHY_Ready to SP0:OOB_COMINIT (see 5.14.4.10)).
09h to 1Fh	Reserved		
SAS arbitration-related phy events (20h to 3Fh)			
20h	Received address frame error count	WC	Number of times the phy detected an invalid address frame (see 6.10) (e.g., because of a CRC error).
21h	Transmitted abandon-class OPEN_REJECT count	WC	Number of times the phy received an OPEN address frame and transmitted an abandon-class OPEN_REJECT (see 6.2.6.10). In expander devices, forwarded OPEN_REJECTs shall not be counted.
22h	Received abandon-class OPEN_REJECT count	WC	Number of times the phy originated an OPEN address frame and received an abandon-class OPEN_REJECT (see 6.2.6.10). In expander devices, OPEN_REJECTs in response to forwarded OPEN address frames shall not be counted.
23h	Transmitted retry-class OPEN_REJECT count	WC	Number of times the phy received an OPEN address frame and transmitted a retry-class OPEN_REJECT (see 6.2.6.10). In expander devices, forwarded OPEN_REJECTs shall not be counted.
24h	Received retry-class OPEN_REJECT count	WC	Number of times the phy originated an OPEN address frame and received a retry-class OPEN_REJECT (see 6.2.6.10). In expander devices, OPEN_REJECTs in response to forwarded OPEN address frames shall not be counted.
25h	Received AIP (WAITING ON PARTIAL) count	WC	Number of times the phy received an AIP (WAITING ON PARTIAL) or AIP (RESERVED WAITING ON PARTIAL). In expander devices, forwarded AIPs shall be counted.
26h	Received AIP (WAITING ON CONNECTION) count	WC	Number of times the phy received an AIP (WAITING ON CONNECTION). In expander devices, forwarded AIPs shall be counted.
27h	Transmitted BREAK count	WC	Number of times the phy transmitted a BREAK primitive sequence that was not a response to a BREAK primitive sequence it received (e.g., a Close Timeout was detected by the SL state machines interfacing to the SMP target port).
<sup>a</sup> The Type column indicates the source type: a) WC = wrapping counter; b) PVD = peak value detector; and c) N/A = not applicable. <sup>b</sup> This standard also defines a saturating counter that counts this event (see 9.2.8.1 and 9.4.3.11).			

**Table 46 – PHY EVENT SOURCE field (part 3 of 5)**

Code	Name	Type <sup>a</sup>	Description
28h	Received BREAK count	WC	Number of times the phy received a BREAK primitive sequence that was not a response to a BREAK primitive sequence that it transmitted.
29h	Break Timeout count	WC	Number of times the phy transmitted a BREAK primitive sequence and did not receive a BREAK primitive sequence or BREAK_REPLY primitive sequence in response (e.g., as detected by the XL state machine and/or the SL state machines interfacing to the SMP target port).
2Ah	Connection count	WC	Number of connections in which the phy was involved.
2Bh	Peak transmitted pathway blocked count	PVD	Peak value of a PATHWAY BLOCKED COUNT field in an OPEN address frame transmitted by the phy. Since the maximum value of the PATHWAY BLOCKED COUNT field is FFh, only byte 3 of the PHY EVENT field is used.
2Ch	Peak transmitted arbitration wait time	PVD	Peak value of an ARBITRATION WAIT TIME field in an OPEN address frame transmitted by the phy. Since the maximum value of the ARBITRATION WAIT TIME field is FFFFh, only byte 2 and byte 3 of the PHY EVENT field are used.
2Dh	Peak arbitration time	PVD	Peak time in microseconds after transmitting an OPEN address frame that the phy has waited for connection response (e.g., OPEN_ACCEPT or OPEN_REJECT).
2Eh	Peak connection time	PVD	The peak duration, in microseconds, of any connection in which the phy was involved.
2Fh	Persistent connection count	WC	Number of persistent connections (see 4.1.13) in which the phy was involved.
30h to 3Fh	Reserved		
SSP-related phy events (40h to 4Fh)			
40h	Transmitted SSP frame count	WC	Number of SSP frames transmitted.
41h	Received SSP frame count	WC	Number of SSP frames received.
42h	Transmitted SSP frame error count	WC	Number of times the phy was used in a connection involving the SSP target port, transmitted an SSP frame, and received a NAK or an ACK/NAK timeout.
<sup>a</sup> The Type column indicates the source type: a) WC = wrapping counter; b) PVD = peak value detector; and c) N/A = not applicable. <sup>b</sup> This standard also defines a saturating counter that counts this event (see 9.2.8.1 and 9.4.3.11).			

**Table 46 – PHY EVENT SOURCE field** (part 4 of 5)

Code	Name	Type <sup>a</sup>	Description
43h	Received SSP frame error count	WC	Number of times the phy was used in a connection involving the SSP target port, detected an invalid SSP frame, and transmitted a NAK (CRC ERROR) (e.g., because of a CRC error).
44h	Transmitted CREDIT_BLOCKED count	WC	Number of times the phy transmitted a CREDIT_BLOCKED.
45h	Received CREDIT_BLOCKED count	WC	Number of times the phy received a CREDIT_BLOCKED.
46h to 4Fh	Reserved		
STP and SATA-related phy events (50h to 5Fh)			
50h	Transmitted SATA frame count	WC	Number of STP or SATA frames transmitted.
51h	Received SATA frame count	WC	Number of STP or SATA frames received.
52h	SATA flow control buffer overflow count	WC	<p>Number of times the phy's STP flow control buffer (see 6.21.4) has overflowed (e.g., because it received more data dwords than allowed after transmitting SATA_HOLD during an STP connection).</p> <p>In an expander device, this count should be maintained in the expander phy transmitting the SATA_HOLD and receiving the data dwords, but may be maintained in the expander phy receiving the SATA_HOLD and transmitting the data dwords.</p>
53h to 5Fh	Reserved		
SMP-related phy events (60h to 6Fh)			
60h	Transmitted SMP frame count	WC	Number of SMP frames transmitted.
61h	Received SMP frame count	WC	Number of SMP frames received.
62h	Reserved		
<sup>a</sup> The Type column indicates the source type: a) WC = wrapping counter; b) PVD = peak value detector; and c) N/A = not applicable. <sup>b</sup> This standard also defines a saturating counter that counts this event (see 9.2.8.1 and 9.4.3.11).			



**Table 46 – PHY EVENT SOURCE field (part 5 of 5)**

Code	Name	Type <sup>a</sup>	Description
63h	Received SMP frame error count	WC	Number of times the phy was used to access the SMP target port and the SMP target port detected an invalid SMP frame and transmitted a BREAK primitive sequence (e.g., because of a CRC error).
64h to 6Fh	Reserved		
Other (70h to FFh)			
70h to CFh	Reserved		
D0h to FFh	Vendor specific		
<sup>a</sup> The Type column indicates the source type: a) WC = wrapping counter; b) PVD = peak value detector; and c) N/A = not applicable. <sup>b</sup> This standard also defines a saturating counter that counts this event (see 9.2.8.1 and 9.4.3.11).			

## 4.13 Using POWER DISABLE signal to create a power on event

### 4.13.1 Using POWER DISABLE signal to create a power on event overview

The POWER DISABLE signal (see SAS-4) allows a SAS initiator device or an expander device to cause a power on event (see SAM-5) for a SAS target device by:

- 1) asserting the POWER DISABLE signal for the minimum hold time (see SAS-4); and
- 2) negating the POWER DISABLE signal.

### 4.13.2 Discovering POWER DISABLE signal support

A SCSI application client determines if a SAS target device supports the POWER DISABLE signal by verifying that the PWR\_D\_S bit is set to one in the Protocol Specific Port Information VPD page (see 9.2.11.4).

If a SCSI application client has determined that a SAS target device supports the POWER DISABLE signal, then the management application client determines if the management device server in the SAS initiator device or expander device attached to that SAS target device is capable of controlling the POWER DISABLE signal by examining the DISCOVER function response data (see 9.4.3.10) for a phy associated with the SAS target device and verifying that:

- a) the ATTACHED PWR\_DIS CAPABLE bit is set to one; and
- b) the PWR\_DIS CONTROL CAPABLE field is set to 01b.

If the DISCOVER function response data indicates that the management device server in the SAS initiator device or expander device is not capable of controlling the POWER DISABLE signal (i.e., the PWR\_DIS CONTROL CAPABLE field is not set to 01b), then the POWER DISABLE signal may be controlled by a method outside the scope of this standard.

#### 4.13.3 Using a management device server to control the POWER DISABLE signal

If a management device server is capable of controlling the POWER DISABLE signal (see 4.13.2), then the management application client uses the PWR\_DIS CONTROL field in the PHY CONTROL function (see 9.4.3.28) to assert and negate the POWER DISABLE signal as required (see 4.13.1). The management application client may examine the PWR\_DIS SIGNAL field in the DISCOVER function response data (see 9.4.3.10) to verify that the POWER DISABLE signal is at the expected level.

#### 4.14 Management application client model for APTA

APTA shall be implemented on phys that support SAS packet mode and is initiated by the management application client when:

- a) the phy is in the phy ready state (see 5.14.4.10); and
- b) there is no connection established or in the process of being established.

APTA shall be disabled on phys:

- a) on which optical mode is enabled; or
- b) attached to an active cable assembly.

The management application client monitors the SP receiver using a method that is beyond the scope of this standard. SAS phys that support the SAS packet mode implement adaptations that optimize the SP receiver settings (see SAS-4). If the SP receiver determines that a vendor specific limit of adjustment is reached, then the management application client may request the local SP receiver adjust the attached SP transmitter's phy by sending an APTA request to the phy layer state machines.

To allow the SP receiver to adapt to each coefficient change of the SP transmitter settings, the receiver waits until a status response has been received for each request for an APTA coefficient change. The status response indicates that the attached phy's SP transmitter coefficients were updated and a time of greater than 1 ms has occurred (see ).

The receiver algorithm calculation may take several seconds to complete.

The management application client receives the following confirmations:

- a) an Enable APTA confirmation (see 5.14.4.10, 6.18.4.2, and 6.19.3);
- b) an APTA Disabled confirmation (see 5.14.3.2, 5.14.5.2, 6.18.4.2, and 6.19.3) with an argument indicating the reason for disabling APTA (e.g., OOB In Progress, Low Phy Power Condition, or Active Connection);
- c) an Adjustment Complete confirmation (see 5.19.5.4); and
- d) an Attached Phy Terminated APTA confirmation (see 5.19.4.4 and 5.19.5).

The management application client sends the following requests to the phy layer state machines:

- a) an Adjust Attached Transmitter request to start adjustment if a SP receiver indicates APTA is required (see 5.19.5.2); and
- b) a Terminate APTA request if the management application client detects any algorithmic or other error during APTA (see 5.19.4.4). The methods for detecting an error are beyond the scope of this standard.

## 5 Phy layer

### 5.1 Phy layer overview

The phy layer defines 8b10b coding, 128b150b coding, BMC coding, and OOB signals. Phy layer state machines interface between the link layer and the physical layer to perform the phy reset sequence and keep track of dword synchronization.

### 5.2 8b10b coding

#### 5.2.1 8b10b coding overview

All information transferred in the SAS dword mode is encoded into 10-bit characters using 8b10b encoding.

All primitives transferred in the SAS packet mode are encoded into a 2-bit control plus three 10-bit characters (see 5.5.4). The three 10-bit characters are encoded using 8b10b encoding. The 2-bit control and 10-bit characters are placed into an SPL packet payload within an 128b150b structure (see 5.5). The 2-bit control and 10-bit characters are transmitted serially bit-by-bit across the physical link.

Out of all 1 024 possible 10-bit characters:

- a) some of the characters are data characters, representing the 256 possible 8-bit data bytes;
- b) some of the characters are control characters, used for primitives (e.g., frame delimiters) and other control purposes; and
- c) the rest of the characters are invalid characters.

8b10b coding ensures that sufficient transitions are present in the serial bit stream to make clock recovery possible at the receiver. 8b10b coding also increases the likelihood of detecting any single or multiple bit errors that occur during transmission and reception. In addition, some of the control characters of the transmission code contain a distinct bit pattern, called a comma pattern, which assists a receiver in achieving character and dword alignment on the incoming bit stream.

#### 5.2.2 8b10b coding notation conventions

This subclause uses letter notation for describing information bits and control variables. Such notation differs from the bit notation specified by the remainder of this standard. The following text describes the translation process between these notations and provides a translation example. This subclause also describes the conventions used to name valid characters. This text is provided for the purposes of terminology clarification only.

An unencoded information byte is composed of:

- a) eight information bits labeled A, B, C, D, E, F, G, and H. Each information bit contains either a binary zero or a binary one; and
- b) a control variable labeled Z. A control variable has either the value D or the value K:
  - A) D means the information byte is a data byte; and
  - B) K means the information byte is a control byte.

The information bit labeled A corresponds to bit 0 in the numbering scheme of this standard, B corresponds to bit 1, and so on, as shown in table 47. Bit H is the most significant bit of the byte and bit A is the least significant bit of the byte.

**Table 47 – Bit designations**

Bit notation	7	6	5	4	3	2	1	0	Control variable
Unencoded bit notation	H	G	F	E	D	C	B	A	Z

Each valid character is named using the following convention:

Zxx.y

where:

- Z is the control variable of the unencoded information byte. The value of Z is used to indicate whether the character is a data character (i.e., Z = D) or a control character (i.e., Z = K);
- xx is the decimal value of the binary number composed of the bits E, D, C, B, and A of the unencoded information byte in that order; and
- y is the decimal value of the binary number composed of the bits H, G, and F of the unencoded information byte in that order.

Table 48 shows the conversion from byte notation to the character naming convention.

**Table 48 – Conversion from byte notation to character name example**

Byte notation	BCh								
Bit notation	7	6	5	4	3	2	1	0	Control
	1	0	1	1	1	1	0	0	K
Unencoded bit notation	H	G	F	E	D	C	B	A	Z
	1	0	1	1	1	1	0	0	K
Unencoded bit notation reordered to conform with Zxx.y naming convention	Z	E	D	C	B	A	H	G	F
	K	1	1	1	0	0	1	0	1
Character name	K 28 . 5								

Most Kxx.y combinations do not result in valid characters within the 8b10b coding scheme. Only those combinations that result in control characters defined in table 50 (see 5.3.7) are considered valid.

## 5.3 Character encoding and decoding

### 5.3.1 Introduction

This subclause describes how to select valid characters (i.e., 8b10b encoding) and how to check the validity of received characters (i.e., 10b8b decoding), and specifies the ordering rules to be followed when transmitting the bits within a character.

### 5.3.2 Bit transmission order

An information byte is encoded into a 10-bit character containing bits labeled a, b, c, d, e, i, f, g, h, and j. In SAS dword mode, bit a shall be transmitted first, followed by bits b, c, d, e, i, f, g, h, and j, in that order.

In SAS packet mode, the placement of the 10-bit character bits a, b, c, d, e, i, f, g, h, and j within a primitive segment is as described in table 57.

### 5.3.3 Character transmission order

In SAS dword mode, characters within primitives shall be transmitted sequentially beginning with the control character used to distinguish the primitive (e.g., K28.3 or K28.5) and proceeding character by character from left to right within the definition of the primitive until all characters of the primitive are transmitted.

In SAS packet mode, characters within primitives shall be placed within a primitive segment as described in table 57.

### 5.3.4 Frame transmission order

In SAS dword mode, the contents of a frame shall be transmitted sequentially beginning with the primitive used to denote the start of frame (e.g., SOAF, SOF, or SATA\_SOF) and proceeding character-by-character from left to right within the definition of the frame until the primitive used to denote the end of frame (e.g., EOAF, EOF, B\_EOF, or SATA\_EOF) is transmitted.

In SAS packet mode, the contents of a frame shall be transmitted as described in 5.5.

### 5.3.5 Running disparity (RD)

RD is a binary parameter with a negative (-) or positive (+) value. After power on, the transmitter may initialize the current RD to either positive or negative.

Each data character and control character is defined in a table by two columns that represent two characters that may or may not be different, corresponding to the current value of the running disparity (i.e., current RD- or current RD+).

Upon transmission of any character, the transmitter shall calculate a new value for its RD based on the contents of the transmitted character.

After power on, the receiver shall assume either the positive or negative value for its initial RD. Upon reception of any character, the receiver shall determine whether the character is valid or invalid and shall calculate a new value for its RD based on the contents of the received character.

The following rules for RD shall be used to calculate the new RD value for characters that have been transmitted (i.e., the transmitter's RD) and that have been received (i.e., the receiver's RD).

RD for a character shall be calculated on the basis of sub-blocks, where the first six bits (i.e., bits a, b, c, d, e, and i) form one sub-block (i.e., the six-bit sub-block) and the second four bits (i.e., bits f, g, h, and j) form the other sub-block (i.e., the four-bit sub-block). RD has the following properties:

- a) RD at the beginning of the six-bit sub-block is the RD at the end of the preceding character;
- b) RD at the beginning of the four-bit sub-block is the RD at the end of the preceding six-bit sub-block;
- and
- c) RD at the end of the character is the RD at the end of the four-bit sub-block.

RD for the sub-blocks shall be calculated as follows:

- a) if the sub-block contains more ones than zeros, then RD at the end of a sub-block is positive;
- b) if the sub-block contains more zeros than ones, then RD at the end of a sub-block is negative; or
- c) if the sub-block contains equal numbers of zeros and ones, then:
  - A) if it is a six-bit sub-block containing 000111b, then RD at the end of the sub-block is positive;
  - B) if it is a six-bit sub-block containing 111000b, then RD at the end of the sub-block is negative;
  - C) if it is a four-bit sub-block containing 0011b, then RD at the end of the sub-block is positive;
  - D) if it is a four-bit sub-block containing 1100b, then RD at the end of the sub-block is negative; or
  - E) otherwise, RD at the end of the sub-block is the same as at the beginning of the sub-block.

All sub-blocks with equal numbers of zeros and ones have neutral disparity (i.e., the ending disparity is the same as the beginning disparity). In order to limit the run length of zeros or ones across adjacent sub-blocks, the 8b10b code rules specify that sub-blocks encoded as 000111b or 0011b are generated only when the RD at the beginning of the sub-block is positive, ensuring that RD at the end of these sub-blocks is also positive. Likewise, sub-blocks containing 111000b or 1100b are generated only when the RD at the beginning of the sub-block is negative, ensuring that RD at the end of these sub-blocks is also negative.

Running disparity (RD) shall be maintained separately on each physical link in each direction. During a connection (see 4.1.12), expander devices shall convert incoming 10-bit characters to 8-bit bytes and generate the 10-bit character with correct disparity for the output physical link. Phys within a device may or may not begin operation with the same disparity.

### 5.3.6 Data characters

Table 49 defines the data characters (i.e., Dxx.y characters) and shall be used for both generating characters (i.e., encoding) and checking the validity of received characters (i.e., decoding).

Table 49 defines the data characters.

**Table 49 – Data characters (part 1 of 6)**

Name	Data byte		Data character (binary representation)	
	Binary representation (HGF EDCBA)	Hexadecimal representation	Current RD- abcdei fghj	Current RD+ abcdei fghj
D00.0	000 00000	00h	100111 0100	011000 1011
D01.0	000 00001	01h	011101 0100	100010 1011
D02.0	000 00010	02h	101101 0100	010010 1011
D03.0	000 00011	03h	110001 1011	110001 0100
D04.0	000 00100	04h	110101 0100	001010 1011
D05.0	000 00101	05h	101001 1011	101001 0100
D06.0	000 00110	06h	011001 1011	011001 0100
D07.0	000 00111	07h	111000 1011	000111 0100
D08.0	000 01000	08h	111001 0100	000110 1011
D09.0	000 01001	09h	100101 1011	100101 0100
D10.0	000 01010	0Ah	010101 1011	010101 0100
D11.0	000 01011	0Bh	110100 1011	110100 0100
D12.0	000 01100	0Ch	001101 1011	001101 0100
D13.0	000 01101	0Dh	101100 1011	101100 0100
D14.0	000 01110	0Eh	011100 1011	011100 0100
D15.0	000 01111	0Fh	010111 0100	101000 1011
D16.0	000 10000	10h	011011 0100	100100 1011
D17.0	000 10001	11h	100011 1011	100011 0100
D18.0	000 10010	12h	010011 1011	010011 0100
D19.0	000 10011	13h	110010 1011	110010 0100
D20.0	000 10100	14h	001011 1011	001011 0100
D21.0	000 10101	15h	101010 1011	101010 0100

Table 49 – Data characters (part 2 of 6)

Name	Data byte		Data character (binary representation)	
	Binary representation (HGF EDCBA)	Hexadecimal representation	Current RD- abcdei fghj	Current RD+ abcdei fghj
D22.0	000 10110	16h	011010 1011	011010 0100
D23.0	000 10111	17h	111010 0100	000101 1011
D24.0	000 11000	18h	110011 0100	001100 1011
D25.0	000 11001	19h	100110 1011	100110 0100
D26.0	000 11010	1Ah	010110 1011	010110 0100
D27.0	000 11011	1Bh	110110 0100	001001 1011
D28.0	000 11100	1Ch	001110 1011	001110 0100
D29.0	000 11101	1Dh	101110 0100	010001 1011
D30.0	000 11110	1Eh	011110 0100	100001 1011
D31.0	000 11111	1Fh	101011 0100	010100 1011
D00.1	001 00000	20h	100111 1001	011000 1001
D01.1	001 00001	21h	011101 1001	100010 1001
D02.1	001 00010	22h	101101 1001	010010 1001
D03.1	001 00011	23h	110001 1001	110001 1001
D04.1	001 00100	24h	110101 1001	001010 1001
D05.1	001 00101	25h	101001 1001	101001 1001
D06.1	001 00110	26h	011001 1001	011001 1001
D07.1	001 00111	27h	111000 1001	000111 1001
D08.1	001 01000	28h	111001 1001	000110 1001
D09.1	001 01001	29h	100101 1001	100101 1001
D10.1	001 01010	2Ah	010101 1001	010101 1001
D11.1	001 01011	2Bh	110100 1001	110100 1001
D12.1	001 01100	2Ch	001101 1001	001101 1001
D13.1	001 01101	2Dh	101100 1001	101100 1001
D14.1	001 01110	2Eh	011100 1001	011100 1001
D15.1	001 01111	2Fh	010111 1001	101000 1001
D16.1	001 10000	30h	011011 1001	100100 1001
D17.1	001 10001	31h	100011 1001	100011 1001
D18.1	001 10010	32h	010011 1001	010011 1001
D19.1	001 10011	33h	110010 1001	110010 1001
D20.1	001 10100	34h	001011 1001	001011 1001
D21.1	001 10101	35h	101010 1001	101010 1001
D22.1	001 10110	36h	011010 1001	011010 1001
D23.1	001 10111	37h	111010 1001	000101 1001
D24.1	001 11000	38h	110011 1001	001100 1001
D25.1	001 11001	39h	100110 1001	100110 1001
D26.1	001 11010	3Ah	010110 1001	010110 1001
D27.1	001 11011	3Bh	110110 1001	001001 1001
D28.1	001 11100	3Ch	001110 1001	001110 1001
D29.1	001 11101	3Dh	101110 1001	010001 1001
D30.1	001 11110	3Eh	011110 1001	100001 1001
D31.1	001 11111	3Fh	101011 1001	010100 1001
D00.2	010 00000	40h	100111 0101	011000 0101
D01.2	010 00001	41h	011101 0101	100010 0101
D02.2	010 00010	42h	101101 0101	010010 0101
D03.2	010 00011	43h	110001 0101	110001 0101
D04.2	010 00100	44h	110101 0101	001010 0101
D05.2	010 00101	45h	101001 0101	101001 0101
D06.2	010 00110	46h	011001 0101	011001 0101
D07.2	010 00111	47h	111000 0101	000111 0101
D08.2	010 01000	48h	111001 0101	000110 0101
D09.2	010 01001	49h	100101 0101	100101 0101
D10.2	010 01010	4Ah	010101 0101	010101 0101
D11.2	010 01011	4Bh	110100 0101	110100 0101

Table 49 – Data characters (part 3 of 6)

Name	Data byte		Data character (binary representation)	
	Binary representation (HGF EDCBA)	Hexadecimal representation	Current RD- abcdei fghj	Current RD+ abcdei fghj
D12.2	010 01100	4Ch	001101 0101	001101 0101
D13.2	010 01101	4Dh	101100 0101	101100 0101
D14.2	010 01110	4Eh	011100 0101	011100 0101
D15.2	010 01111	4Fh	010111 0101	101000 0101
D16.2	010 10000	50h	011011 0101	100100 0101
D17.2	010 10001	51h	100011 0101	100011 0101
D18.2	010 10010	52h	010011 0101	010011 0101
D19.2	010 10011	53h	110010 0101	110010 0101
D20.2	010 10100	54h	001011 0101	001011 0101
D21.2	010 10101	55h	101010 0101	101010 0101
D22.2	010 10110	56h	011010 0101	011010 0101
D23.2	010 10111	57h	111010 0101	000101 0101
D24.2	010 11000	58h	110011 0101	001100 0101
D25.2	010 11001	59h	100110 0101	100110 0101
D26.2	010 11010	5Ah	010110 0101	010110 0101
D27.2	010 11011	5Bh	110110 0101	001001 0101
D28.2	010 11100	5Ch	001110 0101	001110 0101
D29.2	010 11101	5Dh	101110 0101	010001 0101
D30.2	010 11110	5Eh	011110 0101	100001 0101
D31.2	010 11111	5Fh	101011 0101	010100 0101
D00.3	011 00000	60h	100111 0011	011000 1100
D01.3	011 00001	61h	011101 0011	100010 1100
D02.3	011 00010	62h	101101 0011	010010 1100
D03.3	011 00011	63h	110001 1100	110001 0011
D04.3	011 00100	64h	110101 0011	001010 1100
D05.3	011 00101	65h	101001 1100	101001 0011
D06.3	011 00110	66h	011001 1100	011001 0011
D07.3	011 00111	67h	111000 1100	000111 0011
D08.3	011 01000	68h	111001 0011	000110 1100
D09.3	011 01001	69h	100101 1100	100101 0011
D10.3	011 01010	6Ah	010101 1100	010101 0011
D11.3	011 01011	6Bh	110100 1100	110100 0011
D12.3	011 01100	6Ch	001101 1100	001101 0011
D13.3	011 01101	6Dh	101100 1100	101100 0011
D14.3	011 01110	6Eh	011100 1100	011100 0011
D15.3	011 01111	6Fh	010111 0011	101000 1100
D16.3	011 10000	70h	011011 0011	100100 1100
D17.3	011 10001	71h	100011 1100	100011 0011
D18.3	011 10010	72h	010011 1100	010011 0011
D19.3	011 10011	73h	110010 1100	110010 0011
D20.3	011 10100	74h	001011 1100	001011 0011
D21.3	011 10101	75h	101010 1100	101010 0011
D22.3	011 10110	76h	011010 1100	011010 0011
D23.3	011 10111	77h	111010 0011	000101 1100
D24.3	011 11000	78h	110011 0011	001100 1100
D25.3	011 11001	79h	100110 1100	100110 0011
D26.3	011 11010	7Ah	010110 1100	010110 0011
D27.3	011 11011	7Bh	110110 0011	001001 1100
D28.3	011 11100	7Ch	001110 1100	001110 0011
D29.3	011 11101	7Dh	101110 0011	010001 1100
D30.3	011 11110	7Eh	011110 0011	100001 1100
D31.3	011 11111	7Fh	101011 0011	010100 1100
D00.4	100 00000	80h	100111 0010	011000 1101
D01.4	100 00001	81h	011101 0010	100010 1101



Table 49 – Data characters (part 4 of 6)

Name	Data byte		Data character (binary representation)	
	Binary representation (HGF EDCBA)	Hexadecimal representation	Current RD- abcdei fghj	Current RD+ abcdei fghj
D02.4	100 00010	82h	101101 0010	010010 1101
D03.4	100 00011	83h	110001 1101	110001 0010
D04.4	100 00100	84h	110101 0010	001010 1101
D05.4	100 00101	85h	101001 1101	101001 0010
D06.4	100 00110	86h	011001 1101	011001 0010
D07.4	100 00111	87h	111000 1101	000111 0010
D08.4	100 01000	88h	111001 0010	000110 1101
D09.4	100 01001	89h	100101 1101	100101 0010
D10.4	100 01010	8Ah	010101 1101	010101 0010
D11.4	100 01011	8Bh	110100 1101	110100 0010
D12.4	100 01100	8Ch	001101 1101	001101 0010
D13.4	100 01101	8Dh	101100 1101	101100 0010
D14.4	100 01110	8Eh	011100 1101	011100 0010
D15.4	100 01111	8Fh	010111 0010	101000 1101
D16.4	100 10000	90h	011011 0010	100100 1101
D17.4	100 10001	91h	100011 1101	100011 0010
D18.4	100 10010	92h	010011 1101	010011 0010
D19.4	100 10011	93h	110010 1101	110010 0010
D20.4	100 10100	94h	001011 1101	001011 0010
D21.4	100 10101	95h	101010 1101	101010 0010
D22.4	100 10110	96h	011010 1101	011010 0010
D23.4	100 10111	97h	111010 0010	000101 1101
D24.4	100 11000	98h	110011 0010	001100 1101
D25.4	100 11001	99h	100110 1101	100110 0010
D26.4	100 11010	9Ah	010110 1101	010110 0010
D27.4	100 11011	9Bh	110110 0010	001001 1101
D28.4	100 11100	9Ch	001110 1101	001110 0010
D29.4	100 11101	9Dh	101110 0010	010001 1101
D30.4	100 11110	9Eh	011110 0010	100001 1101
D31.4	100 11111	9Fh	101011 0010	010100 1101
D00.5	101 00000	A0h	100111 1010	011000 1010
D01.5	101 00001	A1h	011101 1010	100010 1010
D02.5	101 00010	A2h	101101 1010	010010 1010
D03.5	101 00011	A3h	110001 1010	110001 1010
D04.5	101 00100	A4h	110101 1010	001010 1010
D05.5	101 00101	A5h	101001 1010	101001 1010
D06.5	101 00110	A6h	011001 1010	011001 1010
D07.5	101 00111	A7h	111000 1010	000111 1010
D08.5	101 01000	A8h	111001 1010	000110 1010
D09.5	101 01001	A9h	100101 1010	100101 1010
D10.5	101 01010	AAh	010101 1010	010101 1010
D11.5	101 01011	ABh	110100 1010	110100 1010
D12.5	101 01100	ACh	001101 1010	001101 1010
D13.5	101 01101	ADh	101100 1010	101100 1010
D14.5	101 01110	A Eh	011100 1010	011100 1010
D15.5	101 01111	AFh	010111 1010	101000 1010
D16.5	101 10000	B0h	011011 1010	100100 1010
D17.5	101 10001	B1h	100011 1010	100011 1010
D18.5	101 10010	B2h	010011 1010	010011 1010
D19.5	101 10011	B3h	110010 1010	110010 1010
D20.5	101 10100	B4h	001011 1010	001011 1010
D21.5	101 10101	B5h	101010 1010	101010 1010
D22.5	101 10110	B6h	011010 1010	011010 1010
D23.5	101 10111	B7h	111010 1010	000101 1010

Table 49 – Data characters (part 5 of 6)

Name	Data byte		Data character (binary representation)	
	Binary representation (HGF EDCBA)	Hexadecimal representation	Current RD- abcdei fghj	Current RD+ abcdei fghj
D24.5	101 11000	B8h	110011 1010	001100 1010
D25.5	101 11001	B9h	100110 1010	100110 1010
D26.5	101 11010	BAh	010110 1010	010110 1010
D27.5	101 11011	BBh	110110 1010	001001 1010
D28.5	101 11100	BCh	001110 1010	001110 1010
D29.5	101 11101	BDh	101110 1010	010001 1010
D30.5	101 11110	BEh	011110 1010	100001 1010
D31.5	101 11111	BFh	101011 1010	010100 1010
D00.6	110 00000	C0h	100111 0110	011000 0110
D01.6	110 00001	C1h	011101 0110	100010 0110
D02.6	110 00010	C2h	101101 0110	010010 0110
D03.6	110 00011	C3h	110001 0110	110001 0110
D04.6	110 00100	C4h	110101 0110	001010 0110
D05.6	110 00101	C5h	101001 0110	101001 0110
D06.6	110 00110	C6h	011001 0110	011001 0110
D07.6	110 00111	C7h	111000 0110	000111 0110
D08.6	110 01000	C8h	111001 0110	000110 0110
D09.6	110 01001	C9h	100101 0110	100101 0110
D10.6	110 01010	CAh	010101 0110	010101 0110
D11.6	110 01011	CBh	110100 0110	110100 0110
D12.6	110 01100	CCh	001101 0110	001101 0110
D13.6	110 01101	CDh	101100 0110	101100 0110
D14.6	110 01110	CEh	011100 0110	011100 0110
D15.6	110 01111	CFh	010111 0110	101000 0110
D16.6	110 10000	D0h	011011 0110	100100 0110
D17.6	110 10001	D1h	100011 0110	100011 0110
D18.6	110 10010	D2h	010011 0110	010011 0110
D19.6	110 10011	D3h	110010 0110	110010 0110
D20.6	110 10100	D4h	001011 0110	001011 0110
D21.6	110 10101	D5h	101010 0110	101010 0110
D22.6	110 10110	D6h	011010 0110	011010 0110
D23.6	110 10111	D7h	111010 0110	000101 0110
D24.6	110 11000	D8h	110011 0110	001100 0110
D25.6	110 11001	D9h	100110 0110	100110 0110
D26.6	110 11010	DAh	010110 0110	010110 0110
D27.6	110 11011	DBh	110110 0110	001001 0110
D28.6	110 11100	DCh	001110 0110	001110 0110
D29.6	110 11101	DDh	101110 0110	010001 0110
D30.6	110 11110	DEh	011110 0110	100001 0110
D31.6	110 11111	DFh	101011 0110	010100 0110
D00.7	111 00000	E0h	100111 0001	011000 1110
D01.7	111 00001	E1h	011101 0001	100010 1110
D02.7	111 00010	E2h	101101 0001	010010 1110
D03.7	111 00011	E3h	110001 1110	110001 0001
D04.7	111 00100	E4h	110101 0001	001010 1110
D05.7	111 00101	E5h	101001 1110	101001 0001
D06.7	111 00110	E6h	011001 1110	011001 0001
D07.7	111 00111	E7h	111000 1110	000111 0001
D08.7	111 01000	E8h	111001 0001	000110 1110
D09.7	111 01001	E9h	100101 1110	100101 0001
D10.7	111 01010	EAh	010101 1110	010101 0001
D11.7	111 01011	EBh	110100 1110	110100 1000
D12.7	111 01100	ECh	001101 1110	001101 0001
D13.7	111 01101	EDh	101100 1110	101100 1000

Table 49 – Data characters (part 6 of 6)

Name	Data byte		Data character (binary representation)	
	Binary representation (HGF EDCBA)	Hexadecimal representation	Current RD- abcdei fghj	Current RD+ abcdei fghj
D14.7	111 01110	Eh	011100 1110	011100 1000
D15.7	111 01111	Fh	010111 0001	101000 1110
D16.7	111 10000	F0h	011011 0001	100100 1110
D17.7	111 10001	F1h	100011 0111	100011 0001
D18.7	111 10010	F2h	010011 0111	010011 0001
D19.7	111 10011	F3h	110010 1110	110010 0001
D20.7	111 10100	F4h	001011 0111	001011 0001
D21.7	111 10101	F5h	101010 1110	101010 0001
D22.7	111 10110	F6h	011010 1110	011010 0001
D23.7	111 10111	F7h	111010 0001	000101 1110
D24.7	111 11000	F8h	110011 0001	001100 1110
D25.7	111 11001	F9h	100110 1110	100110 0001
D26.7	111 11010	FAh	010110 1110	010110 0001
D27.7	111 11011	FBh	110110 0001	001001 1110
D28.7	111 11100	FCh	001110 1110	001110 0001
D29.7	111 11101	FDh	101110 0001	010001 1110
D30.7	111 11110	FEh	011110 0001	100001 1110
D31.7	111 11111	FFh	101011 0001	010100 1110

### 5.3.7 Control characters

Table 50 defines the control characters (i.e., Kxx.y characters) and shall be used for both generating characters (i.e., encoding) and checking the validity of received characters (i.e., decoding).

**Table 50 – Control characters**

Name	Control byte		Control character (binary representation) <sup>a</sup>	
	Binary representation (HGF EDCBA)	Hexadecimal representation	Current RD- abcdei fghj	Current RD+ abcdei fghj
K28.0	000 11100	1Ch	001111 0100	110000 1011
K28.1 <sup>b</sup>	001 11100	3Ch	<u>001111</u> 1001	<u>110000</u> 0110
K28.2	010 11100	5Ch	001111 0101	110000 1010
K28.3	011 11100	7Ch	001111 0011	110000 1100
K28.4	100 11100	9Ch	001111 0010	110000 1101
K28.5 <sup>b</sup>	101 11100	BCh	<u>001111</u> 1010	<u>110000</u> 0101
K28.6	110 11100	DCh	001111 0110	110000 1001
K28.7 <sup>b c</sup>	111 11100	FCh	<u>001111</u> 1000	<u>110000</u> 0111
K23.7	111 10111	F7h	111010 1000	000101 0111
K27.7	111 11011	FBh	110110 1000	001001 0111
K29.7	111 11101	FDh	101110 1000	010001 0111
K30.7	111 11110	FEh	011110 1000	100001 0111
<sup>a</sup> Comma patterns, which are two bits of one polarity followed by five bits of the opposite polarity (i.e., 0011111b or 1100000b), are underlined. <sup>b</sup> K28.1, K28.5, and K28.7 are the only characters that contain comma patterns. Comma patterns do not appear in any data characters and do not appear across any adjacent data characters. <sup>c</sup> The K28.7 control character introduces an additional comma pattern starting with bits i and f when followed by any of the following characters: K28.y; D3.y; D11.y; D12.y; D19.y; D20.y; or D28.y, where y is a value in the range 0 to 7, inclusive. None of the other control characters introduce a comma pattern when adjacent to any other character. Therefore, K28.7 is not used, ensuring that comma patterns do not appear in any sequence of characters except the first 7 bits of K28.1 or K28.5.				

The only control characters used in this standard are K28.3, K28.5, and K28.6, as defined in table 51.

**Table 51 – Control character usage**

First character of a dword	Usage in SAS physical links	Usage in SATA physical links
K28.3	Primitives used only inside STP connections	All primitives except ALIGN
K28.5	ALIGN and most primitives defined in this standard	ALIGN
K28.6	Not used	SATA_ERROR

See 6.2 for details on primitives, which use the control characters defined in table 51.

### 5.3.8 Encoding characters in the transmitter

To transmit a data byte, the transmitter shall select the appropriate character from table 49 based on the current value of the transmitter's RD. To transmit a control byte, the transmitter shall select the appropriate character from table 50 based on the current value of the transmitter's RD. After transmitting the character, the transmitter shall calculate a new value for its RD based on that character. This new value shall be used as the transmitter's current RD for the next character transmitted. This process is called 8b10b encoding.

### 5.3.9 Decoding characters in the receiver

After receiving a character, the receiver shall search the character column in table 49 and table 50 corresponding to its current RD to determine the data byte or control byte to which the character corresponds. This process is called 10b8b decoding. If the received character is not found in the proper column, then the character shall be considered invalid and the dword containing the character shall be considered an invalid dword.

Regardless of the received character's validity, the received character shall be used to calculate a new value of RD in the receiver. This new value shall be used as the receiver's current RD for the next received character.

Detection of a code violation in a character does not always indicate that the character in which the code violation was detected is in error. Code violations may result from an error in a previous character that altered the RD of the bit stream but did not result in a detectable error in the character in which the error occurred. Table 52 shows an example of this behavior. Code violation errors may span dword boundaries. Expanders forwarding a dword in which a code violation was detected, forward the dword as an ERROR (see 6.2.6.7).

**Table 52 – Delayed code violation example**

	RD	First character	RD	Second character	RD	Third character	RD
Transmitted character stream	-	D21.1	-	D10.2	-	D23.5	+
Transmitted bit stream	-	101010 1001	-	010101 0101	-	111010 1010	+
Bit stream after error	-	101010 10 <u>1</u> 1 (error in second to last bit)	+	010101 0101	+	111010 1010	+
Decoded character stream	-	D21.0 (rather than D21.1) (not detected as an error)	+	D10.2 (no error)	+	Code violation (although D23.5 was received without error)	+

## 5.4 SAS dword mode bit order

Dwords transmitted in an STP connection shall be transmitted in the bit order specified by SATA.

In SAS dword mode, dwords for connections other than STP connections and outside of connections shall be transmitted in the bit order shown in figure 57.

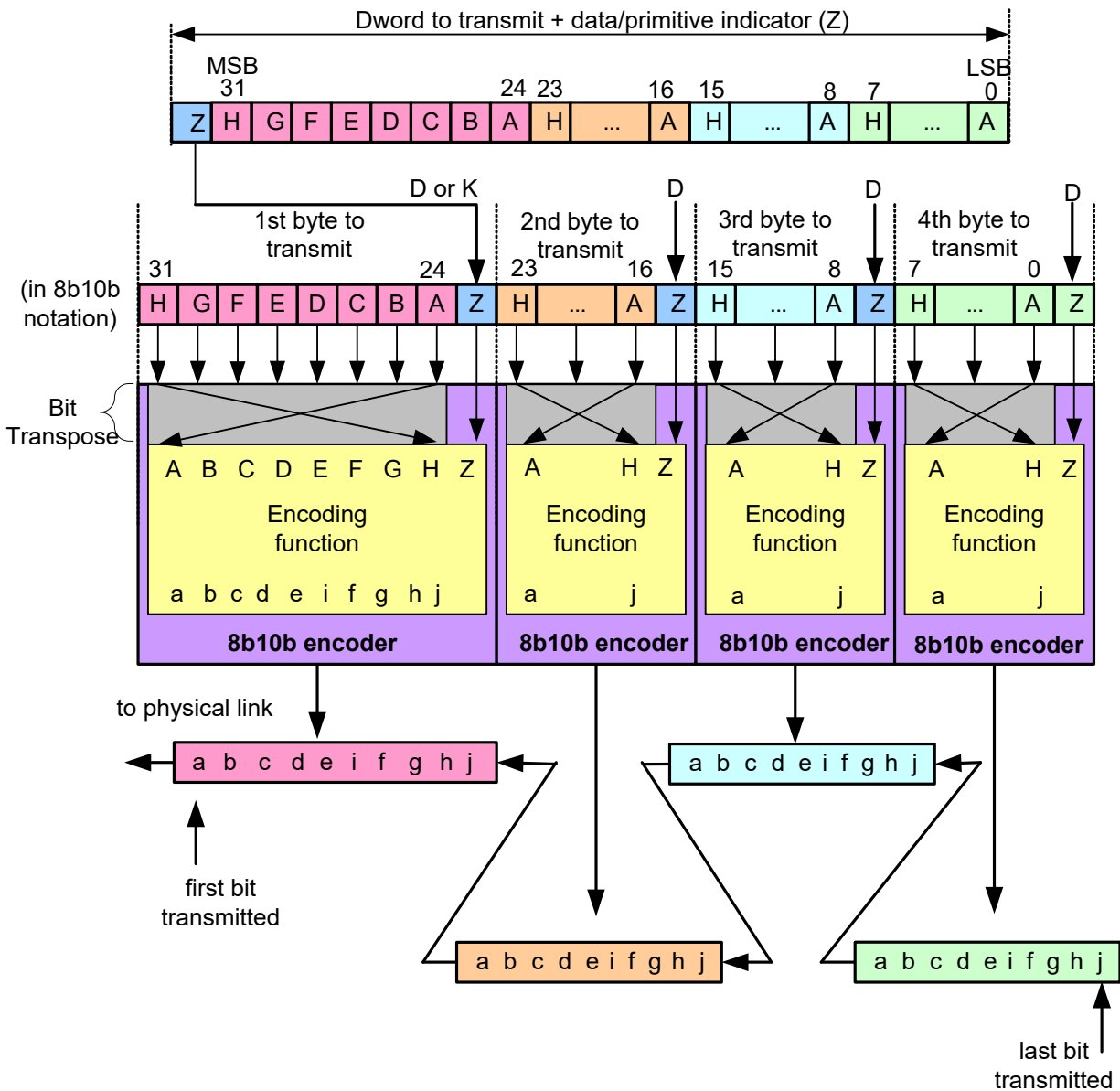


Figure 57 – SAS bit transmission logic

Figure 58 shows the SAS bit reception order.

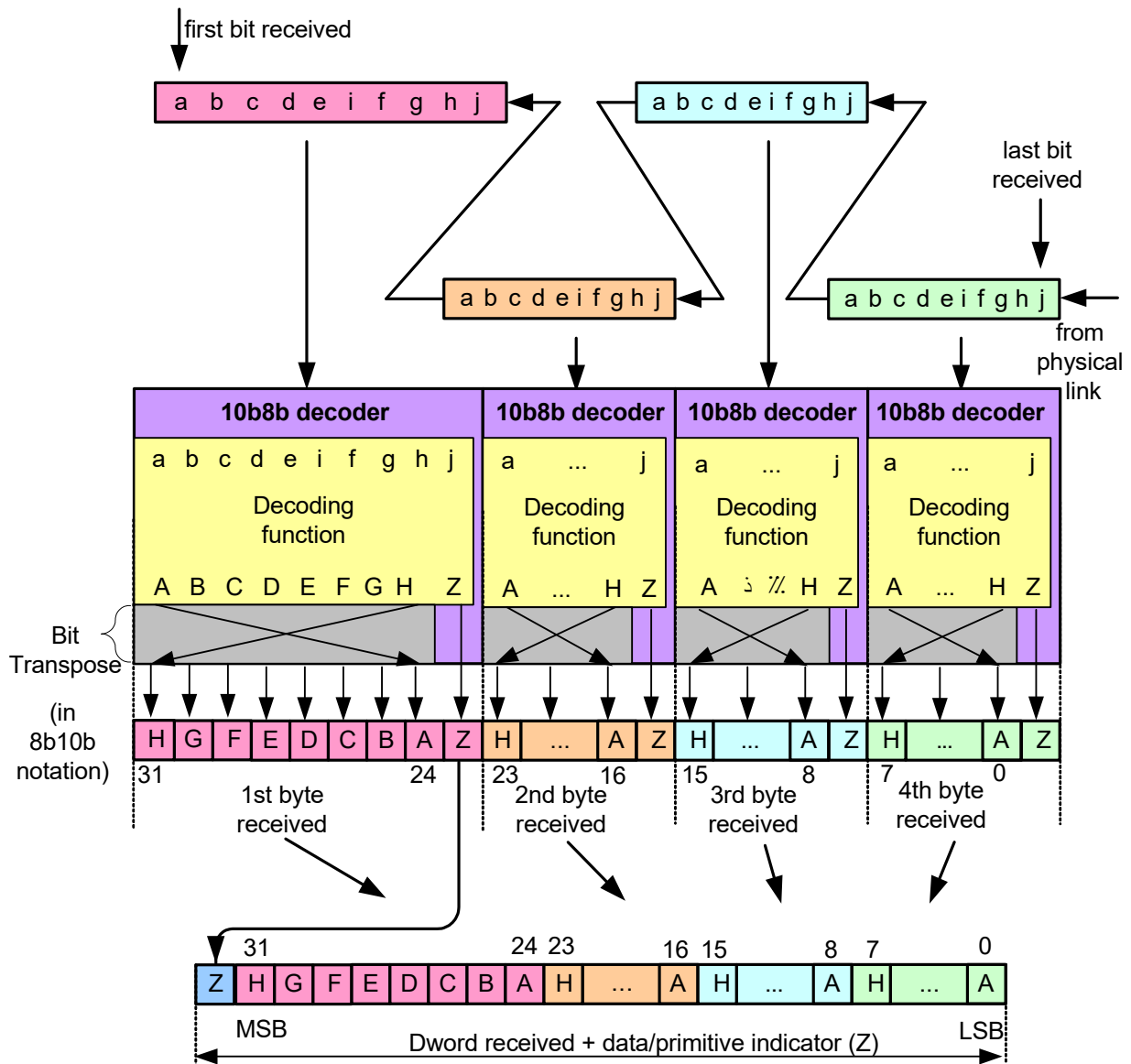


Figure 58 – SAS bit reception logic

## 5.5 SPL packet

### 5.5.1 SPL packet overview

All information transferred in the SAS packet mode is encoded into SPL packets (i.e., 150-bit blocks using 128b150b coding). Each SPL packet comprises:

- an SPL PACKET HEADER field (see table 53);
- an SPL packet payload (see 5.5.2); and
- forward error correction information (see table 53).

SPL packets are transmitted serially across the physical link.



If a transmitter has none of the following to transmit:

- a) primitives;
- b) binary primitives;
- c) extended binary primitives;
- d) SPL frame segments; or
- e) scrambled idle segments,

then the transmitter shall transmit an SPL packet payload containing an idle dword segment (see 6.6).

Within an SPL packet, the SPL frame header specifies the type of the SPL packet payload (see table 54).

An SSP frame, an SMP frame, an STP frame, or an address frame comprises one or more SPL packets.

Integrity of the SPL packet payload is maintained by:

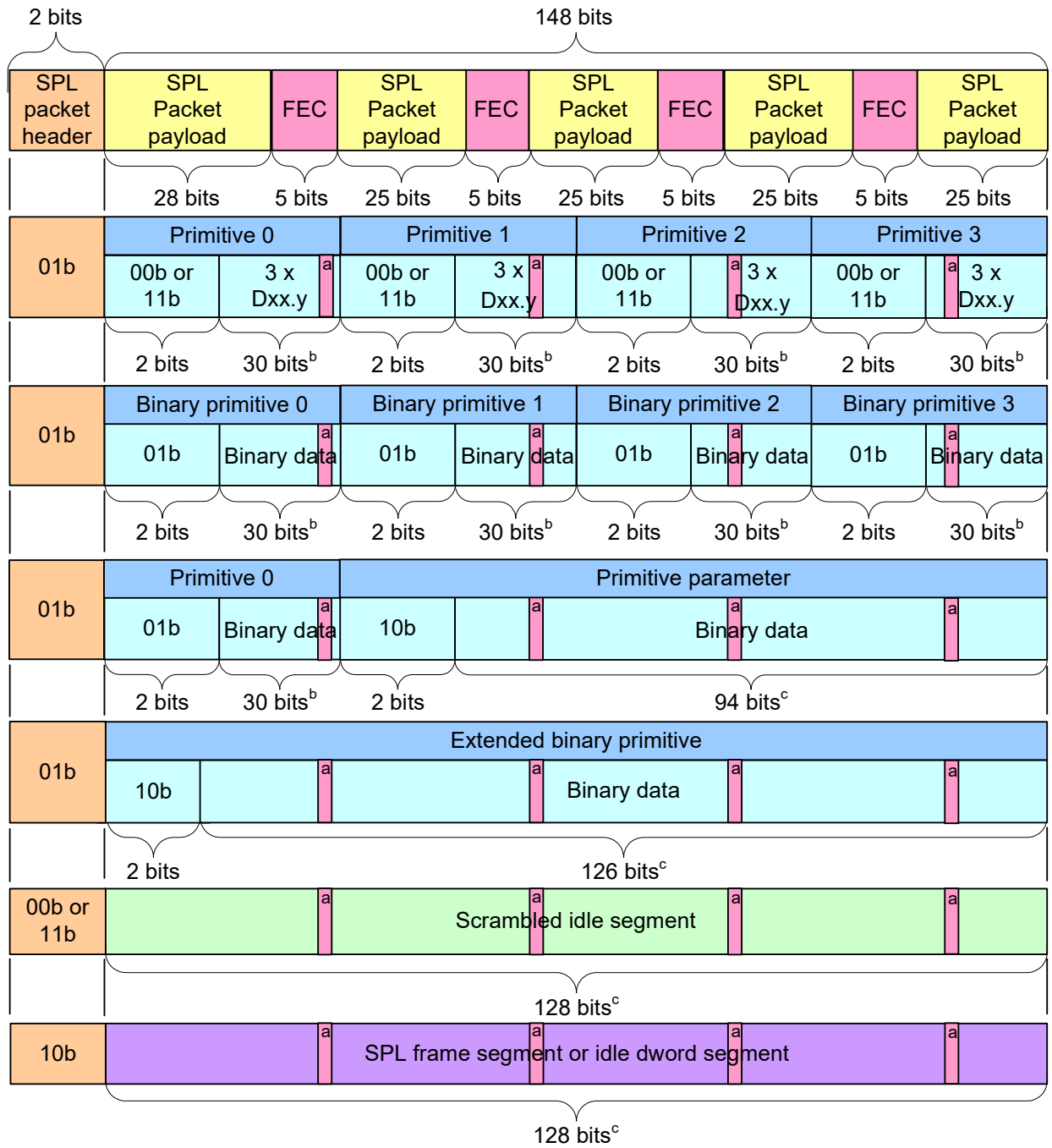
- a) detection of bit errors in the SPL PACKET HEADER field;
- b) defining a Hamming distance of seven or more between primitives;
- c) forward error correction distributed throughout the SPL packet payload; and
- d) CRC protection of an SSP frame, an SMP frame, an STP frame, or an address frame.

### 5.5.2 SPL packet format

The format of the SPL packet is shown in figure 59 and table 53.

When a phy sends or receives an SPL packet:

- a) the SPL PACKET HEADER field specifies the format of the subsequent SPL packet payload;
  - b) the SPL packet payload contains:
    - A) a scrambled idle segment;
    - B) an idle dword segment;
    - C) an SPL frame segment that contains data dwords; or
    - D) a primitive segment that contains:
      - a) primitives and a primitive parameter, if any;
      - b) binary primitives and a primitive parameter, if any; or
      - c) an extended binary primitive;
- and
- c) the SPL packet contains forward error correction information (see 5.5.7 and SAS-4).



FEC = forward error correction

- <sup>a</sup> Five bits of forward error correction information.
- <sup>b</sup> Excludes five bits of forward error correction information.
- <sup>c</sup> Excludes forward error correction information.

**Figure 59 – SPL packet formats with error correction information**

Table 53 – SPL packet

								Bit	
								1	0
								SPL PACKET HEADER	
Byte\Bit	7	6	5	4	3	2	1	0	
0	SPL packet payload								
...									
15									
16	Forward error correction <sup>a</sup>								
17									
18									
<sup>a</sup> The 20-bit forward error correction's position (i.e., byte 16, byte 17, and bits 3 to 0 of byte 18) shown in this table is a logical representation of the position of the forward error correction. The physical position of the forward error correction is distributed throughout the SPL packet payload as described in table 69.									

The SPL PACKET HEADER field (see table 54) specifies the contents of the SPL packet payload.

Table 54 – SPL PACKET HEADER field

Code	Description
00b <sup>a</sup>	The SPL packet payload contains a scrambled idle segment.
01b	The SPL packet payload descriptor contains a primitive segment containing the following that are not scrambled: a) primitives; b) binary primitives c) primitive parameters; or d) extended binary primitive.
10b	The SPL packet payload contains a scrambled: a) segment of an SSP frame; b) segment of an SMP frame; c) segment of an address frame; d) segment of an STP frame; or e) idle dword segment.
11b <sup>a</sup>	The SPL packet payload contains a scrambled idle segment.
<sup>a</sup> The selection of 00b or 11b is as described in 6.8.3.	

The forward error correction contains information that is used by a phy to:

- a) detect errors; and
- b) attempt to correct errors in the SPL packet.

### 5.5.3 SPL packet payload that contains a scrambled idle segment

If the SPL PACKET HEADER field is set to 00b or 11b, then the SPL packet payload contains a scrambled idle segment as defined in table 55.

**Table 55 – SPL packet payload that contains a scrambled idle segment**

Byte\Bit	7	6	5	4	3	2	1	0
0	DWORDS TO BE SCRAMBLED							
...								
15								

The DWORDS TO BE SCRAMBLED field contains a scrambled idle segment.

### 5.5.4 SPL packet payload that contains a primitive segment

If the SPL PACKET HEADER field is set to 01b, then the SPL packet payload contains a primitive segment with:

- a) primitives and primitive parameters, if any, as defined in table 58;
- b) binary primitives and primitive parameters, if any, as defined in table 56; or
- c) an extended binary primitive as defined in table 63.

If the primitive segment contains primitives, binary primitives, or a primitive parameter as described in table 58, then:

- a) for primitives:
  - A) the first character of the primitive (i.e., control character) is represented as a reduced control character and is labeled in table 57 as a reduced control character; and
  - B) the second character, the third character, and the last character of the primitive are positioned as described in table 57;
- b) for a primitive parameter the least significant two bits of the primitive parameter shall be as described in table 65; and
- c) for binary primitives the least significant two bits of the binary primitive shall be as described in table 64 or table 65.

If the primitive segment contains a primitive parameter, then:

- a) the length in dwords of the primitive parameter;
- b) the byte locations of the primitive parameter within the primitive segment; and
- c) the byte locations of the primitive or binary primitive associated with the primitive parameter within the primitive segment,

shall be as described in table 56.

Table 56 – Primitive parameter location within primitive segment

Primitive parameter		Byte locations of associated primitive or binary primitive within primitive segment	Example
Length (dwords)	Byte locations within primitive segment		
1	4 to 7	0 to 3	table 59
	8 to 11	4 to 7	not shown
	12 to 15	8 to 11	table 60
2	4 to 11	0 to 3	table 61
	8 to 15	4 to 7	not shown
3	4 to 15	0 to 3	table 62

Table 57 – Primitive segment primitive data character placement

Byte\Bit	7	6	5	4	3	2	1	0
n	Second character					i	First character <sup>a</sup>	
n+1	f	j			a		d	
n+2	e		Third character			j	a	b
n+3	Last character					j		
Key:								
a = first 8b10b bit of a character (see 5.4)								
j = last 8b10b bit of a character (see 5.4)								
<sup>a</sup> Represents a K28.5 if set to 00b or a K28.3 if set 11b.								

Table 58 – Primitive segment SPL packet payload containing primitives and binary primitives

Byte\Bit	7	6	5	4	3	2	1	0
0	PRIMITIVE0						PRIMITIVE SYNCHRONIZE SELECT	
...								
3								
4	PRIMITIVE1						CONTROL1	
...								
7								
8	PRIMITIVE2						CONTROL2	
...								
11								
12	PRIMITIVE3						CONTROL3	
...								
15								

**Table 59 – Primitive segment SPL packet payload that contains primitives, binary primitives, and a 1-dword primitive parameter in second dword**

Byte\Bit	7	6	5	4	3	2	1	0
0	PRIMITIVE0						PRIMITIVE SYNCHRONIZE SELECT	
...								
3								
4	PRIMITIVE PARAMETER				PARAMETER LENGTH (01b)		CONTROL1	
...								
7								
8	PRIMITIVE2						CONTROL2	
...								
11								
12	PRIMITIVE3						CONTROL3	
...								
15								

**Table 60 – Primitive segment SPL packet payload that contains primitives, binary primitives, and a 1-dword primitive parameter in fourth dword**

Byte\Bit	7	6	5	4	3	2	1	0
0	PRIMITIVE0						PRIMITIVE SYNCHRONIZE SELECT	
...								
3								
4	PRIMITIVE1						CONTROL1	
...								
7								
8	PRIMITIVE2						CONTROL2	
...								
11								
12	PRIMITIVE PARAMETER				PARAMETER LENGTH (01b)		CONTROL3	
...								
15								

**Table 61 – Primitive segment SPL packet payload that contains primitives, binary primitives, and a 2-dword primitive parameter**

Byte\Bit	7	6	5	4	3	2	1	0
0	PRIMITIVE0						PRIMITIVE SYNCHRONIZE SELECT	
...								
3								
4	PRIMITIVE PARAMETER				PARAMETER LENGTH (10b)		CONTROL1	
...								
11								
12	PRIMITIVE3						CONTROL3	
...								
15								

**Table 62 – Primitive segment SPL packet payload that contains a primitive or binary primitive, and 3-dword primitive parameter**

Byte\Bit	7	6	5	4	3	2	1	0
0	PRIMITIVE0						PRIMITIVE SYNCHRONIZE SELECT	
...								
3								
4	PRIMITIVE PARAMETER				PARAMETER LENGTH (11b)		CONTROL1	
...								
15								

**Table 63 – Primitive segment SPL packet payload that contains an extended binary primitives**

Byte\Bit	7	6	5	4	3	2	1	0
0	EXTENDED BINARY PRIMITIVE						PRIMITIVE SYNCHRONIZE SELECT (10b)	
1								
...								
15								

The PRIMITIVE SYNCHRONIZE SELECT field is defined in table 64.

**Table 64 – PRIMITIVE SYNCHRONIZE SELECT field**

Code	Description
00b <sup>a</sup>	The control character for the associated PRIMITIVE0 field is K28.5.
01b	The associated PRIMITIVE0 field is a binary primitive.
10b <sup>b</sup>	The SPL packet payload is formatted as an extended binary primitive as shown in table 63.
11b <sup>a</sup>	The control character for the associated PRIMITIVE0 field is K28.3.
<sup>a</sup> The PRIMITIVE0 field shall contain the 8b10b data characters for the version of the primitive with a starting RD+ disparity (see 5.3.5). <sup>b</sup> The EXTENDED BINARY PRIMITIVE field shall contain 126 bits that specify an extended binary primitive (see 6.4).	

The CONTROL1 field, CONTROL2 field, and CONTROL3 field are defined in table 65.

**Table 65 – CONTROL1 field, CONTROL2 field, and CONTROL3 field**

Code	Description
00b <sup>a</sup>	The control character for the associated PRIMITIVE1 field, PRIMITIVE2 field, or PRIMITIVE3 field is K28.5.
01b	Specifies the associated PRIMITIVE1 field, PRIMITIVE2 field, or PRIMITIVE3 field is a binary primitive.
10b	Specifies the associated PRIMITIVE1 field, PRIMITIVE2 field, or PRIMITIVE3 field is a primitive parameter associated with a primitive or binary primitive (see table 56).
11b <sup>a</sup>	The control character for the associated PRIMITIVE1 field, PRIMITIVE2 field, or PRIMITIVE3 field is K28.3.
<sup>a</sup> The PRIMITIVE1 field, PRIMITIVE2 field, and PRIMITIVE3 field shall contain the 8b10b data characters for the version of the primitives with a starting RD+ disparity (see 5.3.5).	

The PARAMETER LENGTH field contains the length in dwords of:

- a) the PARAMETER LENGTH field;
- b) the CONTROL1 field, CONTROL2 field, or CONTROL3 field; and
- c) the PRIMITIVE PARAMETER field.

If the transmitter has one single primitive sequence or one binary primitive to transmit within a single primitive segment, then the transmitter shall set:

- a) the PRIMITIVE0 field to that single primitive sequence or binary primitive;
- b) the PRIMITIVE1 field to ALIGN (1) or a primitive parameter;
- c) the PRIMITIVE2 field to ALIGN (2) or a primitive parameter; and
- d) the PRIMITIVE3 field to ALIGN (3) or a primitive parameter.

If the transmitter has two single primitive sequences or two binary primitives to transmit within a single primitive segment, then the transmitter shall set:

- a) the PRIMITIVE0 field to one of the single primitive sequences or binary primitives;



- b) the PRIMITIVE1 field to the other single primitive sequence or binary primitive;
- c) the PRIMITIVE2 field to ALIGN (2) or a primitive parameter; and
- d) the PRIMITIVE3 field to ALIGN (3) or a primitive parameter.

If the transmitter has three single primitive sequences or three binary primitives to transmit within a single primitive segment, then the transmitter shall set:

- a) the PRIMITIVE0 field to one of the single primitive sequences or binary primitives;
- b) the PRIMITIVE1 field to another single primitive sequence or binary primitive;
- c) the PRIMITIVE2 field to the last single primitive sequence or binary primitive; and
- d) the PRIMITIVE3 field to ALIGN (3) or a primitive parameter.

If the transmitter has four single primitive sequences or four binary primitives to transmit within a single primitive segment, then the transmitter shall set:

- a) the PRIMITIVE0 field to one of the single primitive sequences or binary primitives;
- b) the PRIMITIVE1 field to another single primitive sequence or binary primitive;
- c) the PRIMITIVE2 field to another single primitive sequence or binary primitive; and
- d) the PRIMITIVE3 field to the last single primitive sequence or binary primitive (see figure 60).

The transmitter shall insert a triple primitive sequence or an extended primitive sequence into two SPL packets as follows:

- 1) in the first SPL packet set:
  - A) the PRIMITIVE0 field to ALIGN (1), a single primitive sequence, or a single binary primitive;
  - B) the PRIMITIVE1 field to ALIGN (2), a single primitive sequence, or a single binary primitive; and
  - C) the PRIMITIVE2 field and the PRIMITIVE3 field to two primitives of the triple primitive sequence or extended primitive sequence (see figure 60);

and
- 2) in the second SPL packet set:
  - A) the PRIMITIVE0 field to one of the primitives of the triple primitive sequence or extended primitive sequence (see figure 64);
  - B) the PRIMITIVE1 field to ALIGN (1), a single primitive sequence, a single binary primitive, or a primitive parameter;
  - C) the PRIMITIVE2 field to ALIGN (2), a single primitive sequence, a single binary primitive, or a primitive parameter; and
  - D) the PRIMITIVE3 field to ALIGN (3), a single primitive sequence, a single binary primitive, or a primitive parameter,

however, any number of SPL packet payloads containing scrambled idle segments or deletable extended binary primitives may be transmitted between the SPL packets containing the triple primitive sequence or the extended primitive sequence.

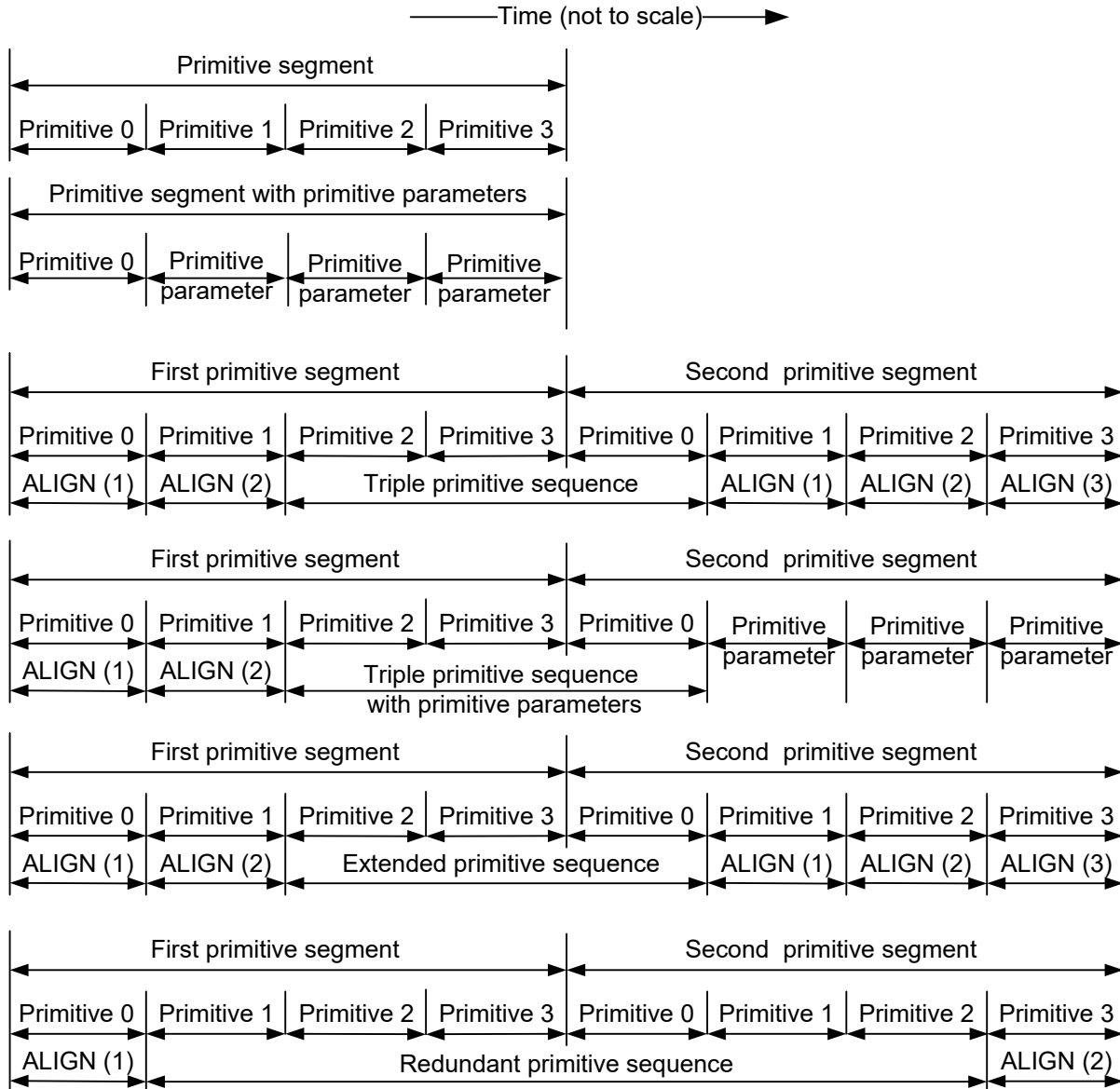
The transmitter shall insert a redundant primitive sequence into two SPL packets as follows:

- 1) in the first SPL packet set:
  - A) the PRIMITIVE0 field to ALIGN (1), a single primitive sequence, or a single binary primitive; and
  - B) the PRIMITIVE1 field, the PRIMITIVE2 field, and the PRIMITIVE3 field to three primitives of the redundant primitive sequence (see figure 60);

and
- 2) in the second SPL packet set:
  - A) the PRIMITIVE0 field, the PRIMITIVE1 field, and the PRIMITIVE2 field to three primitives of the redundant primitive sequence (see figure 60); and
  - B) the PRIMITIVE3 field to ALIGN (2), a single primitive, a single binary primitive, or a primitive parameter,

however, any number of SPL packet payloads containing scrambled idle segments or deletable extended binary primitives may be transmitted between the SPL packets containing the redundant primitive sequence.

As a primitive segment is forwarded through an expander, one or more deletable primitives may be replaced with ALIGN primitives (e.g., substitution of a NOTIFY) (see 6.2.5.3.1). Receivers shall be capable of decoding any combination of ALIGNs and single primitives.



**Figure 60 – Examples of primitive segment alignment**

### 5.5.5 SPL packet payload that contains an SPL frame segment

If the SPL PACKET HEADER field is set to 10b within a frame, then the SPL packet payload contains the SPL frame segment shown in table 66. SPL frame segments are used to transmit portions of frames.

**Table 66 – SPL packet payload that contains an SPL frame segment**

Byte\Bit	7	6	5	4	3	2	1	0
0	SPL FRAME SEGMENT DWORD 0							
...								
3								
4	SPL FRAME SEGMENT DWORD 1							
...								
7								
8	SPL FRAME SEGMENT DWORD 2							
...								
11								
12	SPL FRAME SEGMENT DWORD 3							
...								
15								

The SPL FRAME SEGMENT DWORD 0 field, the SPL FRAME SEGMENT DWORD 1 field, the SPL FRAME SEGMENT DWORD 2 field, and the SPL FRAME SEGMENT DWORD 3 field contain a scrambled (see 6.8.3):

- a) SSP frame segment;
- b) SMP frame segment;
- c) STP frame segment; or
- d) address frame segment.

If the CRC is not the final dword of the SPL frame segment, then pad dwords shall be included to fill any unused dwords between the CRC and the end of the SPL frame segment. The pad dwords shall be set to 00000000h (see 6.20.3.3).

### 5.5.6 SPL packet payload that contains an idle dword segment

If the SPL PACKET HEADER field is set to 10b and is received outside a frame, then the SPL packet payload contains the idle dword segment shown in table 66. Idle dword segments are used to transmit idle dwords on idle physical links (see 6.6).

**Table 67 – SPL packet payload that contains an idle dword segment**

Byte\Bit	7	6	5	4	3	2	1	0
0	SPL IDLE DWORD 0							
...								
3								
4	SPL IDLE DWORD 1							
...								
7								
8	SPL IDLE DWORD 2							
...								
11								
12	SPL IDLE DWORD 3							
...								
15								

The SPL IDLE DWORD 0 field, the SPL IDLE DWORD 1 field, the SPL IDLE DWORD 2 field, and the SPL IDLE DWORD 3 field each contain a scrambled (see 6.8.3) idle dword.

### 5.5.7 Forward error correction

#### 5.5.7.1 Forward error correction overview

When in SAS packet mode, the transmitter and receiver shall use for forward error correction purposes a Reed Solomon (n, k) code where the relationship of parameters in the code is as follows:

$$0 < k < n < 2^m + 2$$

where:

n is 30;  
k is 26; and  
m is 5.

The smallest possible number of differences between two information sequences of the Reed Solomon code is:

$$d_{\min} = ((n - k) + 1) = ((2 \times t) + 1)$$

where:

n is 30;  
k is 26;  
t is 2; and  
d<sub>min</sub> is 5.

Table 68 defines the notation and associated equations used in Reed Solomon encoding and decoding used in this standard.

**Table 68 – Reed Solomon code notation and definitions**

Notation	Definition
$G(x)$	<p>Generator Polynomial:</p> $G(x) = (x - \alpha) \times (x - \alpha^2) \times (x - \alpha^3) \times (x - \alpha^4) = x^4 + (\alpha^{24} \times x^3) + (\alpha^{19} \times x^2) + (\alpha^{29} \times x) + \alpha^{10}$ <p>where:</p> <p><math>\alpha</math> is a root of the primitive polynomial <math>1 + x^2 + x^5</math> over the <math>GF(2^5)</math>.</p>
$M(x)$	<p>Message to be encoded (i.e., 2-bit SPL PACKET HEADER field plus 128-bit SPL packet payload):</p> $M(x) = m_0 + (m_1 \times x) + (m_2 \times x^2) + \dots + (m_{25} \times x^{25})$ <p>where:</p> <p><math>m_i</math> is a message symbol and the index <math>i</math> denotes that <math>m_i</math> is more significant than <math>m_{(i-1)}</math>.</p>
$P(x)$	<p>Remainder parity check symbols (i.e., 20-bit forward error correction):</p> $P(x) = \frac{x^4 \times M(x)}{G(x)} = p_0 + (p_1 \times x) + (p_2 \times x^2) + (p_3 \times x^3)$ <p>where:</p> <p><math>p_i</math> is a parity symbol and the index <math>i</math> denotes that <math>p_i</math> is more significant than <math>p_{(i-1)}</math>.</p>
$T(x)$	<p>Transmitted codeword polynomial:</p> $T(x) = P(x) + (x^4 \times M(x))$
$E(x)$	<p>Errors induced:</p> $E(x) = R(x) - T(x)$
$R(x)$	<p>Received codeword polynomial, including errors:</p> $R(x) = T(x) + E(x)$
$E_r(x)$	<p>Receiver's estimate of errors:</p> $E_r(x) = e_0 + (e_1 \times x) + (e_2 \times x^2) + \dots + (e_{29} \times x^{29})$ <p>where:</p> <p><math>e_i</math> is an error symbol and the index <math>i</math> denotes that <math>e_i</math> is more significant than <math>e_{(i-1)}</math>.</p> <p>Non-zero <math>e_i</math> define the error values and locations as determined by the Reed Solomon decode process.</p>
$M_r(x)$	<p>Decoded message based on receiver's estimate of errors:</p> $M_r(x) = \frac{(R(x) - E_r(x))}{x^4}$
<p>Key:</p> <p>GF = Galois field (i.e., a mathematical field that contains a finite number of elements)</p>	

### 5.5.7.2 Forward error correction encoding

The Reed Solomon code utilizes a generator polynomial  $G(x)$  to encode the message symbols  $M(x)$  and produce the parity check symbols  $P(x)$ , as described in table 68. The original message symbols  $M(x)$  and the parity check symbols  $P(x)$  comprise the entire 30-symbol codeword  $T(x)$  that is transmitted.

The message symbols occupy the mathematical higher order 26 symbols and the parity occupies the mathematical lower order four symbols of the 30-symbol codeword  $T(x)$ . The specific ordering of transmitted symbols is as shown in figure 61 and table 68.

The generator polynomial  $G(x)$  is irreducible and has a degree of four that is equal to the number of parity symbols. The parity symbols  $P(x)$  are generated from the remainder of the message  $M(x)$  divided by the generator polynomial  $G(x)$ , as described in table 68.

Figure 61 shows the forward error correction encoding process to generate  $T(x)$  and  $T(x)$  is subsequently passed to the physical link with ordering as shown in the figure 61 and table 69.

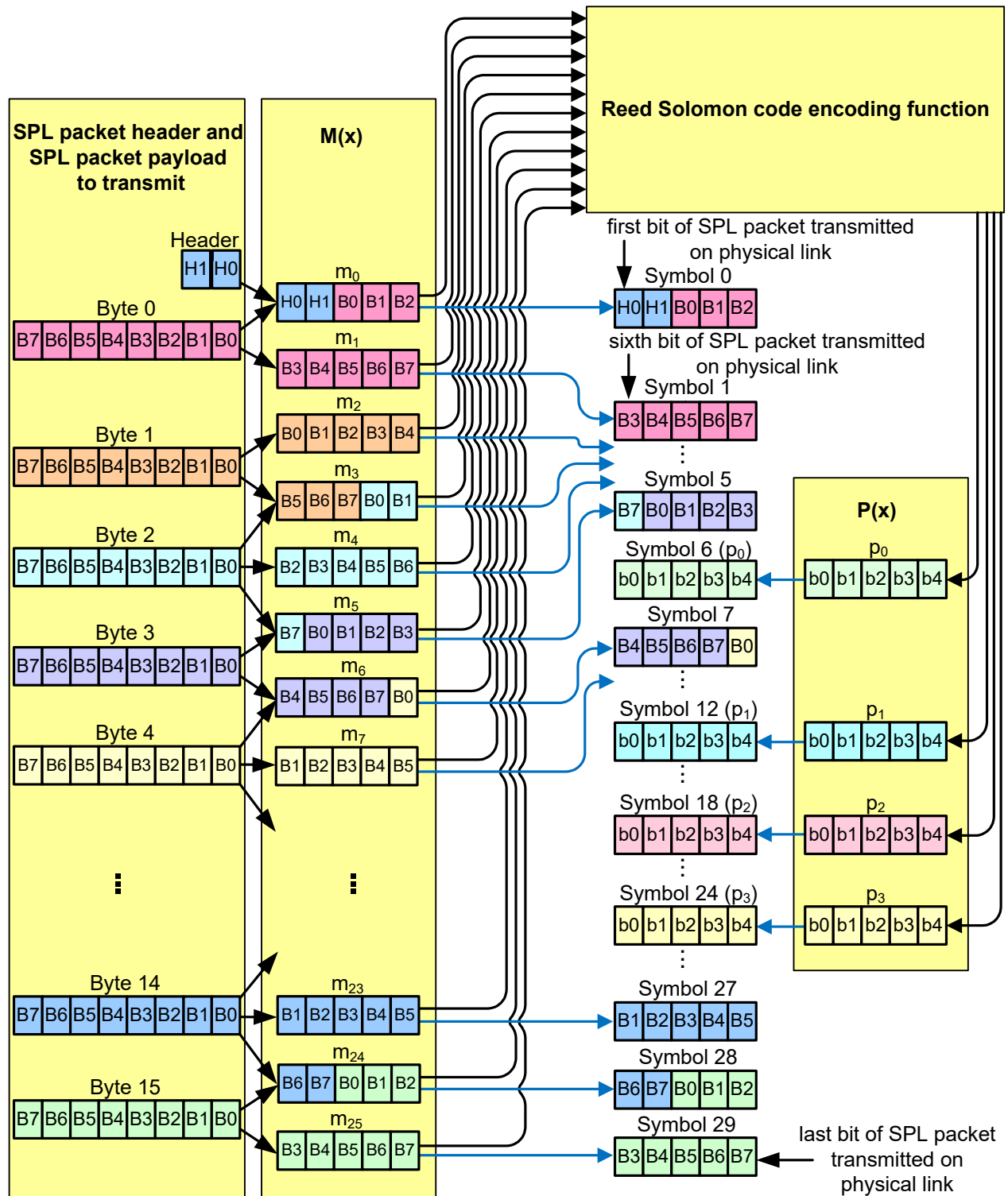


Figure 61 – Forward error correction encoding and transmission

Table 69 specifies the full set of source data and ordering of symbols sent on the physical link.

**Table 69 – Ordering of parity and message symbols transmitted**

Transmit symbol order <sup>a</sup>	Symbol type	Symbol label	Symbol source <sup>b</sup>				
			bit 0	bit 1	bit 2	bit 3	bit 4
0	Message	m <sub>0</sub>	H0	H1	D(0, 0)	D(0, 1)	D(0, 2)
1	Message	m <sub>1</sub>	D(0, 3)	D(0, 4)	D(0, 5)	D(0, 6)	D(0, 7)
2	Message	m <sub>2</sub>	D(1, 0)	D(1, 1)	D(1, 2)	D(1, 3)	D(1, 4)
3	Message	m <sub>3</sub>	D(1, 5)	D(1, 6)	D(1, 7)	D(2, 0)	D(2, 1)
4	Message	m <sub>4</sub>	D(2, 2)	D(2, 3)	D(2, 4)	D(2, 5)	D(2, 6)
5	Message	m <sub>5</sub>	D(2, 7)	D(3, 0)	D(3, 1)	D(3, 2)	D(3, 3)
6	Parity	p <sub>0</sub>	P(0, 0)	P(0, 1)	P(0, 2)	P(0, 3)	P(0, 4)
7	Message	m <sub>6</sub>	D(3, 4)	D(3, 5)	D(3, 6)	D(3, 7)	D(4, 0)
8	Message	m <sub>7</sub>	D(4, 1)	D(4, 2)	D(4, 3)	D(4, 4)	D(4, 5)
9	Message	m <sub>8</sub>	D(4, 6)	D(4, 7)	D(5, 0)	D(5, 1)	D(5, 2)
10	Message	m <sub>9</sub>	D(5, 3)	D(5, 4)	D(5, 5)	D(5, 6)	D(5, 7)
11	Message	m <sub>10</sub>	D(6, 0)	D(6, 1)	D(6, 2)	D(6, 3)	D(6, 4)
12	Parity	p <sub>1</sub>	P(1, 0)	P(1, 1)	P(1, 2)	P(1, 3)	P(1, 4)
13	Message	m <sub>11</sub>	D(6, 5)	D(6, 6)	D(6, 7)	D(7, 0)	D(7, 1)
14	Message	m <sub>12</sub>	D(7, 2)	D(7, 3)	D(7, 4)	D(7, 5)	D(7, 6)
15	Message	m <sub>13</sub>	D(7, 7)	D(8, 0)	D(8, 1)	D(8, 2)	D(8, 3)
16	Message	m <sub>14</sub>	D(8, 4)	D(8, 5)	D(8, 6)	D(8, 7)	D(9, 0)
17	Message	m <sub>15</sub>	D(9, 1)	D(9, 2)	D(9, 3)	D(9, 4)	D(9, 5)
18	Parity	p <sub>2</sub>	P(2, 0)	P(2, 1)	P(2, 2)	P(2, 3)	P(2, 4)
19	Message	m <sub>16</sub>	D(9, 6)	D(9, 7)	D(10, 0)	D(10, 1)	D(10, 2)
20	Message	m <sub>17</sub>	D(10, 3)	D(10, 4)	D(10, 5)	D(10, 6)	D(10, 7)
21	Message	m <sub>18</sub>	D(11, 0)	D(11, 1)	D(11, 2)	D(11, 3)	D(11, 4)
22	Message	m <sub>19</sub>	D(11, 5)	D(11, 6)	D(11, 7)	D(12, 0)	D(12, 1)
23	Message	m <sub>20</sub>	D(12, 2)	D(12, 3)	D(12, 4)	D(12, 5)	D(12, 6)
24	Parity	p <sub>3</sub>	P(3, 0)	P(3, 1)	P(3, 2)	P(3, 3)	P(3, 4)
25	Message	m <sub>21</sub>	D(12, 7)	D(13, 0)	D(13, 1)	D(13, 2)	D(13, 3)
26	Message	m <sub>22</sub>	D(13, 4)	D(13, 5)	D(13, 6)	D(13, 7)	D(14, 0)
27	Message	m <sub>23</sub>	D(14, 1)	D(14, 2)	D(14, 3)	D(14, 4)	D(14, 5)
28	Message	m <sub>24</sub>	D(14, 6)	D(14, 7)	D(15, 0)	D(15, 1)	D(15, 2)
29	Message	m <sub>25</sub>	D(15, 3)	D(15, 4)	D(15, 5)	D(15, 6)	D(15, 7)
Key: H0 = bit 0 of the SPL PACKET HEADER field H1 = bit 1 of the SPL PACKET HEADER field D(x, y) = SPL packet payload byte x, bit y P(x, y) = parity symbol x, bit y							
<sup>a</sup> Symbols are transmitted on a physical link in order from 0 to 29. <sup>b</sup> Bits within symbols are transmitted on a physical link from LSB (i.e., bit 0) to MSB (i.e., bit 4).							



### 5.5.7.3 Forward error correction decoding

The codeword  $T(x)$  is transmitted and certain errors  $E(x)$  may corrupt the original codeword to produce the codeword with errors  $R(x)$  at the receiver, as described in table 68.

The receiver's Reed Solomon code decoding function makes an estimate of the error locations  $Er(x)$  and if two or fewer symbols have errors, then  $Er(x) = E(x)$ . Non-zero coefficients of  $Er(x)$  define the error locations and the error values, both of which are needed to decode the original transmitted message. If a decoding failure is identified (e.g., the decoding function detects that three symbols are in error), then correction is not attempted.

Once  $Er(x)$  is determined, it is subsequently subtracted from the received codeword to yield the original codeword, as described in table 68. The original message  $M(x)$  is obtained by removing the parity check symbols  $P(x)$  and then dividing out the original shift from encoding.

If the decoder detects no errors, then the decoder shall send a Decode Success message to the phy's receiver that received the SPL packet.

After the decoder computes the estimate of the error locations:

- a) if a decode failure is identified, then the decoder shall:
  - A) not perform error correction (i.e., set  $Er(x) = 0$ ); and
  - B) send a Decode Failure message to the phy's receiver that received the SPL packet;or
- b) if a decode failure is not identified, then the decoder shall:
  - 1) determine the error locations;
  - 2) determine the error values;
  - 3) evaluate error locations and error values;
  - 4) perform error correction; and
  - 5) send a Decode Success message to the phy's receiver that received the SPL packet.

Multiple algorithms are suitable for identifying the error locations and error values of a Reed Solomon code. The selection of an algorithm is outside the scope of this standard.

The  $R(x)$  codeword is received from the physical link in the order given in figure 62 and then subsequently decoded to produce the error corrected SPL packet.

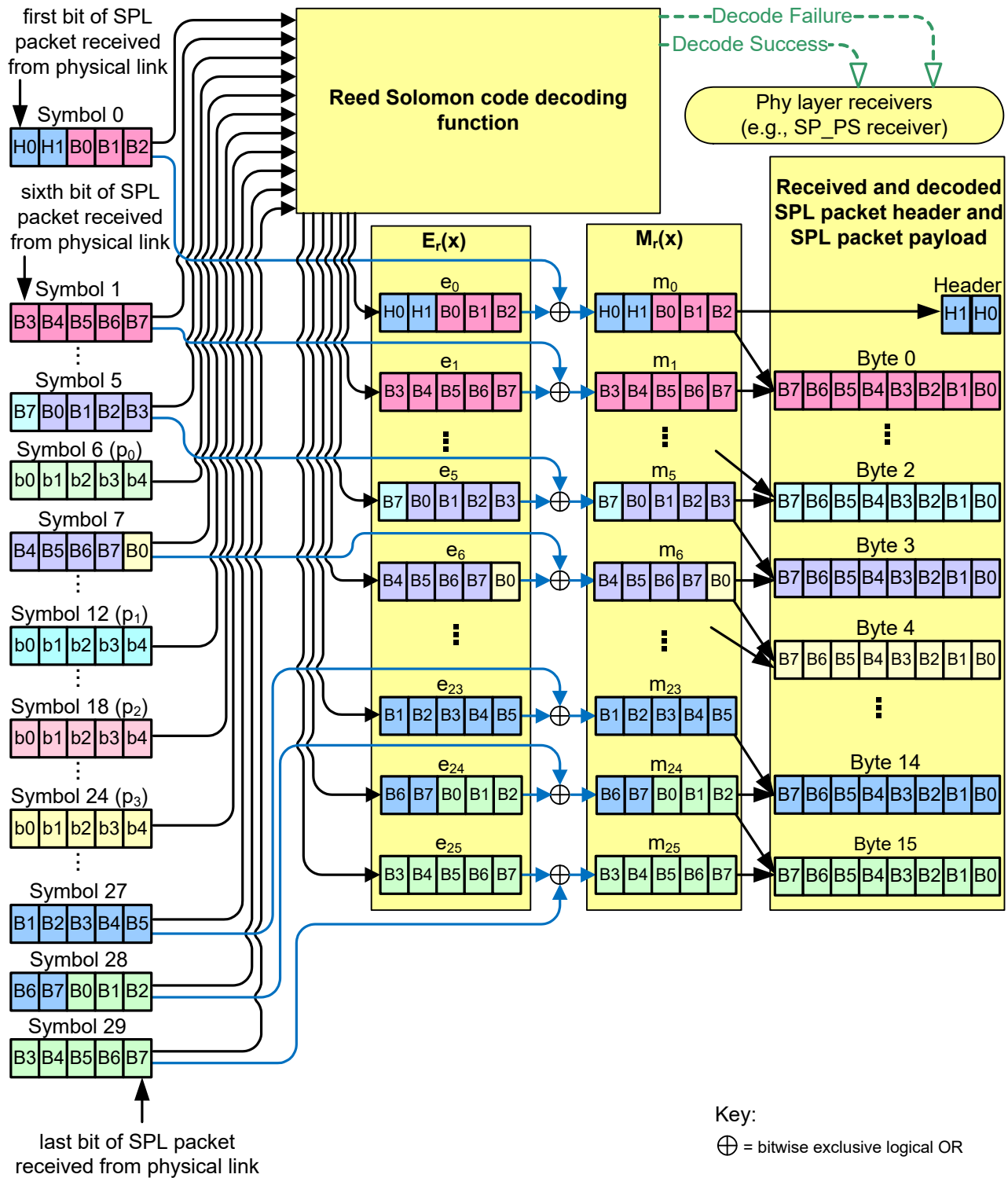


Figure 62 – Forward error correction reception and decoding

## 5.6 Dwords, primitives, binary primitives, extended binary primitives, data dwords, SPL frame segments, and invalid dwords

If the phy is in the SAS dword mode, then:

- a) all characters transferred in SAS are grouped into four-character sequences called dwords;
- b) a data dword is a dword that contains four data characters with correct disparity;
- c) a dword containing an invalid character shall be considered an invalid dword;
- d) a primitive is a dword whose first character is K28.3 or K28.5 and whose remaining three characters are data characters with correct disparity; and
- e) primitives are defined with both negative and positive starting RD (see 5.3.5). SAS defines primitives starting with K28.5 (see 6.2.6 and 6.2.7). SATA defines primitives starting with K28.3 and K28.5, which are used in SAS during STP connections (see 6.2.8).

If the phy is in the SAS packet mode, then:

- a) all bytes transferred in SAS are grouped into four-byte sequences called dwords;
- b) an SPL frame segment contains four scrambled dwords;
- c) an unscrambled primitive segment (see 5.5.4) contains:
  - A) primitives (see 6.2) and primitive parameter (see 5.5.4), if any;
  - B) binary primitives (see 6.3) and primitive parameter (see 5.5.4), if any; or
  - C) an extended binary primitive (see 6.4);and
- d) forward error correction (see 5.5.7) consists of 20 unscrambled bits.

## 5.7 Out of band (OOB) signals

### 5.7.1 OOB signals overview

For the timing and timing characteristics of transmitted and received OOB signals see SAS-4.

### 5.7.2 Transmission of OOB signals

Figure 63 describes OOB signal transmission by the SP transmitter (see 5.14.2 and 5.18.2). The COMWAKE Transmitted, COMINIT Transmitted, and COMSAS Transmitted messages are sent to the SP state machine (see 5.14).

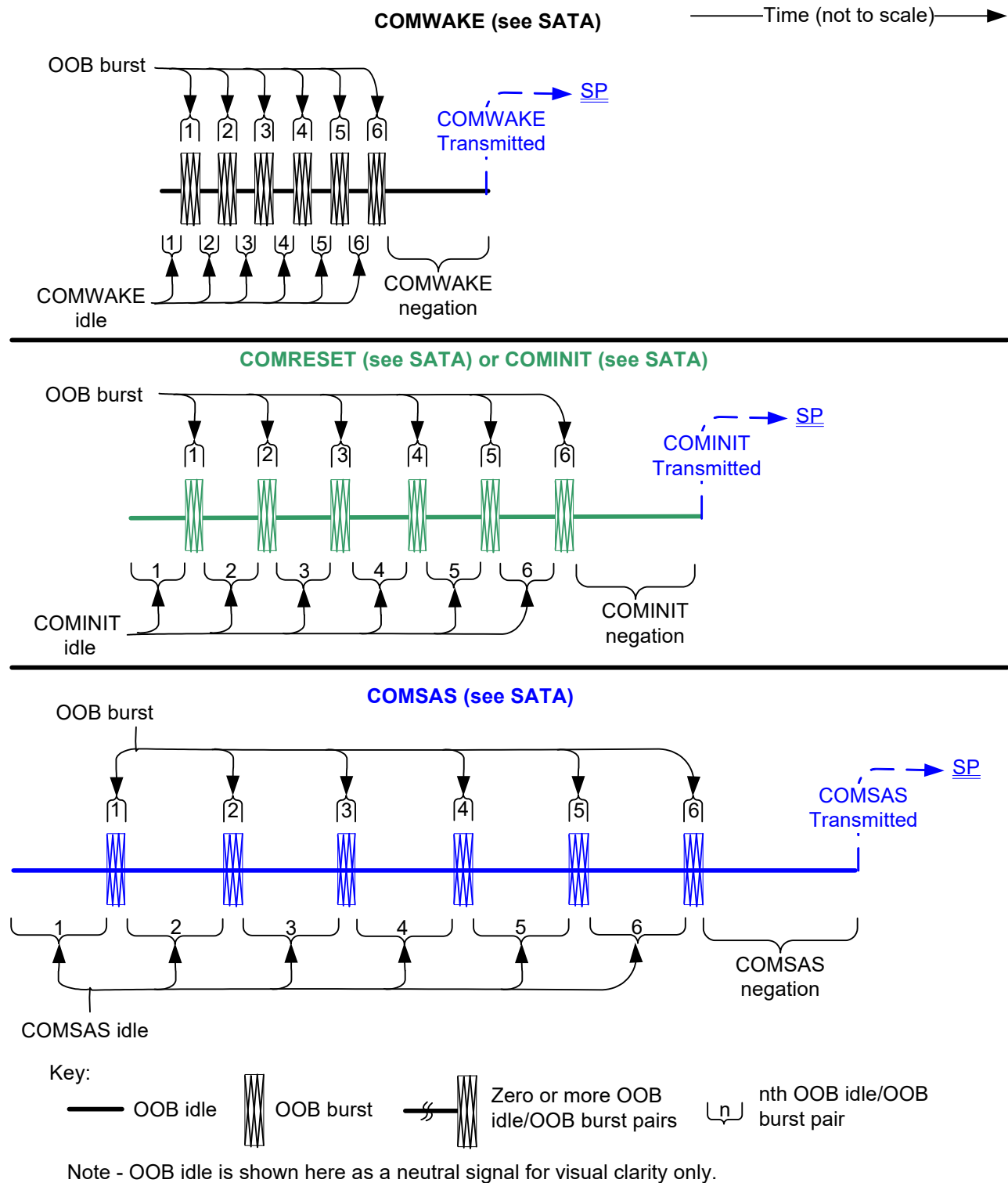


Figure 63 – OOB signal transmission

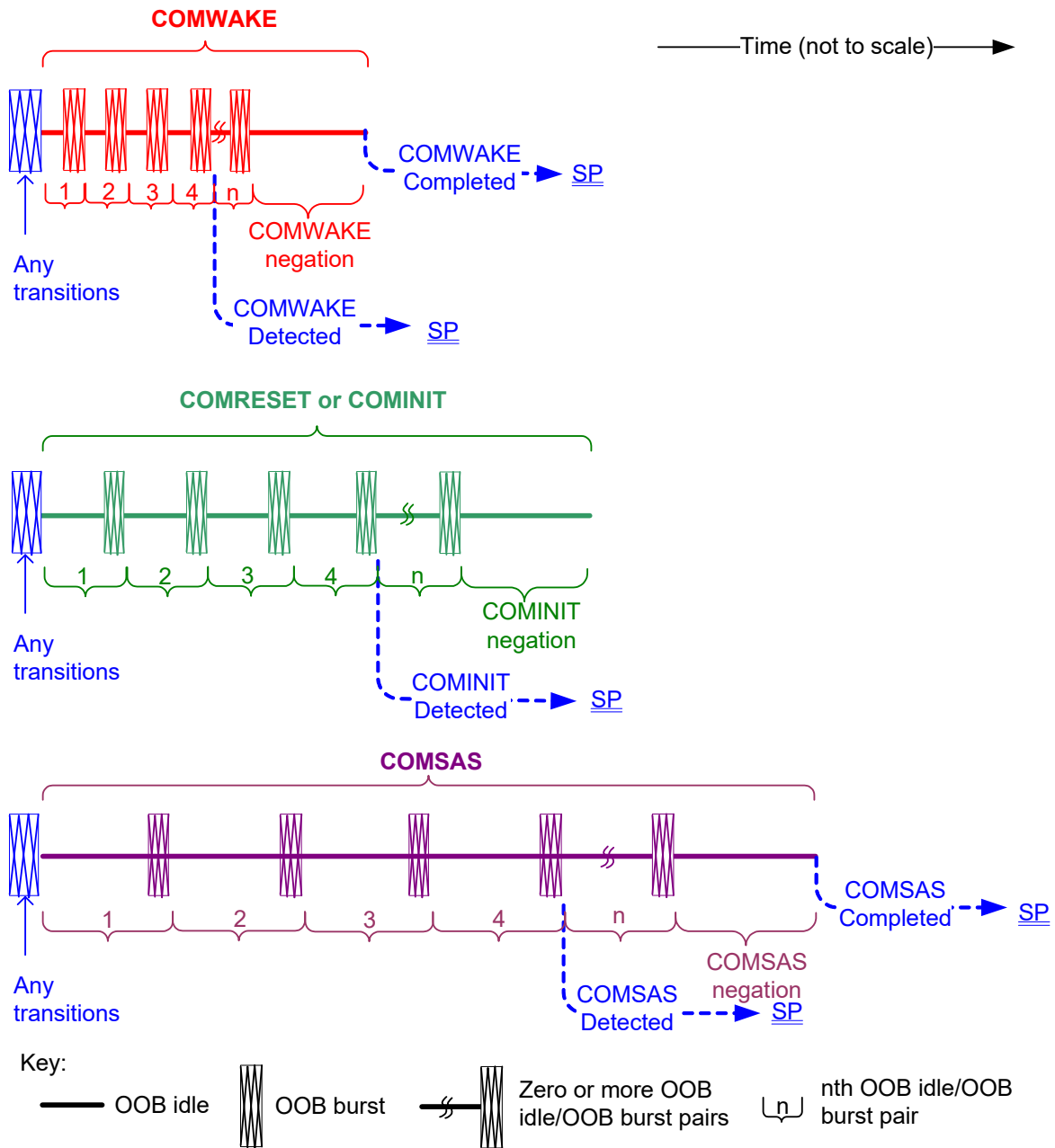
### 5.7.3 Receiver detection of OOB signals

The detection of OOB signals is disabled and enabled by the SP state machine (see 5.14.4.13).

If detection of OOB signals is disabled, then the SP receiver (see 5.14.2 and 5.18.3) shall ignore all OOB signals.

If detection of OOB signals is enabled and either optical mode is enabled or D.C. mode is enabled, then a receiver device shall detect an OOB signal after receiving four consecutive idle time/burst time pairs (see figure 64) while the SP\_DWS state machine (see 5.15) has not achieved dword synchronization (see 5.14.4.10 and 5.14.6.8). If detection of OOB signals is enabled and optical mode is enabled, then a receiver device shall detect an OOB signal after receiving four consecutive idle time/burst time pairs while the SP\_DWS state machine has achieved dword synchronization. It is not an error to receive more than four idle time/burst time pairs. A receiver device shall not detect the same OOB signal again until it has detected the corresponding negation time (e.g., a COMINIT negation time for a COMINIT (see SATA)) or has detected a different OOB signal (e.g., if a receiver device that previously detected COMINIT receives four COMWAKE idle time/burst time pairs, then that receiver device detects COMWAKE (see SATA) and may then detect COMINIT at a later time).

Figure 64 describes OOB signal detection by the SP receiver (see 5.14.2 and 5.14.3). The COMWAKE Detected, COMWAKE Completed, COMINIT Detected, COMSAS Detected, and COMSAS Completed messages are sent to the SP state machine (see 5.14) to indicate that an OOB signal has been partially or fully detected.



Note - OOB idle is shown here as a neutral signal for visual clarity only.

**Figure 64 – OOB signal detection**

Expander devices shall not forward OOB signals. An expander device shall run the link reset sequence independently on each physical phy.

### 5.7.4 SATA port selection signal

For the timing characteristics of transmitted SATA port selection signal see SAS-4.

See 5.14.7 and 9.4.3.28 for information on usage of the SATA port selection signal.

### 5.7.5 Phy power conditions

During a low phy power condition (see 4.10.1) the phy shall transmit D.C. idle (see SAS-4). A phy that is in a low phy power condition shall meet the phy wakeup timeout requirements shown in table 85.

## 5.8 Phy capabilities bits

Table 70 defines the SNW-3 (see 5.11.4.2.3.3) phy capabilities. For each bit defined as reserved, the phy shall transmit a zero (i.e., OOB idle) and shall ignore the received value. Byte 0 shall be transmitted first and byte 3 shall be transmitted last. Within each byte, bit 7 shall be transmitted first and bit 0 shall be transmitted last (e.g., overall, the START bit is transmitted first and the PARITY bit is transmitted last).

**Table 70 – SNW-3 phy capabilities**

Byte\Bit	7	6	5	4	3	2	1	0
0	START (1b)	TX SSC TYPE	Reserved	Reserved	REQUESTED LOGICAL LINK RATE			
1	Supported settings							
	G1 WITHOUT SSC	G1 WITH SSC	G2 WITHOUT SSC	G2 WITH SSC	G3 WITHOUT SSC	G3 WITH SSC	G4 WITHOUT SSC	G4 WITH SSC
2	Supported settings							
	G5 WITHOUT SSC	G5 WITH SSC	Reserved					
3	Reserved							PARITY

The START bit shall be set as shown in table 70 for the SNW-3 phy capabilities.

A TX SSC TYPE bit set to one indicates that the phy's transmitter uses center-spreading SSC while SSC is enabled (e.g., the phy is an expander phy) (see SAS-4). A TX SSC TYPE bit set to zero indicates that the phy's transmitter uses down-spreading SSC while SSC is enabled (e.g., the phy is a SAS phy) or that the phy does not support SSC.

NOTE 11 - The phy's receiver uses the TX SSC TYPE bit to optimize its clock data recovery circuitry (see SAS-4).

The REQUESTED LOGICAL LINK RATE field indicates if the phy supports multiplexing (see 5.20) and, if so, then the logical link rate that the phy is requesting.

Table 71 defines the requested logical link rate based on the transmitted and received REQUESTED LOGICAL LINK RATE fields.

**Table 71 – Requested logical link rate**

<b>Transmitted REQUESTED LOGICAL LINK RATE field</b>	<b>Received REQUESTED LOGICAL LINK RATE field</b>	<b>Requested logical link rate</b>
0h (i.e., no multiplexing)	Any	Negotiated physical link rate
8h (i.e., 1.5 Gbit/s)	8h (i.e., 1.5 Gbit/s)	1.5 Gbit/s
	9h (i.e., 3 Gbit/s)	
	Ah (i.e., 6 Gbit/s)	
	Bh (i.e., 12 Gbit/s)	
	Ch (i.e., 22.5 Gbit/s)	
	Dh to Fh (i.e., future rates)	
9h (i.e., 3 Gbit/s)	8h (i.e., 1.5 Gbit/s)	1.5 Gbit/s
	9h (i.e., 3 Gbit/s)	3 Gbit/s
	Ah (i.e., 6 Gbit/s)	
	Bh (i.e., 12 Gbit/s)	
	Ch (i.e., 22.5 Gbit/s)	
	Dh to Fh (i.e., future rates)	
Ah (i.e., 6 Gbit/s)	8h (i.e., 1.5 Gbit/s)	1.5 Gbit/s
	9h (i.e., 3 Gbit/s)	3 Gbit/s
	Ah (i.e., 6 Gbit/s)	6 Gbit/s
	Bh (i.e., 12 Gbit/s)	
	Ch (i.e., 22.5 Gbit/s)	
	Dh to Fh (i.e., future rates)	
Bh (i.e., 12 Gbit/s)	8h (i.e., 1.5 Gbit/s)	1.5 Gbit/s
	9h (i.e., 3 Gbit/s)	3 Gbit/s
	Ah (i.e., 6 Gbit/s)	6 Gbit/s
	Bh (i.e., 12 Gbit/s)	12 Gbit/s
	Ch (i.e., 22.5 Gbit/s)	
	Dh to Fh (i.e., future rates)	
Ch (i.e., 22.5 Gbit/s)	8h (i.e., 1.5 Gbit/s)	1.5 Gbit/s
	9h (i.e., 3 Gbit/s)	3 Gbit/s
	Ah (i.e., 6 Gbit/s)	6 Gbit/s
	Bh (i.e., 12 Gbit/s)	12 Gbit/s
	Ch (i.e., 22.5 Gbit/s)	22.5 Gbit/s
	Dh to Fh (i.e., future rates)	



Table 72 defines:

- a) whether or not multiplexing is enabled;
- b) the operating mode of the physical link (i.e., SAS dword or SAS packet mode); and
- c) the negotiated logical link rate based on the requested logical link rate (see table 71) and the negotiated physical link rate (see 5.11.4.2.4).

**Table 72 – Multiplexing negotiation**

Requested logical link rate (see table 71)	Negotiated physical link rate	Multiplexing	Mode	Negotiated logical link rate
1.5 Gbit/s	1.5 Gbit/s	Disabled	SAS dword	1.5 Gbit/s
	3 Gbit/s	Enabled	SAS dword	1.5 Gbit/s
	6 Gbit/s			3 Gbit/s
	12 Gbit/s			6 Gbit/s
3 Gbit/s	1.5 Gbit/s	Disabled	SAS dword	1.5 Gbit/s
	3 Gbit/s			3 Gbit/s
	6 Gbit/s	Enabled	SAS dword	3 Gbit/s
	12 Gbit/s			6 Gbit/s
6 Gbit/s	1.5 Gbit/s	Disabled	SAS dword	1.5 Gbit/s
	3 Gbit/s			3 Gbit/s
	6 Gbit/s			6 Gbit/s
	12 Gbit/s	Enabled	SAS dword	6 Gbit/s
12 Gbit/s	1.5 Gbit/s	Disabled	SAS dword	1.5 Gbit/s
	3 Gbit/s			3 Gbit/s
	6 Gbit/s			6 Gbit/s
	12 Gbit/s			12 Gbit/s
22.5 Gbit/s	1.5 Gbit/s	Disabled	SAS dword	1.5 Gbit/s
	3 Gbit/s			3 Gbit/s
	6 Gbit/s			6 Gbit/s
	12 Gbit/s			12 Gbit/s
	22.5 Gbit/s		SAS packet	22.5 Gbit/s
Negotiated physical link rate	1.5 Gbit/s	Disabled	SAS dword	1.5 Gbit/s
	3 Gbit/s			3 Gbit/s
	6 Gbit/s			6 Gbit/s
	12 Gbit/s			12 Gbit/s
	22.5 Gbit/s		SAS packet	22.5 Gbit/s

The supported settings bits include the G1 WITHOUT SSC bit, the G1 WITH SSC bit, the G2 WITHOUT SSC bit, the G2 WITH SSC bit, the G3 WITHOUT SSC bit, the G3 WITH SSC bit, the G4 WITHOUT SSC bit, the G4 WITH SSC bit the G5 WITHOUT SSC bit, and the G5 WITH SSC bit.

A G1 WITHOUT SSC bit set to one indicates that the phy supports G1 (i.e., 1.5 Gbit/s) without SSC and that SAS dword mode is enabled. A G1 WITHOUT SSC bit set to zero indicates that the phy does not support G1 without SSC.

A G1 WITH SSC bit set to one indicates that the phy supports G1 (i.e., 1.5 Gbit/s) with SSC and that SAS dword mode is enabled. A G1 WITH SSC bit set to zero indicates that the phy does not support G1 with SSC.

A G2 WITHOUT SSC bit set to one indicates that the phy supports G2 (i.e., 3 Gbit/s) without SSC and that SAS dword mode is enabled. A G2 WITHOUT SSC bit set to zero indicates that the phy does not support G2 without SSC.

A G2 WITH SSC bit set to one indicates that the phy supports G2 (i.e., 3 Gbit/s) with SSC and that SAS dword mode is enabled. A G2 WITH SSC bit set to zero indicates that the phy does not support G2 with SSC.

A G3 WITHOUT SSC bit set to one indicates that the phy supports G3 (i.e., 6 Gbit/s) without SSC and that SAS dword mode is enabled. A G3 WITHOUT SSC bit set to zero indicates that the phy does not support G3 without SSC.

A G3 WITH SSC bit set to one indicates that the phy supports G3 (i.e., 6 Gbit/s) with SSC and that SAS dword mode is enabled. A G3 WITH SSC bit set to zero indicates that the phy does not support G3 with SSC.

A G4 WITHOUT SSC bit set to one indicates that the phy supports G4 (i.e., 12 Gbit/s) without SSC, that transmitter training is enabled, and that SAS dword mode is enabled. A G4 WITHOUT SSC bit set to zero indicates that the phy does not support G4 without SSC.

A G4 WITH SSC bit set to one indicates that the phy supports G4 (i.e., 12 Gbit/s) with SSC, that transmitter training is enabled, and that SAS dword mode is enabled. A G4 WITH SSC bit set to zero indicates that the phy does not support G4 with SSC.

A G5 WITHOUT SSC bit set to one indicates that the phy supports G5 (i.e., 22.5 Gbit/s) without SSC, that transmitter training is enabled, and that SAS packet mode is enabled. A G5 WITHOUT SSC bit set to zero indicates that the phy does not support G5 without SSC.

A G5 WITH SSC bit set to one indicates that the phy supports G5 (i.e., 22.5 Gbit/s) with SSC, that transmitter training is enabled, and that SAS packet mode is enabled. A G5 WITH SSC bit set to zero indicates that the phy does not support G5 with SSC.

Table 73 defines the priority of the supported settings bits.

**Table 73 – Supported settings bit priorities**

Priority	Bit
Highest	G5 WITH SSC bit <sup>a</sup>
...	G5 WITHOUT SSC bit <sup>a</sup>
...	G4 WITH SSC bit <sup>a</sup>
...	G4 WITHOUT SSC bit <sup>a</sup>
...	G3 WITH SSC bit
...	G3 WITHOUT SSC bit
...	G2 WITH SSC bit
...	G2 WITHOUT SSC bit
...	G1 WITH SSC bit
Lowest	G1 WITHOUT SSC bit
<sup>a</sup> If optical mode is enabled or there is an active cable assembly attached to the phy, then transmitter training is disabled (see 5.14.4.12.1), APTA is disabled (see 5.12), and the phy's transmitter coefficients shall be set to a default value. The determination of the default value is outside the scope of this standard.	

The PARITY bit provides for error detection for the SNW-3 phy capabilities bits. The PARITY bit shall be set to one or zero such that the total number of SNW-3 phy capabilities bits that are set to one is even, including the START bit and the PARITY bit. If the PARITY bit received is incorrect based upon the received SNW phy capabilities bits, then the parity is bad and the phy shall consider a phy reset problem (see 5.11.4.2.4) to have occurred.

Table 74 lists some example SNW-3 phy capabilities values.

**Table 74 – Example SNW-3 phy capabilities values**

Code <sup>a</sup>	Description
80540000h	Down-spreading SSC G1, G2, and G3 with SSC supported
80550001h	Down-spreading SSC G1, G2, G3, and G4 with SCC supported
80FC0001h	Down-spreading SSC G1, G2, and G3 with and without SSC supported
80FF0001h	Down-spreading SSC G1, G2, G3, and G4 with and without SSC supported
803FC001h	Down-spreading SSC G2, G3, G4, and G5 with and without SSC supported
80A80000h	G1, G2, and G3 without SSC supported
80AA0001h	G1, G2, G3, and G4 without SSC supported
802A8001h	G2, G3, G4, and G5 without SSC supported
C0FC0000h	Center-spreading SSC G1, G2, and G3 with and without SSC supported
C0FF0000h	Center-spreading SSC G1, G2, G3, and G4 with and without SSC supported
C03FC000h	Center-spreading SSC G2, G3, G4, and G5 with and without SSC supported
C9FC0000h	Center-spreading SSC Requested 3 Gbit/s logical link rate G1, G2, and G3 with and without SSC supported
C8F00001h	Center-spreading SSC Requested 1.5 Gbit/s logical link rate G1 and G2 with and without SSC supported
<sup>a</sup> Expressed as a 32-bit value with byte 0 bit 7 (i.e., the START bit) as the MSB and byte 3 bit 0 (i.e., the PARITY bit) as the LSB.	

## 5.9 BMC coding

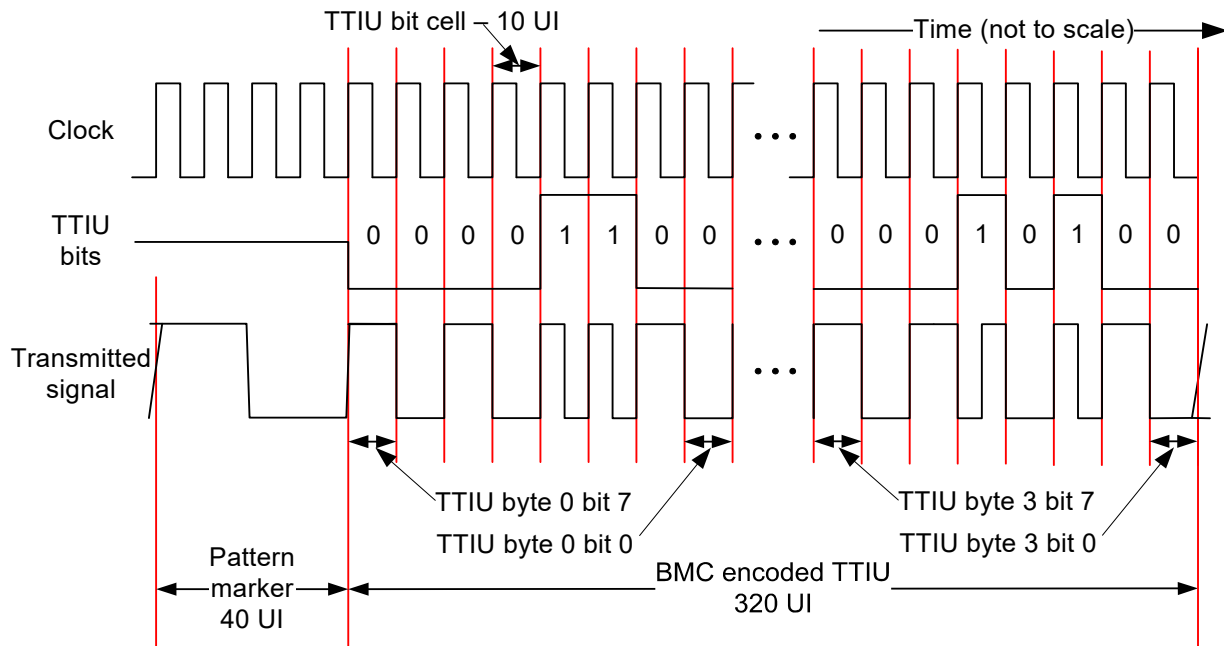
### 5.9.1 BMC coding overview

BMC is a coding method in which TTIU bits (see 5.11.4.2.3.5) and a clock signal are combined to form a single self-synchronizing signal. The self-synchronizing signal is a differential coding that indicates a:

- a) one with a transition at the midpoint of the TTIU bit cell; or
- b) zero with no transition within the TTIU bit cell.

BMC coding ensures that at least one transition occurs for each transmitted TTIU bit allowing the receiver to perform clock recovery.

All TTIU bits are encoded into BMC and transmitted serially bit-by-bit across the physical link as shown in figure 65.



**Figure 65 – TTIU transmitter BMC encoding**

### 5.9.2 TTIU bit cell encoding in the transmitter

A TTIU bit cell shall be 10 UIs.

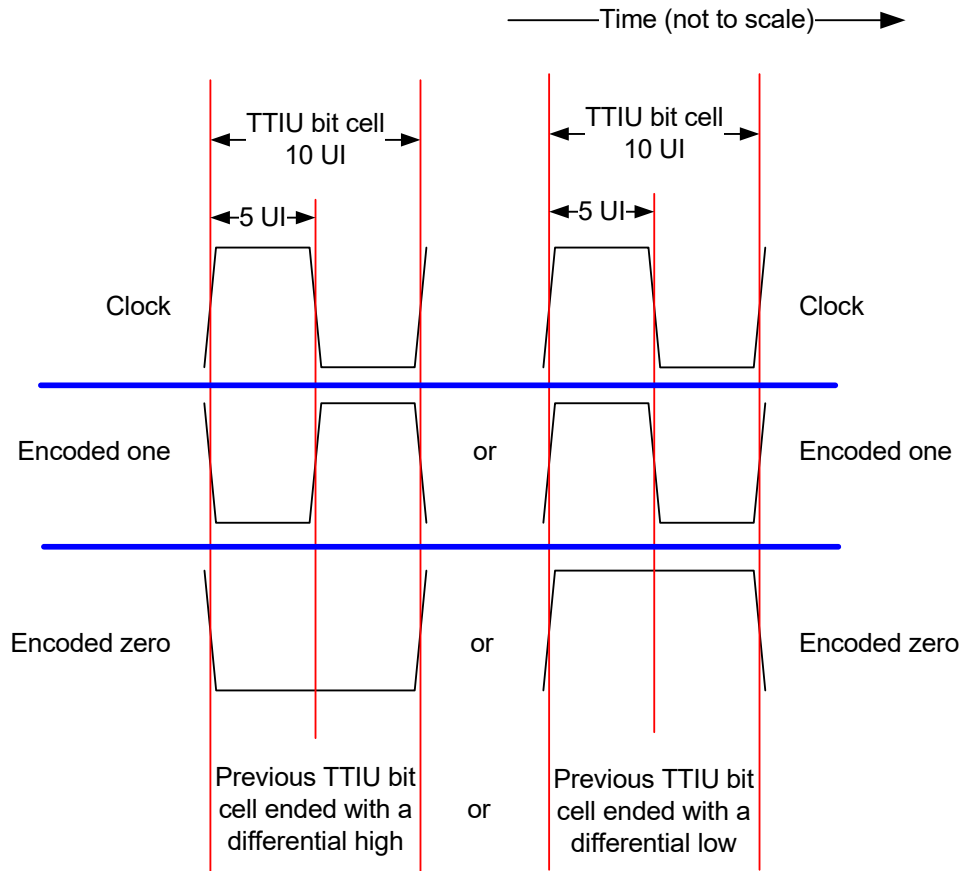
A TTIU bit cell may be formed from a clock (see figure 66) with a period of 10 UIs (e.g.,  $833.\bar{3}$  ps at 12 Gbit/s and  $444.\bar{4}$  ps at 22.5 Gbit/s) with a duty cycle of:

- five UIs differential high followed by five UIs differential low; or
- five UIs differential low followed by five UIs differential high.

The transmitter shall cause a transition (i.e., low to high or high to low) to begin a TTIU bit cell as shown in figure 66.

The transmitter shall encode a one in a TTIU bit cell as a transition five UIs from the beginning of that TTIU bit cell (e.g., a low to high transition of a clock signal) as shown in figure 66.

The transmitter shall encode a zero in a TTIU bit cell by making no transition within a TTIU bit cell as shown in figure 66.



**Figure 66 – TTIU bit cell transmitter encoding**

### 5.9.3 TTIU bit transmission order

A TTIU shall be encoded as 32 consecutive TTIU bit cells as follows:

- 1) the first TTIU bit cell transmitted contains byte 0 bit 7 of the TTIU;
- 2) the eighth TTIU bit cell transmitted contains byte 0 bit 0 of the TTIU;
- 3) the ninth TTIU bit cell transmitted contains byte 1 bit 7 of the TTIU;
- 4) the 16<sup>th</sup> TTIU bit cell transmitted contains byte 1 bit 0 of the TTIU;
- 5) the 17<sup>th</sup> TTIU bit cell transmitted contains byte 2 bit 7 of the TTIU;
- 6) the 24<sup>th</sup> TTIU bit cell transmitted contains byte 2 bit 0 of the TTIU;
- 7) the 25<sup>th</sup> TTIU bit cell transmitted contains byte 3 bit 7 of the TTIU; and
- 8) the 32<sup>nd</sup> TTIU bit cell transmitted contains byte 3 bit 0 of the TTIU.

### 5.9.4 TTIU bit cell decoding in the receiver

After receiving a TTIU bit cell if the receiver detects a transition:

- a) after or at three UIs from the beginning of the TTIU cell; and
- b) before or at seven UIs from the beginning of the TTIU cell,

then the received bit shall be considered a one as shown in figure 67.

After receiving a TTIU bit cell if the receiver detects no transition:

- a) after or at three UIs from the beginning of the TTIU cell; and

b) before or at seven UIs from the beginning of the TTIU cell, then the received bit shall be considered a zero as shown in figure 67.

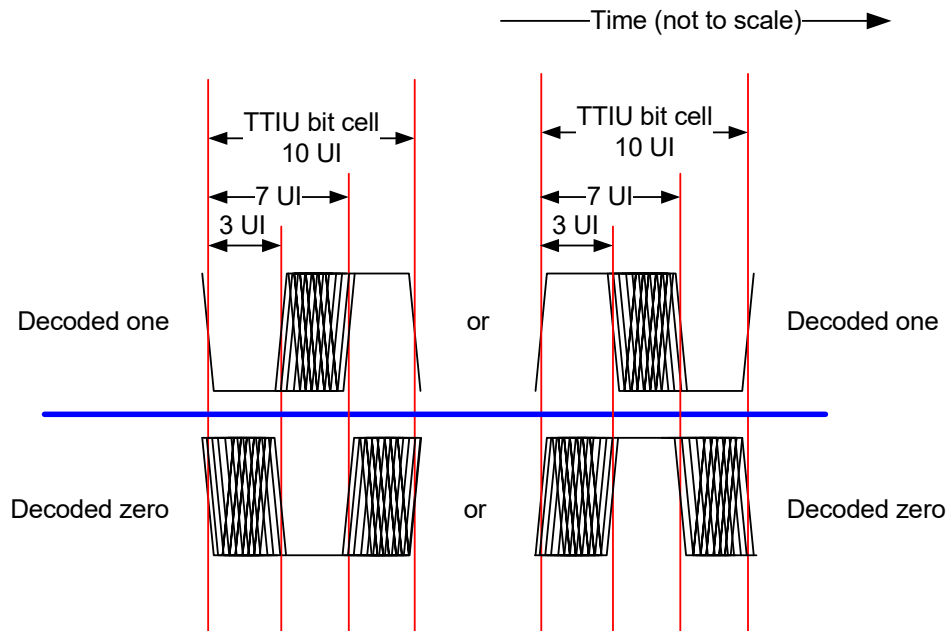


Figure 67 – TTIU bit cell receiver decoding

## 5.10 Train\_Tx-SNW TTIUs

### 5.10.1 Train\_Tx-SNW TTIU format

Table 75 defines the content of each Train\_Tx-SNW TTIU bit.

Table 75 – Train\_Tx-SNW TTIU bit

Value	Transmitted
One	BMC encoded one (see 5.9.2)
Zero	BMC encoded zero (see 5.9.2)

Table 76 defines the Train\_Tx-SNW TTIU. For each Train\_Tx-SNW TTIU bit defined as reserved, the phy shall transmit a zero. Byte 0 shall be transmitted first and byte 3 shall be transmitted last. Within each byte, bit 7 shall be transmitted first and bit 0 shall be transmitted last.

**Table 76 – Train\_Tx-SNW TTIU**

Byte\Bit	7	6	5	4	3	2	1	0
0	PATTERN TYPE							
1								
2				PATTERN TYPE SPECIFIC				
3								

Table 77 defines the PATTERN TYPE field, which defines the format of the PATTERN TYPE SPECIFIC field.

**Table 77 – PATTERN TYPE field**

Code	Name of TTIU	Type	Reference
000b	Control/Status TTIU	M	5.10.2
001b to 110b	Reserved <sup>a</sup>		
111b	Error Response TTIU	M	5.10.3
Key:			
M = TTIU implementation is mandatory.			
<sup>a</sup> If a phy receives a pattern type that is reserved, then that phy shall transmit an Error Response TTIU with the ERROR CODE field set to RESERVED PATTERN TYPE (see 5.18.5.4).			

The PATTERN TYPE SPECIFIC field contains the transmitter training information, the form of which is defined by the PATTERN TYPE field (see table 77).

The total number of bits within a TTIU that are set to zero shall be even.

### 5.10.2 Control/Status TTIU

When the local phy receives a Control/Status TTIU:

- the Training Control word specifies adjustments to the local phy's transmitter coefficients; and
- the Training Status word indicates the status of the attached phy's transmitter.

When the local phy sends a Control/Status TTIU:

- the Training Control word specifies adjustments to the attached phy's transmitter coefficients; and
- the Training Status word indicates the status of the local phy's transmitter.



**Table 78 – Control/Status TTIU**

Byte\Bit	7	6	5	4	3	2	1	0
0	Training Control word							
	PATTERN TYPE (000b)			Reserved	COEFFICIENT SETTINGS		Reserved	
1	Coefficient Request byte							
	Reserved		COEFFICIENT 3 REQUEST		COEFFICIENT 2 REQUEST		COEFFICIENT 1 REQUEST	
2	Training Status word							
	TRAIN COMP	TX INIT	BALANCE	Reserved				
3	Reserved		COEFFICIENT 3 STATUS		COEFFICIENT 2 STATUS		COEFFICIENT 1 STATUS	

If a TTIU bit that is defined as reserved in the Training Control word is set to one or the Training Status word is set to one, then the phy shall transmit an Error Response TTIU (see 5.18.5.4) with the ERROR CODE field set to RESERVED TTIU BIT SET TO ONE.

The PATTERN TYPE field shall be set to the value shown in table 77.

Table 79 defines the COEFFICIENT SETTINGS field.

**Table 79 – COEFFICIENT SETTINGS field**

Code	Name	Description	Type
00b	normal	If a coefficient change is specified (see table 80) and the TRAIN COMP bit is set to zero, then the local phy's transmitter coefficients shall be changed as specified in the: a) COEFFICIENT 1 REQUEST field; b) COEFFICIENT 2 REQUEST field; and c) COEFFICIENT 3 REQUEST field.	M
01b	reference_1	The local phy's transmitter coefficients shall be set to the reference 1 values specified in SAS-4.	M
10b	reference_2	The local phy's transmitter coefficients shall be set to the reference 2 values specified in SAS-4.	M
11b	no_equalization	The local phy's transmitter coefficients shall be set to the no equalization values specified in SAS-4.	M
Key: M = Coefficient settings implementation is mandatory.			

If the COEFFICIENT SETTINGS field is set to 00b and the TRAIN COMP bit is set to zero, then the COEFFICIENT 3 REQUEST field specifies the adjustment, if any, (see table 80) that the local phy's transmitter shall make to its transmitter coefficient 3 (see SAS-4). If the COEFFICIENT SETTINGS field is set to a value other than 00b or the TRAIN COMP bit is set to one, then the COEFFICIENT 3 REQUEST field shall be ignored by the local phy's transmitter.

If the COEFFICIENT SETTINGS field is set to 00b and the TRAIN COMP bit is set to zero, then the COEFFICIENT 2 REQUEST field specifies the adjustment, if any, (see table 80) that the local phy's transmitter shall make to its transmitter coefficient 2 (see SAS-4). If the COEFFICIENT SETTINGS field is set to a value other than 00b or the TRAIN COMP bit is set to one, then the COEFFICIENT 2 REQUEST field shall be ignored by the local phy's transmitter.

If the COEFFICIENT SETTINGS field is set to 00b and the TRAIN COMP bit is set to zero, then the COEFFICIENT 1 REQUEST field specifies the adjustment, if any, (see table 80) that the local phy's transmitter shall make to its transmitter coefficient 1 (see SAS-4). If the COEFFICIENT SETTINGS field is set to a value other than 00b or the TRAIN COMP bit is set to one, then the COEFFICIENT 1 REQUEST field shall be ignored by the local phy's transmitter.

**Table 80 – COEFFICIENT 1 REQUEST field, COEFFICIENT 2 REQUEST field, and COEFFICIENT 3 REQUEST field**

Code	Name	Description	Type
00b	hold	The local phy shall make no adjustment to the specified coefficient.	M
01b	increment	The local phy shall adjust the specified coefficient by one increment. <sup>a</sup>	M
10b	decrement	The local phy shall adjust the specified coefficient by one decrement. <sup>a</sup>	M
11b	Reserved <sup>b</sup>		
Key: M = Coefficient request implementation is mandatory.			
<sup>a</sup> See SAS-4 for the amount of adjustment represented by one increment and one decrement to the coefficient. <sup>b</sup> If a phy receives one coefficient request that is reserved, then that phy shall transmit an Error Response TTIU (see 5.18.5.4) that indicates the coefficient request that was reserved with the ERROR CODE field set to: a) RESERVED COEFFICIENT 1 REQUEST; b) RESERVED COEFFICIENT 2 REQUEST; or c) RESERVED COEFFICIENT 3 REQUEST. If a phy receives more than one coefficient request that is reserved, then that phy shall transmit an Error Response TTIU (see 5.18.5.4) with the ERROR CODE field set to MULTIPLE RESERVED COEFFICIENTS REQUESTED.			

See table 81 for the combinations of coefficient requests that are mandatory or reserved.

**Table 81 – Valid coefficient requests**

Coefficient Request byte							
Code	COEFFICIENT 3 REQUEST field		COEFFICIENT 2 REQUEST field		COEFFICIENT 1 REQUEST field		Type
	Code	Name	Code	Name	Code	Name	
00h	00b	hold	00b	hold	00b	hold	M
01h	00b	hold	00b	hold	01b	increment	M
02h	00b	hold	00b	hold	10b	decrement	M
04h	00b	hold	01b	increment	00b	hold	M
05h	00b	hold	01b	increment	01b	increment	M
08h	00b	hold	10b	decrement	00b	hold	M
0Ah	00b	hold	10b	decrement	10b	decrement	M
10h	01b	increment	00b	hold	00b	hold	M
14h	01b	increment	01b	increment	00b	hold	M
20h	10b	decrement	00b	hold	00b	hold	M
28h	10b	decrement	10b	decrement	00b	hold	M
All others	Reserved <sup>a</sup>						
Key: M = Coefficient Request byte's coefficient request combination implementation is mandatory.							
<sup>a</sup> If a phy receives a Coefficient Request byte with a combination of coefficient requests that is reserved, then that phy shall transmit an Error Response (see table 80).							

A training complete (TRAIN COMP) bit set to one indicates the local phy's receiver has determined the attached phy's transmitter coefficients are set to their optimum value. A TRAIN COMP bit set to zero indicates the local phy's receiver may be requesting the attached phy's transmitter coefficients to be adjusted as indicated in the COEFFICIENT SETTINGS field, COEFFICIENT 3 REQUEST field, COEFFICIENT 2 REQUEST field, and COEFFICIENT 1 REQUEST field.

A transmitter initializing (TX INIT) bit set to one indicates the local phy is initializing and not ready for training. A TX INIT bit set to zero indicates the local phy is ready and may be adjusted by the attached phy's receiver.

The BALANCE bit shall be set to one or zero such that the total number of bits within the TTIU that are set to zero is even, including the BALANCE bit (see 5.18.5.4.1).

The COEFFICIENT 3 STATUS field indicates the status (see table 82) of the local phy's transmitter coefficient 3 (see SAS-4).

The COEFFICIENT 2 STATUS field indicates the status (see table 82) of the local phy's transmitter coefficient 2 (see SAS-4).

The COEFFICIENT 1 STATUS field indicates the status (see table 82) of the local phy's transmitter coefficient 1 (see SAS-4).

**Table 82 – COEFFICIENT 1 STATUS field, COEFFICIENT 2 STATUS field, and COEFFICIENT 3 STATUS field**

Code	Name	Description	Type
00b	ready	The indicated local phy's transmitter coefficient may be adjusted by the attached phy.	M
01b	update complete	The local phy's transmitter has completed the last indicated coefficient adjustment requested by the attached phy's receiver (see 5.18.2.5).	M
10b	minimum	The indicated local phy's transmitter coefficient is at a minimum value (see 5.18.2.7). <sup>a</sup>	M
11b	maximum	The indicated local phy's transmitter coefficient is at a maximum value (see 5.18.2.6). <sup>a</sup>	M
Key: M = Coefficient status value implementation is mandatory.			
<sup>a</sup> See SAS-4 for the minimum value and the maximum value.			

### 5.10.3 Error Response TTIU

The Error Response TTIU is used by a phy to report requests to process:

- a) unsupported optional features;
- b) reserved code values; and
- c) TTIU reserved bits.

**Table 83 – Error Response TTIU**

Byte\Bit	7	6	5	4	3	2	1	0
0	PATTERN TYPE (111b)			Reserved	RECEIVED COEFFICIENT SETTINGS		Reserved	
1	Reserved		RECEIVED COEFFICIENT 3 REQUEST		RECEIVED COEFFICIENT 2 REQUEST		RECEIVED COEFFICIENT 1 REQUEST	
2	Reserved		BALANCE	Reserved				
3	ERROR CODE							

The PATTERN TYPE field shall be set as shown in table 83 for the Error Response TTIU.

The RECEIVED COEFFICIENT SETTINGS field shall be set to the contents of the COEFFICIENT SETTINGS field received in the Training Control word of the Control/Status TTIU that contained any unsupported or illegal request.

The RECEIVED COEFFICIENT 3 REQUEST field shall be set to the contents of the COEFFICIENT 3 REQUEST field received in the Training Control word of the Control/Status TTIU that contained any unsupported or illegal request.

The RECEIVED COEFFICIENT 2 REQUEST field shall be set to the contents of the COEFFICIENT 2 REQUEST field received in the Training Control word of the Control/Status TTIU that contained any unsupported or illegal request.

The RECEIVED COEFFICIENT 1 REQUEST field shall be set to the contents of the COEFFICIENT 1 REQUEST field received in the Training Control word of the Control/Status TTIU that contained any unsupported or illegal request.

The BALANCE bit shall be set to one or zero such that the total number of bits within the TTIU that are set to zero is even, including the BALANCE bit.

The ERROR CODE field is defined in table 84.

**Table 84 – ERROR CODE field**

Code <sup>a</sup>	Name	Description
Error codes returned for all TTUUs		
00h	UNKNOWN	The cause of the error is indeterminate.
01h	UNSUPPORTED PATTERN TYPE	Unsupported transmitter training pattern type requested. The information returned in the Error Response TTUUs does not contain information about the pattern type that was in error (see table 77).
02h	RESERVED PATTERN TYPE	Reserved transmitter training pattern type requested. The information returned in the Error Response TTUUs does not contain information about the pattern type that was in error (see table 77).
03h	RESERVED TTUUs BIT SET TO ONE	Reserved TTUUs bit set to one (e.g., reserved bits in table 78).
04h to 0Fh	Reserved	
Error codes returned for Control/Status TTUUs		
10h	MULTIPLE RESERVED COEFFICIENTS REQUESTED	More than one coefficient request value is set to a reserved value (see table 80).
11h	RESERVED COEFFICIENT 1 REQUEST	Coefficient 1 is set to a reserved value (see table 80).
13h	RESERVED COEFFICIENT 2 REQUEST	Coefficient 2 is set to a reserved value (see table 80).
15h	RESERVED COEFFICIENT 3 REQUEST	Coefficient 3 is set to a reserved value (see table 80).
1Ah	RESERVED COEFFICIENT REQUEST COMBINATION	If the following occurs: a) none of the coefficient requests are set 11b (i.e., reserved) (see table 80); and b) the combination of the coefficient 3 request, coefficient 2 request, and coefficient 1 request is reserved (see table 81).
1Bh to 1Fh	Reserved	
Reserved error codes		
20h to FFh	Reserved	
<sup>a</sup> If more than one error condition is true, then the phy shall use the following priority determine which error condition to report in the ERROR CODE field: 1) UNSUPPORTED PATTERN TYPE or RESERVED PATTERN TYPE; 2) RESERVED TTUUs BIT SET TO ONE; 3) MULTIPLE RESERVED COEFFICIENTS REQUESTED; 4) RESERVED COEFFICIENT 1 REQUEST, RESERVED COEFFICIENT 2 REQUEST, or RESERVED COEFFICIENT 3 REQUEST; 5) RESERVED COEFFICIENT REQUEST COMBINATION; and 6) UNKNOWN.		

## 5.11 Phy reset sequences

### 5.11.1 Phy reset sequences overview

The phy reset sequence consists of:

- 1) an OOB sequence (see 5.11.2.1 and 5.11.4.1);
- 2) a speed negotiation sequence (see 5.11.2.2 and 5.11.4.2); and
- 3) if the physical link is a SAS physical link and multiplexing (see 5.20) is enabled (see table 72), then a multiplexing sequence (see 5.11.4.3).

The phy reset sequence shall only affect the phy, not the port or device containing the phy or other phys in the same port or device.

A phy shall originate a phy reset sequence after:

- a) power on;
- b) hard reset (i.e., receiving a HARD\_RESET primitive sequence before an IDENTIFY address frame) (see 4.4.2);
- c) a management application layer request (see 5.14.1);
- d) losing dword synchronization and not attempting to re-acquire dword synchronization (see 5.14.4.10 and 5.14.6.8);
- e) losing SPL packet synchronization and not being able to re-acquire SPL packet synchronization (see 5.17);
- f) the Receive Identify Timeout timer expires (see 6.12);
- g) a hot-plug timeout (see 5.11.5) occurs for an expander phy;
- h) a hot-plug timeout occurs while in a low phy power condition (see 4.10.1); or
- i) the SNTT timer expires while in a low phy power condition.

A SAS phy may originate a phy reset sequence after a hot-plug timeout (see 5.11.5).

After receiving a HARD\_RESET primitive sequence before an IDENTIFY address frame, a phy should start the phy reset sequence within 250 ms.

Table 85 defines phy reset sequence timing parameters used by the SP state machine (see 5.14).

**Table 85 – Phy reset sequence timing specifications**

Parameter	Minimum	Maximum	Comments
Hot-plug timeout	10 ms	500 ms	The time after which: <ol style="list-style-type: none"> <li>a) an expander phy shall retry an unsuccessful phy reset sequence;</li> <li>b) a SAS initiator phy should retry an unsuccessful phy reset sequence (see 5.11.5); and</li> <li>c) a phy shall initiate a phy reset sequence if that phy does not receive a COMWAKE Completed message as defined in the SP31:SAS_PS_Low_Phy_Power state (see 5.14.5.2).</li> </ol>
Phy wakeup partial timeout	none	10 $\mu$ s	While a phy is in the partial phy power condition (see 4.10.1.3) the time within which the phy shall become active after receiving a Phy Wakeup message (see 5.14.2).
Phy wakeup slumber timeout	none	10 ms	While a phy is in the slumber phy power condition (see 4.10.1.4) the time within which the phy shall become active after receiving a Phy Wakeup message (see 5.14.2).

## 5.11.2 SATA phy reset sequence

### 5.11.2.1 SATA OOB sequence

Figure 68 shows the SATA OOB sequence between a SATA host and SATA device. The SATA OOB sequence is defined by SATA.

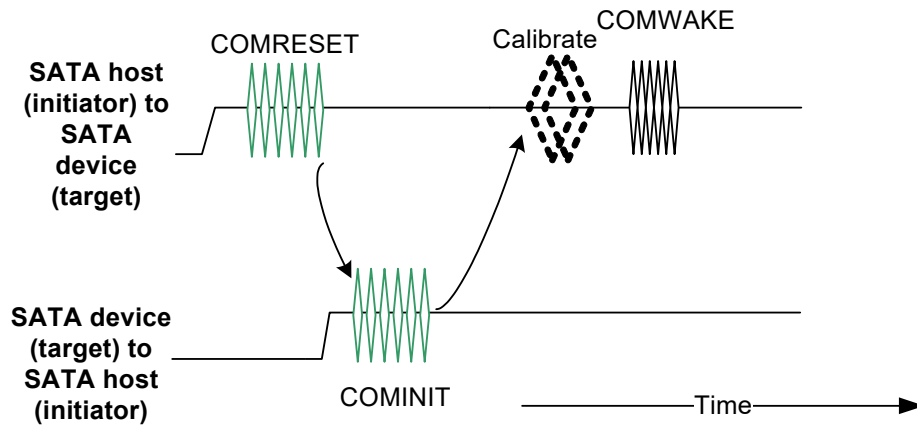


Figure 68 – SATA OOB sequence

### 5.11.2.2 SATA speed negotiation sequence

Figure 69 shows the speed negotiation sequence between a SATA host and SATA device. The SATA speed negotiation sequence is defined by SATA (see SATA).

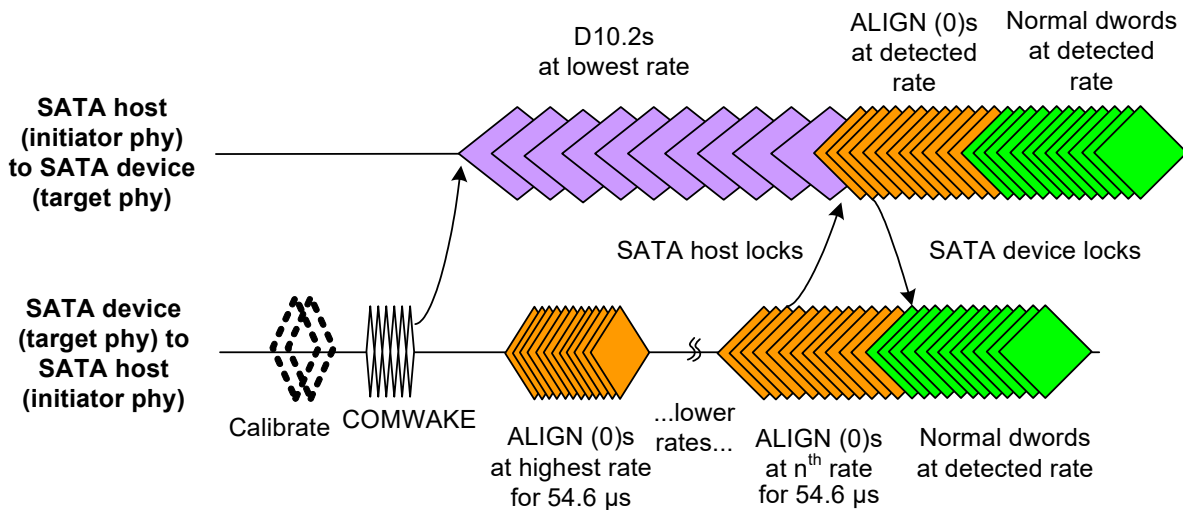


Figure 69 – SATA speed negotiation sequence



Table 86 defines SATA speed negotiation sequence timing parameters used by the SP state machine (see 5.14).

**Table 86 – SATA speed negotiation sequence timing specifications**

Parameter	Time	Comments
Await ALIGN timeout	$873.81\bar{3} \mu\text{s}$ <sup>a</sup>	The minimum time during SATA speed negotiation that a phy shall allow for an ALIGN (0) to be received after detecting COMWAKE Completed.
COMWAKE response time	533 ns <sup>b</sup>	The maximum time during SATA speed negotiation after detecting COMWAKE Completed before which a phy shall start transmitting D10.2 characters.
<sup>a</sup> $873.81\bar{3} \mu\text{s}$ is $32\,768 \times 40 \times \text{nominal OOB I}$ (see SAS-4 and SATA). <sup>b</sup> 533 ns is $200 \times 40 \times \text{nominal OOB I}$ (see SATA).		

The transmitter device shall use SATA signal output levels during the SATA speed negotiation sequence as described in SAS-4.

The phy shall not perform physical link rate tolerance management (see 6.5) during the SATA speed negotiation sequence.

### 5.11.3 SAS to SATA phy reset sequence

SAS initiator phys and expander phys may support SATA (e.g., support being directly attached to a SATA device or a SATA port selector).

To initiate a phy reset sequence a SAS initiator phy or expander phy shall:

- 1) transmit a COMINIT (see SATA); and
- 2) in response to receiving a COMINIT, transmit a COMSAS (see SATA).

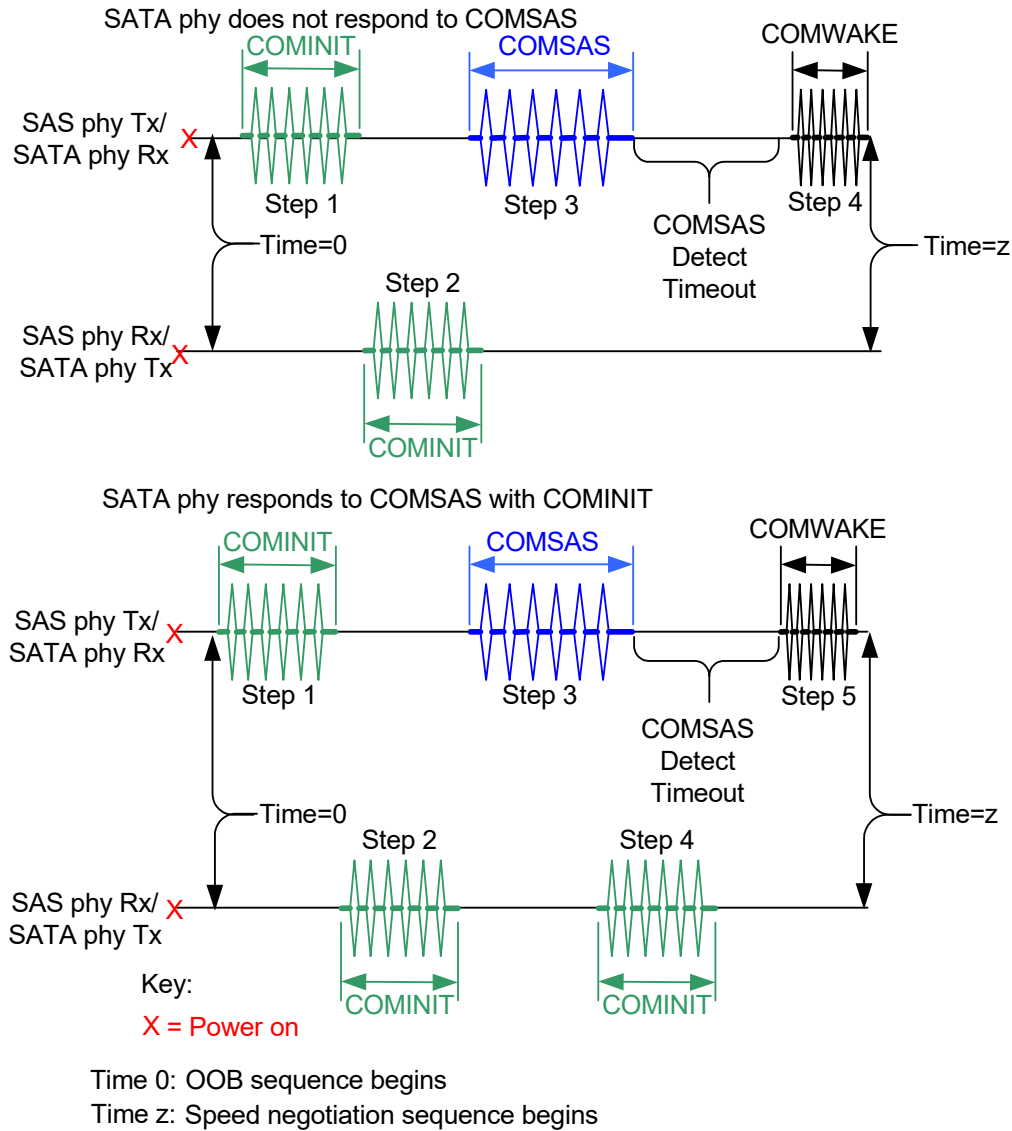
The COMSAS identifies the phy as a SAS phy or expander phy instead of a SATA phy.

If a SATA phy is attached to the physical link, then that SATA phy either:

- a) misinterprets the COMSAS to be a COMRESET (see SATA) and responds with a COMINIT; or
- b) ignores the COMSAS and provides no response within a COMSAS detect timeout.

Either response indicates to the SAS initiator phy or expander phy that a SATA phy is attached. As a result, the SAS initiator phy or expander phy shall transmit COMWAKE (see SATA) and enter the SATA speed negotiation sequence (see 5.11.2.2).

Figure 70 shows an OOB sequence between a SAS phy or expander phy (i.e., a phy compliant with this standard) and a SATA phy (i.e., a phy in a SATA device, defined by SATA). The two possible cases are presented. The first case is that the SATA phy ignores the COMSAS and provides no response within a COMSAS detect timeout. The second case is that the SATA phy misinterprets the COMSAS to be a COMINIT (see SATA). The SP state machine treats these two cases the same (see 5.14.3.9) and determines that a SATA phy is attached after a COMSAS detect timeout. The SATA speed negotiation sequence is entered after COMWAKE is detected.



**Figure 70 – SAS to SATA OOB sequence**

#### 5.11.4 SAS to SAS phy reset sequence

##### 5.11.4.1 SAS OOB sequence

To initiate a SAS OOB sequence a phy transmits a COMINIT (see SATA).

On receipt of a COMINIT the receiving phy either:

- a) if the phy has not yet transmitted a COMINIT, then transmit a COMINIT followed by a COMSAS (see SATA); or

- b) if the phy has transmitted a COMINIT, then transmit a COMSAS.

If the receiving phy does not respond to a COMINIT within the minimum hot-plug timeout (see 5.11.5), then the attached phy may transmit another COMINIT. If repeated, then this results in a livelock.

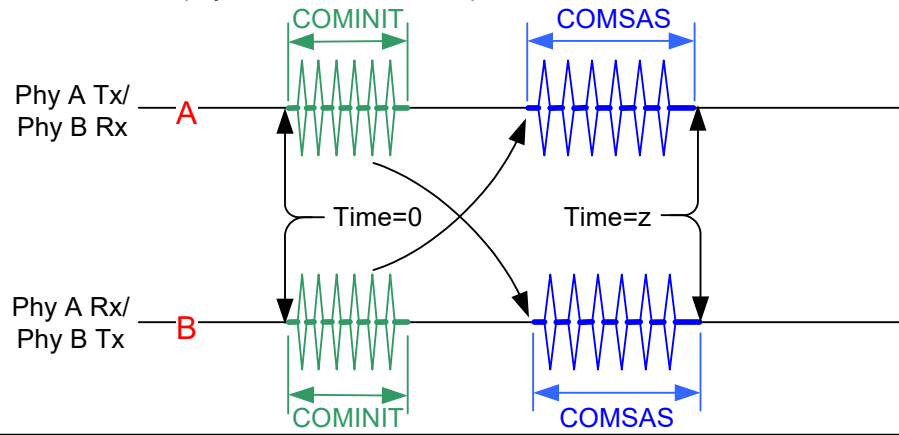
On receipt of a COMSAS, if the receiving phy has not yet transmitted a COMSAS, then the phy transmits a COMSAS.

After completing the transmission of a COMSAS and the successful receipt of a COMSAS the SAS OOB sequence is complete and the SAS speed negotiation sequence begins.

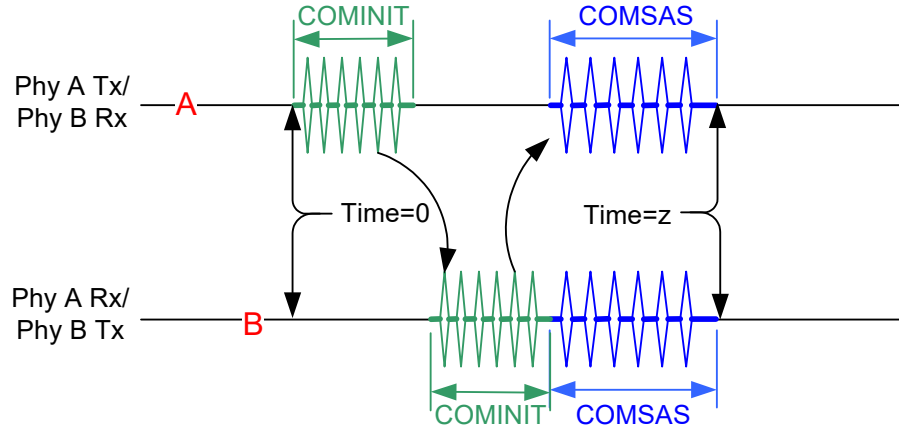
A phy distinguishes between COMINIT and COMSAS and continues with a SAS speed negotiation sequence (see 5.11.4.2) after completing the SAS OOB sequence.

Figure 71 shows several different SAS OOB sequences between phy A and phy B, with phy A starting the SAS OOB sequence at the same time as phy B, before phy B, and before phy B powers on.

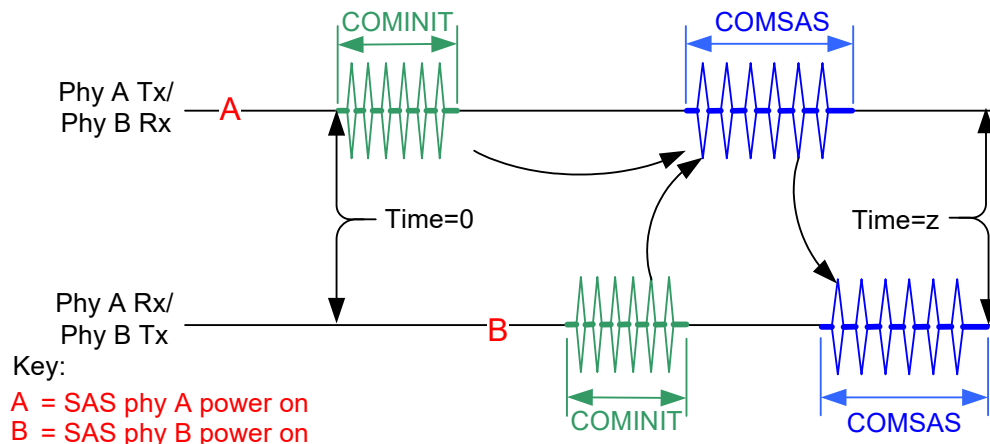
Scenario 1: Both SAS phys start SAS OOB sequence at same time



Scenario 2: SAS phy A starts SAS OOB sequence



Scenario 3: SAS phy B misses SAS phy A s COMINIT



Time 0: SAS phy reset sequence begins

Time z: SAS speed negotiation sequence begins

Figure 71 – SAS to SAS OOB sequence

#### 5.11.4.2 SAS speed negotiation sequence

##### 5.11.4.2.1 SAS speed negotiation sequence overview

The SAS speed negotiation sequence establishes communications between the two phys on a physical link at the highest possible transmission rate.

The SAS speed negotiation sequence is a peer-to-peer negotiation technique that does not assume initiator and target roles. The rules for speed negotiation are the same for both participating phys.

The SAS speed negotiation sequence consists of a set of speed negotiation windows (SNWs). Each SNW is identified by a name (e.g., Speed Negotiation Window-1 or SNW-1).

SNWs conform to one of the following types:

- a) speed negotiation without training (i.e., SNW-1, SNW-2, and Final-SNW) (see 5.11.4.2.3.2);
- b) phy capabilities exchange (i.e., SNW-3) (see 5.11.4.2.3.3);
- c) phy receiver training (i.e., Train\_Rx-SNW) (see 5.11.4.2.3.5); or
- d) phy transmitter training (i.e., Train\_Tx-SNW) (see 5.11.4.2.3.4).

Many of the timing parameters used for defining the SNWs are common to multiple SNW types. All of the timing specifications for all SNW types are defined in 5.11.4.2.2.

Phys may implement a subset of SNWs provided that the subset implements a valid speed negotiation sequence. SAS speed negotiation sequences are defined in 5.11.4.2.4.

The transmitter device shall use SAS signal output levels during the SAS speed negotiation sequence as described in SAS-4.

The phy shall not perform physical link rate tolerance management (see 6.5) during the SAS speed negotiation sequence.

## 5.11.4.2.2 SAS speed negotiation sequence timing specifications

Table 87 defines the timing specifications for the SAS speed negotiation sequence.

**Table 87 – SAS speed negotiation sequence timing specifications** (part 1 of 2)

Parameter	Acronym	Time <sup>a</sup>	Comments
Rate change delay time	RCDT	750 000 OOB <sub>I</sub> <sup>b</sup>	The time the transmitter device shall transmit negotiation idle at the beginning of: a) SNW-1; b) SNW-2; c) SNW-3; d) Final-SNW; and e) Train_Rx-SNW or Train_Tx-SNW.
Speed negotiation transmit time	SNTT	163 840 OOB <sub>I</sub> <sup>c</sup>	During SNW-1, SNW-2, and Final-SNW, the time after RCDT during which ALIGN (0) or ALIGN (1) is transmitted.  During SNW-3, the time after RCDT in which bit cells and OOB idle are transmitted.  During low phy power conditions, the maximum time for a phy to transmit ALIGN (0)s or ALIGN (1)s while in the SP32:SAS_PS_ALIGN0 state or SP33:SAS_PS_ALIGN1 state (see 5.14.5.4).
Speed negotiation lock time	SNLT	153 600 OOB <sub>I</sub> <sup>d</sup>	The maximum time for a phy to reply with ALIGN (1) during SNW-1, SNW-2, and Final-SNW.  During low phy power conditions, the maximum time for a phy to reply with an ALIGN (0) or ALIGN (1) while in the SP32:SAS_PS_ALIGN0 state or SP33:SAS_PS_ALIGN1 state (see 5.14.5.4).
Actual lock time	none		The time during SNW-1, SNW-2, and Final-SNW at which actual dword synchronization occurs to the received ALIGN (0) or ALIGN (1) and the phy begins transmitting ALIGN (1) rather than ALIGN (0).
SNW time	SNWT	913 840 OOB <sub>I</sub> <sup>e</sup>	The duration of SNW-1, SNW-2, SNW-3, or Final-SNW.
<sup>a</sup> OOB <sub>I</sub> is defined in SAS-4. <sup>b</sup> 750 000 OOB <sub>I</sub> (e.g., RCDT) is nominally 500 $\mu$ s (i.e., $18\,750 \times 40$ OOB <sub>I</sub> ). <sup>c</sup> 163 840 OOB <sub>I</sub> (e.g., SNTT) is nominally 109.226 $\mu$ s (i.e., $4\,096 \times 40$ OOB <sub>I</sub> ). <sup>d</sup> 153 600 OOB <sub>I</sub> (e.g., SNLT) is nominally 102.4 $\mu$ s (i.e., $(4\,096 - 256) \times 40$ OOB <sub>I</sub> ). <sup>e</sup> 913 840 OOB <sub>I</sub> (e.g., SNWT) is nominally 609.226 $\mu$ s (i.e., RCDT + SNTT). <sup>f</sup> 2 200 OOB <sub>I</sub> is nominally 1 466.6 ns (i.e., COMWAKE signal time (see SAS-4)). <sup>g</sup> 29 998 080 OOB <sub>I</sub> (e.g., MRTT) is nominally 19.998 72 ms (i.e., $11\,718 \times 64 \times 40$ OOB <sub>I</sub> ). This is the time of the maximum number of complete receiver training patterns that fit into 20 ms. <sup>h</sup> 28 497 920 OOB <sub>I</sub> (e.g., TLT) is nominally 18.998 613 ms (i.e., $11\,132 \times 64 \times 40$ OOB <sub>I</sub> ). This is the time of the maximum number of complete receiver training patterns that fit into 19 ms. <sup>i</sup> 30 748 080 OOB <sub>I</sub> (e.g., MTWT) is nominally 20.498 72 ms (i.e., RCDT + MRTT). <sup>j</sup> 750 000 000 OOB <sub>I</sub> (e.g., MTTT) is nominally 500 ms. <sup>k</sup> 750 750 000 OOB <sub>I</sub> (e.g., MTXT) is nominally 500.5 ms (i.e., MTTT + RCDT).			

Table 87 – SAS speed negotiation sequence timing specifications (part 2 of 2)

Parameter	Acronym	Time <sup>a</sup>	Comments
SNW-3 Bit cell time	none	2 200 OOB <sup>f</sup>	The time to transmit a COMWAKE or OOB idle during SNW-3.
Maximum receiver training time	MRTT	29 998 080 OOB <sup>g</sup>	The maximum time for receiver training to complete during Train_Rx-SNW.
Training lock time	TLT	28 497 920 OOB <sup>h</sup>	The maximum time for a phy to reply with TRAIN_DONE during Train_Rx-SNW.
Train_Rx-SNW time	none		The actual duration of Train_Rx-SNW.
Maximum Train_Rx-SNW time	MTWT	30 748 080 OOB <sup>i</sup>	The maximum duration of Train_Rx-SNW.
Maximum transmitter training time	MTTT	750 000 000 OOB <sup>j</sup>	The maximum time for transmitter training to complete during Train_Tx-SNW.
Maximum Train_Tx time	MTXT	750 750 000 OOB <sup>k</sup>	The maximum duration of Train_Tx-SNW.
<sup>a</sup> OOB <sup>i</sup> is defined in SAS-4. <sup>b</sup> 750 000 OOB <sup>i</sup> (e.g., RCDT) is nominally 500 $\mu$ s (i.e., $18\,750 \times 40$ OOB <sup>i</sup> ). <sup>c</sup> 163 840 OOB <sup>i</sup> (e.g., SNTT) is nominally 109.226 $\mu$ s (i.e., $4\,096 \times 40$ OOB <sup>i</sup> ). <sup>d</sup> 153 600 OOB <sup>i</sup> (e.g., SNLT) is nominally 102.4 $\mu$ s (i.e., $(4\,096 - 256) \times 40$ OOB <sup>i</sup> ). <sup>e</sup> 913 840 OOB <sup>i</sup> (e.g., SNWT) is nominally 609.226 $\mu$ s (i.e., RCDT + SNTT). <sup>f</sup> 2 200 OOB <sup>i</sup> is nominally 1 466.6 ns (i.e., COMWAKE signal time (see SAS-4)). <sup>g</sup> 29 998 080 OOB <sup>i</sup> (e.g., MRTT) is nominally 19.998 72 ms (i.e., $11\,718 \times 64 \times 40$ OOB <sup>i</sup> ). This is the time of the maximum number of complete receiver training patterns that fit into 20 ms. <sup>h</sup> 28 497 920 OOB <sup>i</sup> (e.g., TLT) is nominally 18.998 613 ms (i.e., $11\,132 \times 64 \times 40$ OOB <sup>i</sup> ). This is the time of the maximum number of complete receiver training patterns that fit into 19 ms. <sup>i</sup> 30 748 080 OOB <sup>i</sup> (e.g., MTWT) is nominally 20.498 72 ms (i.e., RCDT + MRTT). <sup>j</sup> 750 000 000 OOB <sup>i</sup> (e.g., MTTT) is nominally 500 ms. <sup>k</sup> 750 750 000 OOB <sup>i</sup> (e.g., MTXT) is nominally 500.5 ms (i.e., MTTT + RCDT).			

#### 5.11.4.2.3 Speed negotiation window (SNW) definitions

##### 5.11.4.2.3.1 SNW definitions overview

During each SNW, a phy shall either:

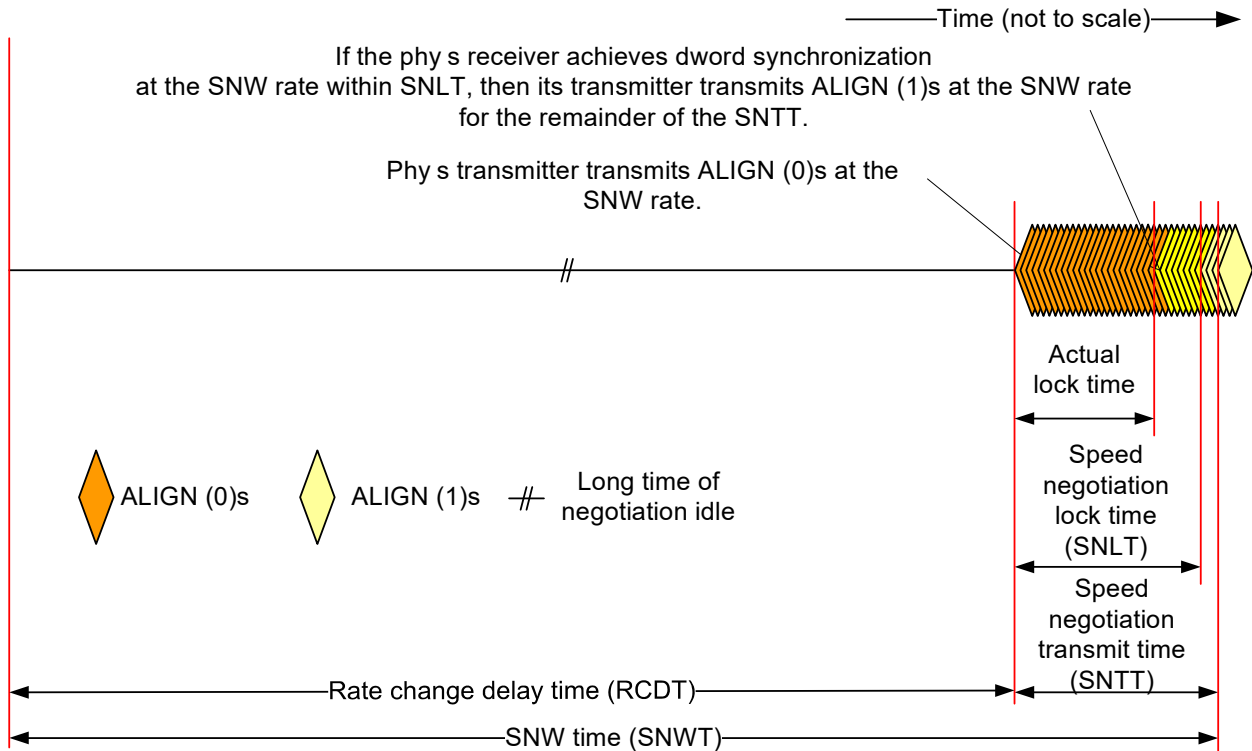
- if it supports the SNW, then transmit and receive as defined for the SNW; or
- if it does not support the SNW, then transmit negotiation idle and ignore the SNW information received.

If a phy supports the SNW and receives the expected transmission, then the SNW is valid. If a phy does not receive the expected transmission from the attached phy, then the SNW is invalid.

#### 5.11.4.2.3.2 SNW-1, SNW-2, and Final-SNW

Figure 72 defines SNW-1, SNW-2, and Final-SNW, including:

- SNW time (SNWT);
- rate change delay time (RCDT);
- speed negotiation transmit time (SNTT);
- speed negotiation lock time (SNLT); and
- actual lock time.



**Figure 72 – SNW-1, SNW-2, and Final-SNW**

If the phy supports the SNW, then it shall transmit:

- negotiation idle for an RCDT; and
- ALIGNs at the SNW rate for the remainder of the SNWT (i.e., for SNTT).

If the phy does not support the SNW, then it shall transmit negotiation idle for the entire SNWT.



Table 88 defines the SNW rate used in SNW-1, SNW-2, and Final-SNW.

**Table 88 – SNW rates used in SNW-1, SNW-2, and Final-SNW**

SNW	SNW rate
SNW-1	1.5 Gbit/s
SNW-2	3 Gbit/s
Final-SNW	Based on SNW-1, SNW-2, and SNW-3 validity: a) 1.5 Gbit/s if SNW-1 is valid and SNW-2 is invalid; or b) 3 Gbit/s if SNW-2 is valid and SNW-3 is invalid.

If the phy supports the SNW, then after RCDT it shall attempt to attain dword synchronization on an incoming series of dwords (e.g., ALIGN (0) or ALIGN (1) primitives) at that rate for the SNLT and:

- if the phy achieves dword synchronization within the SNLT, then it shall change from transmitting ALIGN (0) primitives to transmitting ALIGN (1) primitives for the remainder of the SNTT (i.e., the remainder of the SNW time). The point at which the phy achieves dword synchronization is called the actual lock time; or
- if the phy does not achieve dword synchronization within the SNLT, then it shall continue transmitting ALIGN (0) primitives for the remainder of the SNTT (i.e., the remainder of the SNW time).

At the end of the SNTT:

- if the phy is both transmitting and receiving ALIGN (1) primitives, then it shall consider the SNW to be valid; or
- if the phy is not both transmitting and receiving ALIGN (1) primitives, then it shall consider the SNW to be invalid.

The phy shall disable SSC (see SAS-4) during SNW-1, SNW-2, and Final-SNW.

#### 5.11.4.2.3.3 SNW-3

SNW-3 allows the phys to exchange phy capabilities (see 5.8) to establish phy parameters used in Train\_Rx-SNW and Train\_Tx-SNW.

Figure 73 defines SNW-3, including:

- SNW time (SNWT);
- rate change delay time (RCDT); and
- speed negotiation transmit time (SNTT).

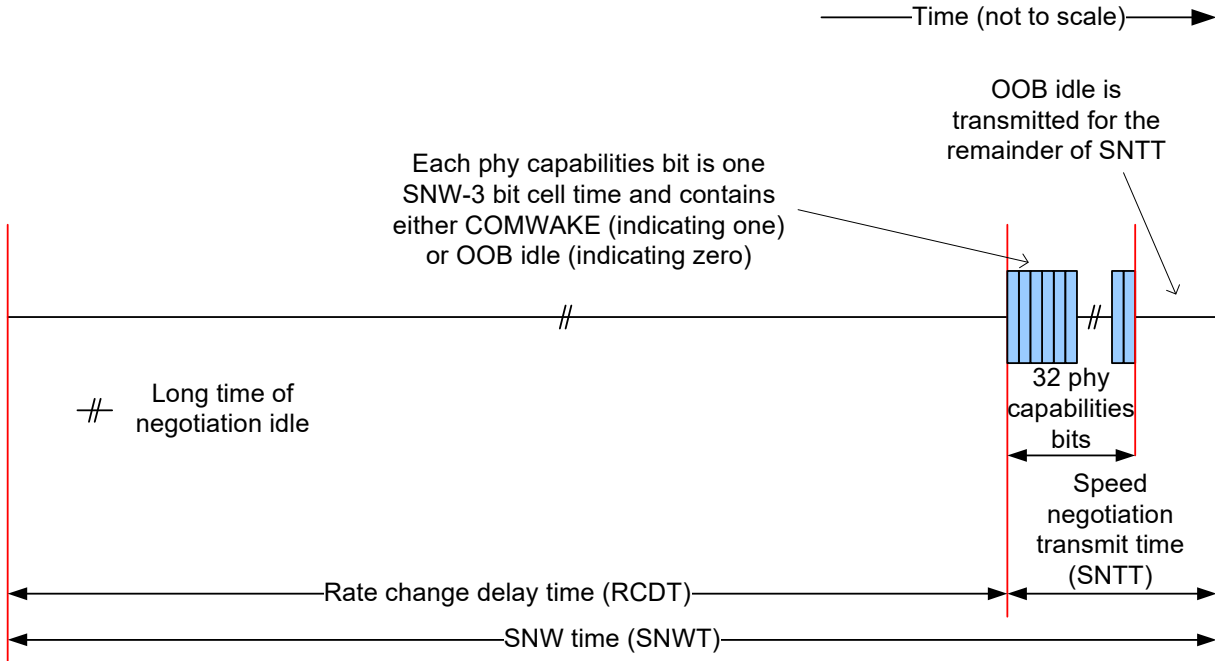
**Figure 73 – SNW-3**

Table 89 defines the content of each phy capabilities bit.

**Table 89 – SNW-3 phy capabilities bit**

Value	Transmitted
One	COMWAKE (see 5.7)
Zero	OOB idle

If the phy supports SNW-3, then:

- a) the phy shall:
  - 1) transmit negotiation idle for an RCDT;
  - 2) transmit 32 phy capabilities bits (see 5.8); and
  - 3) transmit OOB idle for the remainder of SNTT;
 and
- b) the phy shall receive a 32-bit phy capabilities value from the attached phy.

The phy receives no capabilities bits (i.e., negotiation idle) if the attached phy does not support SNW-3.

If the phy does not support SNW-3, then it shall transmit negotiation idle for the entire SNWT and ignore any phy capabilities bits received.

The first phy capabilities bit (see 5.8) is the START bit and is set to one. Each of the remaining 31 phy capabilities bits is set to one or zero. The receiver shall use the START bit to detect the beginning of the phy capabilities bits and establish the timing for subsequent bits.

The phy shall consider SNW-3 to be valid if it supports SNW-3 and receives at least one supported settings bit set to one (see table 70). If the phy does not support SNW-3 or does not receive at least one supported settings bit set to one, then it shall consider SNW-3 to be invalid.

The phy may transmit with SSC enabled or disabled (see SAS-4) during SNW-3.

#### 5.11.4.2.3.4 Train\_Tx-SNW

##### 5.11.4.2.3.4.1 Phy's transmitter initial condition

If transmitter training is enabled, then:

- if there are trained values, then that phy shall set all of its coefficients to the most recent trained values before originating a phy reset sequence (see 5.11.1); or
- if there are no trained values, then that phy shall set all of its coefficients to a default value before originating a phy reset sequence. The determination of the default value is outside the scope of this standard.

##### 5.11.4.2.3.4.2 Transmitter training

The Train\_Tx-SNW includes the following:

- maximum Train\_Tx-SNW window time (MTXT);
- rate change delay time (RCDT);
- maximum transmitter train time (MTTT); and
- the point at which the transmitter is trained.

Figure 74 defines the Train\_Tx-SNW while the phy is in the SAS dword mode.

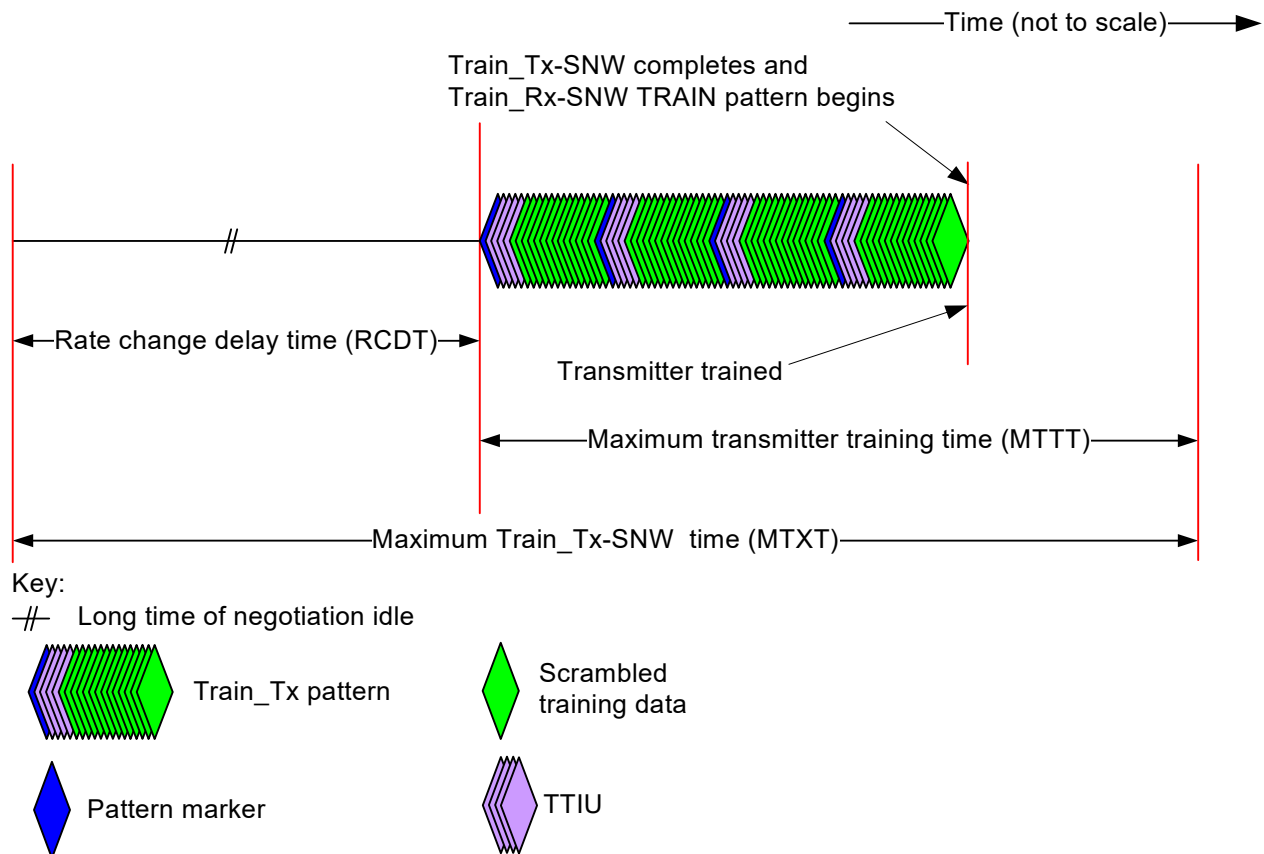
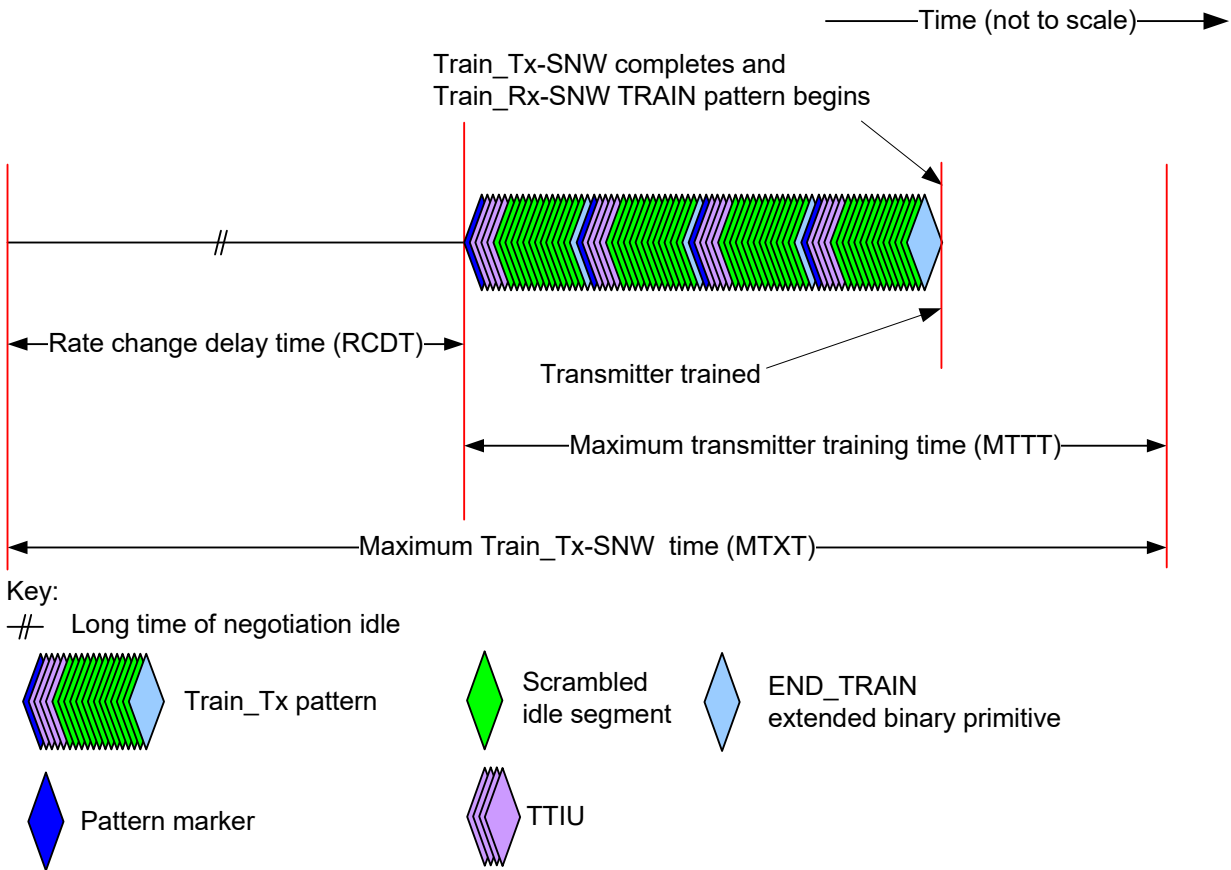


Figure 74 – Train\_Tx-SNW while in the SAS dword mode

Figure 75 defines the Train\_Tx-SNW while the phy is in the SAS packet mode.



**Figure 75 – Train\_Tx-SNW while in the SAS packet mode**

The Train\_Tx-SNW contains transmitter training patterns formed by the Train\_Tx pattern as defined in table 90.

**Table 90 – Transmitter training pattern**

Transmitter training pattern	Mode	Description
Train_Tx pattern	SAS dword	Sequence of: 1) pattern marker (see 5.11.4.2.3.4.3); 2) TTIU (see 5.11.4.2.3.5); and 3) 58 data dwords set to 00000000h that are transmitted scrambled and 8b10b encoded.
	SAS packet	Sequence of: 1) pattern marker (see 5.11.4.2.3.4.3); 2) TTIU (see 5.11.4.2.3.5); 3) 59 SPL packet payloads containing scrambled idle segments; and 4) one END_TRAIN.

The scrambler is the same as that defined for the link layer (see 6.8) and shall be initialized at the end of RCDT. The scrambler shall not be reinitialized for the remainder of the Train\_Tx-SNW.

The phy shall start transmitting Train\_Tx patterns at the end of RCDT. The number of Train\_Tx patterns transmitted is determined by the time required for the phys to complete transmitter training.

After RCDT, the local phy's receiver shall attempt to train the attached phy's transmitter as follows:

- a) if the local phy and the attached phy complete transmitter training within MTTT, then the phy shall start Train\_Rx-SNW (see 5.11.4.2.3.5). At the point the phy completes transmitter training the phy shall consider the Train\_Tx-SNW to be valid; or
- b) if the local phy and the attached phy do not complete transmitter training within MTTT, then the phy shall consider the Train\_Tx-SNW to be invalid.

The phy shall not transmit primitives during Train\_Tx-SNW.

During the Train\_Tx-SNW the phy's transmitter shall transmit a pattern marker (see figure 76) at the start of each Train\_Tx pattern as defined in table 90.

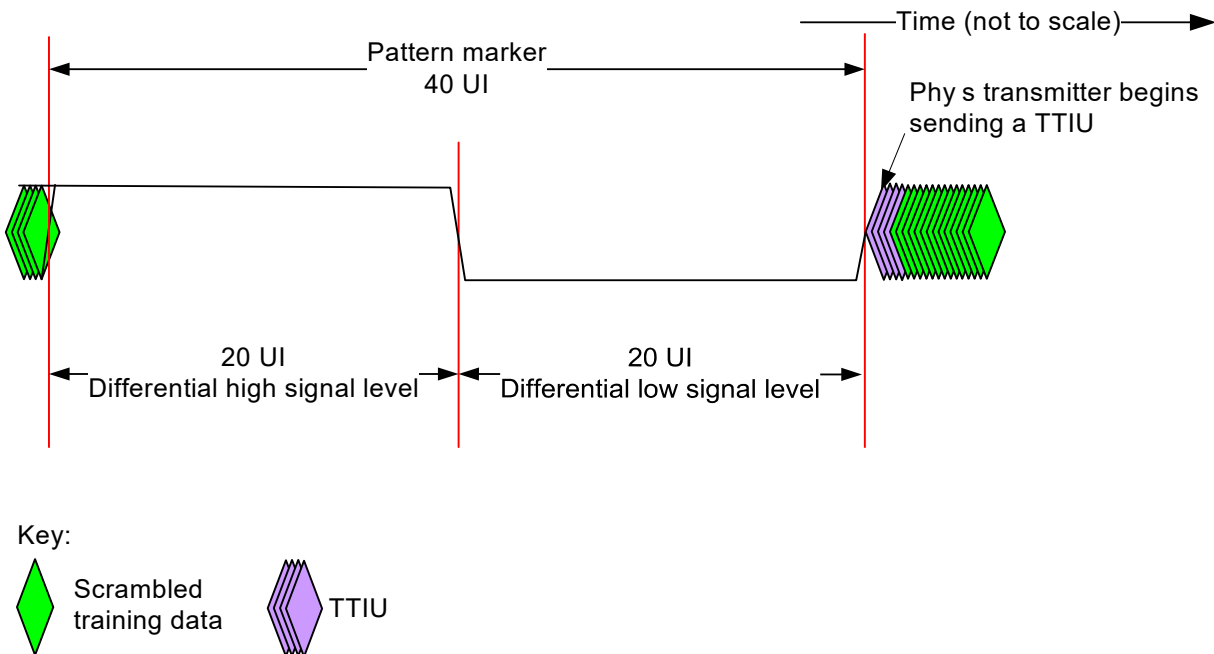
#### 5.11.4.2.3.4.3 Pattern marker

The pattern marker specifies the start of a TTIU.

A pattern marker is formed by a:

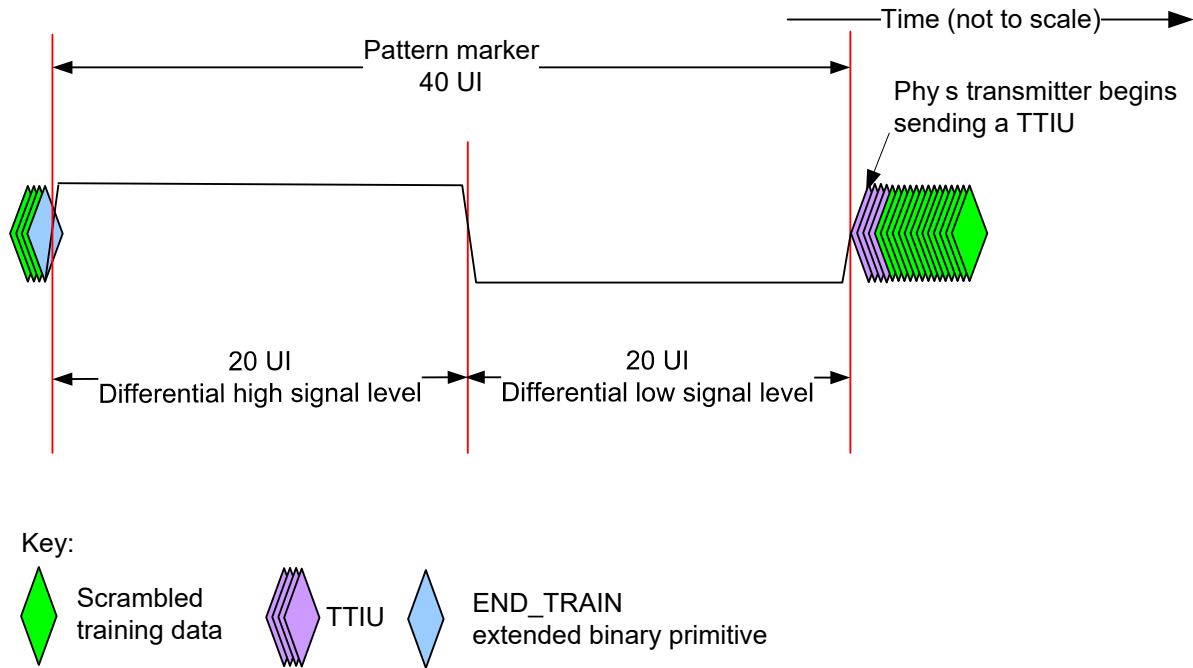
- 1) 20 UI (e.g., 1.6 ns at 12 Gbit/s and 0.88 ns at 22.5 Gbit/s) differential high signal level; and
- 2) 20 UI (e.g., 1.6 ns at 12 Gbit/s and 0.88 ns at 22.5 Gbit/s) differential low signal level.

Figure 76 shows the pattern marker while the phy is in the SAS dword mode.



**Figure 76 – Pattern marker transmission while in the SAS dword mode**

Figure 77 shows the pattern marker while the phy is in the SAS packet mode.



**Figure 77 – Pattern marker transmission while in the SAS packet mode**

#### 5.11.4.2.3.4.4 Pattern marker detection while in SAS dword mode

If the SAS dword mode is enabled, then during the Train\_Tx-SNW if the phy's receiver detects:

- 1) a differential high signal level for greater than or equal to 19 UIs and less than or equal to 24 UIs; and
- 2) a differential low signal level for greater than or equal to 19 UIs and less than or equal to 21 UIs,

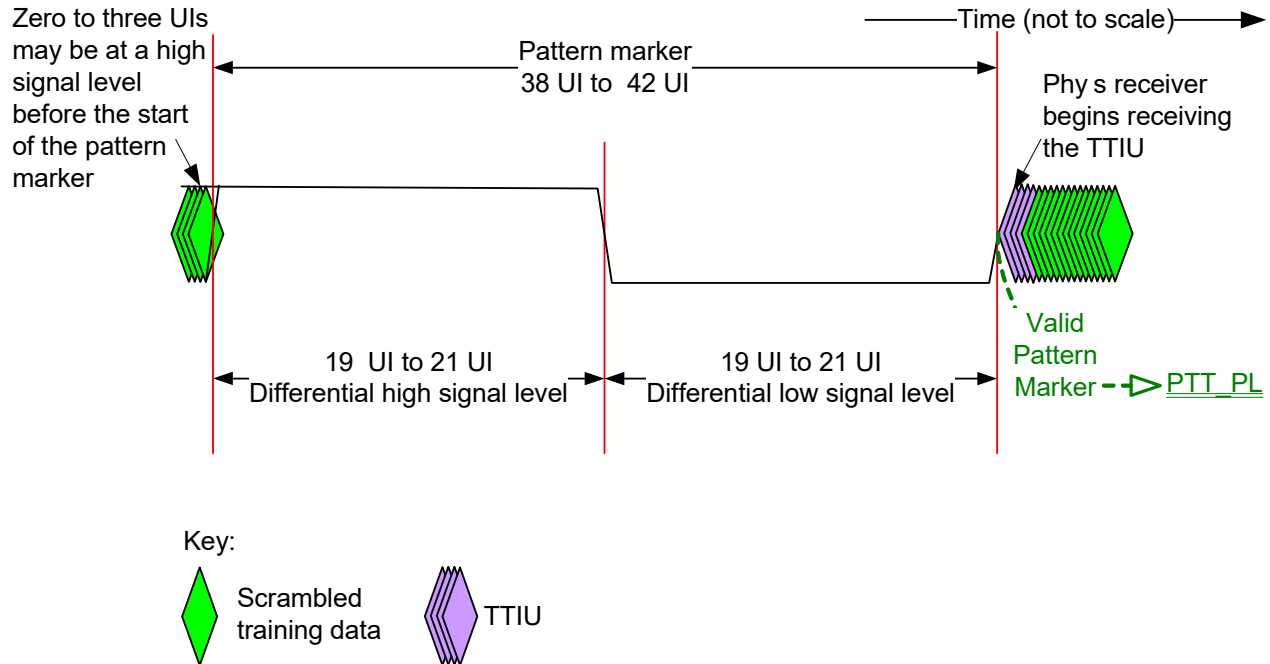
then the phy's receiver shall consider the pattern marker to be valid (see figure 78).

If the SAS dword mode is enabled, then during the Train\_Tx-SNW if the phy's receiver detects:

- a) a differential high signal level for greater than or equal to 13 UIs and less than 19 UIs;
- b) a differential high signal level for greater than 24 UIs;
- c) a differential low signal level for greater than or equal to 13 UIs and less than 19 UIs; or
- d) a differential low signal level for greater than 21 UIs,

then the phy's receiver shall consider the pattern marker to be invalid.

If the phy's receiver does not detect a pattern marker at the beginning of a Train\_Tx pattern then the pattern marker shall be invalid.



**Figure 78 – Valid pattern marker detection while in SAS dword mode**

#### 5.11.4.2.3.4.5 Pattern marker detection while in SAS packet mode

If the SAS packet mode is enabled, then during the Train\_Tx-SNW if the phy's receiver detects:

- 1) a differential high signal level for greater than or equal to 19 UIs and less than or equal to 21 UIs; and
- 2) a differential low signal level for greater than or equal to 19 UIs and less than or equal to 21 UIs,

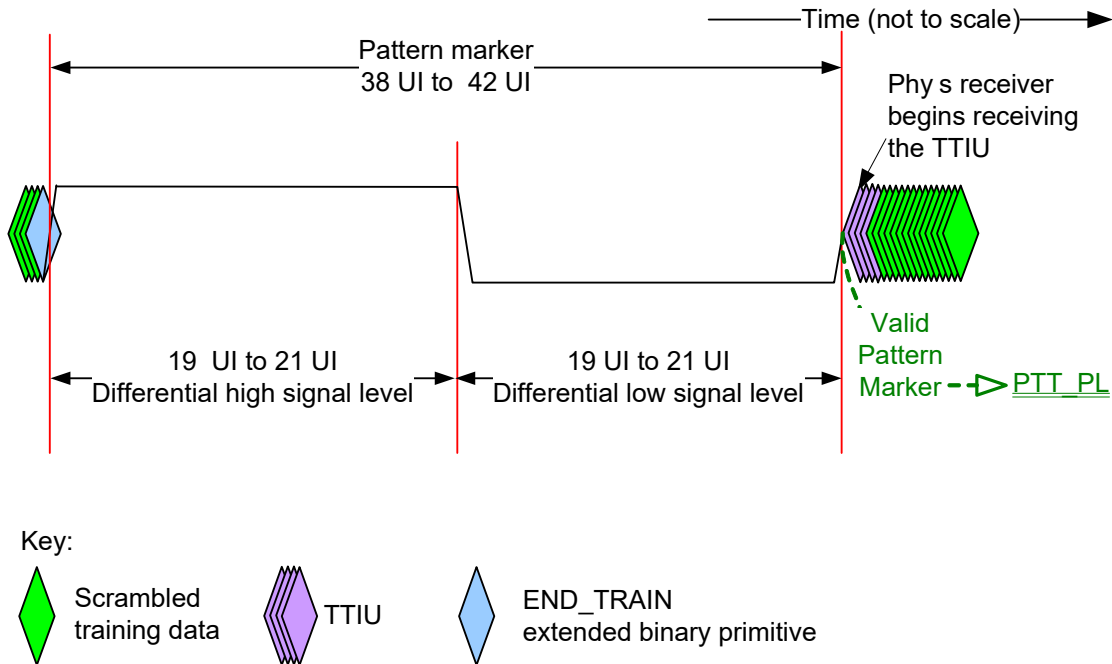
then the phy's receiver shall consider the pattern marker to be valid (see figure 79).

If the SAS packet mode is enabled, then during the Train\_Tx-SNW if the phy's receiver detects:

- a) a differential high signal level for greater than or equal to 13 UIs and less than 19 UIs;
- b) a differential high signal level for greater than 21 UIs;
- c) a differential low signal level for greater than or equal to 13 UIs and less than 19 UIs; or
- d) a differential low signal level for greater than 21 UIs,

then the phy's receiver shall consider the pattern marker to be invalid.

If the phy's receiver does not detect a pattern marker at the beginning of a Train\_Tx pattern then the pattern marker shall be invalid.



**Figure 79 – Valid pattern marker detection while in SAS packet mode**

#### 5.11.4.2.3.5 Train\_Rx-SNW while in SAS dword mode

Figure 80 defines the Train\_Rx-SNW, including:

- maximum Train\_Rx-SNW window time (MTWT);
- rate change delay time (RCDT);
- maximum receiver train time (MRTT);
- train lock time (TLT); and
- the point at which the receiver is trained and dword synchronization is acquired.



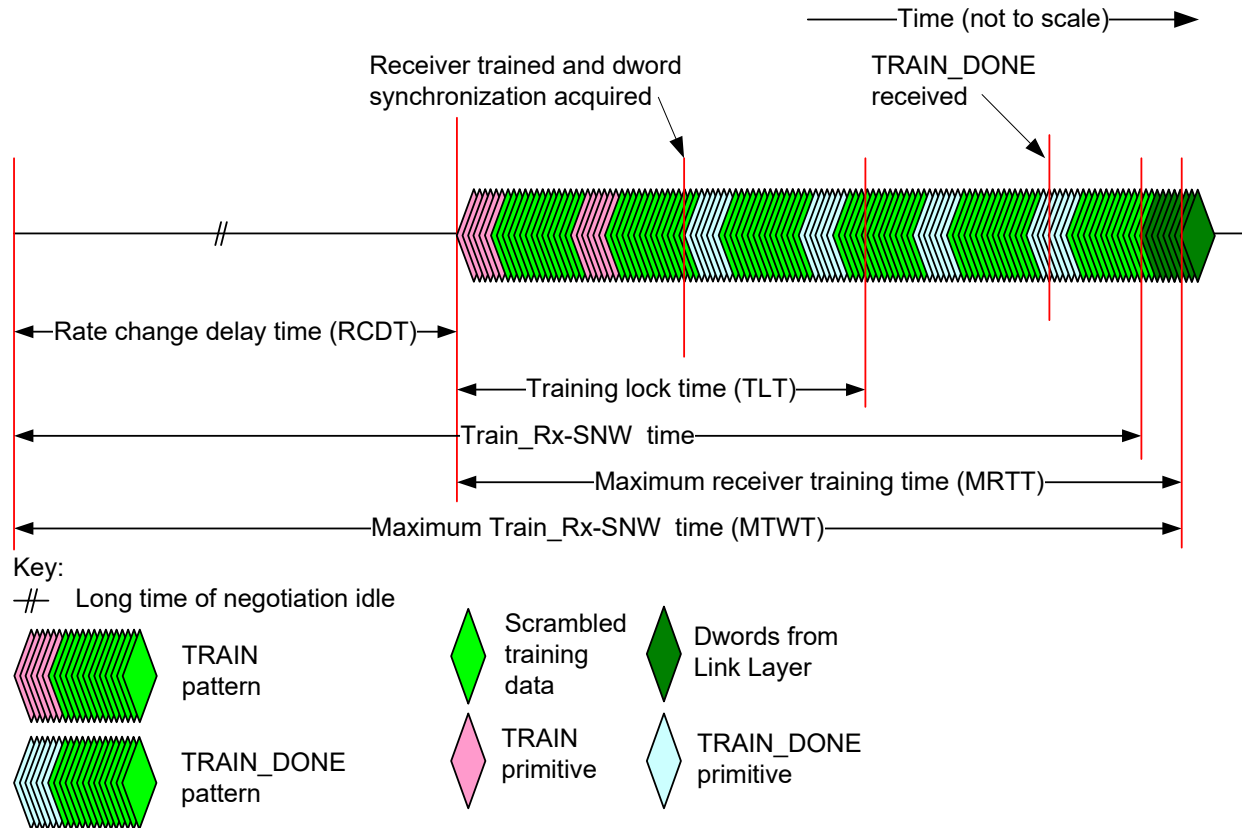


Figure 80 – Train\_Rx-SNW while in SAS dword mode

The Train\_Rx-SNW contains receiver training patterns formed by TRAIN and TRAIN\_DONE (see 6.2) as defined in table 91.

Table 91 – Receiver training patterns while in SAS dword mode

Receiver training pattern	Description
TRAIN pattern	Sequence of: 1) TRAIN primitive sequence; and 2) 58 data dwords set to 00000000h that are transmitted scrambled and 8b10b encoded.
TRAIN_DONE pattern	Sequence of: 1) TRAIN_DONE primitive sequence; and 2) 58 data dwords set to 00000000h that are transmitted scrambled and 8b10b encoded.

The scrambler is the same as that defined for the link layer (see 6.8). If there is no Train\_Tx-SNW, then the scrambler shall be initialized at the end of RCDT. If there is a Train\_Tx-SNW, then the scrambler shall be initialized at the end of Train\_Tx-SNW. The scrambler shall not be reinitialized for the remainder of the Train\_Rx-SNW.

If there is no Train\_Tx-SNW, then the phy shall start transmitting TRAIN patterns at the end of RCDT. If a Train\_Tx-SNW occurs, then the phy shall start transmitting TRAIN patterns at the end of the transmitter training (see 5.11.4.2.3.4.2). The first TRAIN pattern may have either starting disparity. The number of TRAIN patterns transmitted is determined by the time required for the phy's receiver to complete training and acquire dword synchronization. The phy shall transmit at least one TRAIN pattern and shall transmit a minimum of four TRAIN\_DONE patterns:

- a) if the phy achieves dword synchronization within the TLT, then, after completing transmission of the current TRAIN pattern, the phy shall change from transmitting TRAIN patterns to transmitting TRAIN\_DONE patterns for the remainder of the Train\_Rx-SNW window time (i.e., the remainder of the SNW time); or
- b) if the phy does not achieve dword synchronization within the TLT, then the phy shall continue transmitting TRAIN patterns for the remainder of the MRTT (i.e., the remainder of the SNW time).

The phy shall not compare the received data characters to the expected transmitted data characters in the receiver training pattern.

If the phy:

- a) transmits four or more TRAIN\_DONE patterns; and
- b) receives a minimum of one TRAIN\_DONE primitive sequence before MRTT,

then the phy shall:

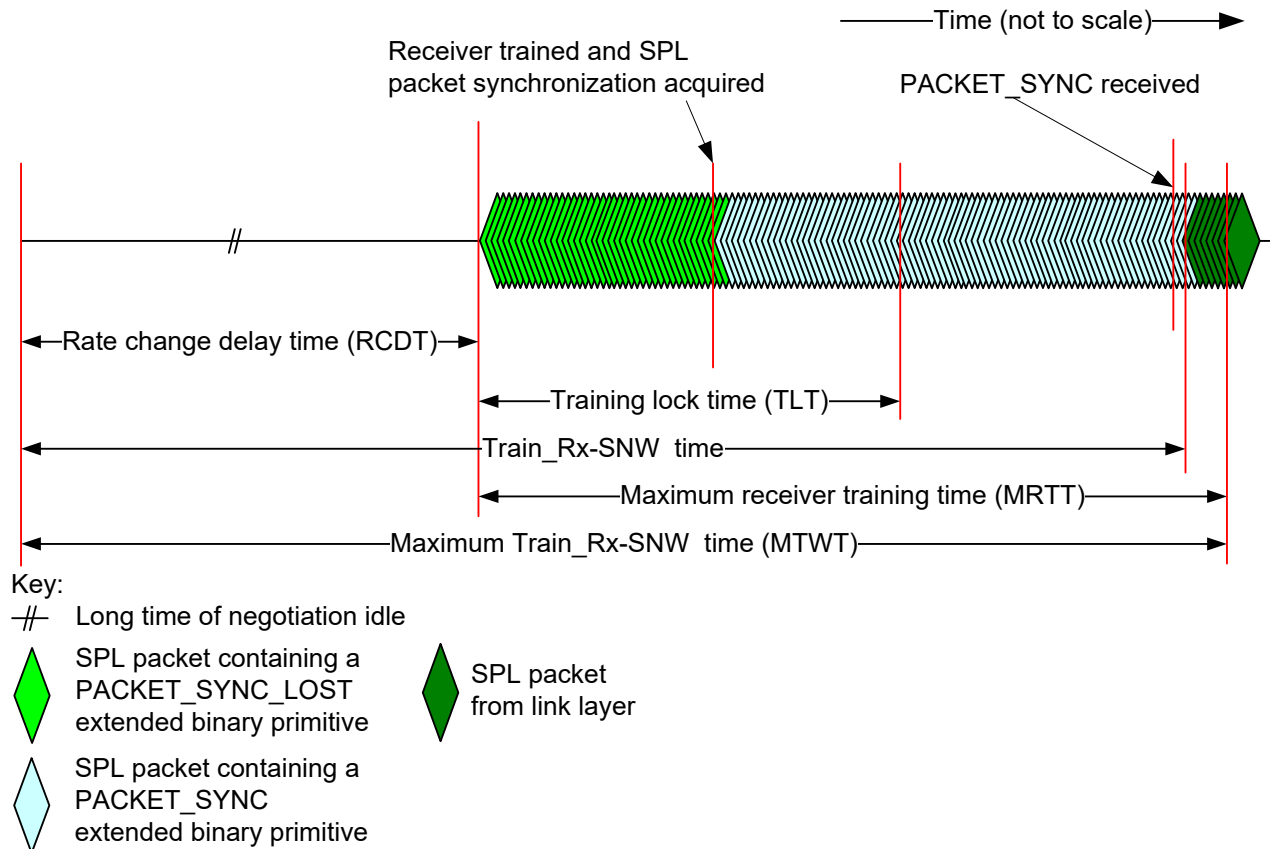
- a) after completing transmission of the current TRAIN\_DONE pattern, transmit at least one more TRAIN\_DONE pattern, stop transmitting TRAIN\_DONE patterns, and start transmitting dwords from the link layer; and
- b) consider the Train\_Rx-SNW to be valid.

If the phy does not receive a TRAIN\_DONE primitive sequence before MRTT and transmits four or more TRAIN\_DONE patterns, then it shall consider the Train\_Rx-SNW to be invalid.

#### 5.11.4.2.3.6 Train\_Rx-SNW while in SAS packet mode

Figure 81 defines the Train\_Rx-SNW, including:

- a) maximum Train\_Rx-SNW window time (MTWT);
- b) rate change delay time (RCDT);
- c) maximum receiver train time (MRTT);
- d) train lock time (TLT); and
- e) the point at which:
  - A) the receiver is trained; and
  - B) SPL packet synchronization is acquired.



**Figure 81 – Train\_Rx-SNW while in SAS packet mode**

The Train\_Rx-SNW contains receiver training extended binary primitives formed by PACKET\_SYNC\_LOST and PACKET\_SYNC (see 6.4).

The scrambler is the same as that defined for the link layer (see 6.8). If there is no Train\_Tx-SNW, then the scrambler shall be initialized at the end of RCDT.

If there is no Train\_Tx-SNW, then the phy shall start transmitting PACKET\_SYNC\_LOSTs at the end of RCDT. If a Train\_Tx-SNW occurs, then the phy shall start transmitting PACKET\_SYNC\_LOSTs at the end of the transmitter training (see 5.11.4.2.3.4.2). The number of PACKET\_SYNC\_LOSTs transmitted is determined by the time required for the phy's receiver to complete training and acquire SPL packet synchronization. The phy shall transmit a minimum of one PACKET\_SYNC\_LOST and shall transmit a minimum of four PACKET\_SYNCs. If the phy's receiver:

- a) achieves SPL packet synchronization within the TLT, then, after completing transmission of the current PACKET\_SYNC\_LOST, the phy shall change from transmitting PACKET\_SYNC\_LOSTs to transmitting PACKET\_SYNCs for the remainder of the Train\_Rx-SNW window time (i.e., the remainder of the SNW time); or
- b) does not achieve SPL packet synchronization within the TLT, then the phy shall continue transmitting PACKET\_SYNC\_LOSTs for the remainder of the MRTT (i.e., the remainder of the SNW time).

If the phy:

- a) transmits four or more PACKET\_SYNCs; and
- b) receives a minimum of one PACKET\_SYNC before MRTT,

then the phy shall:

- a) after completing transmission of the current PACKET\_SYNC:
  - 1) transmit at least one more PACKET\_SYNC;

- 2) stop transmitting PACKET\_SYNCs;
  - 3) reinitialize the scrambler; and
  - 4) start transmitting SPL packets from the link layer;
- and

- b) consider the Train\_Rx-SNW to be valid.

If the phy does not receive a PACKET\_SYNC before MRTT and transmits four or more PACKET\_SYNCs, then it shall consider the Train\_Rx-SNW to be invalid.

#### 5.11.4.2.4 SAS speed negotiation sequence

The SAS speed negotiation sequence consists of a set of SNWs (see 5.11.4.2.3) in the order shown in figure 82.

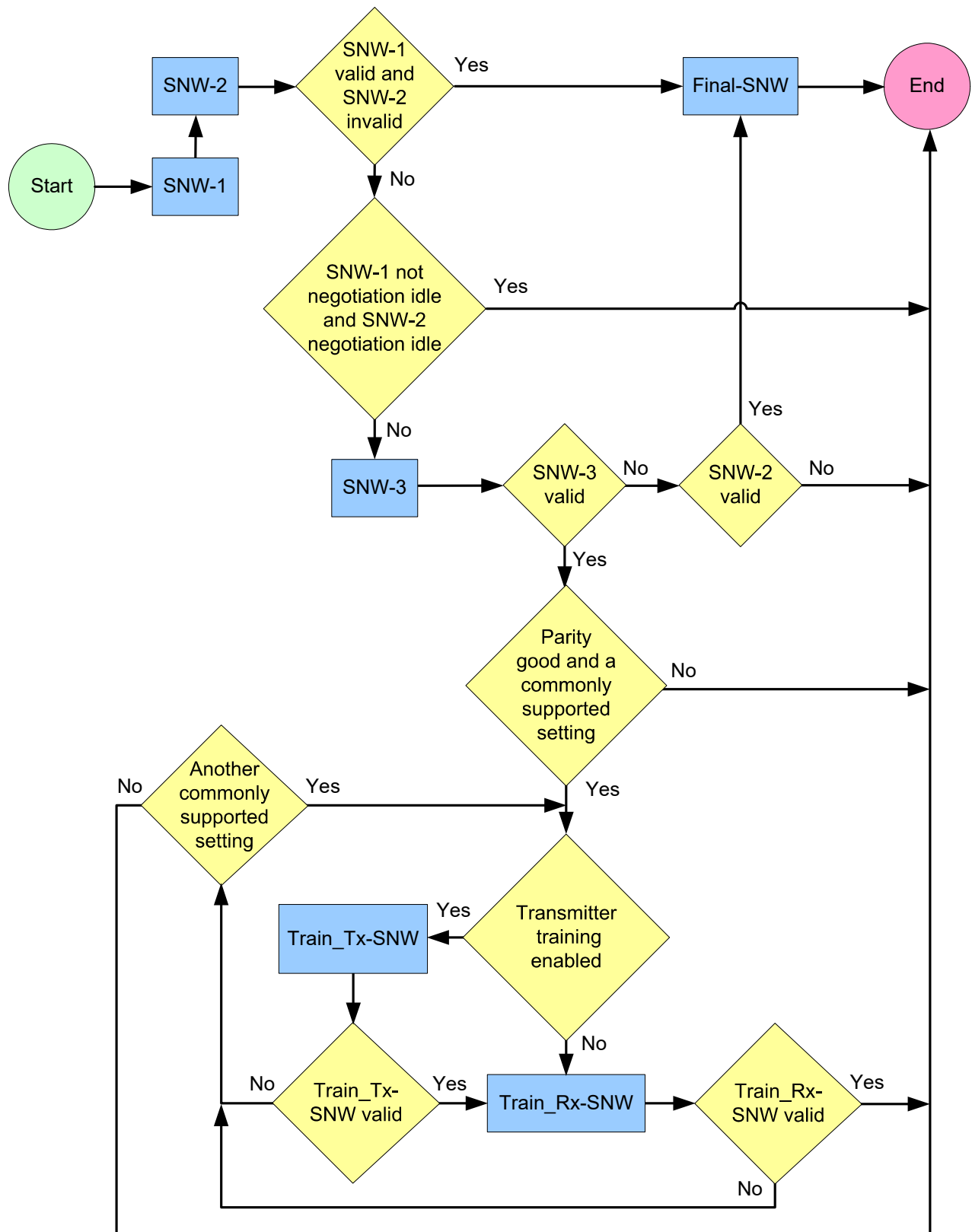


Figure 82 – SAS speed negotiation sequence SNW flowchart

A phy shall not support the following combinations of supported SNWs:

- a) no SNWs; and
- b) SNW-1 and SNW-3 only.

NOTE 12 - If SNW-1 is successful and the combination of SNW-1 and SNW-3 only is used, then the phy is not able to reach SNW-3.

A phy shall detect whether the physical link is negotiation idle during SNW-1 and SNW-2, even if the phy does not support that SNW. If the phy detects:

- a) SNW-1 is not negotiation idle; and
- b) SNW-2 is negotiation idle,

then the phy shall:

- a) end the speed negotiation sequence without progressing to SNW-3 as shown in figure 82; and
- b) transmit negotiation idle and ignore the SNW information received during SNW-3 as defined in the SP state machine (see 5.14.4.3).

NOTE 13 - This avoids causing an attached phy compliant with SAS-1.1 to misdetect a SATA port selector.

Train\_Rx-SNW and Train\_Tx-SNW are based on the highest priority commonly supported setting that has not been tried based on the outgoing and incoming SNW-3 supported settings bits (see 5.11.4.2.3).

If a Train\_Rx-SNW or Train\_Tx-SNW is invalid and there are additional, untried, commonly supported settings exchanged during SNW-3, then a new Train\_Rx-SNW and Train\_Tx-SNW, if any, shall be performed based on the next highest, untried, commonly supported settings.

A phy reset problem occurs:

- a) after Final-SNW, if Final-SNW is invalid (see 5.14.4.9);
- b) after SNW-3, if SNW-3 is valid and the parity is bad (see 5.14.4.11);
- c) during a Train\_Tx-SNW, if the MTTT timer expires or the Pattern Lock Lost Timeout timer expires and there are no additional, untried, commonly supported settings (see 5.14.4.13); or
- d) after a Train\_Rx-SNW, if the Train\_Rx-SNW is invalid and there are no additional, untried, commonly supported settings (see 5.14.4.14).

Phy reset problems terminate the SAS speed negotiation sequence and are counted and reported in the PHY RESET PROBLEM COUNT field in the SMP REPORT PHY ERROR LOG page (see 9.4.3.11) and the Protocol Specific Port log page (see 9.2.8.1).

#### 5.11.4.2.5 SAS speed negotiation sequence examples

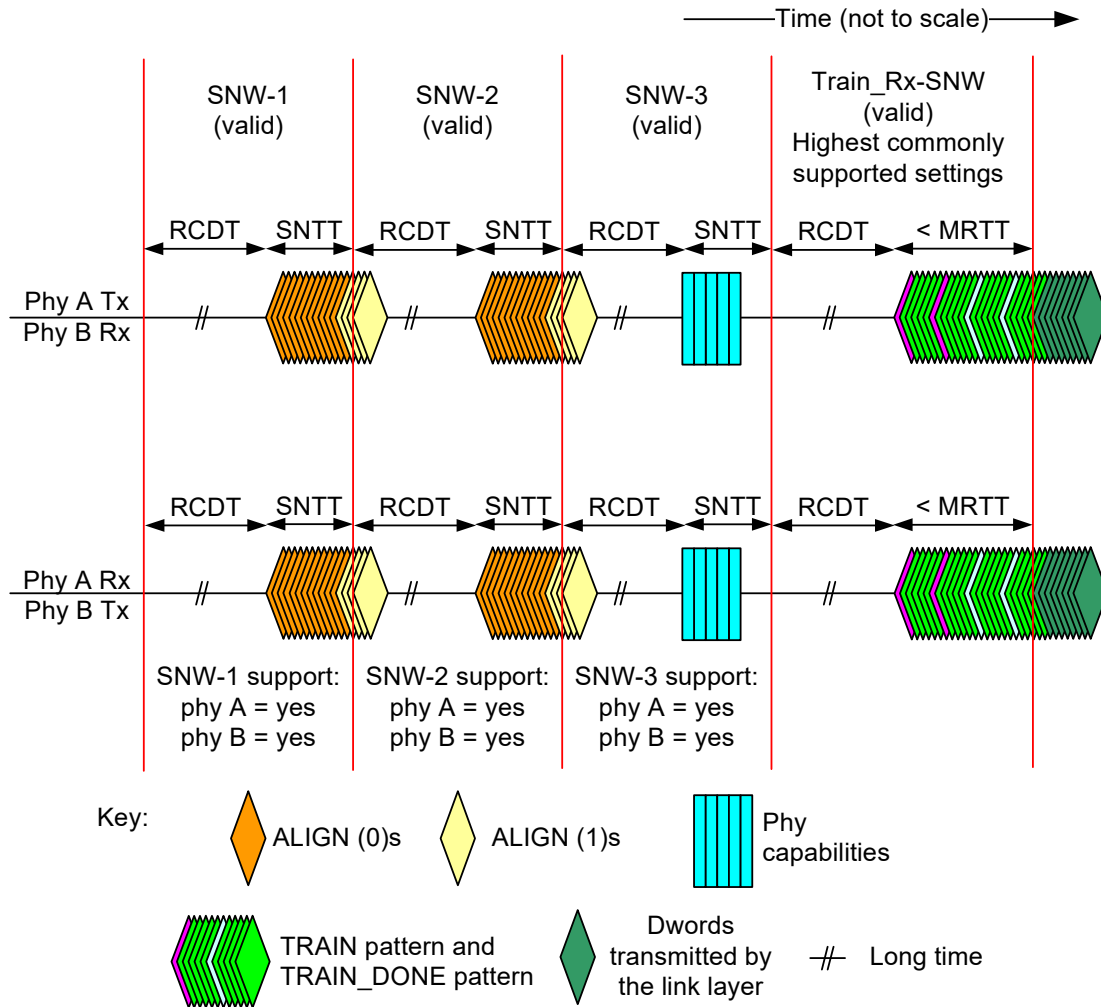
Figure 83 shows speed negotiation between a phy A and a phy B where both phys participate in:

- 1) SNW-1, supported by both phys;
- 2) SNW-2, supported by both phys;
- 3) SNW-3, supported by both phys; and
- 4) Train\_Rx-SNW.

After phy A and phy B detect:

- a) SNW-1 valid;
- b) SNW-2 valid; and
- c) SNW-3 valid,

the phys proceed to Train\_Rx-SNW negotiating based on SNW-3 phy capabilities bits.



**Figure 83 – SAS speed negotiation sequence (both phys SNW-1 through Train\_Rx-SNW with no Train\_Tx-SNW)**

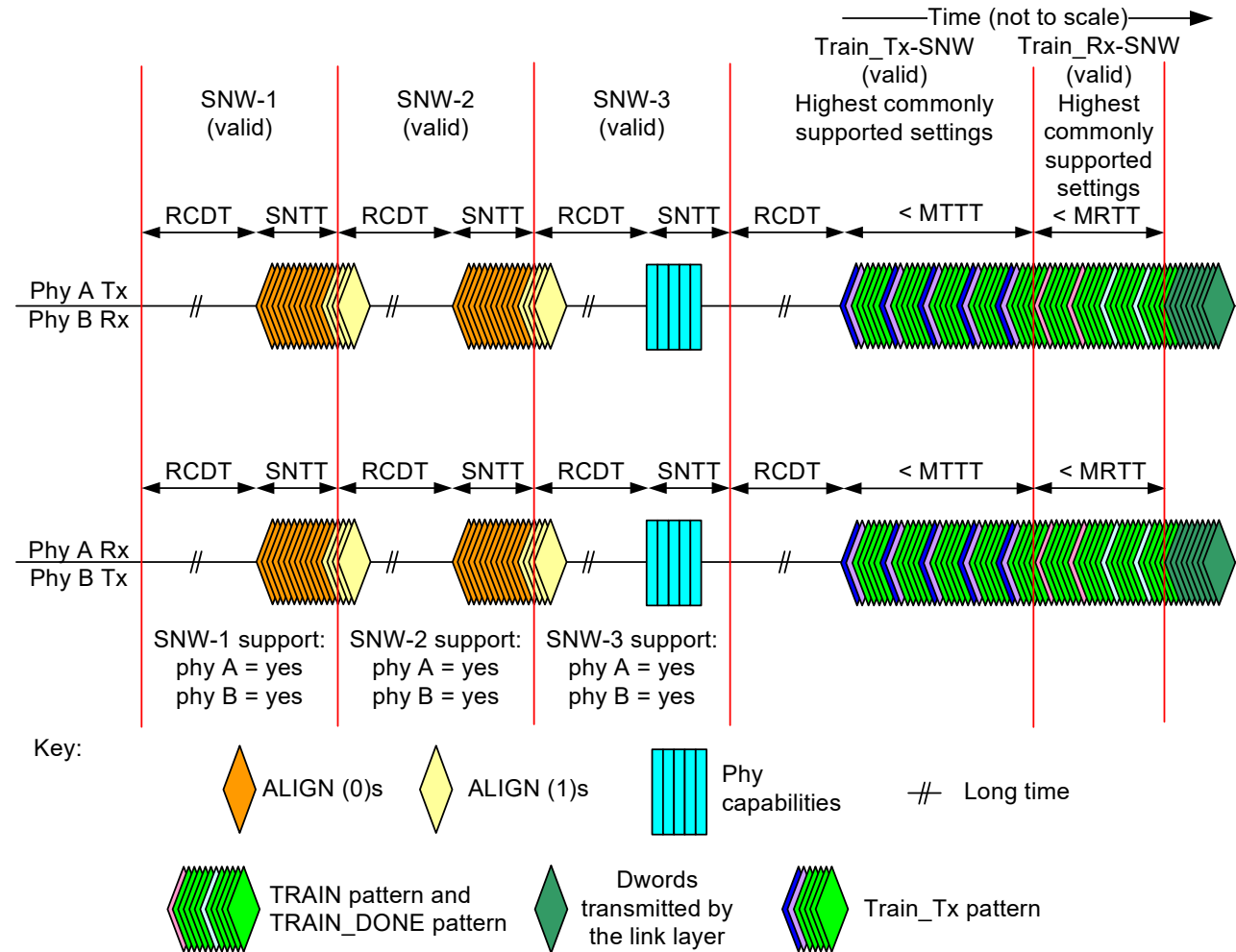
Figure 84 shows speed negotiation between a phy A and a phy B where both phys participate in:

- 1) SNW-1, supported by both phys;
- 2) SNW-2, supported by both phys;
- 3) SNW-3, supported by both phys;
- 4) Train\_Tx-SNW; and
- 5) Train\_Rx-SNW.

After phy A and phy B detect:

- a) SNW-1 valid;
- b) SNW-2 valid; and
- c) SNW-3 valid,

the phys proceed to Train\_Tx-SNW and Train\_Rx-SNW based on SNW-3 phy capabilities bits.



**Figure 84 – SAS speed negotiation sequence (both phys SNW-1 through Train\_Rx-SNW with Train\_Tx-SNW)**

Figure 85 shows speed negotiation between a phy A and phy B where phys participate in:

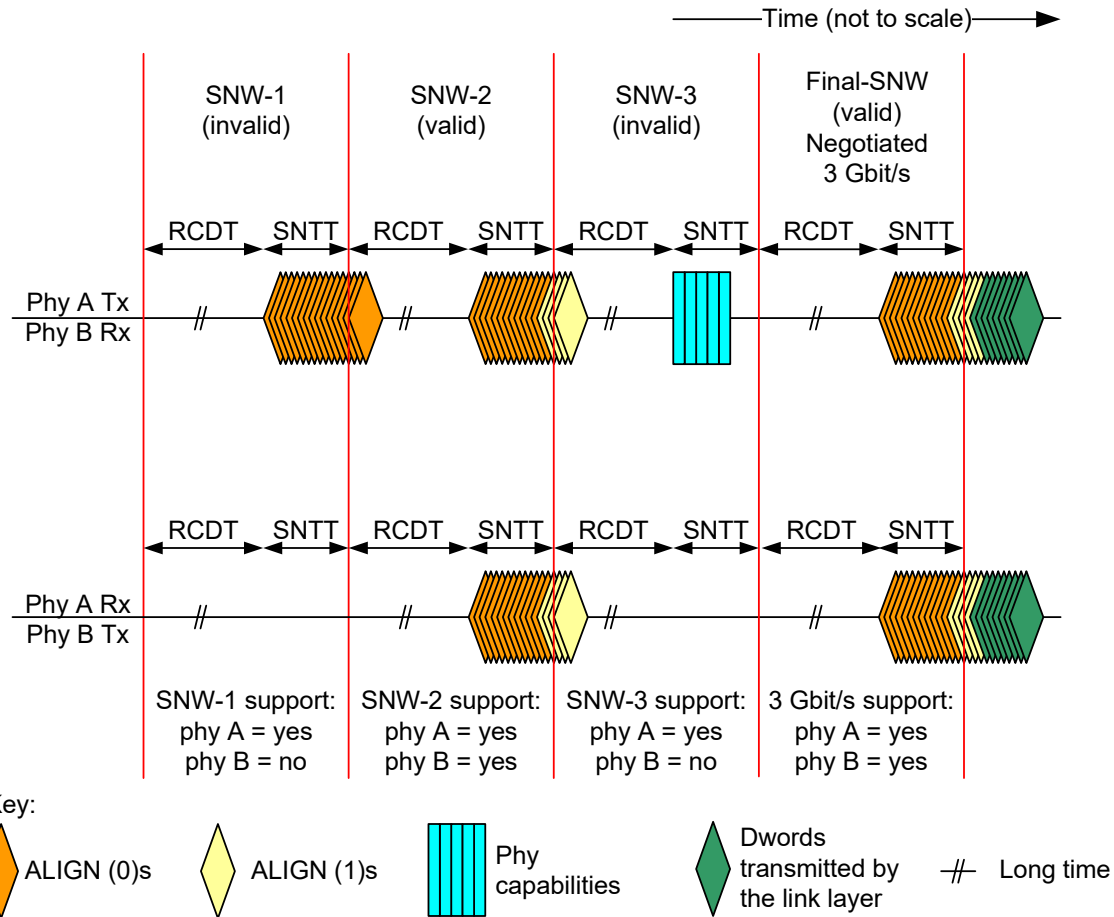
- 1) SNW-1, supported by phy A but not by phy B;
- 2) SNW-2, supported by both phys;
- 3) SNW-3, supported by phy A but not by phy B; and
- 4) Final-SNW negotiating 3 Gbit/s.

After phy A and phy B detect:

- a) SNW-1 invalid;
- b) SNW-2 valid; and
- c) SNW-3 invalid,

the phys proceed to Final-SNW negotiating 3 Gbit/s.





**Figure 85 – SAS speed negotiation sequence (phy A: SNW-1 through SNW-3, phy B: SNW-2 only)**

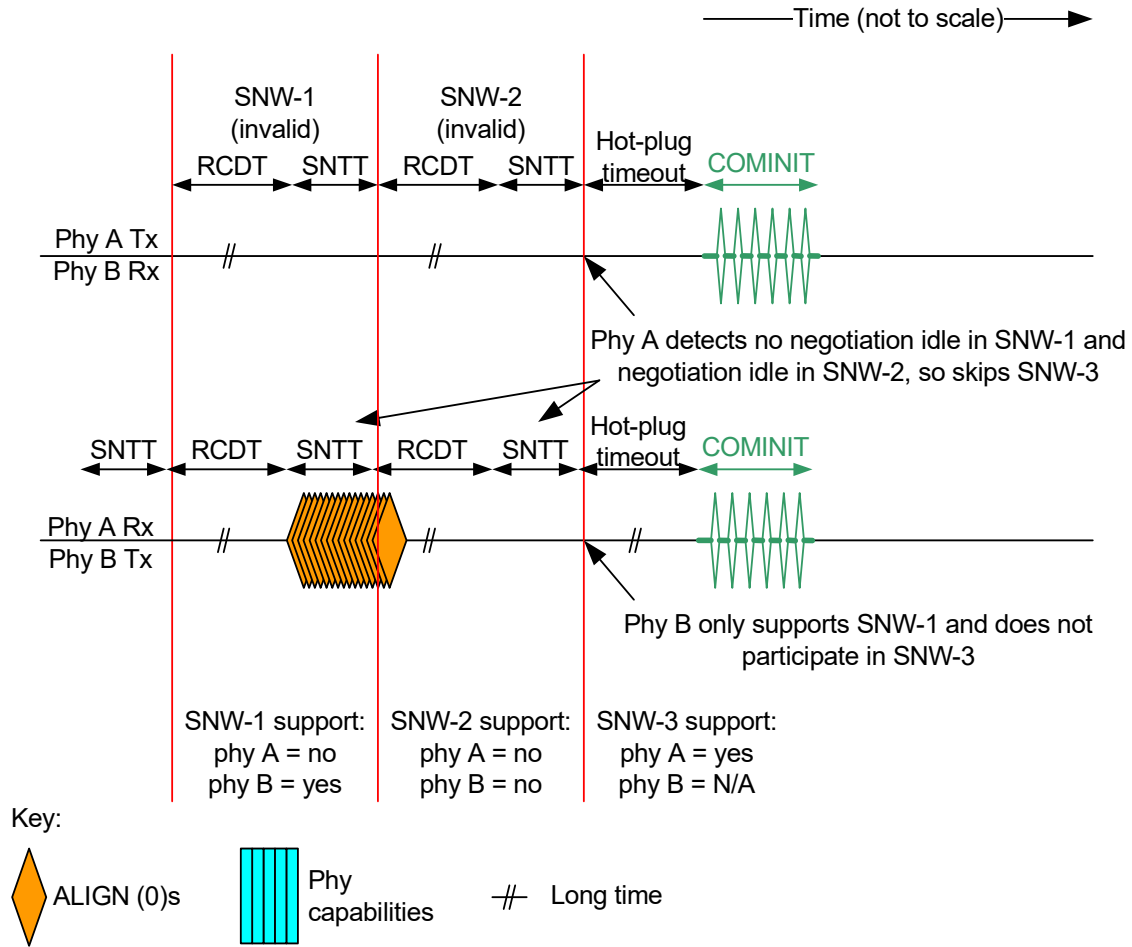
Figure 86 shows speed negotiation between a phy A and phy B where the phys participate in:

- 1) SNW-1, supported by phy B but not by phy A; and
- 2) SNW-2, supported by neither phy.

After phy A and phy B detect:

- a) SNW-1 invalid; and
- b) SNW-2 invalid,

phy A detects SNW-3 invalid.



**Figure 86 – SAS speed negotiation sequence (phy A: SNW-3 only, phy B: SNW-1 only)**

Figure 87 shows a speed negotiation sequence where phy B does not achieve dword synchronization during Final-SNW, creating a phy reset problem. If this occurs, then the handshake is not complete and the phy reset sequence is retried.

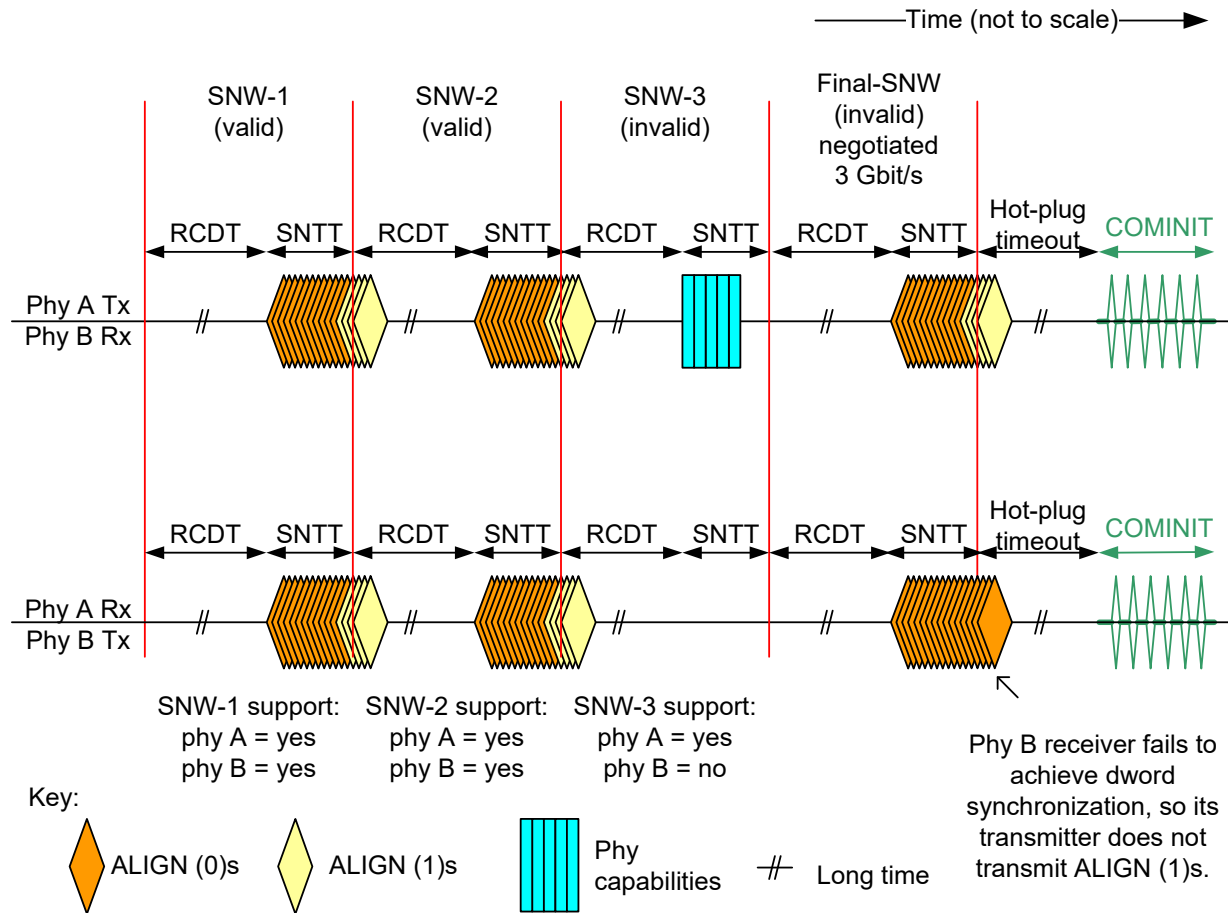


Figure 87 – SAS speed negotiation sequence - phy reset problem in Final-SNW

Figure 88 shows a speed negotiation sequence in which a phy reset problem is encountered in SNW-3 because the phys do not exchange the phy capabilities bits properly (e.g., due to a parity error).

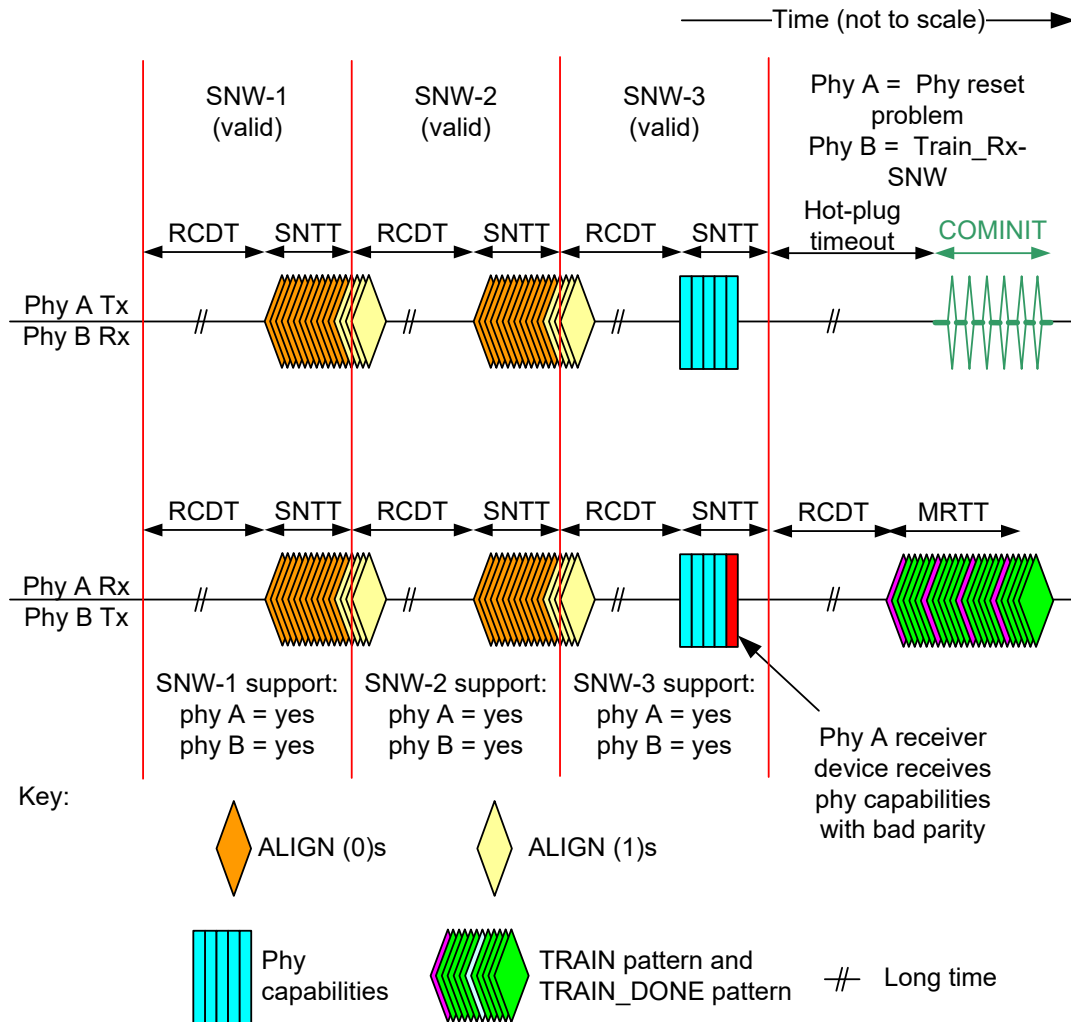


Figure 88 – SAS speed negotiation sequence - phy reset problem in SNW-3

Figure 89 shows a speed negotiation sequence in which a phy reset problem is encountered in Train\_Rx-SNW because either phy does not complete training within MRTT. This example assumes that only one commonly supported setting is exchanged in the phy capabilities bits.

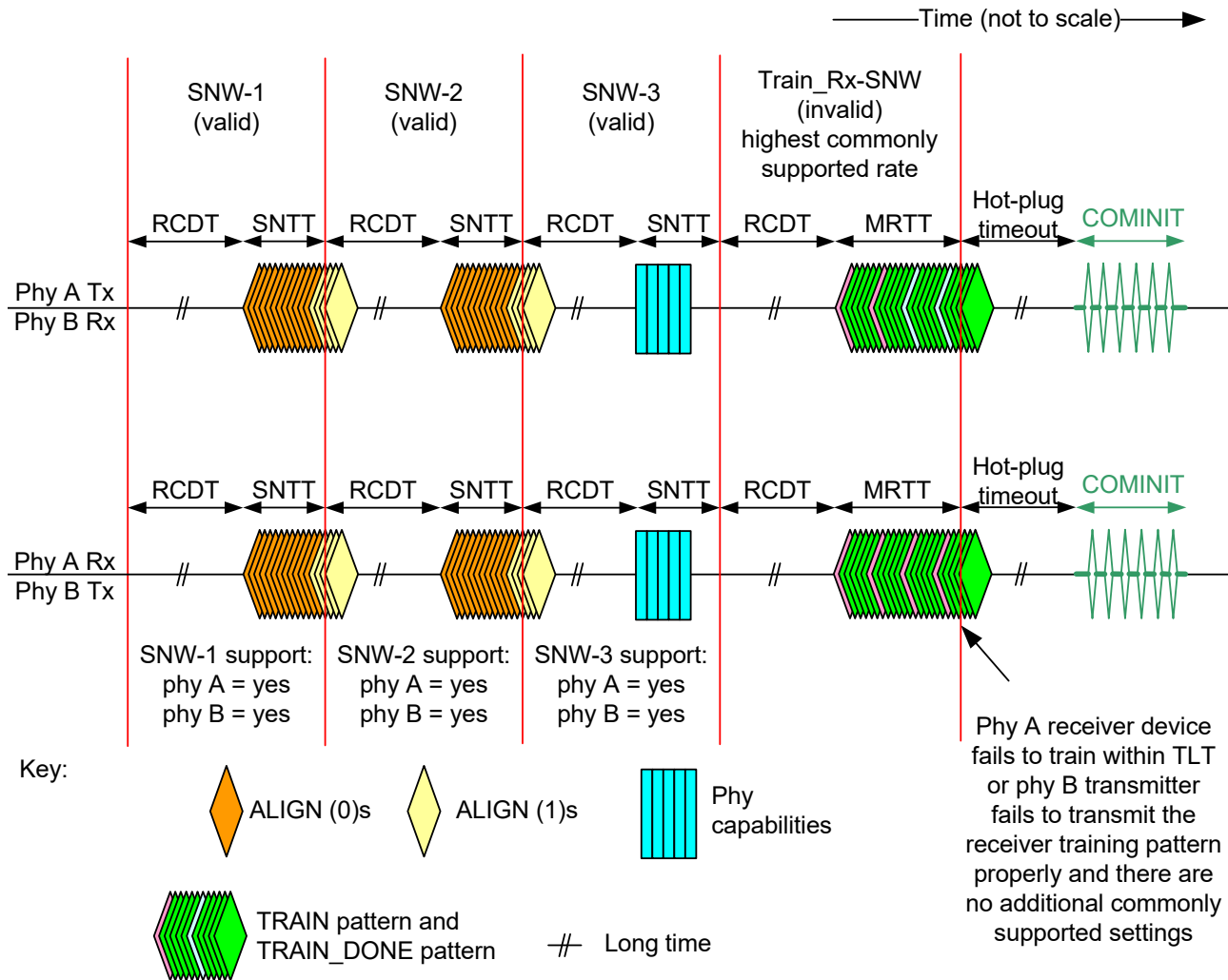
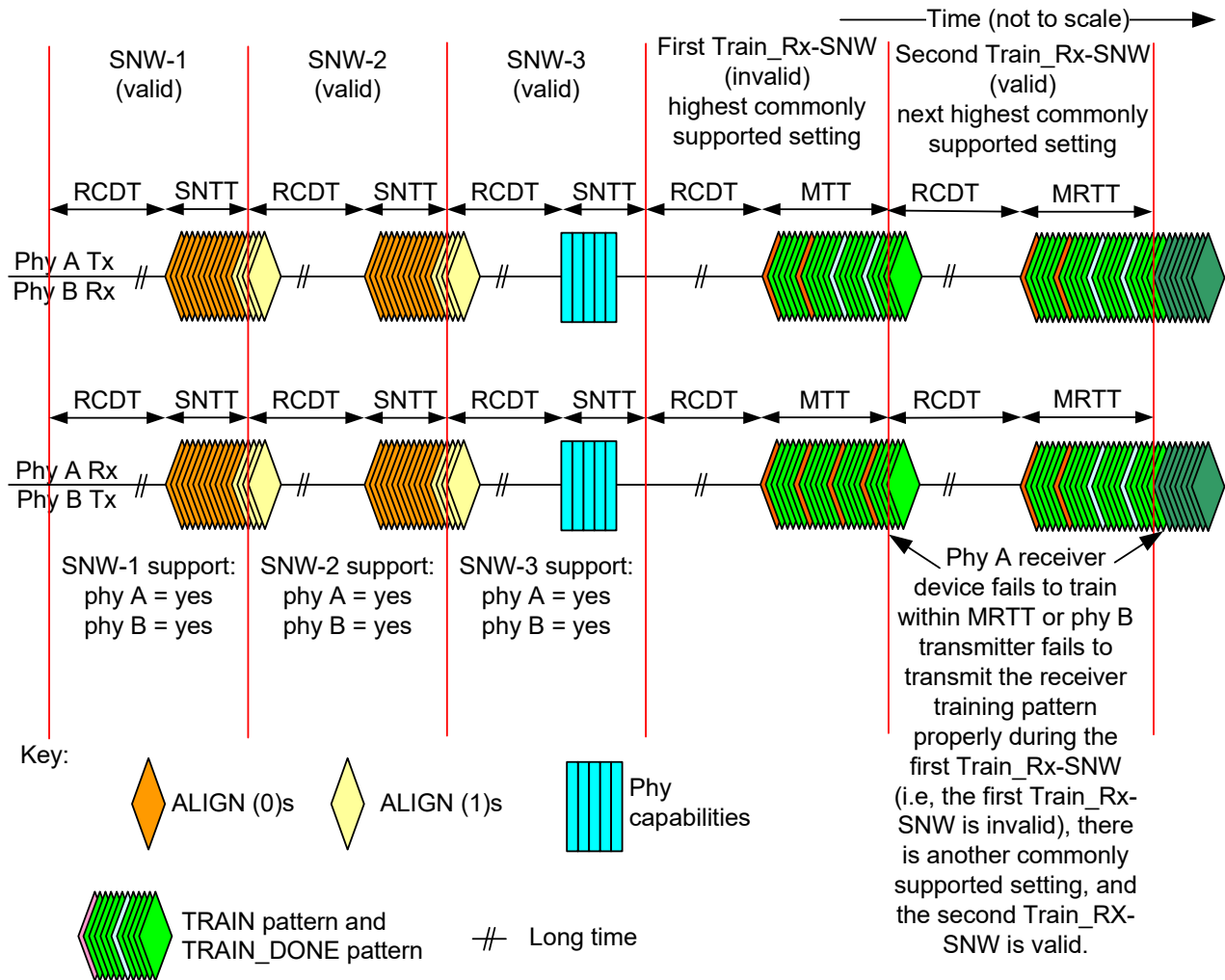


Figure 89 – SAS speed negotiation sequence - phy reset problem in Train\_Rx-SNW

Figure 90 shows two Train\_Rx-SNWs, where supported settings bits are exchanged that contain more than one commonly supported setting and the Train\_Rx-SNW using the highest commonly supported setting is invalid, so a second Train\_Rx-SNW is performed using the next highest commonly supported setting.



**Figure 90 – SAS speed negotiation sequence - multiple Train\_Rx-SNWs**

For more examples of SAS speed negotiations, see Annex B.

#### 5.11.4.2.6 Train\_Tx pattern sequence

##### 5.11.4.2.6.1 Train\_Tx pattern sequence overview

If SNW-3 indicates commonly supported settings that require transmitter training (see table 73), then the local phy and the attached phy shall exchange a sequence of Train\_Tx patterns that cause each phy's transmitter to be trained. The local phy and the attached phy exchange information in TTUUs that:

- indicate the current status of the local phy's transmitter; and
- specify adjustments to the attached phy's transmitter.

The sequencing of adjustments to the transmitter is defined in the PTT state machines (see 5.18).

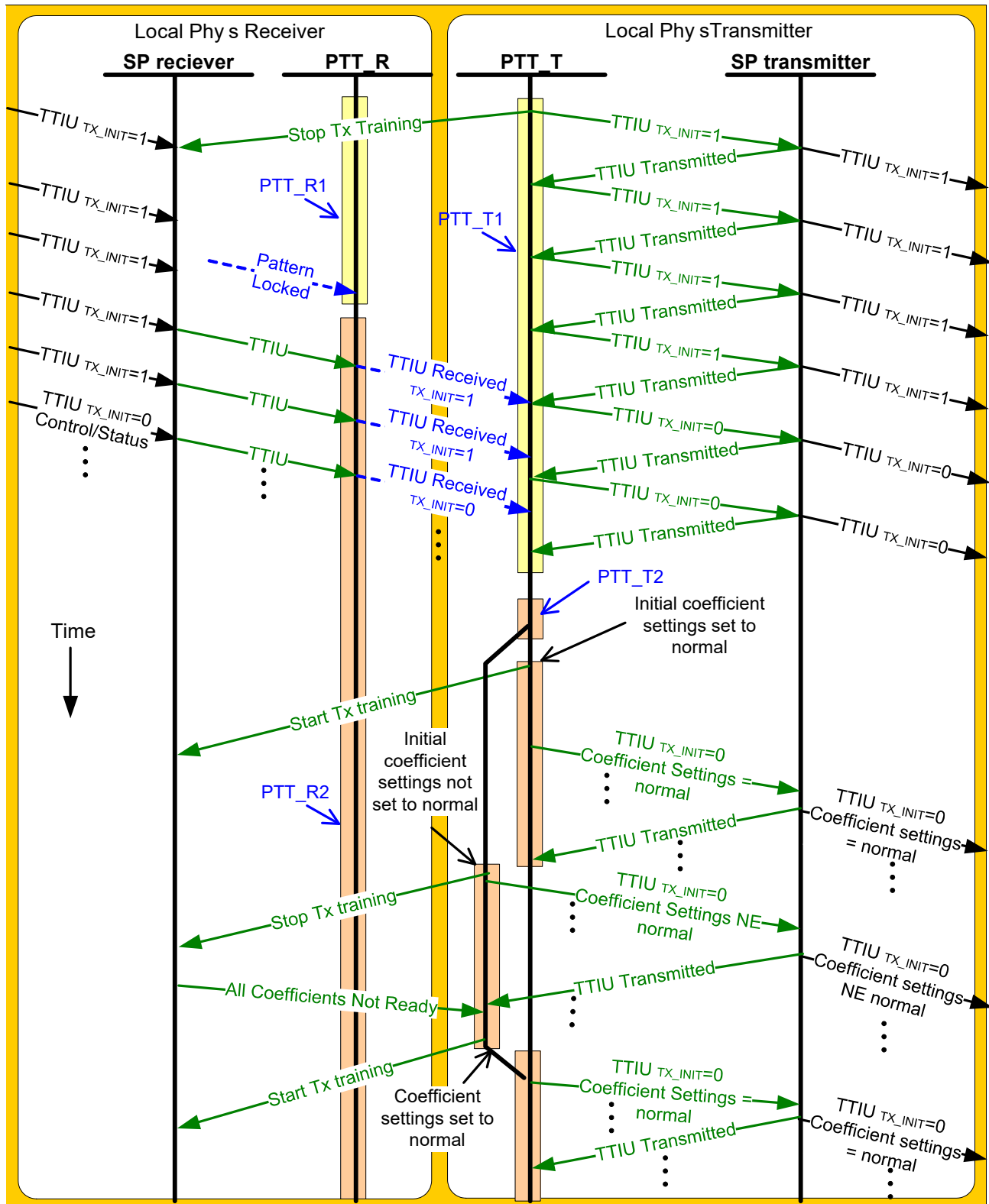
#### 5.11.4.2.6.2 Train\_Tx pattern initial sequence

After RCDT during a Train\_Tx-SNW, (see 5.11.4.2.3.4) the local phy and the attached phy both begin transmitting and receiving Train\_Tx patterns. The point at which a transmitter training pattern lock occurs at the local phy's receiver may occur before or after a transmitter training pattern lock occurs at the attached phy's receiver.

The PTT\_T state machine (see 5.18.4.1) is informed:

- a) that the local phy's receiver is receiving TTIs when the PTT\_T1 state (see 5.18.4.3) receives a TTIU Received message from the PTT\_R2 state (see 5.18.5.4); and
- b) that the attached phy's receiver is receiving TTIs when the PTT\_T1 state receives a TTIU Received (Attached Transmitter Initialized) message from the PTT\_R2 state.

Figure 91 shows an example of an initial sequence of a local phy in which pattern lock occurs before the attached phy achieves pattern lock.



Key: Start Tx Training = Attached Physical Transmitter Training (Start) message

Stop Tx Training = Attached Physical Transmitter Training (Stop) message

Processing PTT state name  $\longrightarrow$

Information sent or received from a SP receiver or SP transmitter 

**Figure 91 – Local phy achieves pattern lock before the attached phy achieves pattern lock**





### 5.11.4.2.6.3 Train\_Tx pattern handshake sequence

#### 5.11.4.2.6.3.1 Train\_Tx pattern handshake sequence overview

During a Train\_Tx-SNW (see 5.11.4.2.3.4) after the local phy and the attached phy have both completed initialization (i.e., the local phy has received valid TTIs and verified that the attached phy is receiving valid TTIs) the attached phy's receiver begins training the local phy's transmitter.

The attached phy's receiver may request that the local phy's transmitter coefficients be:

- a) held at the current value;
- b) set to a no\_equalization value;
- c) set to a reference\_1 value;
- d) set to a reference\_2 value;
- e) incremented (see table 81); or
- f) decremented (see table 81).

#### 5.11.4.2.6.3.2 Attached phy's receiver increment or decrement request

The attached phy's receiver only requests changes to a local phy's transmitter coefficient if the local phy's transmitter is indicating a status of ready on all the local phy's transmitter coefficients (see 5.18.3).

The local phy's transmitter responds to the attached phy's receiver request by adjusting the specified coefficient as requested. After the specified coefficient is adjusted the local phy's transmitter indicates the completion of the coefficient adjustment by indicating to the attached phy's receiver a status of:

- a) update complete;
- b) minimum; or
- c) maximum.

After the attached phy's receiver requests one or more of the local phy's transmitter coefficients be updated, the attached phy does not attempt to do analysis of a training pattern until after it has received a status of update complete, maximum, or minimum for all the local phy's transmitter coefficients that were updated (see 5.18.3).

The processing of the attached phy's receiver coefficient adjustment requests is handled by the:

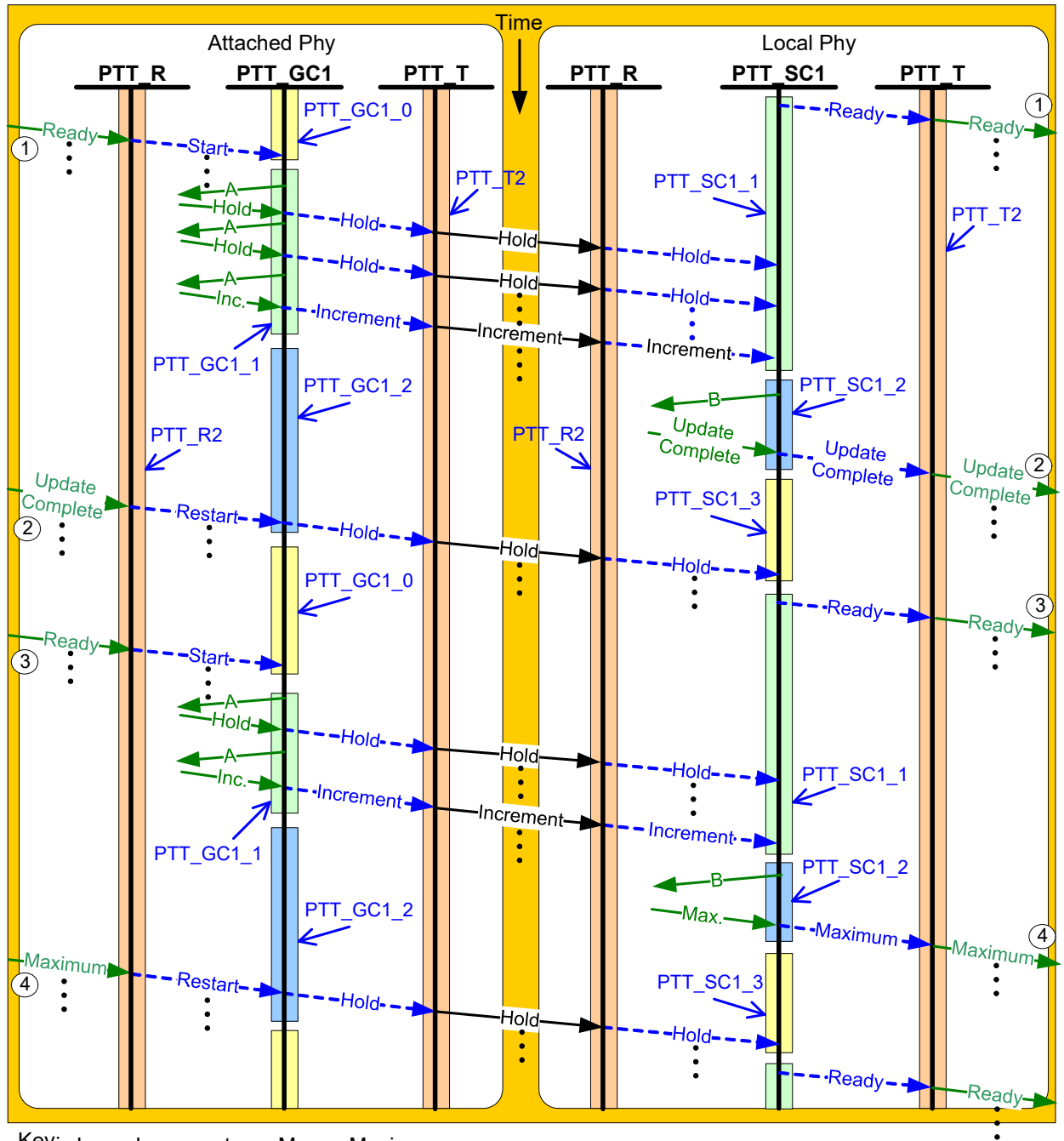
- a) PTT\_GC1 state machine (see 5.18.9);
- b) PTT\_GC2 state machine (see 5.18.10); and
- c) PTT\_GC3 state machine (see 5.18.11).

The processing of the local phy's transmitter coefficient adjustments is handled by the:

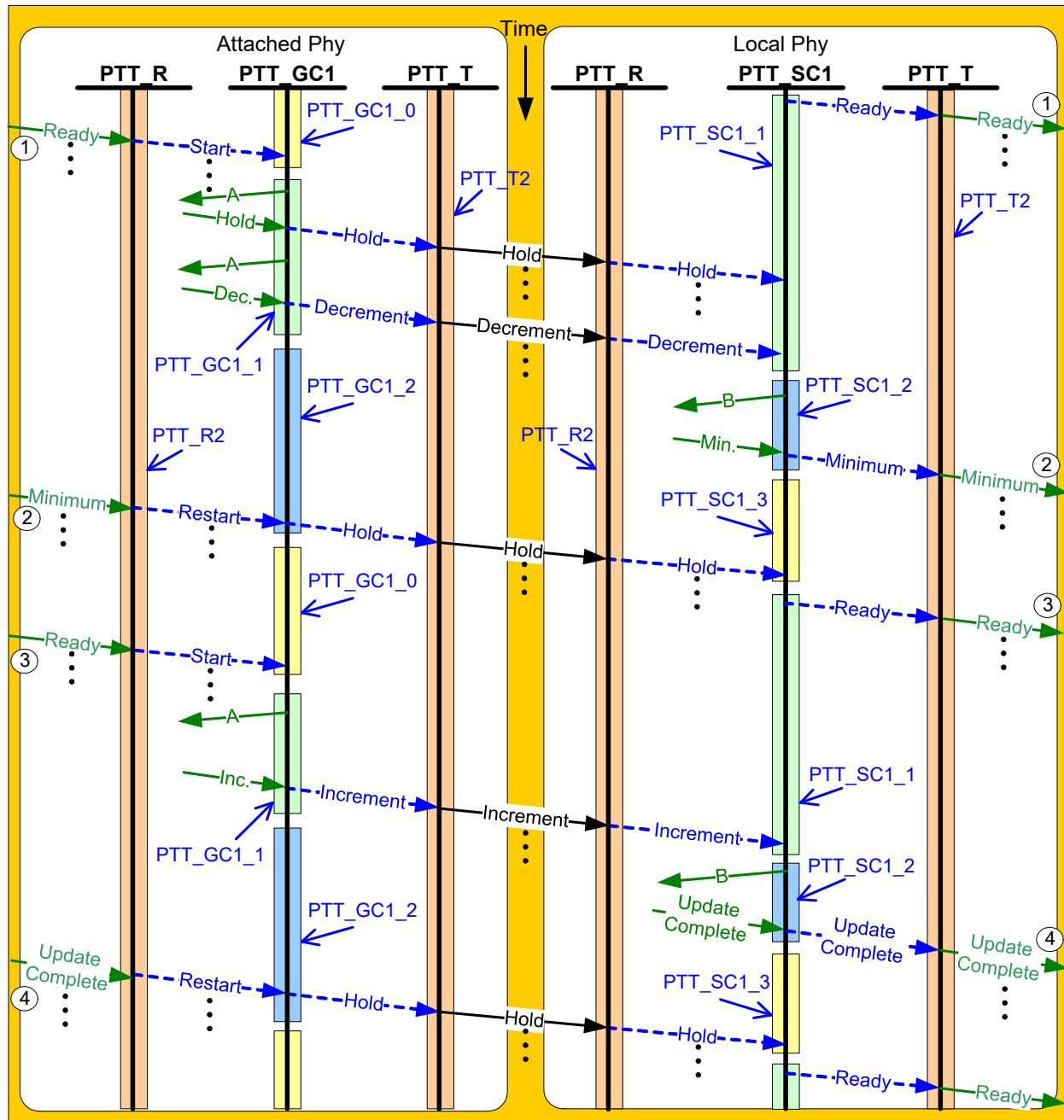
- a) PTT\_SC1 state machine (see 5.18.6);
- b) PTT\_SC2 state machine (see 5.18.7); and
- c) PTT\_SC3 state machine (see 5.18.8).

Figure 93 shows an example of a handshake sequence in which the attached phy's receiver requests two increments to coefficient 1 where the response to the first increment is an update complete and the response to the second increment is a maximum.

Figure 94 shows an example of a handshake sequence in which the attached phy's receiver requests a decrement to coefficient 1 where the response to the decrement is a minimum followed by a request for an increment where the response to the increment is an update complete.



**Figure 93 – Attached receiver handshake sequence (requesting two increments to coefficient 1)**



Key: Inc. = Increment    Dec. = Decrement    Max. = Maximum

—A—> Get Current Coefficient 1 message to local SP receiver

—B—> Adjust Coefficient 1 message to local SP transmitter

Processing PTT state name —>

Information sent or received from a SP receiver or SP transmitter —>

Note 1 - Not shown in this figure is the attached phy's receiver PTT\_R state machine sending coefficient 1 status (i.e., ready, update complete, minimum, or maximum) to the attached SP receiver.

Note 2 - Not shown in this figure is the attached SP receiver or local SP receiver.

Note 3 - Not shown in this figure is the attached SP transmitter or local SP transmitter.

**Figure 94 – Attached receiver handshake sequence (requesting one decrement and one increment to coefficient 1)**

#### 5.11.4.2.6.3.3 Attached phy's receiver reference\_1, reference\_2, or no\_equalization request

The attached phy's receiver only requests a local phy's transmitter coefficients be set to a reference\_1 value, reference\_2 value, or no\_equalization value after the local phy's transmitter indicates a status of ready for all the local phy's transmitter coefficients (see 5.18.3).

The local phy's transmitter responds to the attached phy's receiver request by adjusting all the coefficients to the reference\_1 value, reference\_2 value, or no\_equalization value. As each coefficient is adjusted the local phy's transmitter indicates the completion of the coefficient adjustment by indicating to the attached phy's receiver a status of:

- a) update complete;
- b) minimum; or
- c) maximum.

After the attached phy's receiver requests a local phy's transmitter be set to its reference\_1 value, reference\_2 value, or no\_equalization value the attached phy does not attempt to start the analysis of a training pattern until it has received, for all the coefficients (see 5.18.3), a status of:

- a) update complete;
- b) minimum; or
- c) maximum.

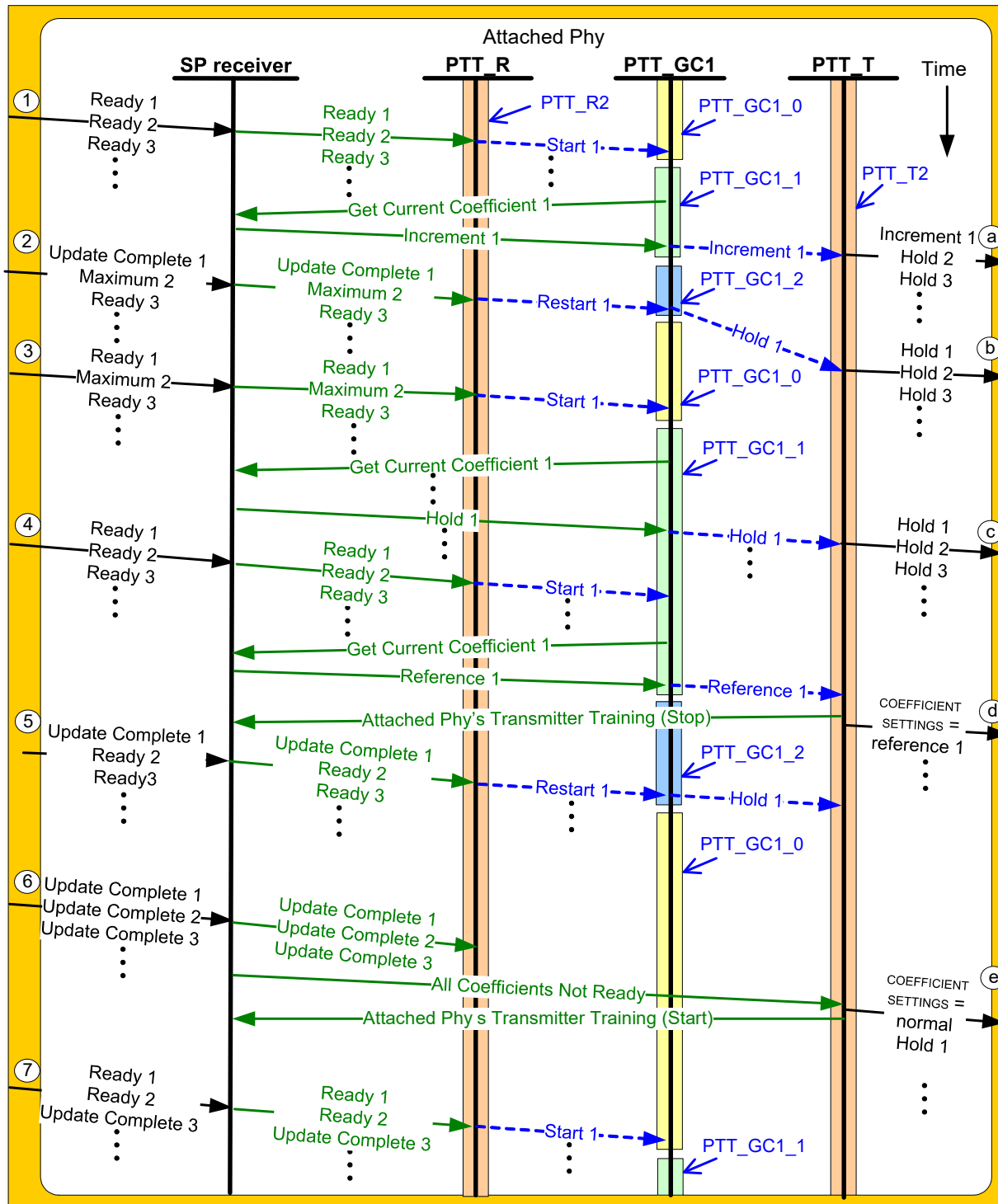
The processing of the attached phy's receiver coefficient setting requests is handled by the:

- a) PTT\_GC1 state machine (see 5.18.9);
- b) PTT\_GC2 state machine (see 5.18.10); and
- c) PTT\_GC3 state machine (see 5.18.11).

The processing of the local phy's transmitter coefficient setting is handled by the:

- a) PTT\_SC1 state machine (see 5.18.6);
- b) PTT\_SC2 state machine (see 5.18.7); and
- c) PTT\_SC3 state machine (see 5.18.8).

Figure 95 shows an example of an attached phy's side and figure 96 shows an example of a local phy's side of a handshake sequence in which the attached phy's receiver requests an increment to coefficient 1 followed by a request from the attached phy's receiver to set the local phy's transmitter to its no equalization value. The time sequencing between figure 95 and figure 96 is indicated using circled numbers and circled letters.



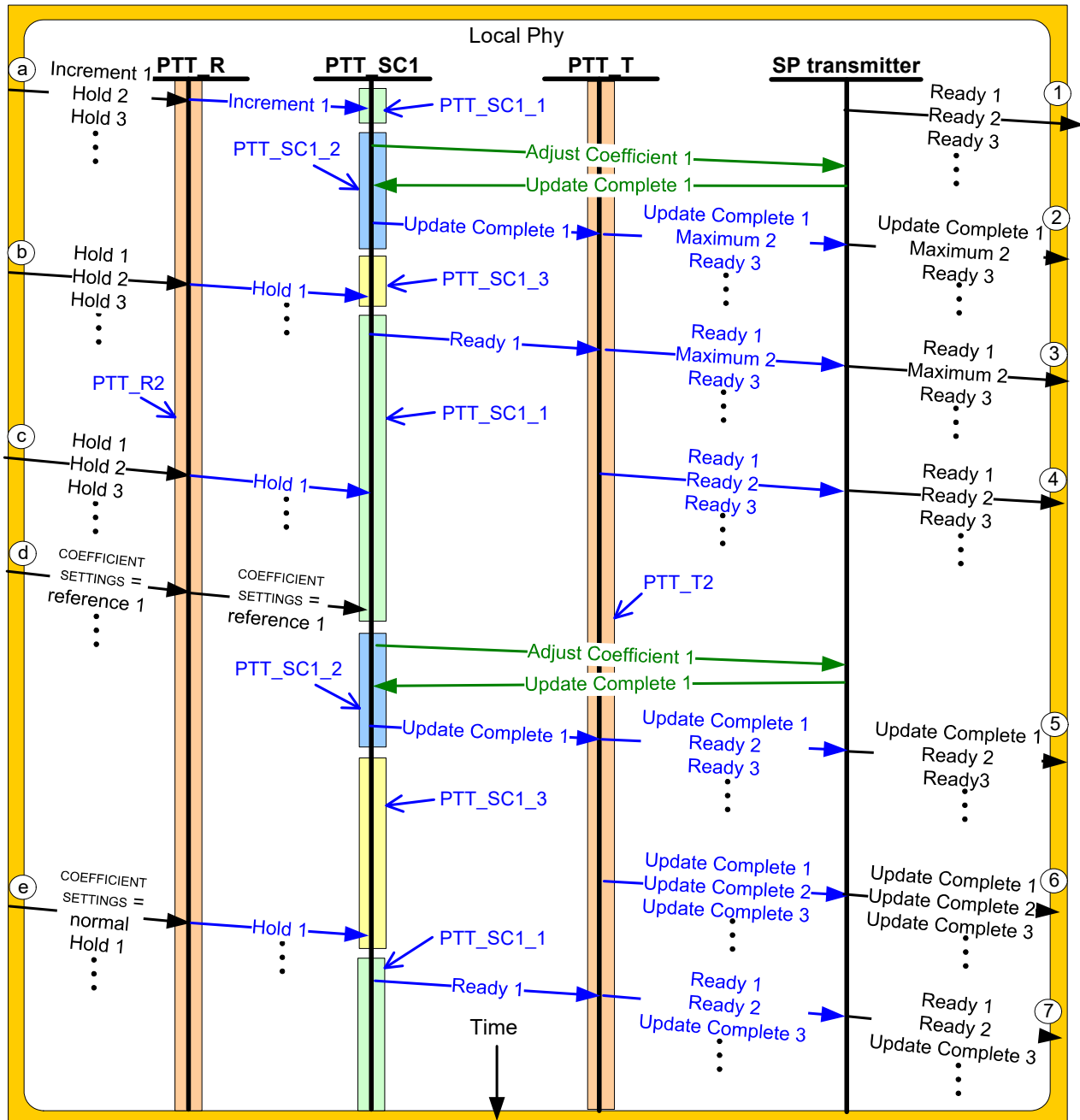
Key: Processing PTT state name →

Information sent or received from a SP receiver or SP transmitter →

Note 1 - Not shown in this figure is the attached phy's receiver PTT\_R state machine sending coefficient 1 status (i.e., ready, update complete, minimum, or maximum) to the attached SP receiver.

Note 2 - Not shown in this figure are all the messages related to the coefficient 2 increment.

**Figure 95 – Handshake sequence to set local phy's receiver coefficients to no\_equalization values (attached phy)**



**Figure 96 – Handshake sequence to set local phy's receiver coefficients to no\_equalization values (local phy)**

#### 5.11.4.2.6.4 Train\_Tx pattern completion sequence

The Train\_Tx-SNW (see 5.11.4.2.3.4) completes when:

- the local phy's receiver determines attached phy's transmitter is trained (see 5.18.3); and
- the attached phy's receiver indicates the local phy's transmitter is trained (see 5.18.5.4.1).

Figure 97 shows an example of a local phy's receiver completion of training before the attached phy's receiver indicates it has completed training.

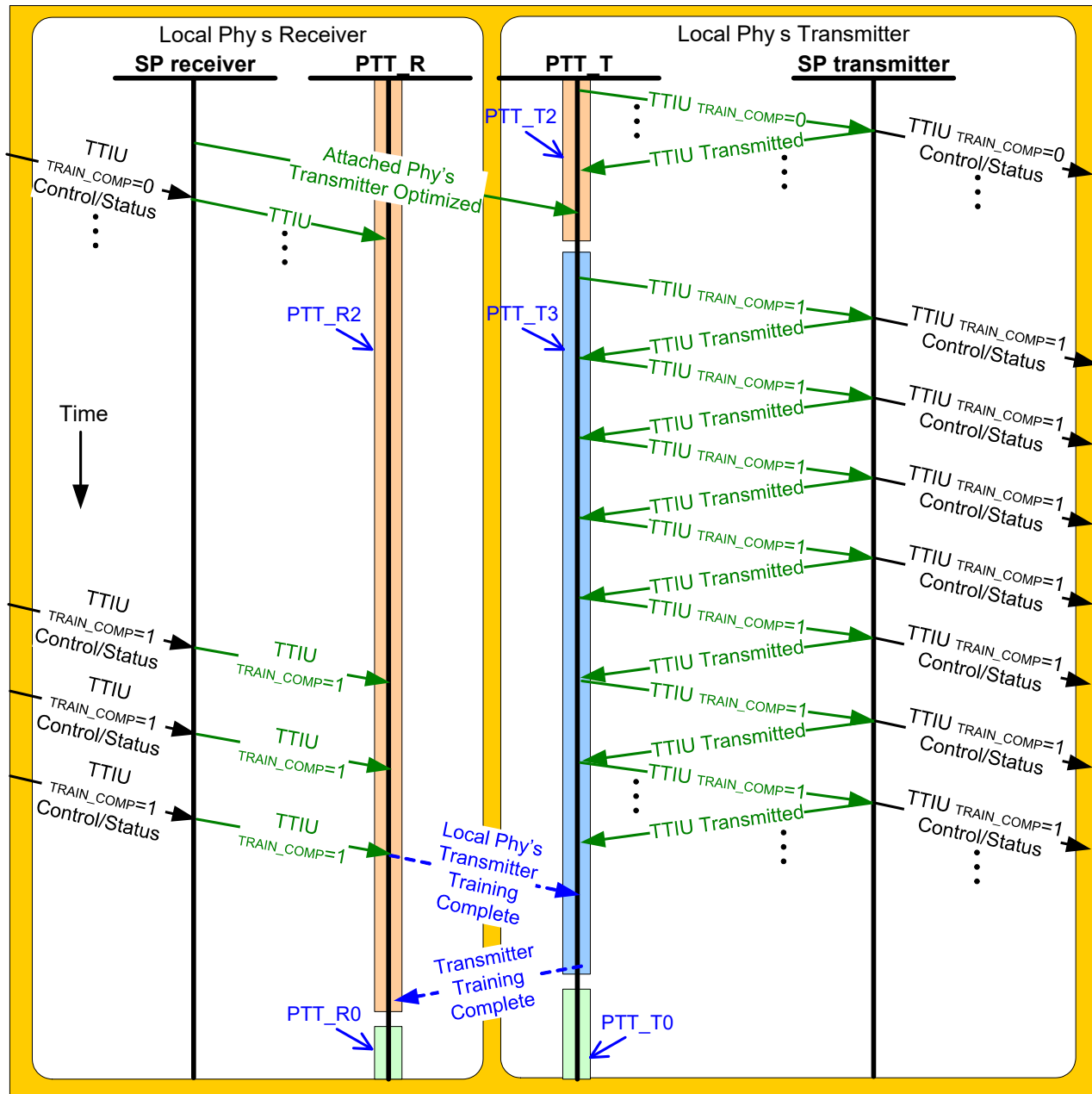


Figure 97 – Local phy's receiver indicates completion of training before the attached phy's receiver completes training



Figure 98 shows an example of an attached phy's receiver indication of completion of training before the local phy's receiver completes training.

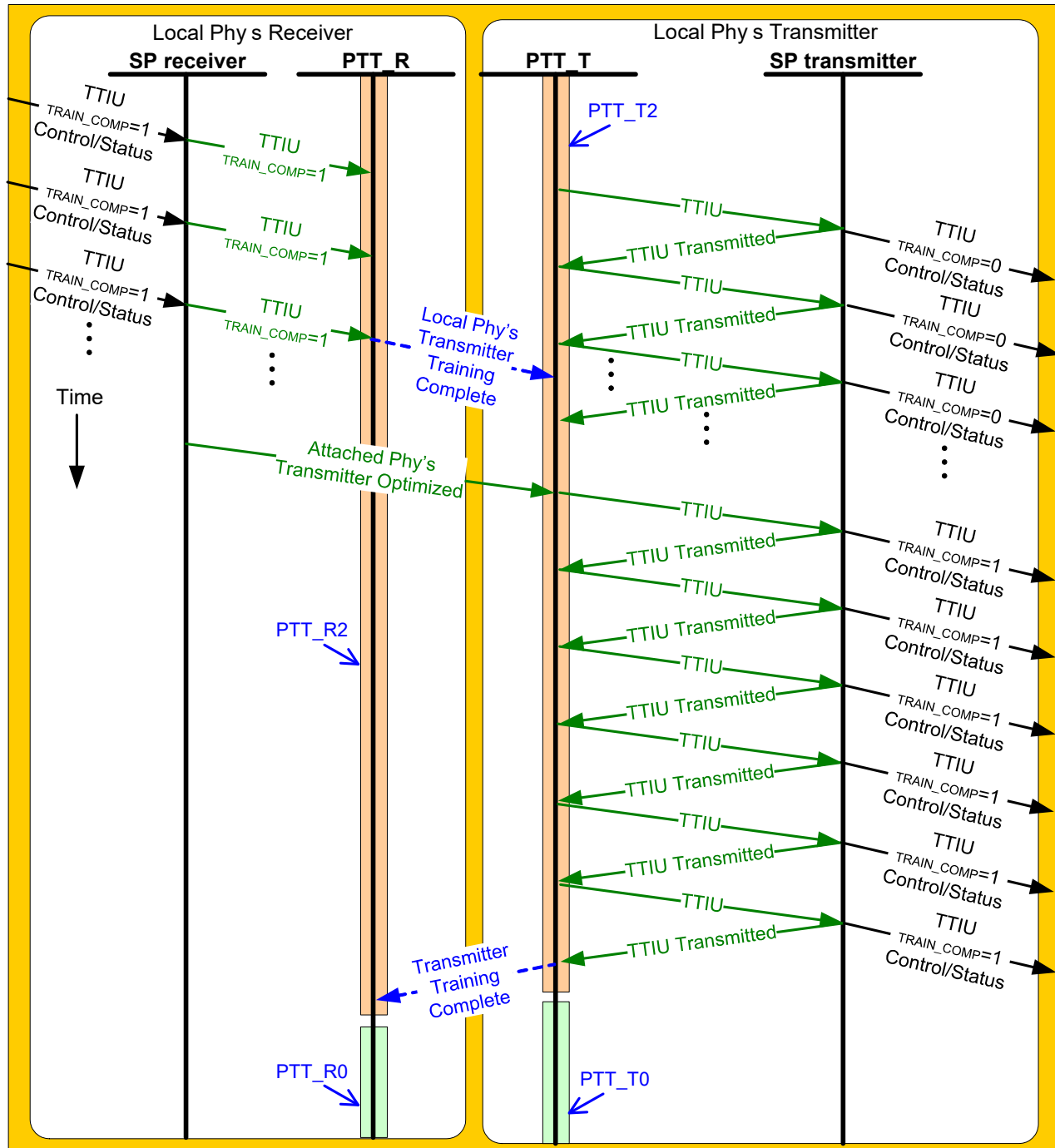


Figure 98 – Attached phy's receiver indicates completion of training before the local phy's receiver completes training

**5.11.4.2.6.5 Invalid TTIU sequence**

If the PTT\_R state machine (see 5.18.5.4.1) detects an unsupported or reserved value in a TTIU, then the PTT\_R state machine sends a:

- a) Transmit Error Response message to the PTT\_T state machine; and
- b) Cancel message to the PTT\_SC1 state machine, PTT\_SC2 state machine, and PTT\_SC3 state machine.

If the PTT\_T2 state (see 5.18.4.4.4) or PTT\_T3 state (see 5.18.4.5.5) receives a Transmit Error Response message, then that state:

- 1) builds an Error Response TTIU;
- 2) transmits an Error Response TTIU; and
- 3) transmits a Control/Status TTIU.

If the PTT\_R2 state machine (see 5.18.5.4.1) receives an Error Response TTIU, then the PTT\_R state machine sends a:

- a) Transmitter Control Failed message to the SP receiver; and
- b) Cancel message to the PTT\_GC1 state machine, PTT\_GC2 state machine, and PTT\_GC3 state machine.

The diagram illustrates the timing of PTT signaling between an Attached PHY and a Local PHY. The central vertical axis represents Time. The Attached PHY has PTT\_R and PTT\_T lines, while the Local PHY has PTT\_R and PTT\_T lines. The sequence of events is as follows:

- Control/Status TTU (valid)** is sent from the Attached PHY PTT\_T to the Local PHY PTT\_R.
- Error Response TTU** is sent from the Local PHY PTT\_T to the Attached PHY PTT\_R.
- Control/Status TTU (valid)** is sent from the Local PHY PTT\_T to the Local PHY PTT\_R.

Intermediate messages and delays are also shown:

- Control/Status TTU (invalid)** and **Invalid Control/Status TTU discarded** are sent from the Attached PHY PTT\_T to the Local PHY PTT\_R.
- Transmit Error Response Cancel** (dashed blue line) is sent from the Local PHY PTT\_T to the Local PHY PTT\_R.
- Transmit Error Response** (dashed blue line) is sent from the Local PHY PTT\_T to the Local PHY PTT\_R.
- Invalid Control/Status TTU discarded** is sent from the Local PHY PTT\_T to the Local PHY PTT\_R.
- Control/Status TTU (valid)** is sent from the Local PHY PTT\_T to the Local PHY PTT\_R.

Delays  $PTT_{R2}$  and  $PTT_{T2}$  are indicated for the PTT\_R and PTT\_T lines respectively. The diagram also shows **Control/Status TTU (valid)** and **Error Response TTU** messages between the Attached PHY and the Local PHY.

<sup>b</sup> Cancel message to PTT\_SC1 state machine, PTT\_SC2 state machine, and PTT\_SC3 state machine.

### Figure 99 – Processing an invalid TTIU

#### 5.11.4.3 Multiplexing sequence

The multiplexing sequence is:

- 1) MUX (LOGICAL LINK 0);
- 2) MUX (LOGICAL LINK 1);

- 3) MUX (LOGICAL LINK 0);
- 4) MUX (LOGICAL LINK 1);
- 5) MUX (LOGICAL LINK 0); and
- 6) MUX (LOGICAL LINK 1).

The phy shall not transmit deletable primitives for physical link rate tolerance management (see 6.5) during the multiplexing sequence.

If SNW-3 indicates multiplexing is not enabled, then the phy shall not transmit the multiplexing sequence.

The phy shall assign the incoming logical links to its logical phys based on the first MUX it receives:

- a) MUX (LOGICAL LINK 0) indicates the position of logical link 0 and indicates the next dword is in logical link 1; or
- b) MUX (LOGICAL LINK 1) indicates the position of logical link 1 and indicates the next dword is in logical link 0.

The phy shall handle errors during the multiplexing sequence (i.e., after receiving the first MUX) as follows:

- a) if the phy loses dword synchronization, then it shall restart the link reset sequence rather than attempt to reestablish dword synchronization;
- b) if the phy receives a dword that is not a MUX before receiving the MUX expected in that position, then it shall discard the dword;
- c) if the phy receives an invalid dword, then it shall discard the dword; or
- d) if the phy receives a MUX that does not match the MUX expected in that position (i.e., it receives MUX (LOGICAL LINK 1) on logical link 0 or receives MUX (LOGICAL LINK 0) on logical link 1), then the phy shall restart the link reset sequence.

#### **5.11.5 Phy reset sequence after devices are attached**

Since SATA and SAS signal cable connectors do not include power lines, it is not possible to detect the physical insertion of the signal cable connector onto a plug. It may also not be possible to detect physical insertion of a device in non-cabled environments. As a result, every time a phy reset sequence is originated:

- a) expander phys that are enabled but not active shall originate a new phy reset sequence repeatedly, with no more than a hot-plug timeout (see table 85 in 5.11.1) between each attempt, until a speed negotiation sequence completes without error;
- b) SAS initiator phys should originate a new phy reset sequence after every hot-plug timeout; and
- c) SAS target phys should not originate a new phy reset sequence after their first attempt.

Figure 100 shows how two phys complete the phy reset sequence if the phys are not attached at power on. In this example, phy A and phy B are attached some time before phy B's second hot-plug timeout occurs. Phy B's OOB detection circuitry detects a COMINIT (see SATA) after the attachment and therefore phy B transmits COMSAS, since it has both transmitted and received a COMINIT. Upon receiving COMSAS (see SATA), phy A transmits its own COMSAS. The SAS speed negotiation sequence follows.

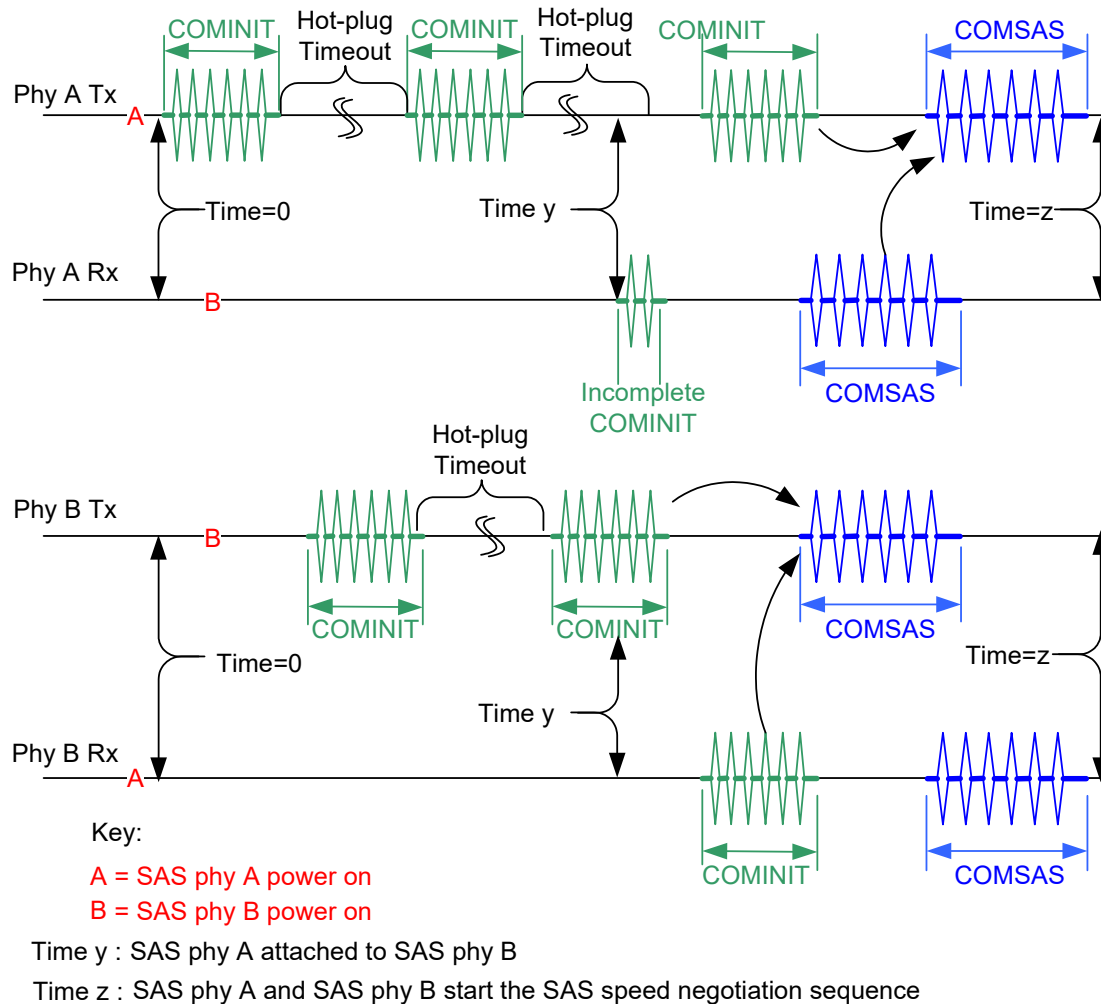


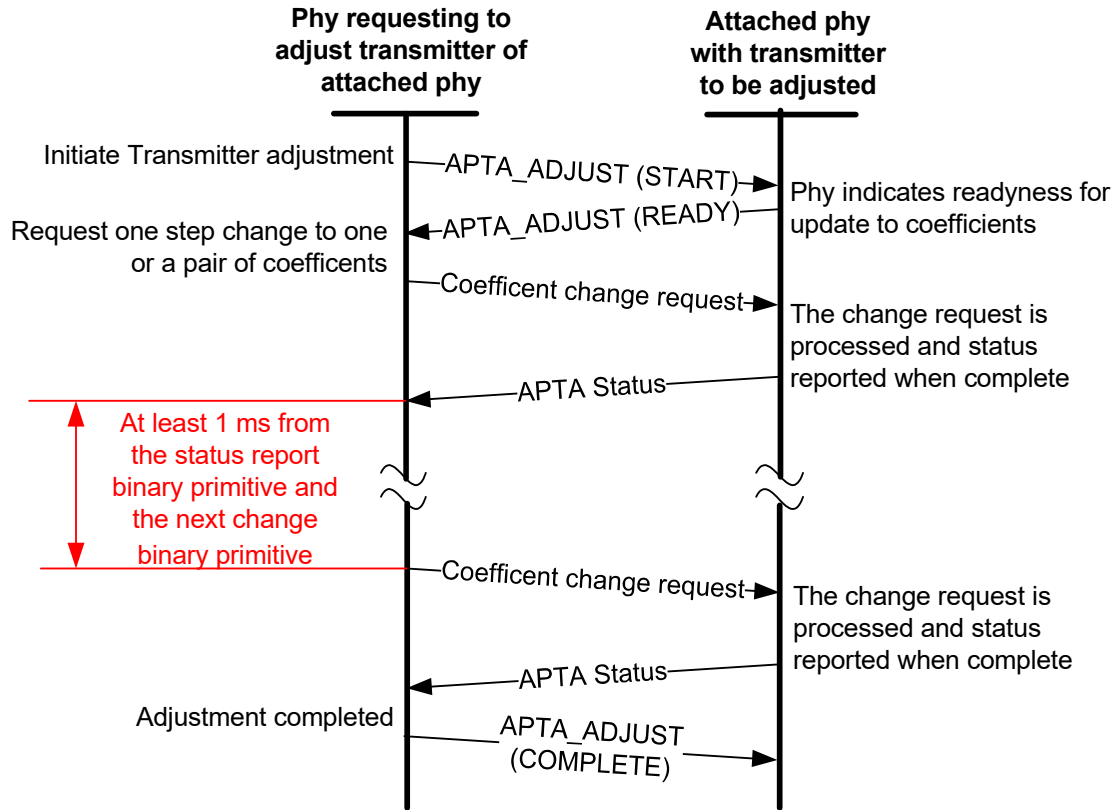
Figure 100 – Hot-plug and the phy reset sequence

## 5.12 APTA

If SAS packet mode is enabled, optical mode is disabled, and there is no active cable assembly attached to the phy, then APTA is a method that may be used to adjust SP transmitter coefficients without requiring a phy reset sequence (see 5.11). APTA binary primitives are exchanged between attached phys to request changes to the SP transmitter coefficient settings and to report SP transmitter status without causing a reset sequence.

To allow the SP receiver to adapt to changes in the SP transmitter settings, the time between each change request is at least 1 ms (see 5.19.6). By making only one change request at each time and allowing the SP receiver to adjust to each change over a long period of active data bits, the SP receiver equalizes itself to the revised transmitted signal before the SP receiver makes a calculation for the next change request, if any.

Figure 101 shows the exchange of APTA binary primitives to initiate, change, and complete the SP transmitter adjustment.



**Figure 101 – SP transmitter adjustment procedure**

APTA binary primitives are used to:

- request start of active adjustment;
- send change requests of the attached phy's SP transmitter;
- report status;
- terminate adjustment; and
- notify that active adjustment is complete.

If the SP receiver determines that received signal quality is not optimal for the receiver to recover the transmitted signal using a vendor specific algorithm, then the SP receiver requests the management application layer to start an APTA process (see 4.14).

Phy layer APTA (PAPTA) state machines process APTA binary primitives and coefficient change requests by the SP receiver (see 5.14.2).

## 5.13 Phy power condition sequences

### 5.13.1 Transitioning from the active phy power condition to a low phy power condition

See 6.13 for the sequence to transition from the active phy power condition to a low phy power condition.

### 5.13.2 Transitioning from a low phy power condition to the active phy power condition

Figure 102 shows the sequence to transition from a low phy power condition to the active phy power condition.

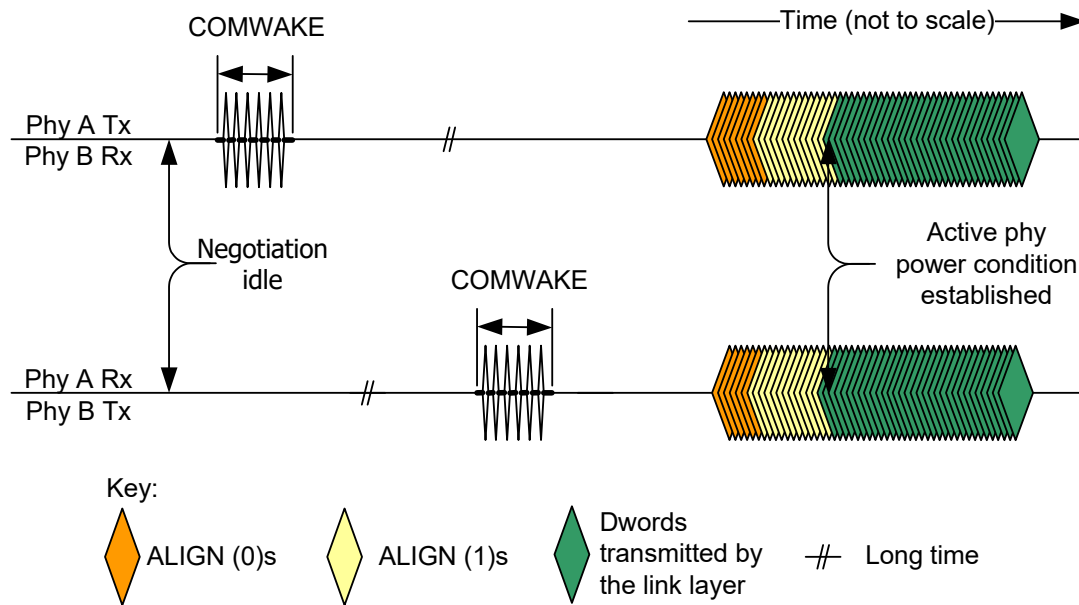


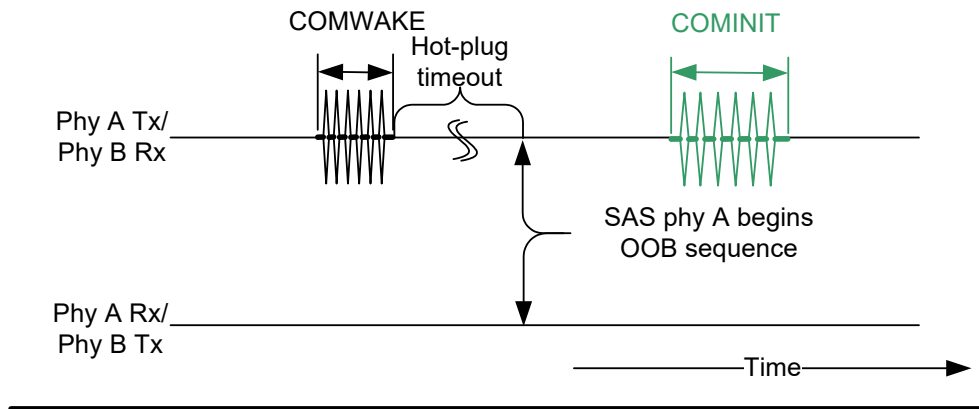
Figure 102 – Transition to active phy power condition

### 5.13.3 Events during low phy power condition

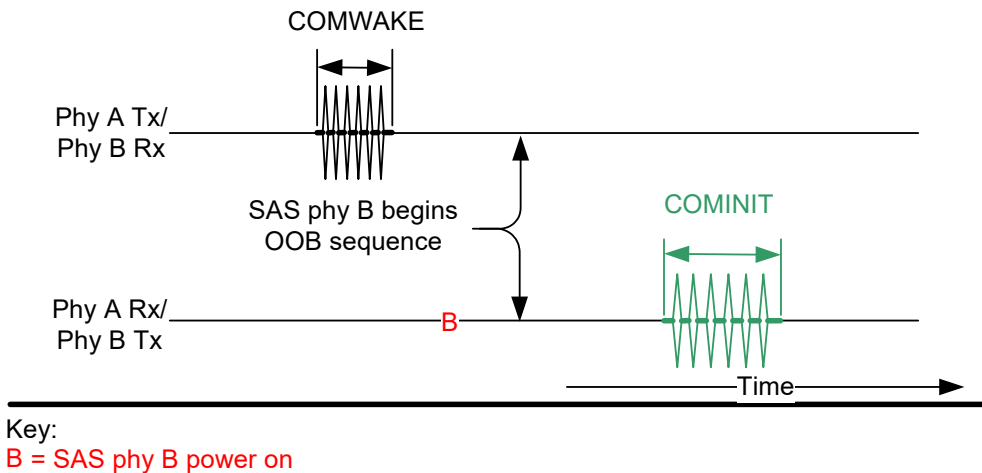
Figure 103 shows examples of responses to the following events that may occur during the transition from a low phy power condition to the active phy power condition:

- no response to COMWAKE within a hot plug timeout; and
- power on occurs after COMWAKE and before the hot plug timeout.

Sequence 1: no response to COMWAKE within a hot plug timeout



Sequence 2: power on occurs after COMWAKE and before the hot plug timeout



Key:

B = SAS phy B power on

**Figure 103 – Hot plug and low phy power condition**

The sequence for a no response to COMWAKE within a hot plug timeout depicted in sequence 1 in figure 103 proceeds as follows:

- 1) phy A transmits COMWAKE;
- 2) phy A detects no COMWAKE within a hot plug timeout; and
- 3) phy A transmits an OOB sequence.

The sequence for a power on occurs after COMWAKE and before the hot plug timeout depicted in sequence 2 in figure 103 proceeds as follows:

- 1) phy A transmits COMWAKE;
- 2) phy A detects an OOB sequence from phy B within a hot plug timeout; and
- 3) phy A transmits an OOB sequence.



## 5.14 SP (phy layer) state machine

### 5.14.1 SP state machine overview

The SP state machine controls the phy reset sequence. This state machine consists of the following sets of states:

- a) OOB sequence (OOB) states;
- b) SAS speed negotiation (SAS) states;
- c) SAS phy power condition (PS) states; and
- d) SATA host emulation (SATA) states.

This state machine consists of the following states:

- a) SP0:OOB\_COMINIT (see 5.14.3.2) (initial state);
- b) SP1:OOB\_AwaitCOMX (see 5.14.3.3);
- c) SP2:OOB\_NoCOMSASTimeout (see 5.14.3.4);
- d) SP3:OOB\_AwaitCOMINIT\_Sent (see 5.14.3.5);
- e) SP4:OOB\_COMSAS (see 5.14.3.6);
- f) SP5:OOB\_AwaitCOMSAS\_Sent (see 5.14.3.7);
- g) SP6:OOB\_AwaitNoCOMSAS (see 5.14.3.8);
- h) SP7:OOB\_AwaitCOMSAS (see 5.14.3.9);
- i) SP8:SAS\_Start (see 5.14.4.3);
- j) SP9:SAS\_WindowNotSupported (see 5.14.4.4);
- k) SP10:SAS\_AwaitALIGN (see 5.14.4.5);
- l) SP11:SAS\_AwaitALIGN1 (see 5.14.4.6);
- m) SP12:SAS\_AwaitSNW (see 5.14.4.7);
- n) SP13:SAS\_Pass (see 5.14.4.8);
- o) SP14 SAS\_Fail (see 5.14.4.9);
- p) SP15:SAS\_PHY\_Ready (see 5.14.4.10);
- q) SP16:SATA\_COMWAKE (see 5.14.6.2);
- r) SP17:SATA\_AwaitCOMWAKE (see 5.14.6.3);
- s) SP18:SATA\_AwaitNoCOMWAKE (see 5.14.6.4);
- t) SP19:SATA\_AwaitALIGN (see 5.14.6.5);
- u) SP20:SATA\_AdjustSpeed (see 5.14.6.6);
- v) SP21:SATA\_TransmitALIGN (see 5.14.6.7);
- w) SP22:SATA\_PHY\_Ready (see 5.14.6.8);
- x) SP23:SATA\_PM\_Partial (see 5.14.6.9);
- y) SP24:SATA\_PM\_Slumber (see 5.14.6.10);
- z) SP25:SATA\_PortSel (see 5.14.7.2);
- aa) SP26:SATA\_SpinupHold (see 5.14.8.2);
- ab) SP27:SAS\_Settings (see 5.14.4.11);
- ac) SP28:SAS\_TrainSetup (see 5.14.4.12);
- ad) SP29:SAS\_Train\_Rx (see 5.14.4.14);
- ae) SP30:SAS\_TrainingDone (see 5.14.4.15);
- af) SP31:SAS\_PS\_Low\_Phy\_Power (see 5.14.5.2);
- ag) SP32:SAS\_PS\_ALIGN0 (see 5.14.5.3);
- ah) SP33:SAS\_PS\_ALIGN1 (see 5.14.5.4);
- ai) SP34:SAS\_Train\_Tx (see 5.14.4.13); and
- aj) SP35:SAS\_PS\_Sync (see 5.14.5.5).

This state machine receives the following requests from the management application layer or a layer outside the scope of this standard:

- a) Management Reset from the management application layer;
- b) Disable Phy from the management application layer;
- c) Transmit SATA Port Selection Signal from the management application layer;
- d) Enter Partial request from SATA link layer, if any;
- e) Enter Slumber from the SATA link layer, if any; and

- f) Exit from the SATA link layer, if any.

This state machine sends the following confirmations to the management application layer:

- a) Transmitter Training Started;
- b) Enable APTA; and
- c) APTA Disabled with an argument of OOB In Progress or Low Phy Power Condition.

This state machine shall start in the SP0:OOB\_COMINIT state after:

- a) a power on;
- b) a hard reset;
- c) receiving a Management Reset request (e.g., as a result of processing an SMP PHY CONTROL function requesting a phy operation of LINK RESET or HARD RESET); or
- d) receiving a Disable Phy request (e.g., as a result of processing an SMP PHY CONTROL function requesting a phy operation of DISABLE).

If the phy supports SATA port selectors, then this state machine shall transition to the SP25:SATA\_PortSel state whenever it receives a Transmit SATA Port Selection Signal request.

This state machine sends the following messages to the SP\_DWS state machine (see 5.15):

- a) Start DWS; and
- b) Stop DWS.

This state machine sends the following messages to the SP\_PS state machine (see 5.16):

- a) Start PS; and
- b) Stop PS.

This state machine sends the following messages to the SP\_ReSync state machine (see 5.17):

- a) Resynchronize; and
- b) Stop Resync.

This state machine receives the following messages from the SP\_DWS state machine:

- a) DWS Lost; and
- b) DWS Reset.

This state machine receives the following messages from the SP\_ReSync state machine:

- a) PS Reset; and
- b) Restart PS.

This state machine sends the following messages to the PTT state machines (see 5.18):

- a) Transmitter Training (Enable); and
- b) Transmitter Training (Disable).

This state machine receives the following message from the PTT\_T state machine (see 5.18.4):

- a) Transmitter Training Complete.

This state machine receives the following messages from the PTT\_PL state machine (see 5.18.12):

- a) Pattern Lock Lost; and
- b) Pattern Locked.

This state machine shall maintain the timers listed in table 92.

**Table 92 – SP state machine timers**

Timer	Initial value
COMSAS Detect Timeout timer	COMSAS detect timeout (see SAS-4)
Await ALIGN Timeout timer	Await ALIGN timeout (see table 86)
Hot-Plug Timeout timer	Hot plug timeout (see table 85)
RCDT timer	RCDT (see table 87)
SNLT timer	SNLT (see table 87)
SNTT timer	SNTT (see table 87)
TLT timer	TLT (see table 87)
MRTT timer	MRTT (see table 87)
MTTT timer	MTTT (see table 87)
Pattern Lock Lost Timeout timer	1 ms
Scrambler Initialization timer	Vendor specific period for initialization of the scrambler timeout

The SP state machine shall maintain the state machine variables defined in table 93.

**Table 93 – SP state machine variables**

State machine variable	Description
MgmtReset	This state machine variable is used to determine whether a Management Reset request has been received. Any SP state that receives a Management Reset request shall set this state machine variable to one before making a transition to the SP0:OOB_COMINIT state (see 5.14.3.2). Any SP state that receives a power on or a hard reset shall set this state machine variable to zero before making a transition to the SP0:OOB_COMINIT state.
Current SNW	This state machine variable is used to determine the current SNW (e.g., SNW-1, SNW-2, SNW-3, Final-SNW, or Unsupported Phy Attached).
ResetStatus	If the phy status is available through any of the following: a) the SMP DISCOVER response (see 9.4.3.10); b) the SMP DISCOVER LIST response (see 9.4.3.15); c) the Phy Control And Discover mode page (see 9.2.7.5); or d) the Protocol Specific Port log page (see 9.2.8.1), then this state machine variable is used to determine the NEGOTIATED PHYSICAL LINK RATE field and/or the NEGOTIATED LOGICAL LINK RATE field.
Commonly Supported Settings	If the phy supports SNW-3, then this state machine variable contains the commonly supported settings.
COMWAKE_Received	If the phy supports SATA port selectors, then this state machine variable is used to determine whether a COMWAKE Detected message was received in the SP0:OOB_COMINIT state or the SP1:OOB_AwaitCOMX state since the last time the SP0:OOB_COMINIT state was entered.
SASPhyPwrCond	If the phy supports low phy power conditions, then this state machine variable is used to determine: a) the current phy power condition (see 4.10 and 5.7.5); and b) the PHY POWER CONDITION field in the SMP DISCOVER response (see table 328).

#### 5.14.2 SP transmitter and SP receiver

The SP transmitter transmits OOB signals, dwords, and SPL packets on the physical link based on messages from the SP state machine (see 5.14).

See 5.18.2 for SP transmitter requirements and see 5.18.3 for SP receiver requirements while PTT state machines are processing Train\_Tx-SNW.

The Phy Wakeup Complete message shall be sent after the SP transmitter becomes active as a result of receiving a Phy Wakeup message.

The SP transmitter receives the following messages from the SP state machine:

- Transmit COMINIT;
- Transmit COMSAS;
- Transmit COMWAKE;
- Transmit SATA Port Selection Signal;
- Transmit D10.2;
- Set Physical Link Rate with an argument specifying the physical link rate (e.g., 1.5 Gbit/s, 3 Gbit/s, 6 Gbit/s, 12 Gbit/s, or 22.5 Gbit/s);

- g) Set SSC with an argument of On or Off;
- h) Transmit ALIGN with an argument indicating the specific type (e.g., Transmit ALIGN (0));
- i) Transmit Phy Capabilities Bits;
- j) Transmit TRAIN Pattern;
- k) Transmit TRAIN\_DONE Pattern;
- l) Transmit PACKET\_SYNC\_LOST;
- m) Transmit PACKET\_SYNC;
- n) Transmit MUX Sequence;
- o) Transmit OOB Idle;
- p) Enter Partial Phy Power Condition;
- q) Enter Slumber Phy Power Condition; and
- r) Phy Wakeup.

If the local SAS phy is adjusting the attached SP transmitter coefficients, then the SP transmitter receives the following messages from the PAPTA\_L\_A state machine that adjusts the attached SP transmitter:

- a) Transmit APTA\_ADJUST with an argument of Start, Complete, or Terminate;
- b) Transmit APTA\_COEFFICIENT\_1 with an argument of Increment or Decrement;
- c) Transmit APTA\_COEFFICIENT\_2 with an argument of Increment or Decrement;
- d) Transmit APTA\_COEFFICIENT\_3 with an argument of Increment or Decrement;
- e) Transmit APTA\_COEFFICIENTS\_1\_2 with an argument of Increment or Decrement; and
- f) Transmit APTA\_COEFFICIENTS\_2\_3 with an argument of Increment or Decrement.

If the attached SAS phy is adjusting the local SP transmitter coefficients, then the SP transmitter receives the following messages from the PAPTA\_A\_L state machine to report status of the local SP transmitter:

- a) Transmit APTA\_ADJUST with an argument of Ready or Terminate;
- b) Transmit APTA\_COEFFICIENT\_1 with an argument of Updated, Maximum, or Minimum;
- c) Transmit APTA\_COEFFICIENT\_2 with an argument of Updated, Maximum, or Minimum;
- d) Transmit APTA\_COEFFICIENT\_3 with an argument of Updated, Maximum, or Minimum;
- e) Transmit APTA\_COEFFICIENTS\_1\_2 with an argument of Updated, Maximum, or Minimum; and
- f) Transmit APTA\_COEFFICIENTS\_2\_3 with an argument of Updated, Maximum, or Minimum.

When not otherwise instructed, the SP transmitter transmits negotiation idle.

Upon receiving a Phy Wakeup message, the SP transmitter shall become active (i.e., capable of transmitting OOB signals, dwords, and SPL packets) within:

- a) a phy wakeup partial timeout (see table 85), if the phy is in the partial phy power condition; or
- b) a phy wakeup slumber timeout (see table 85), if the phy is in the slumber phy power condition.

Upon receiving a Transmit MUX Sequence message, the SP transmitter transmits:

- 1) MUX (LOGICAL LINK 0);
- 2) MUX (LOGICAL LINK 1);
- 3) MUX (LOGICAL LINK 0);
- 4) MUX (LOGICAL LINK 1);
- 5) MUX (LOGICAL LINK 0); and
- 6) MUX (LOGICAL LINK 1).

Upon receiving a Transmit OOB Idle message, the SP transmitter transmits OOB\_IDLE or D.C. idle as defined in SAS-4.

The SP transmitter shall complete any physical link rate change and SSC change requested with the Set Physical Link Rate message and the Set SSC message within RCDT (see table 87 in 5.11.4.2).

The SP transmitter sends the following messages to the SP state machine:

- a) COMINIT Transmitted;
- b) COMSAS Transmitted;
- c) COMWAKE Transmitted;
- d) SATA Port Selection Signal Transmitted;
- e) TRAIN\_DONE Pattern Transmitted;

- f) PACKET\_SYNC Transmitted;
- g) Phy Capabilities Bits Transmitted;
- h) MUX Sequence Transmitted; and
- i) Phy Wakeup Complete.

The SP transmitter sends the following message to the SP\_ReSync state machine:

- a) PACKET\_SYNC Transmitted.

The SP receiver receives the following messages from the SP state machine:

- a) Set Physical Link Rate with an argument specifying the physical link rate (e.g., 1.5 Gbit/s, 3 Gbit/s, 6 Gbit/s, 12 Gbit/s, or 22.5 Gbit/s);
- b) Receive Phy Capabilities Bits;
- c) Start Rx Training;
- d) Abort Rx Training;
- e) Enable OOB Detection;
- f) Disable OOB Detection;
- g) Enter Partial Phy Power Condition; and
- h) Enter Slumber Phy Power Condition.

The SP receiver receives the following messages from the SP\_DWS state machine (see 5.15):

- a) Sync Acquired; and
- b) Sync Lost.

The SP receiver sends the following messages to the SP state machine indicating OOB signals, dwords, and SPL packets received from the physical link:

- a) COMINIT Detected;
- b) COMSAS Detected;
- c) COMWAKE Detected;
- d) COMSAS Completed;
- e) COMWAKE Completed;
- f) ALIGN Received with an argument indicating the specific type (e.g., ALIGN Received (0));
- g) Phy Capabilities Bits Received with arguments indicating the supported settings bits received and either Good Parity or Bad Parity;
- h) Rx Training Completed;
- i) TRAIN\_DONE Received;
- j) PACKET\_SYNC Received;
- k) PACKET\_SYNC\_LOST Received;
- l) SPL Packet Received; and
- m) Dword Received.

The SP receiver sends the following messages to the SP\_ReSync state machine indicating, dwords, and SPL packets received from the physical link:

- a) PACKET\_SYNC Received; and
- b) PACKET\_SYNC\_LOST Received.

If the phy is in the SAS dword mode, then the ALIGN Received message, Dword Received message, and TRAIN\_DONE Received message are only sent after the SP\_DWS state machine has achieved dword synchronization (i.e., a Sync Acquired message is received).

For SATA speed negotiation, the ALIGN Received (0) message includes an argument containing the physical link rate at which the ALIGN (0) primitives were detected. For SAS speed negotiation, only ALIGNs at the physical link rate specified by the last Set Physical Link Rate message received by the SP receiver cause ALIGN Received messages.

If the attached SAS phy is adjusting the local SP transmitter coefficients, then the SP receiver sends the following messages to the PAPT\_A\_L state machine to request coefficient changes to the local SAS phy's SP transmitter:

- a) Received APTA\_ADJUST with an argument of Start, Complete, or Terminate;

- b) Received APTA\_COEFFICIENT\_1 with an argument of Increment or Decrement;
- c) Received APTA\_COEFFICIENT\_2 with an argument of Increment or Decrement;
- d) Received APTA\_COEFFICIENT\_3 with an argument of Increment or Decrement;
- e) Received APTA\_COEFFICIENTS\_1\_2 with an argument of Increment or Decrement; and
- f) Received APTA\_COEFFICIENTS\_2\_3 with an argument of Increment or Decrement.

If the local SAS phy is adjusting the attached SP transmitter coefficients, then the SP receiver sends the following messages to the PAPTA\_L\_A state machine to report status of the local SAS phy's SP transmitter:

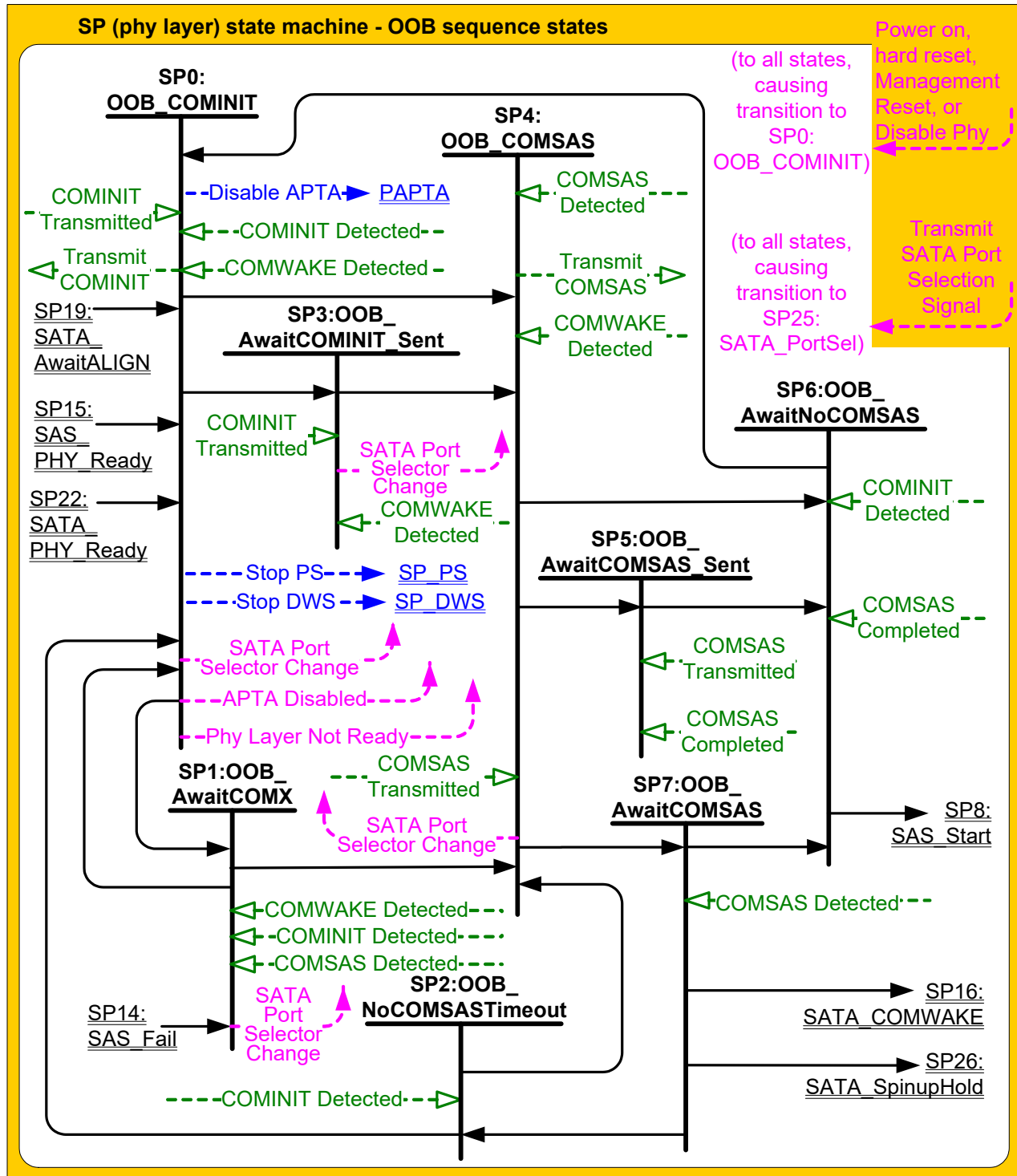
- a) Received APTA\_ADJUST with an argument of Ready or Terminate;
- b) Received APTA\_COEFFICIENT\_1 with an argument of Updated, Maximum, or Minimum;
- c) Received APTA\_COEFFICIENT\_2 with an argument of Updated, Maximum, or Minimum;
- d) Received APTA\_COEFFICIENT\_3 with an argument of Updated, Maximum, or Minimum;
- e) Received APTA\_COEFFICIENTS\_1\_2 with an argument of Updated, Maximum, or Minimum; and
- f) Received APTA\_COEFFICIENTS\_2\_3 with an argument of Updated, Maximum, or Minimum.

The SP transmitter relationship to other transmitters is defined in 4.3.2. The SP receiver relationship to other receivers is defined in 4.3.3.

### 5.14.3 OOB sequence states

### 5.14.3.1 OOB sequence states overview

Figure 104 shows the OOB sequence states. These states are indicated by state names with a prefix of OOB.



**Figure 104 – SP (phy layer) state machine - OOB sequence states**



**5.14.3.2 SP0:OOB\_COMINIT state****5.14.3.2.1 State description**

This state is the initial state for this state machine.

Upon entry into this state, this state shall:

- a) set the COMWAKE\_Received state machine variable to zero;
- b) send an Enable OOB Detection message to the SP receiver;
- c) set the SASPhyPwrCond state machine variable to Active;
- d) send a Stop DWS message to the SP\_DWS state machine;
- e) send a Stop PS message to the SP\_PS state machine;
- f) disable the SAS packet mode;
- g) enable the SAS dword mode;
- h) send a Phy Layer Not Ready confirmation to the link layer;
- i) send an APTA Disabled (OOB In Progress) confirmation to the management application layer;
- j) send a Disable APTA message to the PAPTAs State machines;
- k) set the ATTACHED SATA DEVICE bit to zero in the SMP DISCOVER response (see 9.4.3.10);
- l) if this state was entered due to power on and the phy is an SMP target phy, then set the ATTACHED SATA PORT SELECTOR bit to zero in the SMP DISCOVER response (see 9.4.3.10);
- m) if this state was not entered because of a Disable Phy request and D.C. mode is enabled, then send a Transmit COMINIT message to the SP transmitter; and
- n) if this state was not entered as a result of a Disable Phy request and optical mode is enabled, then:
  - 1) repeatedly send Transmit OOB Idle messages to the SP transmitter for an RCDT time; and
  - 2) send a Transmit COMINIT message to the SP transmitter.

If this state was entered because of a Disable Phy request, then upon entry into this state, this state shall:

- a) ignore COMINIT Detected messages until this state is re-entered due to a power on, hard reset, or Management Reset request; and
- b) set the ResetStatus state machine variable to DISABLED.

If this state was entered due to power on or hard reset, then upon entry into this state, this state shall set the ResetStatus state machine variable to UNKNOWN.

If this state was entered because of a Management Reset request, then upon entry into this state, this state shall:

- a) if the ResetStatus state machine variable is not set to RESET\_IN\_PROGRESS, SPINUP\_HOLD, G1, G2, G3, or G4, then set the ResetStatus state machine variable to UNKNOWN; or
- b) if the ResetStatus state machine variable is set to RESET\_IN\_PROGRESS, SPINUP\_HOLD, G1, G2, G3, or G4, then set the ResetStatus state machine variable to RESET\_IN\_PROGRESS.

If this state was not entered due to a power on, hard reset, Disable Phy, or Management Reset request, then upon entry into this state, this state shall:

- a) if the ResetStatus state machine variable is not set to PHY\_RESET\_PROBLEM, SPINUP\_HOLD, or UNSUPPORTED\_PHY\_ATTACHED, then set the ResetStatus state machine variable to UNKNOWN; or
- b) if the ResetStatus state machine variable is set to PHY\_RESET\_PROBLEM, SPINUP\_HOLD, or UNSUPPORTED\_PHY\_ATTACHED, then not change the ResetStatus state machine variable.

If the phy supports SATA port selectors and this state receives a COMWAKE Detected message, then this state shall:

- a) if the ResetStatus state machine variable is not set to PHY\_RESET\_PROBLEM, SPINUP\_HOLD, or UNSUPPORTED\_PHY\_ATTACHED, then set the ResetStatus state machine variable to PORT\_SELECTOR;
- b) set the COMWAKE\_Received state machine variable to one; and
- c) if the phy is an SMP target phy and the ATTACHED SATA PORT SELECTOR bit is set to zero in the SMP DISCOVER response (see 9.4.3.10), then:

- A) set the ATTACHED SATA PORT SELECTOR bit to one in the SMP DISCOVER response; and
- B) send a SATA Port Selector Change confirmation to the link layer.

This state machine waits for receipt of a COMINIT Transmitted message and/or a COMINIT Detected message.

#### **5.14.3.2.2 Transition SP0:OOB\_COMINIT to SP1:OOB\_AwaitCOMX**

This transition shall occur:

- a) if this state receives a COMINIT Transmitted message and has not received a COMINIT Detected message.

#### **5.14.3.2.3 Transition SP0:OOB\_COMINIT to SP3:OOB\_AwaitCOMINIT\_Sent**

This transition shall occur:

- a) if this state receives a COMINIT Detected message and has not received a COMINIT Transmitted message.

#### **5.14.3.2.4 Transition SP0:OOB\_COMINIT to SP4:OOB\_COMSAS**

This transition shall occur:

- a) if this state receives both a COMINIT Transmitted message and a COMINIT Detected message.

### **5.14.3.3 SP1:OOB\_AwaitCOMX state**

#### **5.14.3.3.1 State description**

Upon entry into this state, this state shall initialize and start the Hot-Plug Timeout timer if this phy is:

- a) an expander phy; or
- b) an initiator phy or target phy implementing the Hot-Plug Timeout timer.

If the phy supports SATA port selectors and this state receives a COMWAKE Detected message, then this state shall:

- a) if the ResetStatus state machine variable is not set to PHY\_RESET\_PROBLEM, SPINUP\_HOLD, or UNSUPPORTED\_PHY\_ATTACHED, then set the ResetStatus state machine variable to PORT\_SELECTOR;
- b) set the COMWAKE\_Received state machine variable to one; and
- c) if the phy is an SMP target phy and the ATTACHED SATA PORT SELECTOR bit is set to zero in the SMP DISCOVER response (see 9.4.3.10), then:
  - A) set the ATTACHED SATA PORT SELECTOR bit to one in the SMP DISCOVER response; and
  - B) send a SATA Port Selector Change confirmation to the link layer.

#### **5.14.3.3.2 Transition SP1:OOB\_AwaitCOMX to SP0:OOB\_COMINIT**

This transition shall occur if the Hot-Plug Timeout timer expires.

If the COMWAKE\_Received state machine variable is set to zero and the ATTACHED SATA PORT SELECTOR bit is set to one in the SMP DISCOVER response (see 9.4.3.10), then the state machine shall, before the transition:

- a) set the ATTACHED SATA PORT SELECTOR bit to zero in the SMP DISCOVER response; and
- b) send a SATA Port Selector Change confirmation to the link layer.

Before the transition, if this state was entered from SP0:OOB\_COMINIT, then this state shall set the ResetStatus state machine variable to UNKNOWN.

**5.14.3.3.3 Transition SP1:OOB\_AwaitCOMX to SP4:OOB\_COMSAS**

This transition shall occur:

- a) after receiving a COMINIT Detected message or a COMSAS Detected message.

If COMSAS Detected was received, then this transition shall include:

- a) a COMSAS Detected argument.

**5.14.3.4 SP2:OOB\_NoCOMSASTimeout state****5.14.3.4.1 State description**

Upon entry into this state, this state shall initialize and start the Hot-Plug Timeout timer if this phy is:

- a) an expander phy; or
- b) an initiator phy or target phy implementing the Hot-Plug Timeout timer.

**5.14.3.4.2 Transition SP2:OOB\_NoCOMSASTimeout to SP0:OOB\_COMINIT**

This transition shall occur:

- a) if the Hot-Plug Timeout timer expires.

**5.14.3.4.3 Transition SP2:OOB\_NoCOMSASTimeout to SP4:OOB\_COMSAS**

This transition shall occur:

- a) after receiving a COMINIT Detected message.

**5.14.3.5 SP3:OOB\_AwaitCOMINIT\_Sent state****5.14.3.5.1 State description**

This state waits for a COMINIT Transmitted message.

If the phy supports SATA port selectors and this state receives a COMWAKE Detected message, then this state shall:

- a) if the ResetStatus state machine variable is not set to PHY\_RESET\_PROBLEM, SPINUP\_HOLD, or UNSUPPORTED\_PHY\_ATTACHED, then set the ResetStatus state machine variable to PORT\_SELECTOR; and
- b) if the phy is an SMP target phy and the ATTACHED SATA PORT SELECTOR bit is set to zero in the SMP DISCOVER response (see 9.4.3.10), then:
  - A) set the ATTACHED SATA PORT SELECTOR bit to one in the SMP DISCOVER response; and
  - B) send a SATA Port Selector Change confirmation to the link layer.

**5.14.3.5.2 Transition SP3:OOB\_AwaitCOMINIT\_Sent to SP4:OOB\_COMSAS**

This transition shall occur:

- a) after receiving a COMINIT Transmitted message.

**5.14.3.6 SP4:OOB\_COMSAS state****5.14.3.6.1 State description**

Upon entry into this state, this state shall send a Transmit COMSAS message to the SP transmitter.

This state waits for receipt of a COMSAS Transmitted message and/or a COMSAS Detected message.

If the phy supports SATA port selectors and this state receives a COMWAKE Detected message, then this state shall:

- a) if the ResetStatus state machine variable is not set to PHY\_RESET\_PROBLEM, SPINUP\_HOLD, or UNSUPPORTED\_PHY\_ATTACHED, then set the ResetStatus state machine variable to PORT\_SELECTOR; and
- b) if the phy is an SMP target phy and the ATTACHED SATA PORT SELECTOR bit is set to zero in the SMP DISCOVER response (see 9.4.3.10), then:
  - A) set the ATTACHED SATA PORT SELECTOR bit to one in the SMP DISCOVER response; and
  - B) send a SATA Port Selector Change confirmation to the link layer.

#### **5.14.3.6.2 Transition SP4:OOB\_COMSAS to SP5:OOB\_AwaitCOMSAS\_Sent**

This transition shall occur:

- a) if this state receives a COMSAS Detected message or this state was entered with a COMSAS Detected argument; and
- b) this state has not received a COMSAS Transmitted message.

If the ATTACHED SATA PORT SELECTOR bit is set to one in the SMP DISCOVER response (see 9.4.3.10), then the state machine shall set the ATTACHED SATA PORT SELECTOR bit to zero in the SMP DISCOVER response and send a SATA Port Selector Change confirmation to the link layer before the transition.

#### **5.14.3.6.3 Transition SP4:OOB\_COMSAS to SP6:OOB\_AwaitNoCOMSAS**

This transition shall occur if this state receives:

- a) a COMSAS Transmitted message; and
- b) a COMSAS Detected message.

If the ATTACHED SATA PORT SELECTOR bit is set to one in the SMP DISCOVER response (see 9.4.3.10), then the state machine shall set the ATTACHED SATA PORT SELECTOR bit to zero in the SMP DISCOVER response and send a SATA Port Selector Change confirmation to the link layer before the transition.

#### **5.14.3.6.4 Transition SP4:OOB\_COMSAS to SP7:OOB\_AwaitCOMSAS**

This transition shall occur if this state:

- a) receives a COMSAS Transmitted message; and
- b) has not received a COMSAS Detected message.

#### **5.14.3.7 SP5:OOB\_AwaitCOMSAS\_Sent state**

##### **5.14.3.7.1 State description**

This state waits for receipt of a COMSAS Transmitted message.

##### **5.14.3.7.2 Transition SP5:OOB\_AwaitCOMSAS\_Sent to SP6:OOB\_AwaitNoCOMSAS**

This transition shall occur:

- a) after receiving a COMSAS Transmitted message.

If this state received a COMSAS Completed message, then it shall include:

- a) a COMSAS Completed argument with the transition.

**5.14.3.8 SP6:OOB\_AwaitNoCOMSAS state****5.14.3.8.1 State description**

This state machine waits for a COMSAS Completed message, which indicates that COMSAS has been received.

**5.14.3.8.2 Transition SP6:OOB\_AwaitNoCOMSAS to SP0:OOB\_COMINIT**

This transition shall occur:

- a) after receiving a COMINIT Detected message.

Before the transition, this state shall set the ResetStatus state machine variable to UNKNOWN.

**5.14.3.8.3 Transition SP6:OOB\_AwaitNoCOMSAS to SP8:SAS\_Start**

This transition shall occur:

- a) after receiving a COMSAS Completed message; or
- b) if a COMSAS Completed argument was received in the transition.

**5.14.3.9 SP7:OOB\_AwaitCOMSAS state****5.14.3.9.1 State description**

Upon entry into this state, this state shall initialize and start the COMSAS Detect Timeout timer.

**5.14.3.9.2 Transition SP7:OOB\_AwaitCOMSAS to SP2:OOB\_NoCOMSASTimeout**

This transition shall occur:

- a) if the phy does not support SATA; and
- b) the COMSAS Detect Timeout timer expires.

The state machine shall set the MgmtReset state machine variable to zero before the transition.

**5.14.3.9.3 Transition SP7:OOB\_AwaitCOMSAS to SP6:OOB\_AwaitNoCOMSAS**

This transition shall occur:

- a) after receiving a COMSAS Detected message.

The state machine shall set the MgmtReset state machine variable to zero before the transition.

The state machine shall set the ATTACHED SATA PORT SELECTOR bit to zero in the SMP DISCOVER response (see 9.4.3.10). If the ATTACHED SATA PORT SELECTOR bit in the SMP DISCOVER response was set to one before to this transition, then the state machine shall send a SATA Port Selector Change confirmation to the link layer before the transition.

**5.14.3.9.4 Transition SP7:OOB\_AwaitCOMSAS to SP16:SATA\_COMWAKE**

This transition shall occur if:

- a) the phy supports SATA; and
- b) the COMSAS Detect Timeout timer expires and:
  - A) the MgmtReset state machine variable is set to one; or
  - B) the phy does not implement SATA spinup hold.

The state machine shall set the MgmtReset state machine variable to zero before the transition.

The state machine shall set the ATTACHED SATA DEVICE bit to one in the SMP DISCOVER response (see 9.4.3.10) before the transition.

#### **5.14.3.9.5 Transition SP7:OOB\_AwaitCOMSAS to SP26:SATA\_SpinupHold**

This transition shall occur if:

- a) the phy supports SATA;
- b) the COMSAS Detect Timeout timer expires;
- c) the phy implements SATA spinup hold; and
- d) the MgmtReset state machine variable is set to zero.

The state machine shall set the ATTACHED SATA DEVICE bit to one in the SMP DISCOVER response (see 9.4.3.10) before the transition.

### **5.14.4 SAS speed negotiation states**

#### **5.14.4.1 SAS speed negotiation states overview**

Figure 105 shows the SAS speed negotiation states, in which the phy has detected that it is attached to a SAS phy or expander phy rather than a SATA phy, and performs the SAS speed negotiation sequence. These states are indicated by state names with a prefix of SAS.

#### **5.14.4.2 Negotiation idle**

SAS speed negotiation states use negotiation idle at the beginning of each SNW.

If D.C. mode is enabled, then a negotiation idle consists of the transmission of D.C. idle.

If optical mode is enabled, then see SAS-4 for the optical mode negotiation idle definition.

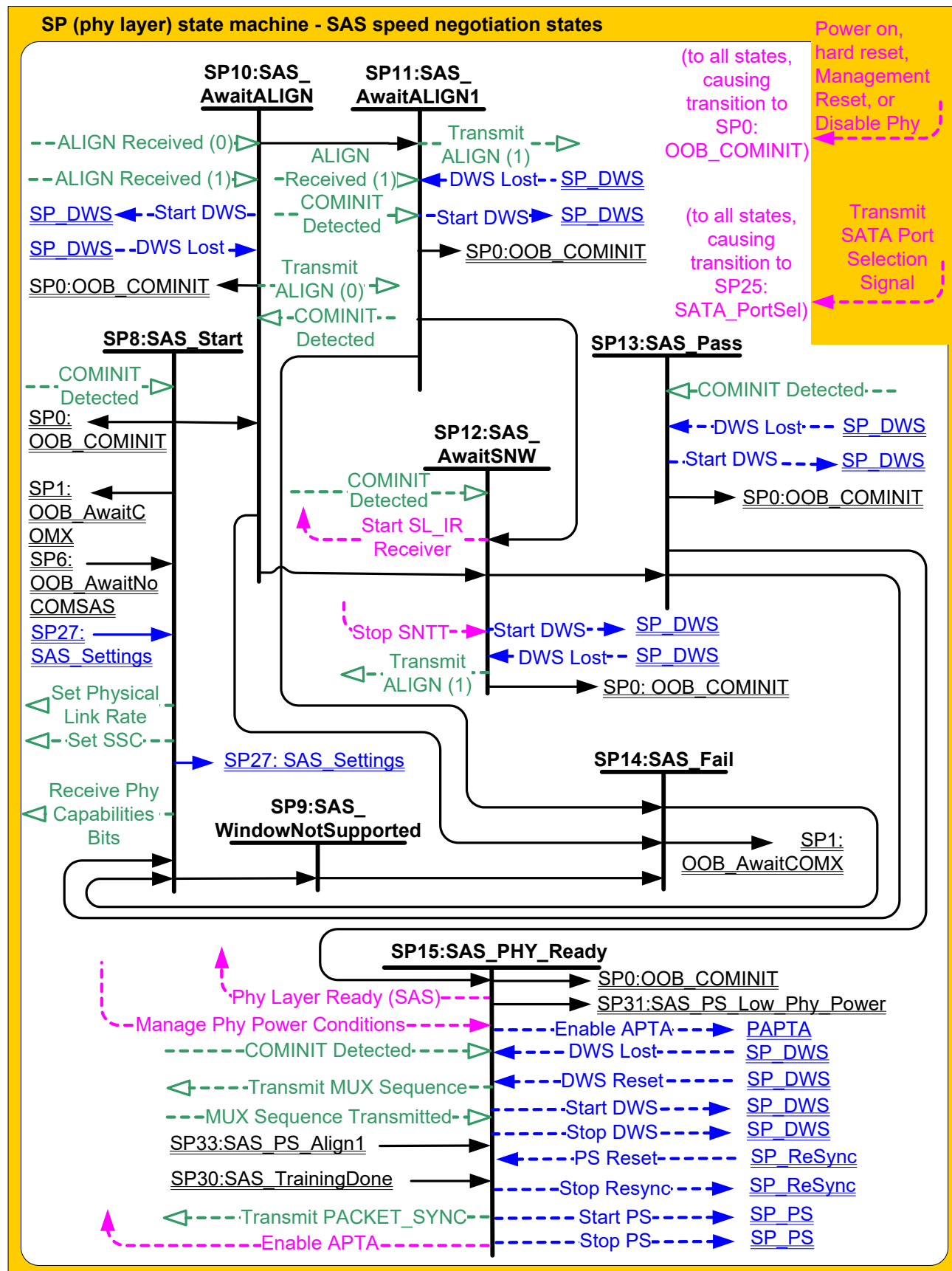


Figure 105 – SP (phy layer) state machine - SAS speed negotiation states

Figure 106 shows the SAS speed negotiation states related to SNW-3, Train\_Tx-SNW, and Train\_Rx-SNW.

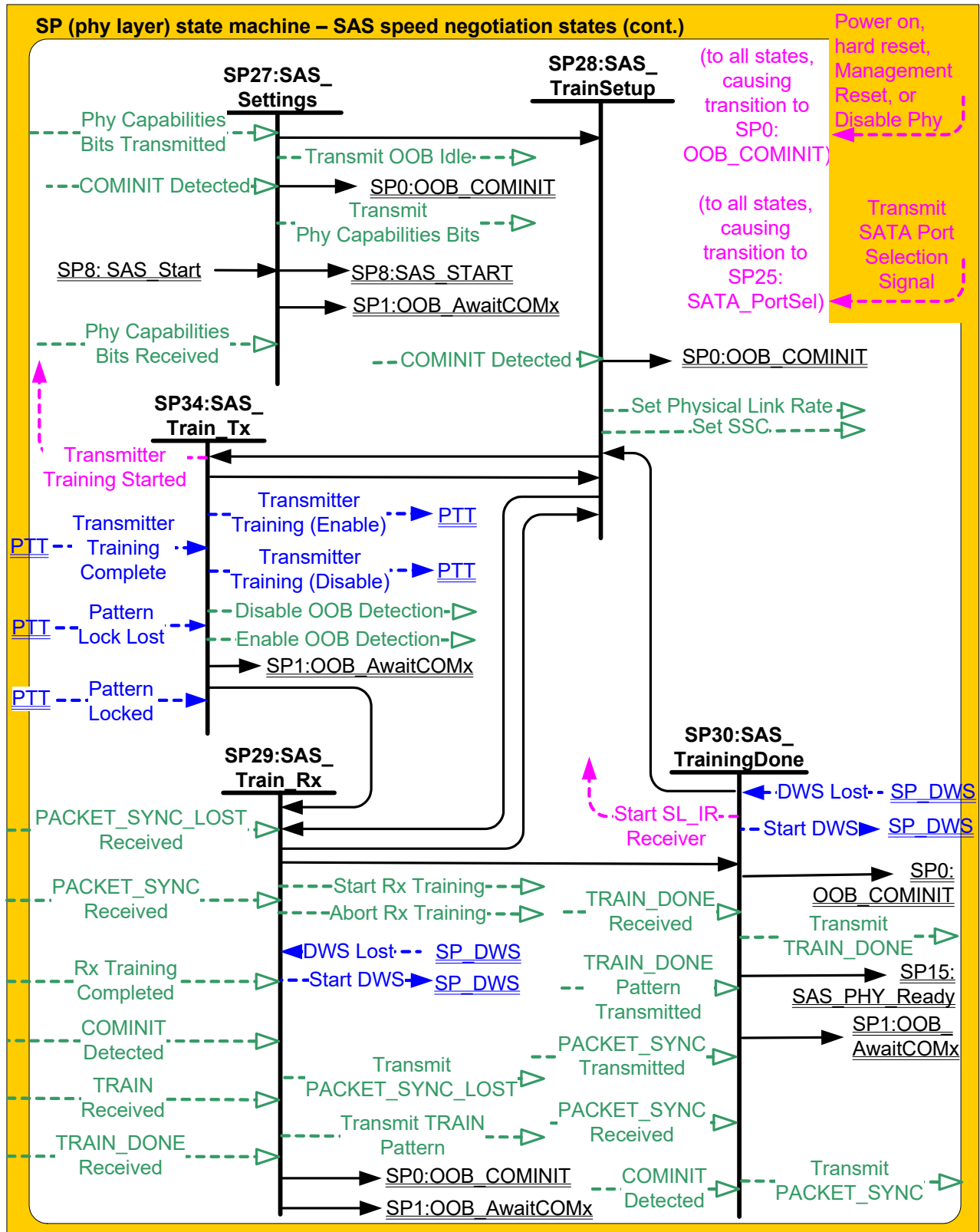


Figure 106 – SP (phy layer) state machine - SAS speed negotiation states for SNW-3 and Train\_Rx-SNW and Train\_Tx-SNW



**5.14.4.3 SP8:SAS\_Start state****5.14.4.3.1 State description**

This is the state in which the SAS speed negotiation sequence begins.

Upon entry into this state, this state shall initialize and start the RCDT timer.

If this state is entered from SP6:OOB\_AwaitNoCOMSAS, then the Current SNW state machine variable shall be set to SNW-1. If this state is not entered from SP6:OOB\_AwaitNoCOMSAS, then the Current SNW state machine variable shall be set to:

- a) SNW-2 if the Current SNW state machine variable is set to SNW-1;
- b) SNW-3 if the Current SNW state machine variable is set to SNW-2, and either SNW-1 is invalid or SNW-2 is valid;
- c) Final-SNW if the Current SNW state machine variable is set to SNW-2, SNW-1 is valid, and SNW-2 is invalid;
- d) Final-SNW if the Current SNW state machine variable is set to SNW-3, SNW-3 is invalid, and SNW-2 is valid; or
- e) Unsupported Phy Attached if the Current SNW state machine variable is set to SNW-3, SNW-3 is invalid, and SNW-2 is invalid.

After the Current SNW state machine variable is updated, if:

- a) the Current SNW state machine variable is not set to Unsupported Phy Attached; and
- b) the SNW specified by the Current SNW state machine variable is supported,

then this state shall send the messages specified in table 94 to the SP transmitter and SP receiver.

**Table 94 – Messages to SP transmitter and SP receiver at start of RCDT**

<b>Current SNW state machine variable</b>	<b>Other conditions</b>	<b>Messages sent to SP transmitter</b>	<b>Messages sent to SP receiver</b>
SNW-1	none	Set Physical Link Rate (1.5 Gbit/s) and Set SSC (Off)	Set Physical Link Rate (1.5 Gbit/s)
SNW-2	none	Set Physical Link Rate (3 Gbit/s) and Set SSC (Off)	Set Physical Link Rate (3 Gbit/s)
SNW-3	none	Set SSC (On) or Set SSC (Off)	Receive Phy Capabilities Bits
Final-SNW	SNW-1 was valid and SNW-2 was invalid	Set Physical Link Rate (1.5 Gbit/s) and Set SSC (Off)	Set Physical Link Rate (1.5 Gbit/s)
	SNW-2 was valid	Set Physical Link Rate (3 Gbit/s) and Set SSC (Off)	Set Physical Link Rate (3 Gbit/s)

**5.14.4.3.2 Transition SP8:SAS\_Start to SP0:OOB\_COMINIT**

This transition shall occur:

- a) after receiving a COMINIT Detected message.

Before the transition, this state shall set the ResetStatus state machine variable to UNKNOWN.

**5.14.4.3.3 Transition SP8:SAS\_Start to SP1:OOB\_AwaitCOMX**

This transition shall occur:

- a) if the Current SNW state machine variable is set to Unsupported Phy Attached.

Before the transition, this state shall set the ResetStatus state machine variable to UNSUPPORTED\_PHY\_ATTACHED.

**5.14.4.3.4 Transition SP8:SAS\_Start to SP9:SAS\_WindowNotSupported**

This transition shall occur:

- a) after the RCDT timer expires if the SNW indicated by the Current SNW state machine variable is not supported.

**5.14.4.3.5 Transition SP8:SAS\_Start to SP10:SAS\_AwaitALIGN**

This transition shall occur after the RCDT timer expires if:

- a) the Current SNW state machine variable is not set to SNW-3; and
- b) the SNW indicated by the Current SNW state machine variable is supported.

**5.14.4.3.6 Transition SP8:SAS\_Start to SP27:SAS\_Settings**

This transition shall occur after the RCDT timer expires if:

- a) the Current SNW state machine variable is set to SNW-3; and
- b) SNW-3 is supported.

**5.14.4.4 SP9:SAS\_WindowNotSupported state****5.14.4.4.1 State description**

Upon entry into this state, this state shall initialize and start the SNTT timer.

**5.14.4.4.2 Transition SP9:SAS\_WindowNotSupported to SP0:OOB\_COMINIT**

This transition should occur:

- a) after receiving a COMINIT Detected message.

Before the transition, this state shall set the ResetStatus state machine variable to UNKNOWN.

**5.14.4.4.3 Transition SP9:SAS\_WindowNotSupported to SP14:SAS\_Fail**

This transition shall occur:

- a) after the SNTT timer expires.

**5.14.4.5 SP10:SAS\_AwaitALIGN state****5.14.4.5.1 State description**

Upon entry into this state, this state shall:

- a) initialize and start the SNTT timer and SNLT timer;
- b) send a Start DWS message to the SP\_DWS state machine; and
- c) repeatedly send Transmit ALIGN (0) messages to the SP transmitter.

Each time this state receives a DWS Lost message, this state may send a Start DWS message to the SP\_DWS state machine to re-acquire dword synchronization without running a new link reset sequence.

**5.14.4.5.2 Transition SP10:SAS\_AwaitALIGN to SP0:OOB\_COMINIT**

This transition shall occur after receiving:

- a) a DWS Lost message, if this state does not send a Start DWS message; or
- b) a COMINIT Detected message.

Before the transition, this state shall set the ResetStatus state machine variable to UNKNOWN.

**5.14.4.5.3 Transition SP10:SAS\_AwaitALIGN to SP11:SAS\_AwaitALIGN1**

This transition shall occur:

- a) if this state receives an ALIGN Received (0) message before the SNLT timer expires.

**5.14.4.5.4 Transition SP10:SAS\_AwaitALIGN to SP12:SAS\_AwaitSNW**

This transition shall occur:

- a) if this state receives an ALIGN Received (1) message before the SNLT timer expires.

**5.14.4.5.5 Transition SP10:SAS\_AwaitALIGN to SP14:SAS\_Fail**

This transition shall occur:

- a) if the SNTT timer expires.

**5.14.4.6 SP11:SAS\_AwaitALIGN1 state****5.14.4.6.1 State description**

This state shall repeatedly send Transmit ALIGN (1) messages to the SP transmitter.

Each time this state receives a DWS Lost message, this state may send a Start DWS message to the SP\_DWS state machine to re-acquire dword synchronization without running a new link reset sequence.

**5.14.4.6.2 Transition SP11:SAS\_AwaitALIGN1 to SP0:OOB\_COMINIT**

This transition shall occur after receiving:

- a) a DWS Lost message, if this state does not send a Start DWS message; or
- b) a COMINIT Detected message.

Before the transition, this state shall set the ResetStatus state machine variable to UNKNOWN.

**5.14.4.6.3 Transition SP11:SAS\_AwaitALIGN1 to SP12:SAS\_AwaitSNW**

This transition shall occur:

- a) if this state receives an ALIGN Received (1) message before the SNTT timer expires.

This indicates that the attached phy has been able to achieve dword synchronization in the current SNW.

**5.14.4.6.4 Transition SP11:SAS\_AwaitALIGN1 to SP14:SAS\_Fail**

This transition shall occur:

- a) if the SNTT timer expires.

This indicates that the attached phy has not been able to achieve dword synchronization in the current SNW.

**5.14.4.7 SP12:SAS\_AwaitSNW state****5.14.4.7.1 State description**

This state shall repeatedly send Transmit ALIGN (1) messages to the SP transmitter.

If the Current SNW state machine variable is set to Final-SNW, then this state shall send a Start SL\_IR Receiver confirmation to the link layer.

Each time this state receives a DWS Lost message, this state may send a Start DWS message to the SP\_DWS state machine to re-acquire dword synchronization without running a new link reset sequence.

This state waits for the SNTT timer to expire or for a Stop SNTT request.

If this state receives a Stop SNTT request, then this state shall stop the SNTT timer.

**5.14.4.7.2 Transition SP12:SAS\_AwaitSNW to SP0:OOB\_COMINIT**

This transition shall occur after receiving:

- a) a DWS Lost message, if this state does not send a Start DWS message; or
- b) a COMINIT Detected message.

Before the transition, this state shall set the ResetStatus state machine variable to UNKNOWN.

**5.14.4.7.3 Transition SP12:SAS\_AwaitSNW to SP13:SAS\_Pass**

This transition shall occur after:

- a) the SNTT timer expires; or
- b) receiving a Stop SNTT request.

**5.14.4.8 SP13:SAS\_Pass state****5.14.4.8.1 State description**

This state determines if:

- a) another SAS SNW is required; or
- b) the SAS speed negotiation sequence is complete.

Each time this state receives a DWS Lost message, this state may send a Start DWS message to the SP\_DWS state machine to re-acquire dword synchronization without running a new link reset sequence.

**5.14.4.8.2 Transition SP13:SAS\_Pass to SP0:OOB\_COMINIT**

This transition shall occur after receiving:

- a) a DWS Lost message, if this state does not send a Start DWS message; or
- b) a COMINIT Detected message.

Before the transition, this state shall set the ResetStatus state machine variable to UNKNOWN.

**5.14.4.8.3 Transition SP13:SAS\_Pass to SP8:SAS\_Start**

This transition shall occur:

- a) if the Current SNW state machine variable is not set to Final-SNW.

**5.14.4.8.4 Transition SP13:SAS\_Pass to SP15:SAS\_PHY\_Ready**

This transition shall occur:

- a) if the Current SNW state machine variable is set to Final-SNW.

**5.14.4.9 SP14:SAS\_Fail state****5.14.4.9.1 State description**

This state determines if:

- a) another SAS SNW is required; or
- b) the SAS speed negotiation sequence is complete.

**5.14.4.9.2 Transition SP14:SAS\_Fail to SP1:OOB\_AwaitCOMX**

This transition shall occur:

- a) if the Current SNW state machine variable is set to Final-SNW.

Before the transition, this state shall set the ResetStatus state machine variable to PHY\_RESET\_PROBLEM.

**5.14.4.9.3 Transition SP14:SAS\_Fail to SP8:SAS\_Start**

This transition shall occur:

- a) if the Current SNW state machine variable is not set to Final-SNW.

**5.14.4.10 SP15:SAS\_PHY\_Ready state****5.14.4.10.1 State description**

This state waits for:

- a) a COMINIT Detected message;
- b) a DWS Lost message;
- c) a DWS Reset message;
- d) a PS Reset message; or
- e) a Manage Phy Power Conditions request.

Upon entry into this state, this state shall:

- a) if multiplexing is:
  - A) enabled (see table 72), then:
    - 1) send a Transmit MUX Sequence message to the SP transmitter; and
    - 2) after receiving MUX Sequence Transmitted, send a Phy Layer Ready (SAS) confirmation to the link layer;
  - or
  - B) not enabled, then send a Phy Layer Ready (SAS) confirmation to the link layer;
- and
- b) if the SP transmitter is transmitting at:
  - A) 1.5 Gbit/s, then set the ResetStatus state machine variable to G1;
  - B) 3 Gbit/s, then set the ResetStatus state machine variable to G2;
  - C) 6 Gbit/s, then set the ResetStatus state machine variable to G3;
  - D) 12 Gbit/s, then set the ResetStatus state machine variable to G4; or
  - E) 22.5 Gbit/s, then:
    - a) send an Enable APTA confirmation to the management application layer;
    - b) send an Enable APTA message to the PAPTA State machines;

- c) set the ResetStatus state machine variable to G5;
- d) send a Start PS message to the SP\_PS state machine;
- e) send a Stop DWS message to the SP\_DWS state machine; and
- f) initialize and start the Scrambler Initialization timer.

When the Scrambler Initialization timer expires, this state shall:

- 1) send a Transmit PACKET\_SYNC message to the SP transmitter; and
- 2) initialize and start the Scrambler Initialization timer.

While in this state, dwords and SPL packets from the link layer are transmitted at the negotiated physical link rate (i.e., the rate established in the previous SNW).

If multiplexing is disabled, then each time this state receives a DWS Lost message, this state may send a Start DWS message to the SP\_DWS state machine to re-acquire dword synchronization without running a new link reset sequence.

If a Manage Phy Power Conditions (Stop DWS) request is received, then this state shall send a Stop DWS message to the SP\_DWS state machine.

If a Manage Phy Power Conditions (Stop PS) request is received, then this state shall send a Stop PS message to the SP\_PS state machine and a Stop Resync message to the SP\_ReSync state machine.

#### **5.14.4.10.2 Transition SP15:SAS\_PHY\_Ready to SP0:OOB\_COMINIT**

If optical mode is enabled or D.C. mode is enabled, then this transition shall occur after receiving:

- a) a DWS Lost message, if this state does not send a Start DWS message;
- b) a DWS Lost message followed by a COMINIT Detected message, if this state does not send a Start DWS message;
- c) a DWS Reset message; or
- d) a PS Reset message.

If optical mode is enabled, then this transition shall not occur after receiving a COMINIT Detected message:

- a) before receiving a DWS Lost message; or
- b) after sending a Start DWS message.

#### **5.14.4.10.3 Transition SP15:SAS\_PHY\_Ready to SP31:SAS\_PS\_Low\_Phy\_Power**

This transition shall occur after this state receives:

- a) a Manage Phy Power Conditions (Enter Partial) request; or
- b) a Manage Phy Power Conditions (Enter Slumber) request.

If this transition is the result of this state receiving a Manage Phy Power Conditions (Enter Partial) request, then the transition shall include:

- a) a Partial argument.

If this transition is the result of this state receiving a Manage Phy Power Conditions (Enter Slumber) request, then the transition shall include:

- a) a Slumber argument.

#### **5.14.4.11 SP27:SAS\_Settings state**

##### **5.14.4.11.1 State description**

This state transmits and receives phy capabilities bits.

Upon entry to this state, this state shall:

- a) initialize and start the SNTT timer;

- b) set the Commonly Supported Settings state machine variable to indicate that there are no commonly supported settings; and
- c) send a Transmit Phy Capabilities Bits message to the SP transmitter.

If a Phy Capabilities Bits Received message is received with the argument of Good Parity, then this state shall set the Commonly Supported Settings state machine variable to the commonly supported settings.

After this state receives a Phy Capabilities Bits Transmitted message this state shall request that OOB idle be transmitted by repeatedly sending Transmit OOB Idle messages to the SP transmitter.

This state waits for the SNTT timer to expire.

#### **5.14.4.11.2 Transition SP27:SAS\_Settings to SP0:OOB\_COMINIT**

This transition shall occur:

- a) after receiving a COMINIT Detected message.

Before the transition, this state shall set the ResetStatus state machine variable to UNKNOWN.

#### **5.14.4.11.3 Transition SP27:SAS\_Settings to SP1:OOB\_AwaitCOMX**

This transition shall occur after the SNTT timer expires if:

- a) a Phy Capabilities Bits Received message is received with an argument of Bad Parity; or
- b) no commonly supported settings exist after the Commonly Supported Settings state machine variable is set as a result of receiving a Phy Capabilities Bits Received message.

Before the transition, this state shall:

- a) if a Phy Capabilities Bits Received message is received with an argument of Bad Parity, then set the ResetStatus state machine variable to PHY\_RESET\_PROBLEM; or
- b) if no commonly supported settings exist after the Commonly Supported Settings state machine variable is set, then set the ResetStatus state machine variable to UNSUPPORTED\_PHY\_ATTACHED.

#### **5.14.4.11.4 Transition SP27:SAS\_Settings to SP8:SAS\_Start**

This transition shall occur if:

- a) the SNTT timer expires; and
- b) a Phy Capabilities Bits Received message is not received during this state.

#### **5.14.4.11.5 Transition SP27:SAS\_Settings to SP28:SAS\_TrainSetup**

This transition shall occur:

- a) after the SNTT timer expires; and
- b) if the Commonly Supported Settings state machine variable indicates there is at least one commonly supported setting.

#### **5.14.4.12 SP28:SAS\_TrainSetup**

##### **5.14.4.12.1 State description**

Upon entry into this state:

- 1) this state shall set the Transmitter Training Enabled argument to:
  - A) no, if:
    - a) optical mode is enabled;
    - b) there is an active cable assembly attached to the phy; or
    - c) the highest priority commonly supported setting indicates G1, G2, or G3;

- or
- B) yes, if optical mode is disabled, there is no active cable assembly attached to the phy, and the highest priority commonly supported setting indicates G4 or a higher priority;
- 2) if the highest priority commonly supported setting indicates:
- A) G1, G2, G3, or G4, then this state shall enable the SAS dword mode; or
  - B) G5 or a higher priority, then this state shall enable the SAS packed mode;
- and
- 3) the phy shall:
- A) initialize and start the RCDT timer; and
  - B) send a Set Physical Link Rate message to the SP transmitter, a Set Physical Link Rate message to the SP receiver, and a Set SSC message to the SP transmitter with the arguments set to reflect the highest priority commonly supported setting contained in the Commonly Supported Settings state machine variable.

After the Set Physical Link Rate messages and Set SSC message are sent, the Commonly Supported Settings state machine variable shall be set to indicate that the selected supported settings bit is no longer in common.

#### **5.14.4.12.2 Transition SP28:SAS\_TrainSetup to SP0:OOB\_COMINIT**

This transition shall occur:

- a) after receiving a COMINIT Detected message.

Before the transition, this state shall set the ResetStatus state machine variable to UNKNOWN.

#### **5.14.4.12.3 Transition SP28:SAS\_TrainSetup to SP29:SAS\_Train\_Rx**

This transition shall occur:

- a) after the RCDT timer expires if the Transmitter Training Enabled argument is set to no.

#### **5.14.4.12.4 Transition SP28:SAS\_TrainSetup to SP34:SAS\_Train\_Tx**

This transition shall occur:

- a) after the RCDT timer expires if the Transmitter Training Enabled argument is set to yes.

#### **5.14.4.13 SP34:SAS\_Train\_Tx state**

##### **5.14.4.13.1 State description**

Upon entry into this state, this state shall:

- a) initialize and start the MTTT timer;
- b) initialize and start the Pattern Lock Lost Timeout timer;
- c) send a Disable OOB Detection message to the SP receiver;
- d) send a Transmitter Training (Enable) message to the PTT state machines (see 5.18); and
- e) send a Transmitter Training Started confirmation to the management application layer.

If this state receives a Pattern Lock Lost message, then this state shall initialize and start the Pattern Lock Lost Timeout timer.

If this state receives a Pattern Locked message, then this state shall stop the Pattern Lock Lost Timeout timer.

If the MTTT timer expires or the Pattern Lock Lost Timeout timer expires, then this state shall send a Transmitter Training (Disable) message to the PTT state machines.

A phy reset problem occurs if:

- a) the MTTT timer expires or the Pattern Lock Lost Timeout timer expires; and



- b) the Commonly Supported Settings state machine variable does not contain additional commonly supported settings.

#### 5.14.4.13.2 Transition SP34:SAS\_Train\_Tx to SP1:OOB\_AwaitCOMX

This transition shall occur:

- a) if a phy reset problem occurs.

Before the transition, this state shall:

- a) set the ResetStatus state machine variable to PHY\_RESET\_PROBLEM;
- b) send an Enable OOB Detection message to the SP receiver;
- c) stop the MTTT timer; and
- d) stop the Pattern Lock Lost Timeout timer.

#### 5.14.4.13.3 Transition SP34:SAS\_Train\_Tx to SP28:SAS\_TrainSetup

This transition shall occur if:

- a) the MTTT timer expires or the Pattern Lock Lost Timeout timer expires; and
- b) the Commonly Supported Settings state machine variable contains additional commonly supported settings.

Before the transition, this state shall:

- a) stop the MTTT timer;
- b) stop the Pattern Lock Lost Timeout timer; and
- c) send an Enable OOB Detection message to the SP receiver.

#### 5.14.4.13.4 Transition SP34:SAS\_Train\_Tx to SP29:SAS\_Train\_Rx

This transition shall occur if:

- a) the MTTT timer has not expired;
- b) the Pattern Lock Lost Timeout timer has not expired; and
- c) this state receives a Transmitter Training Complete message.

Before the transition, this state shall:

- a) stop the MTTT timer;
- b) stop the Pattern Lock Lost Timeout timer; and
- c) send an Enable OOB Detection message to the SP receiver.

#### 5.14.4.14 SP29:SAS\_Train\_Rx state

##### 5.14.4.14.1 State description

Upon entry into this state, this state shall:

- a) initialize and start the MRTT timer;
- b) initialize and start the TLT timer;
- c) send a Start Rx Training message to the SP receiver; and
- d) send a Start DWS message to the SP\_DWS state machine if the phy is in the SAS dword mode.

If the phy is in the SAS dword mode, then:

- a) this state shall repeatedly send Transmit TRAIN Pattern messages to the SP transmitter; and
- b) each time this state receives a DWS Lost message, this state shall send a Start DWS message to the SP\_DWS state machine to re-acquire dword synchronization.

If the phy is in the SAS packet mode, then this state shall repeatedly send Transmit PACKET\_SYNC\_LOST messages to the SP transmitter.

If the MRTT timer expires, then this state shall send an Abort Rx Training message to the SP receiver.

A phy reset problem occurs if:

- a) the MRTT timer expires; and
- b) the Commonly Supported Settings state machine variable does not contain additional commonly supported settings.

#### **5.14.4.14.2 Transition SP29:SAS\_Train\_Rx to SP0:OOB\_COMINIT**

This transition shall occur after receiving a COMINIT Detected message.

Before the transition, this state shall:

- a) set the ResetStatus state machine variable to UNKNOWN;
- b) stop the TLT timer; and
- c) stop the MRTT timer.

#### **5.14.4.14.3 Transition SP29:SAS\_Train\_Rx to SP1:OOB\_AwaitCOMX**

This transition shall occur if a phy reset problem occurs.

Before the transition, this state shall:

- a) set the ResetStatus state machine variable to PHY\_RESET\_PROBLEM;
- b) stop the TLT timer; and
- c) stop the MRTT timer.

#### **5.14.4.14.4 Transition SP29:SAS\_Train\_Rx to SP28:SAS\_TrainSetup**

This transition shall occur if:

- a) the MRTT timer expires; and
- b) the Commonly Supported Settings state machine variable contains additional commonly supported settings.

Before the transition, this state shall:

- a) stop the TLT timer; and
- b) stop the MRTT timer.

#### **5.14.4.14.5 Transition SP29:SAS\_Train\_Rx to SP30:SAS\_TrainingDone**

If the phy is in the SAS dword mode, then this transition shall occur if:

- a) the TLT timer has not expired;
- b) this state receives an Rx Training Completed message;
- c) dword synchronization is acquired; and
- d) this state receives a TRAIN Received message or a TRAIN\_DONE Received message.

If the phy is in the SAS dword mode and a TRAIN\_DONE Received message was received, then the transition shall include a TRAIN\_DONE Received argument.

If the phy is in the SAS packet mode, then this transition shall occur if:

- a) the TLT timer has not expired;
- b) this state receives an Rx Training Completed message; and
- c) this state receives a PACKET\_SYNC\_LOST Received message or a PACKET\_SYNC Received message.

If the phy is in the SAS packet mode and a PACKET\_SYNC Received message was received, then the transition shall include a PACKET\_SYNC Received argument.

Before the transition, this state shall stop the TLT timer.

**5.14.4.15 SP30:SAS\_TrainingDone state****5.14.4.15.1 State description**

If the phy is in the SAS dword mode, then this state shall repeatedly send Transmit TRAIN\_DONE Pattern messages to the SP transmitter.

Each time this state receives a DWS Lost message, this state may send a Start DWS message to the SP\_DWS state machine to re-acquire dword synchronization without running a new link reset sequence.

If the phy is in the SAS packet mode, then this state shall repeatedly send Transmit PACKET\_SYNC messages to the SP transmitter.

This state waits for the MRTT timer to expire, TRAIN\_DONE Received message, or a PACKET\_SYNC Received message from the receiver.

If this state receives a TRAIN\_DONE Received message or PACKET\_SYNC Received message, then this state shall stop the MRTT timer.

This state shall send a Start SL\_IR Receiver confirmation to the link layer when a TRAIN\_DONE Received message or PACKET\_SYNC Received message is received.

A phy reset problem occurs if:

- a) a TRAIN\_DONE Received message or PACKET\_SYNC Received message is not received before the MRTT timer expires; and
- b) the Commonly Supported Settings state machine variable does not contain additional commonly supported settings.

**5.14.4.15.2 Transition SP30:SAS\_TrainingDone to SP0:OOB\_COMINIT**

This transition shall occur after receiving:

- a) a DWS Lost message if this state does not send a Start DWS message; or
- b) a COMINIT Detected message.

Before the transition, this state shall:

- a) set the ResetStatus state machine variable to UNKNOWN; and
- b) stop the MRTT timer.

**5.14.4.15.3 Transition SP30:SAS\_TrainingDone to SP1:OOB\_AwaitCOMX**

This transition shall occur:

- a) if a phy reset problem occurs.

Before the transition, this state shall set the ResetStatus state machine variable to PHY\_RESET\_PROBLEM.

**5.14.4.15.4 Transition SP30:SAS\_TrainingDone to SP28:SAS\_TrainSetup**

This transition shall occur if:

- a) the MRTT timer expires; and
- b) the Commonly Supported Settings state machine variable contains additional commonly supported settings.

**5.14.4.15.5 Transition SP30:SAS\_TrainingDone to SP15:SAS\_PHY\_Ready**

This transition shall occur if the phy is in the SAS dword mode and this state receives at least four TRAIN\_DONE Pattern Transmitted messages and either:

- a) receives:
  - 1) a TRAIN\_DONE Received message before the MRTT timer expires; and

2) at least one TRAIN\_DONE Pattern Transmitted message;

or

b) was entered with a TRAIN\_DONE Received argument.

This transition shall occur if the phy is in the SAS packet mode and this state receives at least four PACKET\_SYNC Transmitted messages and either:

a) receives:

1) a PACKET\_SYNC Received message before the MRTT timer expires; and

2) at least one additional PACKET\_SYNC Transmitted message;

or

b) was entered with a PACKET\_SYNC Received argument.

Before the transition, this state shall stop the MRTT timer.

### **5.14.5 SAS phy power conditions states**

#### **5.14.5.1 SAS phy power conditions states overview**

Figure 107 shows the SAS phy power conditions states. These states are entered when a phy is requested to enter a low phy power condition and process the actions that return a phy from a low phy power condition to the active phy power condition.

These states are indicated by state names with a prefix of SAS\_PS.

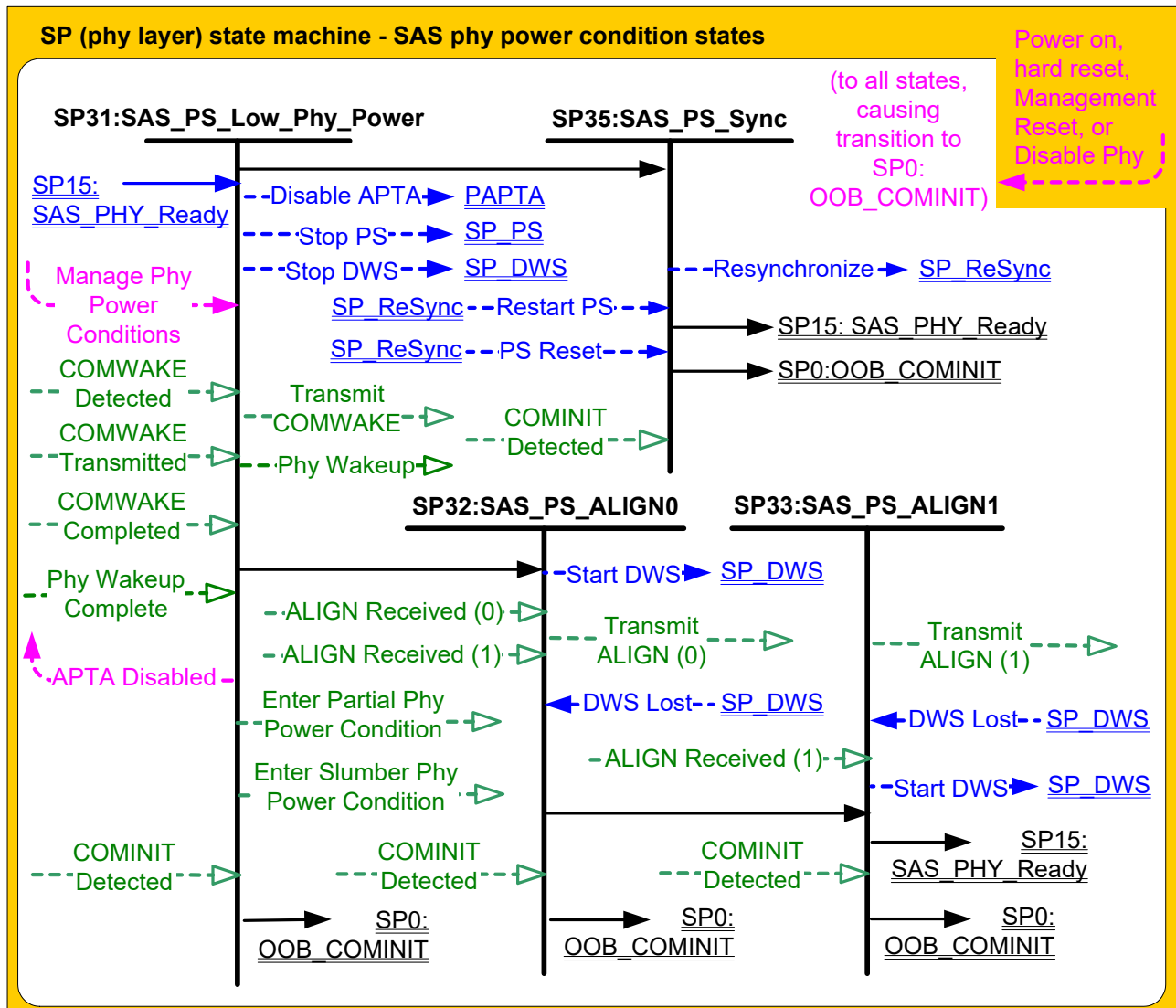


Figure 107 – SP (phy layer) state machine - SAS phy power condition states

#### 5.14.5.2 SP31:SAS\_PS\_Low\_Phy\_Power state

##### 5.14.5.2.1 State description

Upon entry into this state, this state shall:

- send an APTA Disabled (Low Phy Power Condition) confirmation to the management application layer;
- send a Disable APTA message to the PAPTAP State machines;
- send a Stop DWS message to the SP\_DWS state machine;
- send a Stop PS message to the SP\_PS state machine; and
- if applicable, then save any vendor specific information for the SP transmitter and SP receiver (e.g., determined from the previous Train\_Rx-SNW and Train\_Tx-SNW with the arguments set to the same values as those for the previous entry into the SP28:SAS\_TrainSetup state (see 5.14.4.12)).

If this state is entered with a Partial argument, then:

- a) this state shall send an Enter Partial Phy Power Condition message to the SP transmitter and SP receiver;
- b) the phy shall enter the partial phy power condition (see 4.10.1.3); and
- c) this state shall set the SASPhyPwrCond state machine variable to Partial.

If this state is entered with a Slumber argument, then:

- a) this state shall send an Enter Slumber Phy Power Condition message to the SP transmitter and SP receiver;
- b) the phy shall enter the slumber phy power condition (see 4.10.1.4); and
- c) this state shall set the SASPhyPwrCond state machine variable to Slumber.

If this state receives a Manage Phy Power Conditions (Exit) request or a COMWAKE Detected message, then this state shall:

- 1) send a Phy Wakeup message to the SP transmitter;
- 2) wait for a Phy Wakeup Completed message; and
- 3) send a Transmit COMWAKE message to the SP transmitter.

While in this state, if a Manage Phy Power Conditions (Exit) request and a COMWAKE Detected message are received, then this state shall only send one Transmit COMWAKE message to the SP transmitter.

#### **5.14.5.2.2 Transition SP31:SAS\_PS\_Low\_Phy\_Power to SP0:OOB\_COMINIT**

This transition shall occur if:

- a) this state:
  - A) receives a COMWAKE Transmitted message; and
  - B) does not receive a COMWAKE Completed message within a hot-plug timeout (see table 85);
 or
- b) this state receives a COMINIT Detected message.

#### **5.14.5.2.3 Transition SP31:SAS\_PS\_Low\_Phy\_Power to SP32:SAS\_PS\_ALIGN0**

If the phy is in the SAS dword mode, then this transition shall occur after this state receives:

- a) a COMWAKE Transmitted message; and
- b) a COMWAKE Completed message.

#### **5.14.5.2.4 Transition SP31:SAS\_PS\_Low\_Phy\_Power to SP35:SAS\_PS\_Sync**

If the phy is in the SAS packet mode, then this transition shall occur after this state receives:

- a) a COMWAKE Transmitted message; and
- b) a COMWAKE Completed message.

### **5.14.5.3 SP32:SAS\_PS\_ALIGN0 state**

#### **5.14.5.3.1 State description**

Upon entry into this state, this state shall:

- 1) initialize and start the SNTT timer and the SNLT timer;
- 2) send a Set Physical Link Rate message to the SP transmitter and to the SP receiver and send a Set SSC message to the SP transmitter with the arguments set to those determined from the last SNW;
- 3) if applicable, then restore any vendor specific information for the SP transmitter and SP receiver (see 5.14.5.2.1);
- 4) send a Start DWS message to the SP\_DWS state machine; and
- 5) repeatedly send Transmit ALIGN (0) messages to the SP transmitter.

Each time this state receives a DWS Lost message, this state may send a Start DWS message to re-acquire dword synchronization without running a new link reset sequence.

#### **5.14.5.3.2 Transition SP32:SAS\_PS\_ALIGN0 state to SP0:OOB\_COMINIT**

This transition shall occur after this state:

- a) receives a DWS Lost message, if this state does not send a Start DWS message;
- b) receives a COMINIT Detected message; or
- c) the SNTT timer expires.

Before the transition, this state shall stop the SNLT timer and the SNTT timer.

#### **5.14.5.3.3 Transition SP32:SAS\_PS\_ALIGN0 to SP33:SAS\_PS\_ALIGN1**

This transition shall occur:

- a) if this state receives an ALIGN Received (0) message or an ALIGN Received (1) message before the SNLT timer expires; and
- b) after this state has sent at least three Transmit ALIGN (0) messages.

Before the transition, this state shall stop the SNLT timer.

#### **5.14.5.4 SP33:SAS\_PS\_ALIGN1 state**

##### **5.14.5.4.1 State description**

Upon entry into this state, this phy shall repeatedly send Transmit ALIGN (1) messages to the SP transmitter.

Each time this state receives a DWS Lost message, this state may send a Start DWS message to re-acquire dword synchronization without running a new link reset sequence.

If this state receives an ALIGN Received (1) message before the SNTT timer expires, then this state shall set the SASPhyPwrCond state machine variable to Active.

##### **5.14.5.4.2 Transition SP33:SAS\_PS\_ALIGN1 state to SP0:OOB\_COMINIT**

This transition shall occur after this state:

- a) receives a DWS Lost message, if this state does not send a Start DWS message;
- b) receives a COMINIT Detected message; or
- c) the SNTT timer expires.

Before the transition, this state shall stop the SNLT timer and the SNTT timer.

##### **5.14.5.4.3 Transition SP33:SAS\_PS\_ALIGN1 state to SP15:SAS\_PHY\_Ready**

This transition shall occur:

- a) if this state receives an ALIGN Received (1) message before the SNTT timer expires;
- b) after this state has sent at least three Transmit ALIGN (1) messages;
- c) after this state has set the SASPhyPwrCond state machine variable to Active; and
- d) after this state has stopped the SNLT timer and the SNTT timer.

#### **5.14.5.5 SP35:SAS\_PS\_Sync state**

##### **5.14.5.5.1 State description**

Upon entry into this state, this state shall:

- 1) send a Set Physical Link Rate message to the SP transmitter and to the SP receiver and send a Set SSC message to the SP transmitter with the arguments set to those determined from the last SNW;

- 2) if applicable, then restore any vendor specific information for the SP transmitter and SP receiver (see 5.14.5.2.1); and
- 3) send a Resynchronize message to the SP\_ReSync state machine.

#### **5.14.5.5.2 Transition SP35:SAS\_PS\_SYNC state to SP0:OOB\_COMINIT**

This transition shall occur after this state receives:

- a) a PS Reset message; or
- b) a COMINIT Detected message.

#### **5.14.5.5.3 Transition SP35:SAS\_PS\_SYNC state to SP15:SAS\_PHY\_Ready**

This transition shall occur after this state receives:

- a) a Restart PS message.

### **5.14.6 SATA host emulation states**

#### **5.14.6.1 SATA host emulation states overview**

Figure 108 shows the SATA host emulation states, in which the phy has detected that it is attached to a SATA phy and behaves as if it were a SATA host phy initiating the SATA speed negotiation sequence. These states are indicated by state names with a prefix of SATA.

The power management states defined in this standard are for SAS initiator phys or expander phys that support SATA. SATA low phy power conditions may be enabled in expander phys using the SMP PHY CONTROL function (see 9.4.3.28).



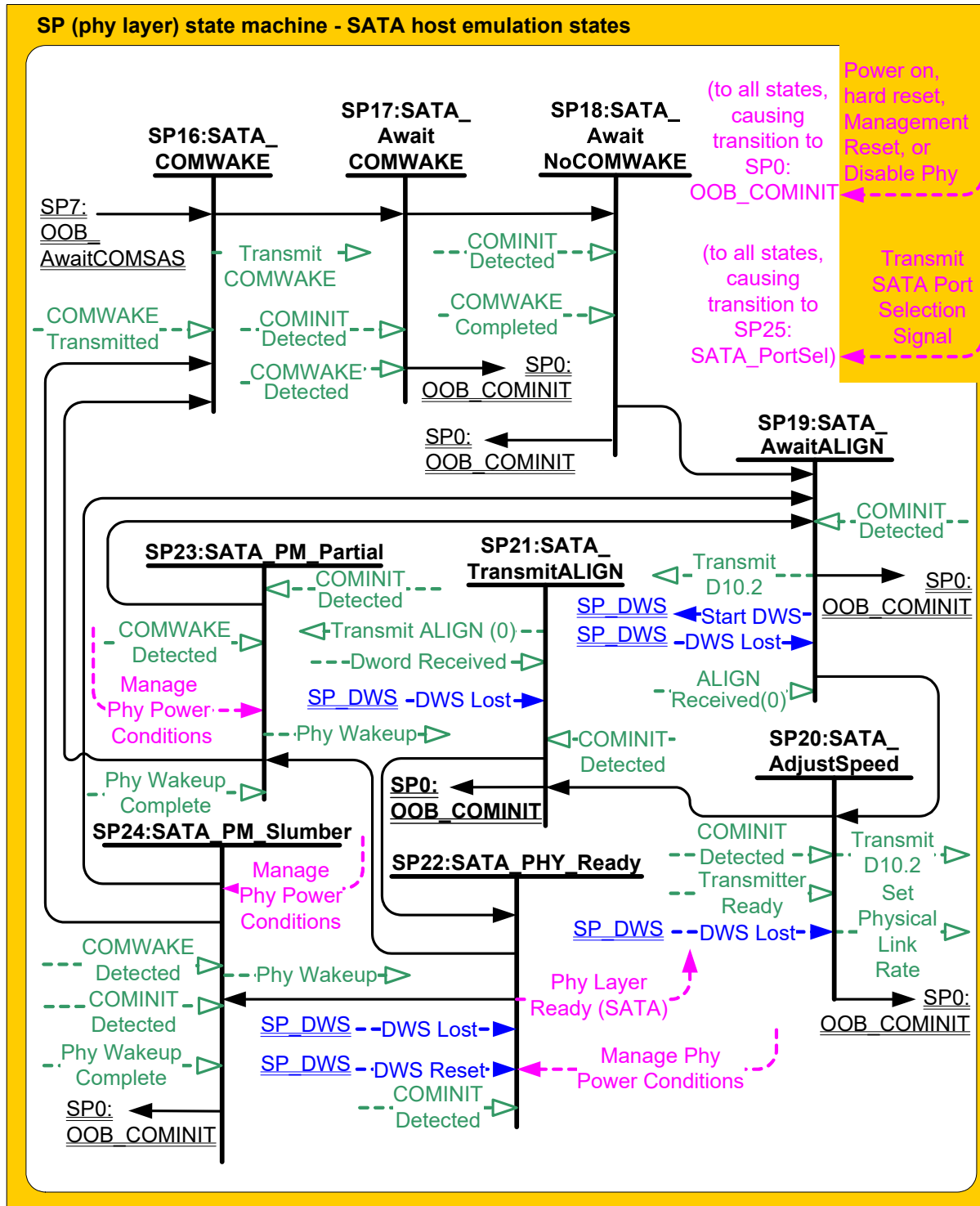


Figure 108 – SP (phy layer) state machine - SATA host emulation states

**5.14.6.2 SP16:SATA\_COMWAKE state****5.14.6.2.1 State description**

This state shall send a Transmit COMWAKE message to the SP transmitter and wait for a COMWAKE Transmitted message.

**5.14.6.2.2 Transition SP16:SATA\_COMWAKE to SP0:OOB\_COMINIT**

This transition shall occur:

- a) after receiving a COMINIT Detected message.

**5.14.6.2.3 Transition SP16:SATA\_COMWAKE to SP17:SATA\_AwaitCOMWAKE**

This transition shall occur:

- a) after receiving a COMWAKE Transmitted message.

**5.14.6.3 SP17:SATA\_AwaitCOMWAKE state****5.14.6.3.1 State description**

This state waits for a COMWAKE Detected message to be received.

**5.14.6.3.2 Transition SP17:SATA\_AwaitCOMWAKE to SP0:OOB\_COMINIT**

This transition shall occur:

- a) after receiving a COMINIT Detected message.

**5.14.6.3.3 Transition SP17:SATA\_AwaitCOMWAKE to SP18:SATA\_AwaitNoCOMWAKE**

This transition shall occur:

- a) after receiving a COMWAKE Detected message.

**5.14.6.4 SP18:SATA\_AwaitNoCOMWAKE state****5.14.6.4.1 State description**

This state waits for a COMWAKE Completed message.

**5.14.6.4.2 Transition SP18:SATA\_AwaitNoCOMWAKE to SP0:OOB\_COMINIT**

This transition shall occur:

- a) after receiving a COMINIT Detected message.

Before the transition, this state shall set the ResetStatus state machine variable to UNKNOWN.

**5.14.6.4.3 Transition SP18:SATA\_AwaitNoCOMWAKE to SP19:SATA\_AwaitALIGN**

This transition shall occur:

- a) after receiving a COMWAKE Completed message.

**5.14.6.5 SP19:SATA\_AwaitALIGN state****5.14.6.5.1 State description**

Upon entry into this state, this state shall send a Start DWS message to the SP\_DWS state machine.

This state shall:

- a) repeatedly send Transmit D10.2 messages to the SP transmitter;
- b) initialize and start the Await ALIGN Timeout timer; and
- c) wait for an ALIGN Received (0) message to be received or for the Await ALIGN Timeout timer to expire.

The phy shall start transmitting D10.2 characters no later than a COMWAKE response time (see 5.11.2.2) after entry into this state.

**5.14.6.5.2 Transition SP19:SATA\_AwaitALIGN to SP0:OOB\_COMINIT**

This transition shall occur:

- a) if the Await ALIGN Timeout timer expires;
- b) after receiving a DWS Lost message; or
- c) after receiving a COMINIT Detected message.

Before the transition, this state shall:

- a) if the Await ALIGN Timeout timer expires, then set the ResetStatus state machine variable to UNSUPPORTED\_PHY\_ATTACHED;
- b) after receiving a DWS Lost message, set the ResetStatus state machine variable to UNKNOWN; or
- c) after receiving a COMINIT Detected message, set the ResetStatus state machine variable to UNKNOWN.

**5.14.6.5.3 Transition SP19:SATA\_AwaitALIGN to SP20:SATA\_AdjustSpeed**

This transition shall occur:

- a) if this state receives an ALIGN Received (0) message before the Await ALIGN Timeout timer expires.

The ALIGN Received (0) message indicates an ALIGN (0) was received at any of the physical link rates supported by this phy.

**5.14.6.6 SP20:SATA\_AdjustSpeed state****5.14.6.6.1 State description**

This state waits for the SP transmitter to adjust to the same physical link rate of the ALIGNs that were detected by the receiver circuitry.

This state shall:

- 1) send a Set Physical Link Rate message to the SP transmitter with an argument specifying the physical link rate of the ALIGNs that were detected by the receiver circuitry; and
- 2) repeatedly send Transmit D10.2 messages to the SP transmitter.

**5.14.6.6.2 Transition SP20:SATA\_AdjustSpeed to SP0:OOB\_COMINIT**

This transition shall occur after receiving:

- a) a DWS Lost message; or
- b) a COMINIT Detected message.

Before the transition, this state shall set the ResetStatus state machine variable to UNKNOWN.

**5.14.6.6.3 Transition SP20:SATA\_AdjustSpeed to SP21:SATA\_TransmitALIGN**

This transition shall occur:

- a) after receiving a Transmitter Ready message.

**5.14.6.7 SP21:SATA\_TransmitALIGN state****5.14.6.7.1 State description**

This state shall repeatedly send Transmit ALIGN (0) messages to the SP transmitter.

**5.14.6.7.2 Transition SP21:SATA\_TransmitALIGN to SP0:OOB\_COMINIT**

This transition shall occur after receiving:

- a) a DWS Lost message; or
- b) a COMINIT Detected message.

Before the transition, this state shall set the ResetStatus state machine variable to UNKNOWN.

**5.14.6.7.3 Transition SP21:SATA\_TransmitALIGN to SP22:SATA\_PHY\_Ready**

This transition shall occur:

- a) after receiving three consecutive Dword Received messages containing primitives other than ALIGN (0).

**5.14.6.8 SP22:SATA\_PHY\_Ready state****5.14.6.8.1 State description**

While in this state dwords from the link layer are transmitted at the negotiated physical link rate (i.e., the rate established in the previous state).

Upon entry into this state, this state shall:

- a) if the SP transmitter is transmitting at 1.5 Gbit/s, then set the ResetStatus state machine variable to G1;
- b) if the SP transmitter is transmitting at 3 Gbit/s, then set the ResetStatus state machine variable to G2; or
- c) if the SP transmitter is transmitting at 6 Gbit/s, then set the ResetStatus state machine variable to G3.

This state shall send a Phy Layer Ready (SATA) confirmation to the link layer.

This state waits for a COMINIT Detected message, a DWS Lost message, or a DWS Reset message.

Each time this state receives a DWS Lost message, this state may send a Start DWS message to the SP\_DWS state machine to re-acquire dword synchronization without running a new link reset sequence.

**5.14.6.8.2 Transition SP22:SATA\_PHY\_Ready to SP0:OOB\_COMINIT**

This transition shall occur after receiving:

- a) a DWS Lost message, if this state does not send a Start DWS message;
- b) a DWS Lost message followed by a COMINIT Detected message, if this state does not send a Start DWS message; or
- c) a DWS Reset message.

**5.14.6.8.3 Transition SP22:SATA\_PHY\_Ready to SP23:SATA\_PM\_Partial**

This transition shall occur:

- a) after receiving a Manage Phy Power Conditions (Enter Partial) request.

**5.14.6.8.4 Transition SP22:SATA\_PHY\_Ready to SP24:SATA\_PM\_Slumber**

This transition shall occur:

- a) after receiving a Manage Phy Power Conditions (Enter Slumber) request.

**5.14.6.9 SP23:SATA\_PM\_Partial state****5.14.6.9.1 State description**

Upon entry into this state, this state shall set the SASPhyPwrCond state machine variable to Partial.

This state waits for a COMWAKE Detected message or a Manage Phy Power Conditions (Exit) request.

When this state receives a COMWAKE Detected message or a Manage Phy Power Conditions (Exit) request this state shall send a Phy Wakeup message to the SP transmitter.

**5.14.6.9.2 Transition SP23:SATA\_PM\_Partial to SP0:OOB\_COMINIT**

This transition shall occur:

- a) after receiving a COMINIT Detected message.

Before the transition, this state shall set the ResetStatus state machine variable to UNKNOWN.

**5.14.6.9.3 Transition SP23:SATA\_PM\_Partial to SP16:SATA\_COMWAKE**

This transition shall occur:

- a) if this state receives a Manage Phy Power Conditions (Exit) request; and
- b) after this state receives a Phy Wakeup Complete message.

**5.14.6.9.4 Transition SP23:SATA\_PM\_Partial to SP19:SATA\_AwaitALIGN**

This transition shall occur after this state receives a:

- a) Phy Wakeup Complete message; and
- b) COMWAKE Completed message.

**5.14.6.10 SP24:SATA\_PM\_Slumber state****5.14.6.10.1 State description**

Upon entry into this state, this state shall set the SASPhyPwrCond state machine variable to Slumber.

This state waits for a COMWAKE Detected message or a Manage Phy Power Conditions (Exit) request.

When this state receives a COMWAKE Detected message or a Manage Phy Power Conditions (Exit) request this state shall send a Phy Wakeup message to the SP transmitter.

**5.14.6.10.2 Transition SP24:SATA\_PM\_Slumber to SP0:OOB\_COMINIT**

This transition shall occur:

- a) after receiving a COMINIT Detected message.

Before the transition, this state shall set the ResetStatus state machine variable to UNKNOWN.

#### 5.14.6.10.3 Transition SP24:SATA\_PM\_Slumber to SP16:SATA\_COMWAKE

This transition shall occur:

- if this state receives a Manage Phy Power Conditions (Exit) request; and
- after this state receives a Phy Wakeup Complete message.

#### 5.14.6.10.4 Transition SP24:SATA\_PM\_Slumber to SP19:SATA\_AwaitALIGN

This transition shall occur after this state receives a:

- Phy Wakeup Complete message; and
- COMWAKE Completed message.

### 5.14.7 SATA port selector state SP25:SATA\_PortSel

#### 5.14.7.1 State description

Figure 109 shows the SP25:SATA\_PortSel state. This state controls transmission of the SATA port selection signal when a specified phy processes a Transmit SATA Port Selection Signal request.

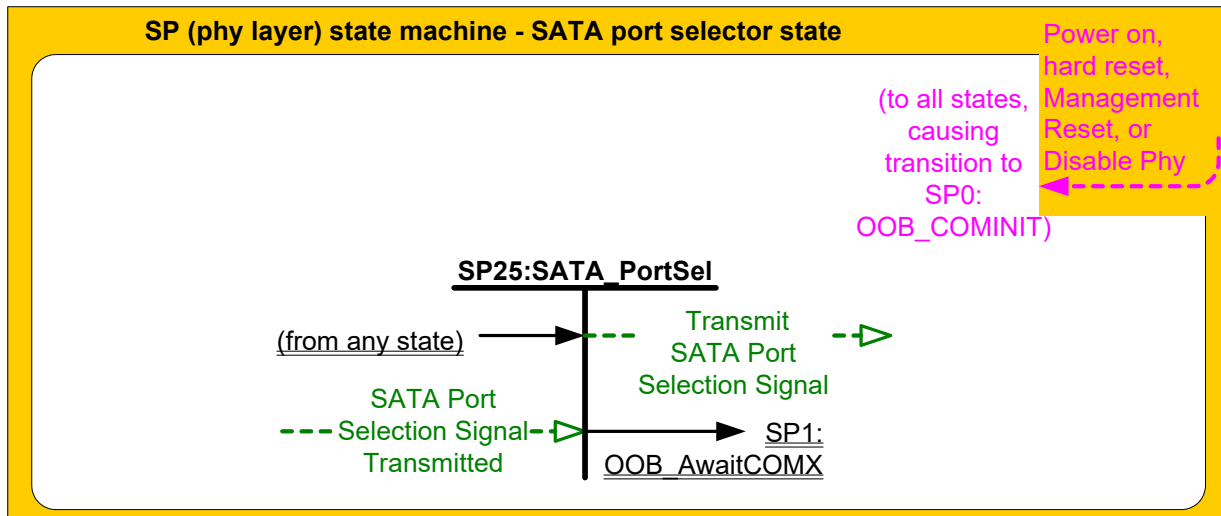


Figure 109 – SP (phy layer) state machine – SATA port selector state

Upon entry into this state, this state shall:

- set the ResetStatus state machine variable to UNKNOWN;
- send a Transmit SATA Port Selection Signal message to the SP transmitter;
- set the ATTACHED SATA PORT SELECTOR bit to zero in the SMP DISCOVER response (see 9.4.3.10); and
- set the ATTACHED SATA DEVICE bit to zero in the SMP DISCOVER response.

#### 5.14.7.2 Transition SP25:SATA\_PortSel to SP1:OOB\_AwaitCOMX

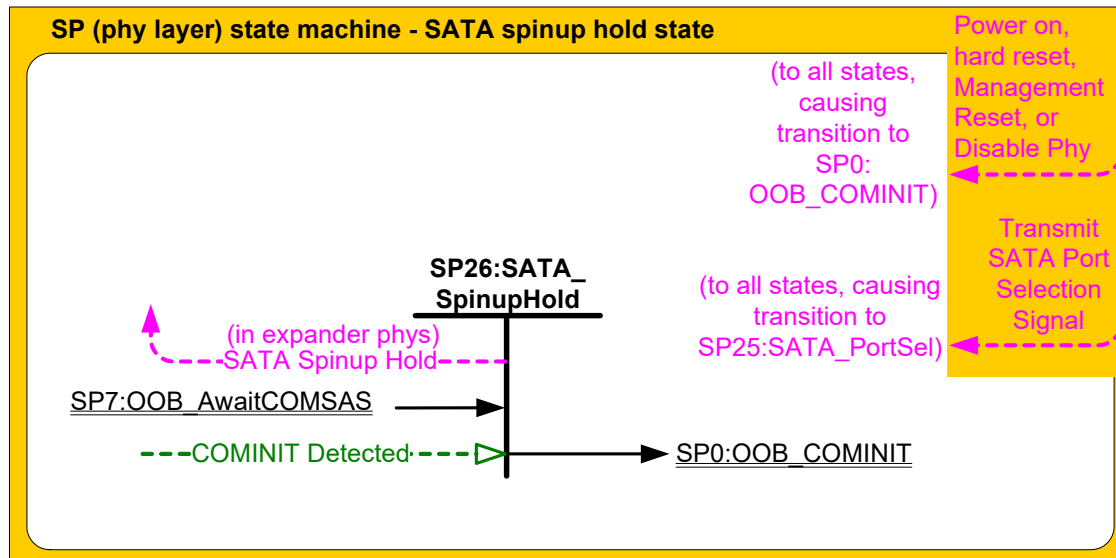
This transition shall occur:

- after receiving a SATA Port Selection Signal Transmitted message.

### 5.14.8 SATA spinup hold state SP26:SATA\_SpinupHold

#### 5.14.8.1 State description

Figure 110 shows the SP26:SATA\_SpinupHold state.



**Figure 110 – SP (phy layer) state machine - SATA spinup hold state**

Upon entry into this state, this state shall:

- a) if the ResetStatus state machine variable is set to SPINUP\_HOLD, then not change the ResetStatus state machine variable; or
- b) if the ResetStatus state machine variable is not set to SPINUP\_HOLD, then:
  - A) set the ResetStatus state machine variable to SPINUP\_HOLD; and
  - B) if this state machine is in an expander phy, then send a SATA Spinup Hold confirmation to the link layer.

#### 5.14.8.2 Transition SP26:SATA\_SpinupHold to SP0:OOB\_COMINIT

This transition shall occur:

- a) if this state receives a COMINIT Detected message.

## 5.15 SP\_DWS (phy layer dword synchronization) state machine

### 5.15.1 SP\_DWS state machine overview

Each phy includes an SP\_DWS state machine and an SP\_DWS receiver.

The SP\_DWS state machine establishes the same dword boundaries at the receiver as at the attached transmitter by searching for control characters. The SP\_DWS receiver monitors and decodes the incoming data stream and forces K28.5 characters into the first character position to perform dword alignment when requested by the SP\_DWS state machine. K28.5 characters with either positive or negative disparity shall be accepted. The SP\_DWS receiver continues to reestablish dword alignment by forcing received K28.5 characters into the first character position until a K28.5-based primitive (i.e., K28.5, Dxx.y, Dxx.y, Dxx.y) with correct disparity on each data character is detected. The resultant primitives, dwords, and valid dword indicators (e.g., encoding error indicators) are sent to this state machine to enable it to determine the dword synchronization policy.

After dword synchronization has been achieved, this state machine evaluates dwords that are received. When an invalid dword is detected, receipt of two valid dwords are required to nullify the effect of receiving the invalid dword. When four invalid dwords are detected without nullification, dword synchronization is considered lost.

While dword synchronization is lost, the data stream received is invalid and dwords shall not be passed to the link layer.

This state machine consists of the following states:

- a) SP\_DWS0:AcquireSync (see 5.15.3) (initial state);
- b) SP\_DWS1:Valid1 (see 5.15.4);
- c) SP\_DWS2:Valid2 (see 5.15.5);
- d) SP\_DWS3:SyncAcquired (see 5.15.6);
- e) SP\_DWS4:Lost1 (see 5.15.7);
- f) SP\_DWS5:Lost1Recovered (see 5.15.8);
- g) SP\_DWS6:Lost2 (see 5.15.9);
- h) SP\_DWS7:Lost2Recovered (see 5.15.10);
- i) SP\_DWS8:Lost3 (see 5.15.11); and
- j) SP\_DWS9:Lost3Recovered (see 5.15.12).

This state machine receives the following requests from the management application layer:

- a) Management Reset; and
- b) Disable Phy.

This state machine shall start in or transition to the SP\_DWS0:AcquireSync state after:

- a) power on;
- b) hard reset;
- c) receiving a Management Reset request (e.g., as a result of processing an SMP PHY CONTROL function requesting a phy operation of LINK RESET or HARD RESET);
- d) receiving a Disable Phy request (e.g., as a result of processing an SMP PHY CONTROL function requesting a phy operation of DISABLE); or
- e) receiving a Stop DWS message from the SP state machine.

This state machine receives the following messages from the SP state machine (see 5.14):

- a) Start DWS; and
- b) Stop DWS.

This state machine sends the following messages to the SP state machine:

- a) DWS Lost; and
- b) DWS Reset.





### 5.15.2 SP\_DWS receiver

The SP\_DWS receiver receives the following messages from the SP\_DWS state machine:

- a) Find Dword; and
- b) Sync Acquired.

The SP\_DWS receiver sends the following messages to the SP\_DWS state machine indicating dwords received from the physical link:

- a) Dword Received (Primitive);
- b) Dword Received (Data Dword);
- c) Dword Received (Invalid); and
- d) Incorrect Mux Received.

When the SP\_DWS receiver receives a Sync Acquired message, the SP\_DWS receiver shall send the most recently received primitive and all subsequent dwords to the link layer state machine receivers (e.g., SL\_IR, SL, SSP, SMP, and XL) through the elasticity buffer (see 6.5) as Dword Received confirmations. If multiplexing is enabled (see table 72), then the SP\_DWS receiver shall use the first incoming MUX to determine the logical phy to which it sends each Dword Received confirmation and shall not send a Dword Received confirmations until it receives the first incoming MUX.

Upon receiving a Find Dword message, the SP\_DWS receiver shall monitor the input data stream and force each K28.5 character detected into the first character position of a possible dword. If the next three characters are data characters with correct disparity, then the SP\_DWS receiver shall send the dword as a Dword Received (Primitive) message to the SP\_DWS state machine. Until the SP\_DWS receiver receives another Find Dword message, for every four characters that it receives the SP\_DWS receiver shall:

- a) send a Dword Received (Invalid) message to the SP\_DWS state machine if the dword is an invalid dword;
- b) send a Dword Received (Primitive) message to the SP\_DWS state machine if the dword is a primitive (i.e., the dword contains a K28.5 character in the first character position followed by three data characters); or
- c) send a Dword Received (Data Dword) message to the SP\_DWS state machine if the dword is a data dword (i.e., it is not an invalid dword or a primitive).

The SP\_DWS receiver relationship to other receivers is defined in 4.3.3.

### 5.15.3 SP\_DWS0:AcquireSync state

#### 5.15.3.1 State description

This is the initial state of this state machine.

After receiving a Start DWS message, this state shall:

- a) send a Find Dword message to the SP\_DWS receiver;
- b) send a Sync Lost message to the SP receiver; and
- c) initialize and start the DWS Reset Timeout timer.

If this state is entered from SP\_DWS1:Valid1 or SP\_DWS2:Valid2, then:

- a) this state shall send a Find Dword message to the SP\_DWS receiver;
- b) this state shall send a Sync Lost message to the SP receiver; and
- c) the DWS Reset Timeout timer shall continue running if this state was entered without a Timer Expired argument.

If the DWS Reset Timeout timer expires or this state was entered with a Timer Expired argument, then this state may send a DWS Reset message to the SP state machine (e.g., if the phy chooses to initiate a new link reset sequence because dword synchronization has been lost for too long).

This state shall not send a DWS Reset message to the SP until the DWS Reset Timeout timer expires or this state was entered with a Timer Expired argument.

**5.15.3.2 Transition SP\_DWS0:AcquireSync to SP\_DWS1:Valid1**

This transition shall occur:

- a) after sending a Find Dword message; and
- b) after receiving a Dword Received (Primitive) message.

**5.15.4 SP\_DWS1:Valid1 state****5.15.4.1 State description**

This state is reached after one valid primitive has been received. This state waits for a second valid primitive or an invalid dword.

The DWS Reset Timeout timer shall continue running.

**5.15.4.2 Transition SP\_DWS1:Valid1 to SP\_DWS0:AcquireSync**

This transition shall occur:

- a) after receiving a Dword Received (Invalid) message; or
- b) after the DWS Reset Timeout timer expires.

If the DWS Reset Timeout timer has expired, then the transition shall include a Timer Expired argument.

**5.15.4.3 Transition SP\_DWS1:Valid1 to SP\_DWS2:Valid2**

This transition shall occur:

- a) after receiving a Dword Received (Primitive) message.

**5.15.5 SP\_DWS2:Valid2 state****5.15.5.1 State description**

This state is reached after two valid primitives have been received without adjusting the dword synchronization. This state waits for a third valid primitive or an invalid dword.

The DWS Reset Timeout timer shall continue running.

**5.15.5.2 Transition SP\_DWS2:Valid2 to SP\_DWS0:AcquireSync**

This transition shall occur:

- a) after receiving a Dword Received (Invalid) message; or
- b) after the DWS Reset Timeout timer expires.

If the DWS Reset Timeout timer has expired, then the transition shall include a Timer Expired argument.

**5.15.5.3 Transition SP\_DWS2:Valid2 to SP\_DWS3:SyncAcquired**

This transition shall occur:

- a) after receiving a Dword Received (Primitive) message.

Before the transition, this state shall stop the DWS Reset Timeout timer.

### **5.15.6 SP\_DWS3:SyncAcquired state**

#### **5.15.6.1 State description**

This state is reached after three valid primitives have been received without adjusting the dword synchronization.

Upon entry into this state, this state shall send a Sync Acquired message to the SP\_DWS receiver and the SP receiver.

This state waits for a Dword Received (Invalid) message, which indicates a potential loss of dword synchronization, or an Incorrect Mux Received message, which indicates that dword synchronization is lost.

If an Incorrect Mux Received message is received, then this state shall send a DWS Lost message to the SP state machine.

#### **5.15.6.2 Transition SP\_DWS3:SyncAcquired to SP\_DWS0:AcquireSync**

This transition shall occur:

- a) after sending a DWS Lost message.

#### **5.15.6.3 Transition SP\_DWS3:SyncAcquired to SP\_DWS4:Lost1**

This transition shall occur:

- a) after receiving a Dword Received (Invalid) message.

### **5.15.7 SP\_DWS4:Lost1 state**

#### **5.15.7.1 State description**

This state is reached when one invalid dword has been received and not nullified. This state waits for a Dword Received message or an Incorrect Mux Received message.

If an Incorrect Mux Received message is received, then this state shall send a DWS Lost message to the SP state machine.

#### **5.15.7.2 Transition SP\_DWS4:Lost1 to SP\_DWS0:AcquireSync**

This transition shall occur:

- a) after sending a DWS Lost message.

#### **5.15.7.3 Transition SP\_DWS4:Lost1 to SP\_DWS5:Lost1Recovered**

This transition shall occur after receiving:

- a) a Dword Received (Data Dword) message; or
- b) a Dword Received (Primitive) message.

#### **5.15.7.4 Transition SP\_DWS4:Lost1 to SP\_DWS6:Lost2**

This transition shall occur:

- a) after receiving a Dword Received (Invalid) message.

**5.15.8 SP\_DWS5:Lost1Recovered state****5.15.8.1 State description**

This state is reached when a valid dword has been received after one invalid dword had been received. This state waits for a Dword Received message or an Incorrect Mux Received message.

If an Incorrect Mux Received message is received, then this state shall send a DWS Lost message to the SP state machine.

**5.15.8.2 Transition SP\_DWS5:Lost1Recovered to SP\_DWS0:AcquireSync**

This transition shall occur:

- a) after sending a DWS Lost message.

**5.15.8.3 Transition SP\_DWS5:Lost1Recovered to SP\_DWS3:SyncAcquired**

This transition shall occur after receiving:

- a) a Dword Received (Data Dword) message; or
- b) a Dword Received (Primitive) message.

**5.15.8.4 Transition SP\_DWS5:Lost1Recovered to SP\_DWS6:Lost2**

This transition shall occur:

- a) after receiving a Dword Received (Invalid) message.

**5.15.9 SP\_DWS6:Lost2 state****5.15.9.1 State description**

This state is reached when two invalid dwords have been received and not nullified. This state waits for a Dword Received message or an Incorrect Mux Received message.

If an Incorrect Mux Received message is received, then this state shall send a DWS Lost message to the SP state machine.

**5.15.9.2 Transition SP\_DWS6:Lost2 to SP\_DWS0:AcquireSync**

This transition shall occur:

- a) after sending a DWS Lost message.

**5.15.9.3 Transition SP\_DWS6:Lost2 to SP\_DWS7:Lost2Recovered**

This transition shall occur after receiving:

- a) a Dword Received (Data Dword) message; or
- b) a Dword Received (Primitive) message.

**5.15.9.4 Transition SP\_DWS6:Lost2 to SP\_DWS8:Lost3**

This transition shall occur:

- a) after receiving a Dword Received (Invalid) message.

**5.15.10 SP\_DWS7:Lost2Recovered state****5.15.10.1 State description**

This state is reached when a valid dword has been received after two invalid dwords had been received. This state waits for a Dword Received message or an Incorrect Mux Received message.

If an Incorrect Mux Received message is received, then this state shall send a DWS Lost message to the SP state machine.

**5.15.10.2 Transition SP\_DWS7:Lost2Recovered to SP\_DWS0:AcquireSync**

This transition shall occur:

- a) after sending a DWS Lost message.

**5.15.10.3 Transition SP\_DWS7:Lost2Recovered to SP\_DWS4:Lost1**

This transition shall occur after receiving:

- a) a Dword Received (Data Dword) message; or
- b) a Dword Received (Primitive) message.

**5.15.10.4 Transition SP\_DWS7:Lost2Recovered to SP\_DWS8:Lost3**

This transition shall occur:

- a) after receiving a Dword Received (Invalid) message.

**5.15.11 SP\_DWS8:Lost3 state****5.15.11.1 State description**

This state is reached when three invalid dwords have been received and not nullified. This state waits for a Dword Received message or an Incorrect Mux Received message.

If a Dword Received (Invalid) message is received (i.e., the fourth non-nullified invalid dword is received), then this state shall send a DWS Lost message to the SP state machine.

If an Incorrect Mux Received message is received, then this state shall send a DWS Lost message to the SP state machine.

**5.15.11.2 Transition SP\_DWS8:Lost3 to SP\_DWS0:AcquireSync**

This transition shall occur:

- a) after sending a DWS Lost message.

**5.15.11.3 Transition SP\_DWS8:Lost3 to SP\_DWS9:Lost3Recovered**

This transition shall occur after receiving:

- a) a Dword Received (Data Dword) message; or
- b) a Dword Received (Primitive) message.

**5.15.12 SP\_DWS9:Lost3Recovered state****5.15.12.1 State description**

This state is reached when a valid dword has been received after three invalid dwords had been received. This state waits for a Dword Received message or an Incorrect Mux Received message.

If a Dword Received (Invalid) message is received (i.e., the fourth non-nullified invalid dword is received), then this state shall send a DWS Lost message to the SP state machine.

If an Incorrect Mux Received message is received, then this state shall send a DWS Lost message to the SP state machine.

#### 5.15.12.2 Transition SP\_DWS9:Lost3Recovered to SP\_DWS0:AcquireSync

This transition shall occur:

- a) after sending a DWS Lost message.

#### 5.15.12.3 Transition SP\_DWS9:Lost3Recovered to SP\_DWS6:Lost2

This transition shall occur after receiving:

- a) a Dword Received (Data Dword) message; or
- b) a Dword Received (Primitive) message.

### 5.16 SP\_PS (phy layer SPL packet synchronization) state machine

#### 5.16.1 SP\_PS state machine overview

Each phy that supports SAS packet mode includes an SP\_PS state machine and an SP\_PS receiver.

The SP\_PS receiver monitors the incoming data stream and forces the SPL packet header into the correct position to form an SPL packet.

After SPL packet synchronization has been achieved (see 5.17) and two SPL packets with no decode failures have been received, the SP\_PS state machine monitors for decode failures (see 5.5.7.3). If a decode failure is detected, then receipt of two consecutive SPL packets with no decode failure are required to nullify the effect of receiving a decode failure. If four consecutive decode failures are detected, then SPL packet synchronization is considered lost.

While SPL packet synchronization is lost, the data stream received is invalid and SPL packets are not passed to the link layer.

This state machine consists of the following states:

- a) SP\_PS0:AcquireSync (see 5.16.3) (initial state);
- b) SP\_PS1:Valid1 (see 5.16.4);
- c) SP\_PS2:SyncAcquired (see 5.16.5);
- d) SP\_PS3:Lost1 (see 5.16.6);
- e) SP\_PS4:LostRecovered (see 5.16.7);
- f) SP\_PS5:Lost2 (see 5.16.8); and
- g) SP\_PS6:Lost3 (see 5.17).

This state machine receives the following requests from the management application layer:

- a) Management Reset; and
- b) Disable Phy.

This state machine shall start in or transition to the SP\_PS0:AcquireSync state after:

- a) power on;
- b) hard reset;
- c) receiving a Management Reset request (e.g., as a result of processing an SMP PHY CONTROL function requesting a phy operation of LINK RESET or HARD RESET); or
- d) receiving a Disable Phy request (e.g., as a result of processing an SMP PHY CONTROL function requesting a phy operation of DISABLE).

This state machine receives the following messages from the SP state machine (see 5.14):

- a) Start PS; and
- b) Stop PS.

This state machine sends the following message to the SP\_ReSync state machine (see 5.17):

- a) Resynchronize.

This state machine receives the following message from the SP\_ReSync state machine:

- a) Restart PS; and
- b) Stop PS.

This state machine shall maintain the timers listed in table 96.

**Table 96 – SP\_PS state machine timers**

Timer	Initial value
PS Reset Timeout timer	1 ms



Figure 112 shows the SP\_PS state machine.

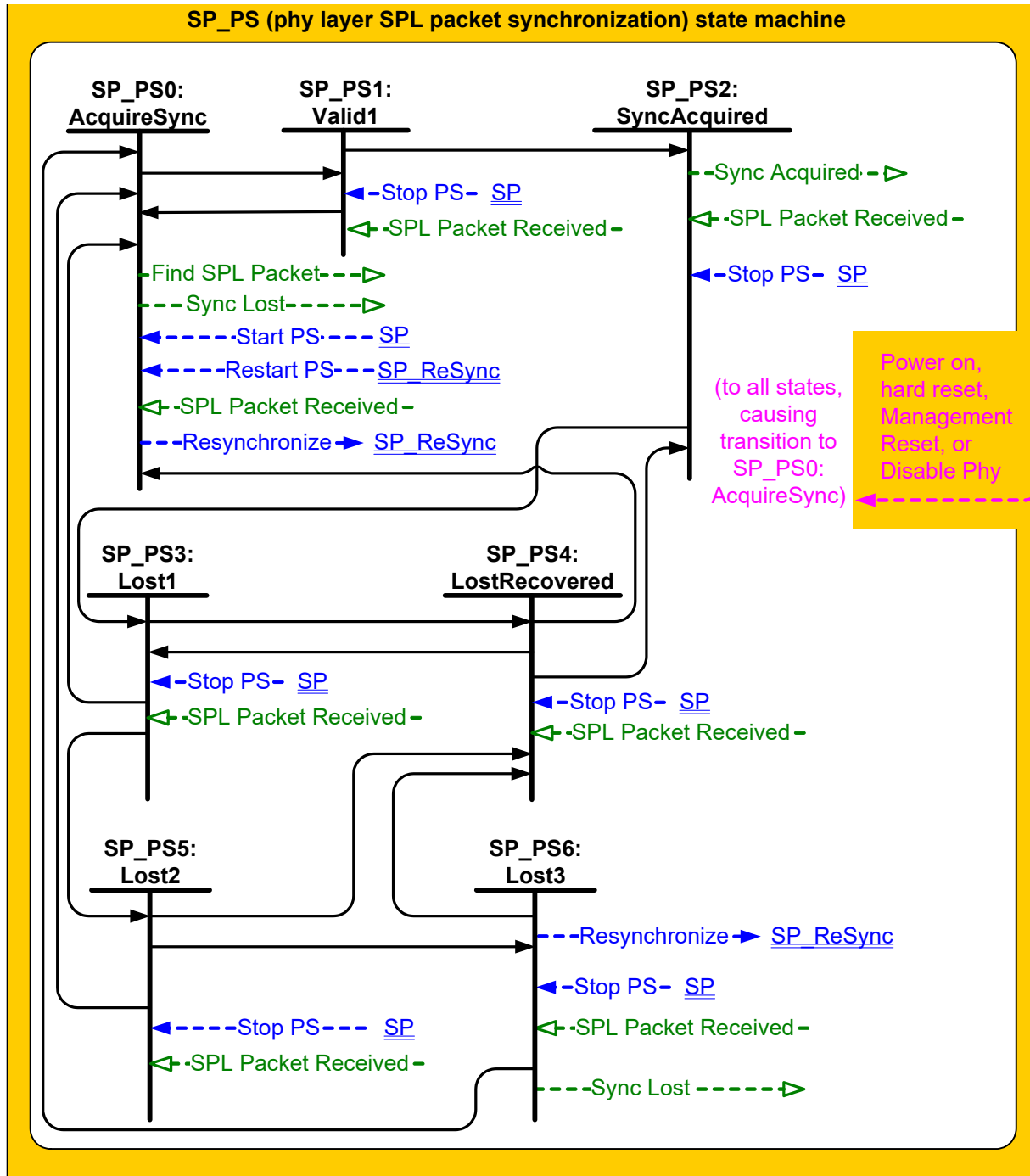


Figure 112 – SP\_PS (phy layer SPL packet synchronization) state machine

### 5.16.2 SP\_PS receiver

The SP\_PS receiver receives the following messages from the SP\_PS state machine:

- Find SPL Packet;
- Sync Lost; and
- Sync Acquired.

The SP\_PS receiver sends the following messages to the SP\_PS state machine indicating SPL packets received from the physical link:

- a) SPL Packet Received (Valid); and
- b) SPL Packet Received (Invalid).

Upon receiving a Find SPL Packet message, the SP\_PS receiver shall wait for a Decode Success message. When a Decode Success message is received the SP\_PS receiver shall send an SPL Packet Received message to the SP\_PS state machine. Until the SP\_PS receiver receives another Find SPL Packet message or Sync Lost message, for every SPL packet that it receives the SP\_PS receiver shall:

- a) send an SPL Packet Received (Invalid) message to the SP\_PS state machine if a Decode Failure message (see 5.5.7.3) is received for the received SPL packet; or
- b) send an SPL Packet Received (Valid) message to the SP\_PS state machine if a Decode Success message is received for the received SPL packet.

If the SP\_PS receiver receives a Sync Lost message, then the SP\_PS receiver shall stop sending data dwords, primitives, binary primitives, extended binary primitives, and invalid dwords to the link layer state machine receivers.

Received SPL packets are separated into data dwords, primitives, binary primitives, extended binary primitives, primitive parameters, and scrambled dwords as described in figure 114.

When the SP\_PS receiver receives a Sync Acquired message, the SP\_PS receiver shall send to the link layer state machine receivers (e.g., SL\_IR, SL, SSP, SMP, and XL) through the elasticity buffer (see 6.5) the following confirmations in the order received in the SPL packet:

- a) four Dword Received (Data Dword) confirmations if the received SPL packet indicates contents of the SPL packet payload contain an SPL frame segment (see 5.5.5) or an idle dword segment;
- b) a QuadDword Received (Extended Binary Primitive) confirmation if the received SPL packet indicates the contents of the SPL packet contain an extended binary primitive (i.e., PRIMITIVE SYNCHRONIZE SELECT field set to 10b (see 5.5.4));
- c) a Dword Received (Invalid Dword) confirmation for all the dwords associated with the SPL packet associated with the a Decode Failure message received for the received SPL packet;
- d) a Dword Received (Binary Primitive) confirmation:
  - A) if the received SPL packet indicates contents of the SPL packet payload contain a primitive segment (see 5.5.4)); and
  - B) for each PRIMITIVE0 field, PRIMITIVE1 field, PRIMITIVE2 field, and PRIMITIVE3 field that have an indication that the field contains a binary primitive (i.e., PRIMITIVE SYNCHRONIZE SELECT field, CONTROL1 field, CONTROL2 field, or CONTROL3 field set to 01b);
- or
- e) one to three Dword Received (Primitive Parameter) confirmations;
  - A) if the received SPL packet indicates contents of the SPL packet payload contain a primitive segment (see 5.5.4)); and
  - B) for each PRIMITIVE1 field, PRIMITIVE2 field, and PRIMITIVE3 field that contains a primitive parameter (i.e., the CONTROL1 field, CONTROL2 field, or CONTROL3 field of the first primitive parameter dword within the primitive segment is set to 10b (see table 56)).

When the SP\_PS receiver receives a Sync Acquired message, the SP\_PS receiver shall translate primitives received in an SPL packet into four 10-bit characters as described in figure 113 and send to the link layer state machine receivers (e.g., SL\_IR, SL, SSP, SMP, and XL):

- a) a Dword Received (Primitive) confirmation that contains the translated primitive; or
- b) a Dword Received (Invalid Dword) confirmation for all the characters associated with the SPL packet that contains any:
  - A) 8b10b data character that has a disparity inconsistent with a primitive stating with RD+ disparity (see 5.3.5); or
  - B) invalid character.

The SP\_PS receiver shall discard all dwords received within an SPL packet that contains a scrambled idle segment (i.e., scrambled dwords).

The SP\_PS receiver relationship to other receivers is defined in 4.3.3.

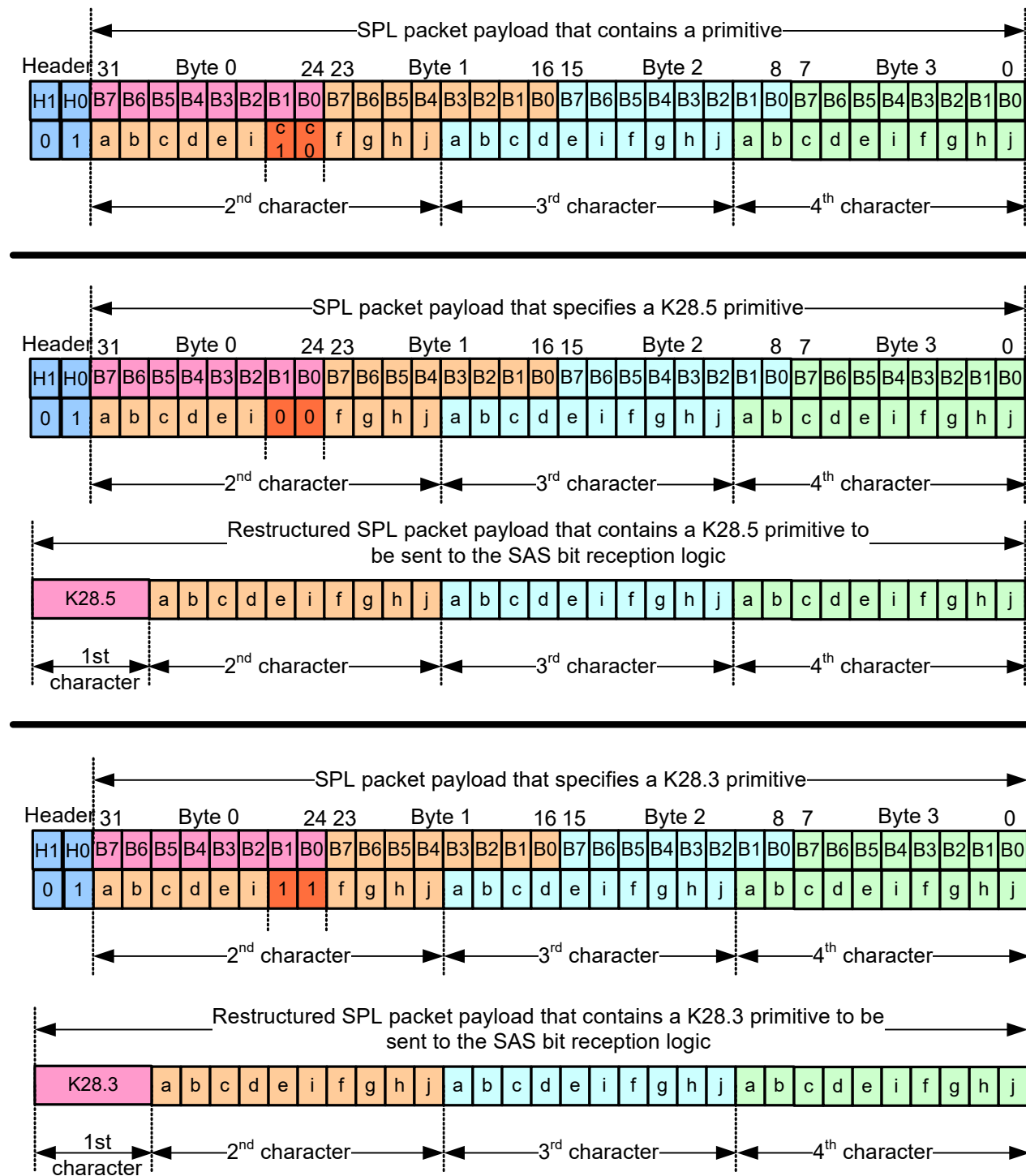


Figure 113 – SPL packet payload primitive decoding

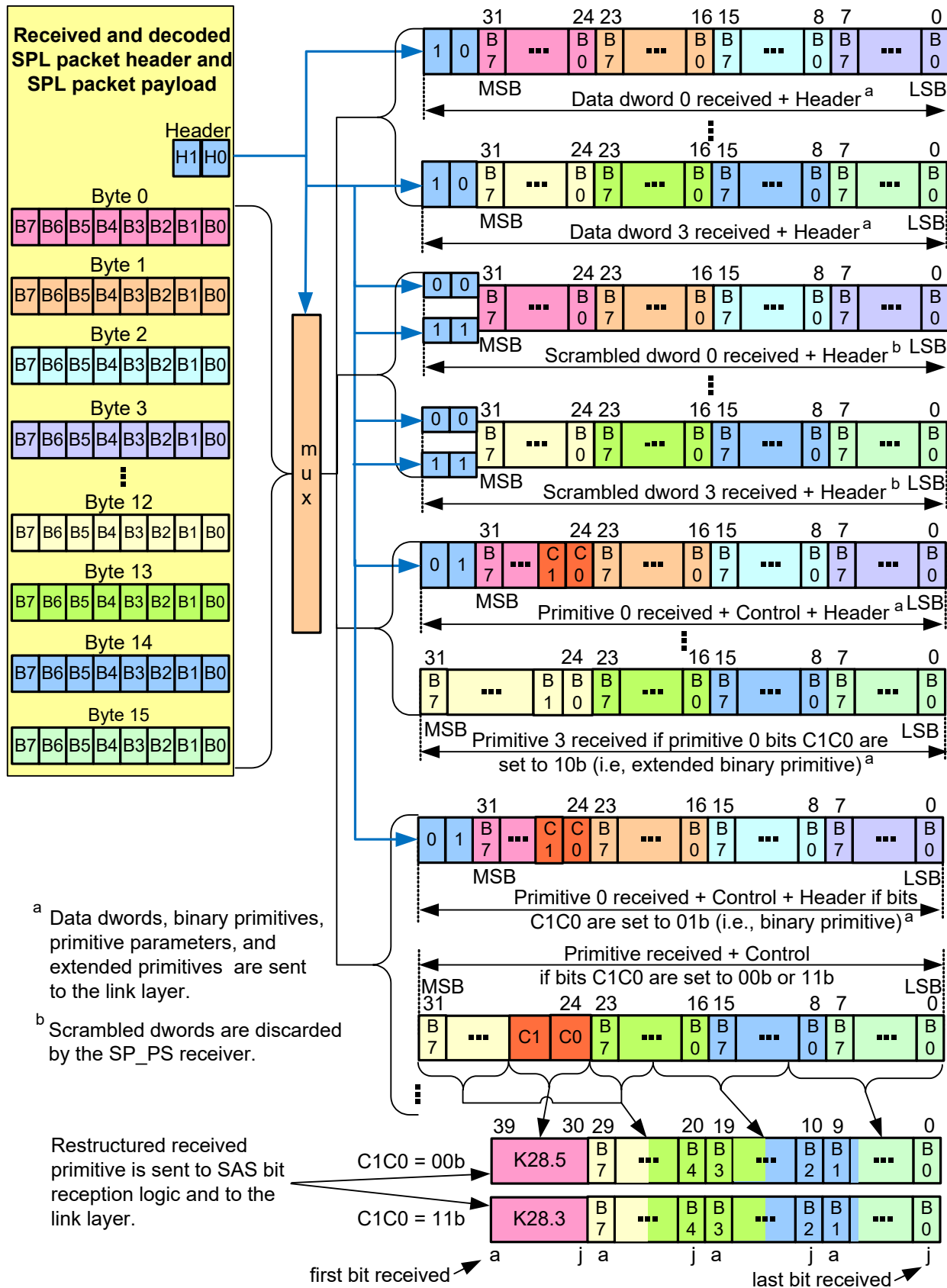


Figure 114 – Unpacking SPL packet

**5.16.3 SP\_PS0:AcquireSync state****5.16.3.1 State description**

This is the initial state of this state machine.

After receiving a Start PS message or Restart PS message, this state shall:

- 1) send a Sync Lost message to the SP\_PS receiver;
- 2) send a Find SPL Packet message to the SP\_PS receiver; and
- 3) initialize and start the PS Reset Timeout timer.

If this state is entered from SP\_PS1:Valid1, then:

- a) this state shall send:
  - 1) a Sync Lost message to the SP\_PS receiver; and
  - 2) a Find SPL Packet message to the SP\_PS receiver;
 and
- b) the PS Reset Timeout timer shall continue running, if this state was entered without a Timer Expired argument.

This state shall send a Resynchronize message to the SP\_ReSync state machine if:

- a) the PS Reset Timeout timer expires; or
- b) this state was entered with a Timer Expired argument.

**5.16.3.2 Transition SP\_PS0:AcquireSync to SP\_PS1:Valid1**

This transition shall occur:

- a) after sending a Find SPL Packet message; and
- b) after receiving an SPL Packet Received (Valid) message.

**5.16.4 SP\_PS1:Valid1 state****5.16.4.1 State description**

This state is reached after one valid SPL packet has been received. This state waits for a second valid SPL packet or an invalid SPL packet.

The PS Reset Timeout timer shall continue running.

**5.16.4.2 Transition SP\_PS1:Valid1 to SP\_PS0:AcquireSync**

This transition shall occur:

- a) after receiving an SPL Packet Received (Invalid) message;
- b) after the PS Reset Timeout timer expires; or
- c) after receiving a Stop PS message.

If the PS Reset Timeout timer has expired, then the transition shall include a Timer Expired argument.

**5.16.4.3 Transition SP\_PS1:Valid1 to SP\_PS2:SyncAcquired**

This transition shall occur:

- a) after receiving an SPL Packet Received (Valid) message.

Before the transition, this state shall stop the PS Reset Timeout timer.

**5.16.5 SP\_PS2:SyncAcquired state****5.16.5.1 State description**

This state is reached after two valid SPL packets have been received.

Upon entry into this state, this state shall send a Sync Acquired message to the SP\_PS receiver and the SP receiver.

This state waits for an SPL Packet Received (Invalid) message, which indicates a potential loss of SPL packet synchronization.

**5.16.5.2 Transition SP\_PS2:SyncAcquired to SP\_PS0:AcquireSync**

This transition shall occur:

- a) after receiving a Stop PS message.

**5.16.5.3 Transition SP\_PS2:SyncAcquired to SP\_PS3:Lost1**

This transition shall occur:

- a) after receiving an SPL Packet Received (Invalid) message.

**5.16.6 SP\_PS3:Lost1 state****5.16.6.1 State description**

This state is reached when one invalid SPL packet has been received. This state waits for an SPL Packet Received message.

**5.16.6.2 Transition SP\_PS3:Lost1 to SP\_PS0:AcquireSync**

This transition shall occur:

- a) after receiving a Stop PS message.

**5.16.6.3 Transition SP\_PS3:Lost1 to SP\_PS5:Lost2**

This transition shall occur:

- a) after receiving an SPL Packet Received (Invalid) message.

**5.16.6.4 Transition SP\_PS3:Lost1 to SP\_PS4:LostRecovered**

This transition shall occur:

- a) after receiving an SPL Packet Received (Valid) message.

**5.16.7 SP\_PS4:LostRecovered state****5.16.7.1 State description**

This state is reached when a valid SPL packet has been received after one invalid SPL packet had been received. This state waits for an SPL Packet Received message.

**5.16.7.2 Transition SP\_PS4:LostRecovered to SP\_PS0:AcquireSync**

This transition shall occur:

- a) after receiving a Stop PS message.

**5.16.7.3 Transition SP\_PS4:LostRecovered to SP\_PS2:SyncAcquired**

This transition shall occur:

- a) after receiving an SPL Packet Received (Valid) message.

**5.16.7.4 Transition SP\_PS4:LostRecovered to SP\_PS3:Lost1**

This transition shall occur:

- a) after receiving an SPL Packet Received (Invalid) message.

**5.16.8 SP\_PS5:Lost2 state****5.16.8.1 State description**

This state is reached when two consecutive invalid SPL packets have been received. This state waits for an SPL Packet Received message.

**5.16.8.2 Transition SP\_PS5:Lost2 to SP\_PS0:AcquireSync**

This transition shall occur:

- a) after receiving a Stop PS message.

**5.16.8.3 Transition SP\_PS5:Lost2 to SP\_PS6:Lost3**

This transition shall occur:

- a) after receiving an SPL Packet Received (Valid) message.

**5.16.9 SP\_PS6:Lost3 state****5.16.9.1 State description**

This state is reached when three consecutive invalid SPL packets have been received. This state waits for an SPL Packet Received message.

If an SPL Packet Received (Invalid) message is received (i.e., the fourth consecutive invalid SPL packet is received), then this state shall send:

- 1) a Sync Lost message to the SP\_PS receiver; and
- 2) a Resynchronize message to the SP\_ReSync state machine.

**5.16.9.2 Transition SP\_PS6:Lost3 to SP\_PS0:AcquireSync**

This transition shall occur:

- a) after sending a Resynchronize message; or
- b) after receiving a Stop PS message.

**5.16.9.3 Transition SP\_PS6:Lost3 to SP\_PS4:LostRecovered**

This transition shall occur:

- a) after receiving an SPL Packet Received (Valid) message.

## 5.17 SP\_ReSync (phy layer resynchronization) state machine

### 5.17.1 SP\_ReSync state machine overview

Each phy that supports SPL packet mode shall include an SP\_ReSync state machine.

The SP\_ReSync state machine:

- a) reestablishes the SPL packet boundaries at the receiver by responding to PACKET\_SYNC\_LOSTs with the transmission of PACKET\_SYNCs; and
- b) requests the reestablishment of SPL packet boundaries between the receiver and the attached transmitter by:
  - 1) transmitting PACKET\_SYNC\_LOSTs; and
  - 2) waiting for the receipt of PACKET\_SYNCs.

This state machine consists of the following states:

- a) SP\_ReSync0:Start (see 5.15.3) (initial state);
- b) SP\_ReSync1:Request (see 5.15.4); and
- c) SP\_ReSync2:Response (see 5.15.6).

This state machine shall start in or transition to the SP\_ReSync0:Start state after:

- a) power on;
- b) hard reset;
- c) receiving a Management Reset request (e.g., as a result of processing an SMP PHY CONTROL function requesting a phy operation of LINK RESET or HARD RESET); or
- d) receiving a Disable Phy request (e.g., as a result of processing an SMP PHY CONTROL function requesting a phy operation of DISABLE).

This state machine receives the following message from the SP state machine (see 5.14):

- a) Stop Resync; and
- b) Restart PS.

This state machine sends the following message to the SP state machine:

- a) PS Reset.

This state machine receives the following message from the SP\_PS state machine (see 5.16):

- a) Resynchronize.

This state machine sends the following message to the SP\_PS state machine (see 5.16):

- a) Restart PS; and
- b) Stop PS.



Figure 115 shows the SP\_ReSync state machine.

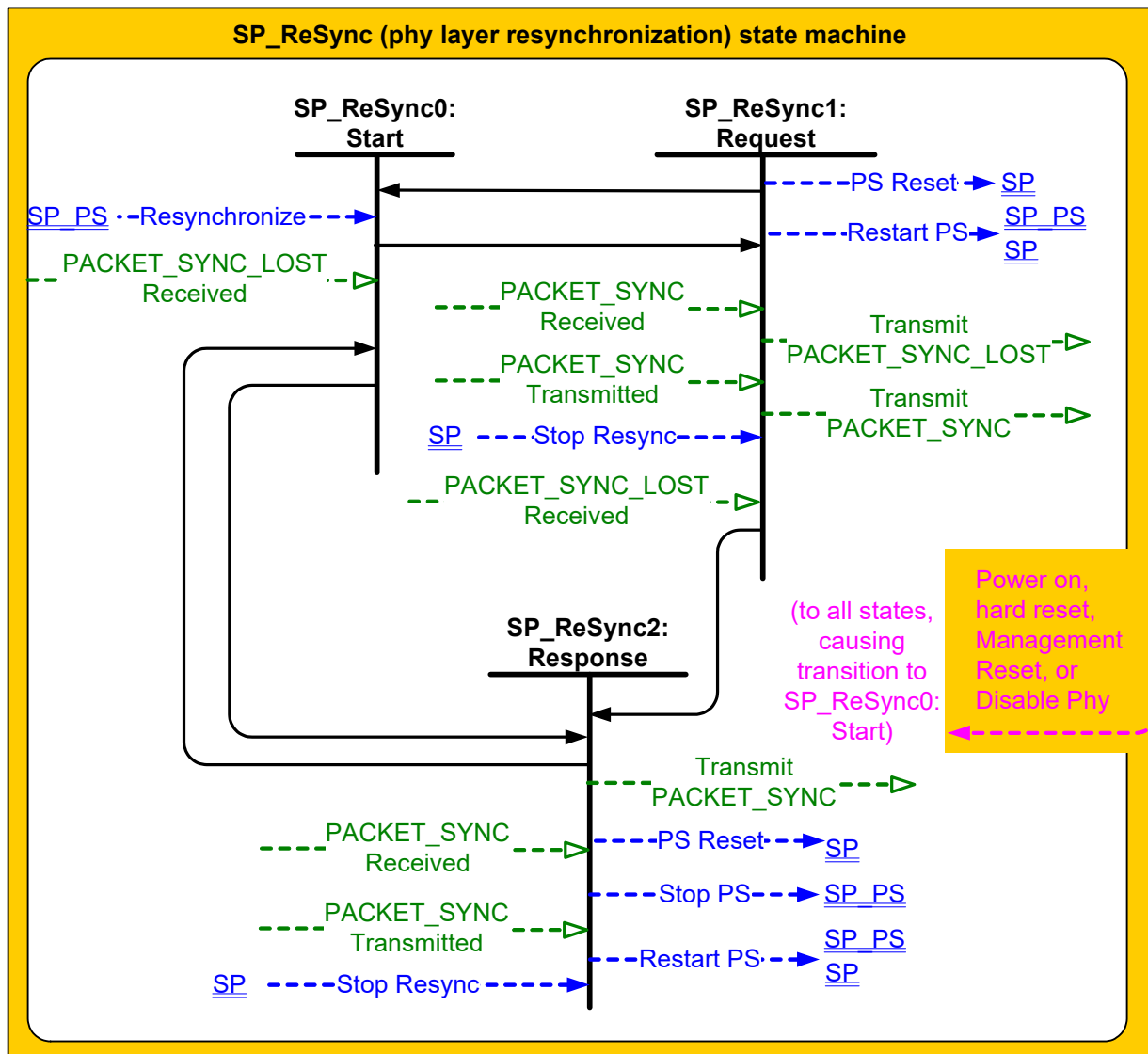


Figure 115 – SP\_ReSync (phy layer resynchronization) state machine

### 5.17.2 SP\_ReSync0:Start

#### 5.17.2.1 State description

This is the initial state of this state machine.

#### 5.17.2.2 Transition SP\_ReSync0:Start to SP\_ReSync1:Request

This transition shall occur:

- after a Resynchronize message is received.

#### 5.17.2.3 Transition SP\_ReSync0:Start to SP\_ReSync2:Response

This transition shall occur:

- after a PACKET\_SYNC\_LOST Received message is received.

**5.17.3 SP\_ReSync1:Request**

This state requests the attached phy perform a resynchronization.

Upon entry into this state, this state shall:

- a) initialize and start the MRTT timer.

This state shall repeatedly send Transmit PACKET\_SYNC\_LOST messages to the SP transmitter (see 5.14.2).

On receiving a PACKET\_SYNC Received message this state shall:

- 1) stop sending Transmit PACKET\_SYNC\_LOST messages;
- 2) stop the MRTT timer; and
- 3) send four Transmit PACKET\_SYNC messages to the SP transmitter.

If the MRTT timer expires, then this state shall send a PS Reset message to the SP state machine.

**5.17.3.1 Transition SP\_ReSync1 to SP\_ReSync0:Start**

This transition shall occur after:

- a) a PS Reset message is sent;
- b) receiving four PACKET\_SYNC Transmitted messages; or
- c) receiving a Stop Resync message.

Before the transition, this state shall:

- a) stop the MRTT timer; and
- b) if four PACKET\_SYNC Transmitted messages were received, then send a Restart PS to the SP\_PS state machine and to the SP state machine.

**5.17.3.2 Transition SP\_ReSync1 to SP\_ReSync2:Response**

This transition shall occur:

- a) after receiving a PACKET\_SYNC\_LOST Received message.

Before the transition, this state shall stop the MRTT timer.

**5.17.4 SP\_ReSync2:Response state****5.17.4.1 State description**

On entry this state shall:

- a) initialize and start the MRTT timer; and
- b) send a Stop PS message to the SP\_PS state machine.

This state shall repeatedly send Transmit PACKET\_SYNC messages to the SP transmitter (see 5.14.2).

This state waits for the MRTT timer to expire or a PACKET\_SYNC Received message from the SP\_PS receiver.

If this state receives a PACKET\_SYNC Received message, then this state shall stop the MRTT timer.

If the MRTT timer expires, then this state shall send a PS Reset message to the SP state machine.

This state achieves SPL packet synchronization if this state receives:

- 1) at least four PACKET\_SYNC Transmitted messages;
- 2) a PACKET\_SYNC Received message before the MRTT timer expires; and
- 3) at least one additional PACKET\_SYNC Transmitted message.

#### 5.17.4.2 Transition SP\_ReSync2:Response state to SP\_ReSync0:Start

This transition shall occur after this state either:

- a) achieves SPL packet synchronization;
- b) sends a PS Reset message; or
- c) receives a Stop Resync message.

Before the transition, this state shall:

- a) stop the MRTT timer; and
- b) if packet synchronization is achieved, then send a Restart PS to the SP\_PS state machine and the SP state machine.

### 5.18 PTT (phy layer transmitter training) state machines

#### 5.18.1 PTT state machines overview

The PTT phy layer contains several state machines that run in parallel to control the physical link during Train\_Tx-SNW. The PTT state machines are as follows:

- a) PTT\_T (phy layer transmitter training transmit pattern) state machine (see 5.18.4);
- b) PTT\_R (phy layer transmitter training receive pattern) state machine (see 5.18.5);
- c) PTT\_SC1 (phy layer transmitter training set transmitter coefficient 1) state machine (see 5.18.6);
- d) PTT\_SC2 (phy layer transmitter training set transmitter coefficient 2) state machine (see 5.18.7);
- e) PTT\_SC3 (phy layer transmitter training set transmitter coefficient 3) state machine (see 5.18.8);
- f) PTT\_GC1 (phy layer transmitter training get transmitter coefficient 1) state machine (see 5.18.9);
- g) PTT\_GC2 (phy layer transmitter training get transmitter coefficient 2) state machine (see 5.18.10);
- h) PTT\_GC3 (phy layer transmitter training get transmitter coefficient 3) state machine (see 5.18.11);
- and
- i) PTT\_PL (phy layer transmitter training pattern lock) state machine (see 5.18.12).

All the PTT state machines shall start in the initial state after receiving a Transmitter Training (Enable) message from the SP state machine (see 5.14.4.13).

All the PTT state machines shall terminate after receiving a Transmitter Training (Disable) message from the SP state machine (see 5.14.4.13).

Each phy that supports transmitter training shall include all the PTT state machines.

If the state machine consists of multiple states, then the initial state is as indicated in the state machine description in this subclause.

Any message, request, or confirmation received by a state that is not referred to in the description of that state shall be ignored.

#### 5.18.2 SP transmitter additions for transmitter training

##### 5.18.2.1 SP transmitter additions for transmitter training overview

The SP transmitter receives the following message from the PTT state machines specifying dwords to transmit:

- a) Transmit Train\_Tx Pattern (see 5.11.4.2.3.4).

The SP transmitter receives the following messages from the PTT state machines:

- a) Set Training Control;
- b) Set Training Status;
- c) Set Transmitter Failed;
- d) Adjust Coefficient 1 with arguments of:

- A) Increment, Decrement, Set No\_Equalization, Set Reference\_1, or Set Reference\_2; and
- B) Single or Dual;
- e) Adjust Coefficient 2 with arguments of:
  - A) Increment, Decrement, Set No\_Equalization, Set Reference\_1, or Set Reference\_2; and
  - B) Single or Dual;
- and
- f) Adjust Coefficient 3 with argument of:
  - A) Increment, Decrement, Set No\_Equalization, Set Reference\_1, or Set Reference\_2; and
  - B) Single or Dual.

See 5.14.2 for SP transmitter requirements while PTT state machines are not processing Train\_Tx-SNW.

The SP transmitter relationship to other transmitters is defined in 4.3.2.

#### 5.18.2.2 TTIU transmit setup

In response to a Set Training Control message the SP transmitter shall set the Training Control word (see 5.10.2) to the contents of the received argument.

In response to a Set Training Status message the SP transmitter shall set the Training Status word (see 5.10.2) to the contents of the received argument.

In response to a Set Transmitter Failed message the SP transmitter shall set the Error Response TTIU (see 5.10.3) to the contents of the received argument.

The SP transmitter shall set the BALANCE bit (see 5.10.2 and 5.10.3) to the correct value before transmitting a Train\_Tx pattern.

The SP transmitter sends the following message to the PTT state machines based on dwords that have been transmitted:

- a) Train\_Tx Pattern Transmitted.

#### 5.18.2.3 No\_equalization, reference\_1, and reference\_2 coefficient settings request

In response to an Adjust Coefficient 1 message, an Adjust Coefficient 2 message, and an Adjust Coefficient 3 message with a Set No\_Equalization argument the SP transmitter shall adjust all the coefficients to the no equalization value (see SAS-4).

In response to an Adjust Coefficient 1 message, an Adjust Coefficient 2 message, and an Adjust Coefficient 3 message with a Set Reference\_1 argument the SP transmitter shall adjust all the coefficients to the reference\_1 value (see SAS-4).

In response to an Adjust Coefficient 1 message, an Adjust Coefficient 2 message, and an Adjust Coefficient 3 message with a Set Reference\_2 argument the SP transmitter shall adjust all the coefficients to the reference\_2 value (see SAS-4).

#### 5.18.2.4 Coefficient limits

The individual coefficient limits and limits associated with relationships between coefficients specified in SAS-4 shall be maintained by the SP transmitter while processing coefficient adjustment requests.

#### 5.18.2.5 Coefficient request result of update complete

##### 5.18.2.5.1 Coefficient request processing

In response to an Adjust Coefficient 1 message, an Adjust Coefficient 2 message, or an Adjust Coefficient 3 message with a Single argument, and an Increment argument or a Decrement argument the SP transmitter shall adjust the specified coefficient if that adjustment results in the specified coefficient being:

- a) less than a maximum value; and

- b) greater than a minimum value (see SAS-4).

In response to an Adjust Coefficient 1 message, an Adjust Coefficient 2 message, or an Adjust Coefficient 3 message with a Dual argument, and an Increment argument or a Decrement argument the SP transmitter shall, after receiving the first Dual argument:

- 1) wait for a second Dual argument;
- 2) process both requests; and
- 3) for each specified coefficient, adjust that coefficient if that adjustment results in:
  - A) that coefficient being:
    - a) less than a maximum value; and
    - b) greater than a minimum value;
 and
  - B) no coefficient being greater than a maximum value or less than a minimum value.

#### 5.18.2.5.2 Coefficient adjustment completes

If the SP transmitter adjusts a specified coefficient, then after the adjustment is complete the SP transmitter shall send to the PTT state machine state that requested the SP transmitter adjustment the message associated with the adjusted coefficient (i.e., Transmitter Adjustment 1 Complete (Update Complete) message, Transmitter Adjustment 2 Complete (Update Complete) message, or Transmitter Adjustment 3 Complete (Update Complete) message).

#### 5.18.2.5.3 No coefficient adjustment

In response to an Adjust Coefficient 1 message, an Adjust Coefficient 2 message, or an Adjust Coefficient 3 message with a Dual argument, if the processing results in:

- a) one specified coefficient being:
  - A) less than a maximum value; and
  - B) greater than a minimum value;
 and
- b) the other specified coefficient being:
  - A) greater than a maximum value; or
  - B) less than a minimum value,

then the SP transmitter shall send the message associated with the specified coefficient (i.e., Transmitter Adjustment 1 Complete (Update Complete) message, Transmitter Adjustment 2 Complete (Update Complete) message, or Transmitter Adjustment 3 Complete (Update Complete) message) that is:

- a) less than a maximum value; and
- b) greater than a minimum value,

to the PTT state machine state that requested the SP transmitter adjustment.

#### 5.18.2.6 Coefficient request result of maximum

##### 5.18.2.6.1 Coefficient request processing

In response to an Adjust Coefficient 1 message, an Adjust Coefficient 2 message, or an Adjust Coefficient 3 message with a Single argument and an Increment argument the SP transmitter shall adjust the specified coefficient if that adjustment results in the specified coefficient being equal to a maximum value (see SAS-4).

In response to an Adjust Coefficient 1 message, an Adjust Coefficient 2 message, or an Adjust Coefficient 3 message with a Dual argument and an Increment argument, the SP transmitter shall, after receiving the first Dual argument:

- 1) wait for a second Dual argument;
- 2) process both requests; and

- 3) for each specified coefficient, adjust that coefficient if that adjustment results in:
  - A) that coefficient being equal to a maximum value; and
  - B) no coefficient being greater than a maximum value.

#### **5.18.2.6.2 Coefficient adjustment completes**

If the SP transmitter adjusts a specified coefficient to a maximum value, then after the adjustment is complete the SP transmitter shall send to the PTT state machine state that requested the SP transmitter adjustment the message associated with the adjusted coefficient (i.e., Transmitter Adjustment 1 Complete (Maximum) message, Transmitter Adjustment 2 Complete (Maximum) message, or Transmitter Adjustment 3 Complete (Maximum) message).

#### **5.18.2.6.3 No coefficient adjustment**

If the processing of the requested SP transmitter adjustment results in an adjustment that is greater than a coefficient's maximum value, then:

- a) no adjustment shall be made to any coefficient; and
- b) the SP transmitter shall send to the PTT state machine state that requested the SP transmitter adjustment the message associated with each specified coefficient that is equal to or greater than a maximum value (i.e., Transmitter Adjustment 1 Complete (Maximum) message, Transmitter Adjustment 2 Complete (Maximum) message, or Transmitter Adjustment 3 Complete (Maximum) message).

#### **5.18.2.7 Coefficient request result of minimum**

##### **5.18.2.7.1 Coefficient request processing**

In response to an Adjust Coefficient 1 message, an Adjust Coefficient 2 message, or an Adjust Coefficient 3 message with a Single argument and a Decrement argument the SP transmitter shall adjust the specified coefficient if that adjustment results in the specified coefficient being equal to a minimum value (see SAS-4).

In response to an Adjust Coefficient 1 message, an Adjust Coefficient 2 message, or an Adjust Coefficient 3 message with a Dual argument and a Decrement argument, the SP transmitter shall, after receiving the first Dual argument:

- 1) wait for a second Dual argument;
- 2) process both requests; and
- 3) for each specified coefficient, adjust that coefficient if that adjustment results in:
  - A) that coefficient being equal to a minimum value; and
  - B) no coefficient being less than a minimum value.

##### **5.18.2.7.2 Coefficient adjustment completes**

If the SP transmitter adjusts a specified coefficient to a minimum value, then after the adjustment is complete the SP transmitter shall send to the PTT state machine state that requested the SP transmitter adjustment the message associated with the adjusted coefficient (i.e., Transmitter Adjustment 1 Complete (Minimum) message, Transmitter Adjustment 2 Complete (Minimum) message, or Transmitter Adjustment 3 Complete (Minimum) message).

##### **5.18.2.7.3 No coefficient adjustment**

If the processing of the requested SP transmitter adjustment results in an adjustment that is less than a coefficient's minimum value then:

- a) no adjustment shall be made to any coefficient; and
- b) the SP transmitter shall send to the PTT state machine state that requested the SP transmitter adjustment the message associated with each specified coefficient that is equal to or less than a minimum value (i.e., Transmitter Adjustment 1 Complete (Minimum) message, Transmitter

Adjustment 2 Complete (Minimum) message, or Transmitter Adjustment 3 Complete (Minimum) message).

### 5.18.3 SP receiver additions for transmitter training

The SP receiver sends the following messages to the PTT state machines indicating dwords received by the SP receiver:

- a) Received TTIU with arguments containing the contents of the received TTIU (e.g., Training Control word and the Training Status word of the Control/Status TTIU, or error response of the Error Response TTIU);
- b) Valid Pattern Marker; and
- c) Invalid Pattern Marker.

The SP receiver sends the following messages to the PTT state machines:

- a) Current Coefficient 1 with an argument of Increment, Decrement, No\_Equalization, Reference\_1, Reference\_2, or Hold;
- b) Current Coefficient 2 with an argument of Increment, Decrement, No\_Equalization, Reference\_1, Reference\_2, or Hold;
- c) Current Coefficient 3 with an argument of Increment, Decrement, No\_Equalization, Reference\_1, Reference\_2, or Hold;
- d) All Coefficients Not Ready; and
- e) Attached Phy's Transmitter Optimized.

The SP receiver receives the following messages from the PTT state machines:

- a) Coefficient 1 Status with an argument of Maximum, Minimum, Update\_Complete, or Ready;
- b) Coefficient 2 Status with an argument of Maximum, Minimum, Update\_Complete, or Ready;
- c) Coefficient 3 Status with an argument of Maximum, Minimum, Update\_Complete, or Ready;
- d) Get Current Coefficient 1;
- e) Get Current Coefficient 2;
- f) Get Current Coefficient 3;
- g) Enable Pattern Marker Detection;
- h) Disable Pattern Marker Detection;
- i) Attached Phy's Transmitter Training with an argument of Start or Stop;
- j) Transmitter Control Failed with an argument containing information on the failure; and
- k) Coefficients Status Updated.

After the SP receiver receives an Attached Phy's Transmitter Training (Start) message, the SP receiver shall respond to a Get Current Coefficient 1 message, Get Current Coefficient 2 message, and Get Current Coefficient 3 message by sending a Current Coefficient 1 message, Current Coefficient 2 message, and Current Coefficient 3 message with an argument (i.e., Increment, Decrement, No\_Equalization, Reference\_1, Reference\_2, or Hold) to the PTT state machine after the SP receiver:

- a) completes the analysis of a transmitter training pattern; and
- b) receives a Control/Status TTIU with an even number of bits set to zero with a coefficient status of ready for all the attached phy's transmitter coefficients.

After the SP receiver sends one or more Current Coefficient 1 message, Current Coefficient 2 message, or Current Coefficient 3 message with an argument of Increment, Decrement, No\_Equalization, Reference\_1, or Reference\_2 the SP receiver shall not attempt analysis of a transmitter training pattern until after it has received a status of update complete, maximum, or minimum for all the attached phy's transmitter coefficients that were updated.

After the SP receiver receives an Attached Phy's Transmitter Training (Stop) message, the SP receiver:

- 1) shall not respond to any Get Current Coefficient 1 messages, Get Current Coefficient 2 messages, or Get Current Coefficient 3 messages;
- 2) waits until a status other than ready has been received on all coefficients; and
- 3) repeatedly send All Coefficients Not Ready messages to the PTT\_T state machine until an Attached Phy's Transmitter Training (Start) message is received.

When the SP receiver determines the attached phy's transmitter training is complete, the SP receiver shall send an Attached Phy's Transmitter Optimized message to the PTT\_T state machine.

If the SP receiver requires the attached phy's transmitter training to be restarted with the no\_equalization value, reference\_1 value, or reference\_2 value, then the SP receiver shall:

- 1) wait until the Coefficient 1 Status message, Coefficient 2 Status message, and Coefficient 3 Status message all specify a Ready argument; and
- 2) send a Current Coefficient 1 message to the PTT\_GC1 state machine, Current Coefficient 2 message to the PTT\_GC2 state machine, and Current Coefficient 3 message to the PTT\_GC3 state machines all with the same argument of No\_Equalization, Reference\_1, or Reference\_2.

The actions taken on receiving a Transmitter Control Failed message are outside the scope of this standard.

See 5.14.2 for SP receiver requirements while PTT state machines are not processing Train\_Tx-SNW.

The SP receiver relationship to other receivers is defined in 4.3.3.

#### **5.18.4 PTT\_T (phy layer transmitter training transmit pattern) state machine**

##### **5.18.4.1 PTT\_T state machine overview**

The PTT\_T state machine's function is to:

- a) transmit the status information and control information to train the attached phy's transmitter;
- b) determine when the local phy's transmitter training is complete; and
- c) determine when the attached phy's transmitter training is complete.

This state machine consists of the following states:

- a) PTT\_T0:Idle (see 5.18.4.2) (initial state);
- b) PTT\_T1:Initialize (see 5.18.4.3);
- c) PTT\_T2:Tx\_Training (see 5.18.4.4); and
- d) PTT\_T3:Local\_Tx\_Training (see 5.18.4.5).

This state machine receives the following request from the management application layer:

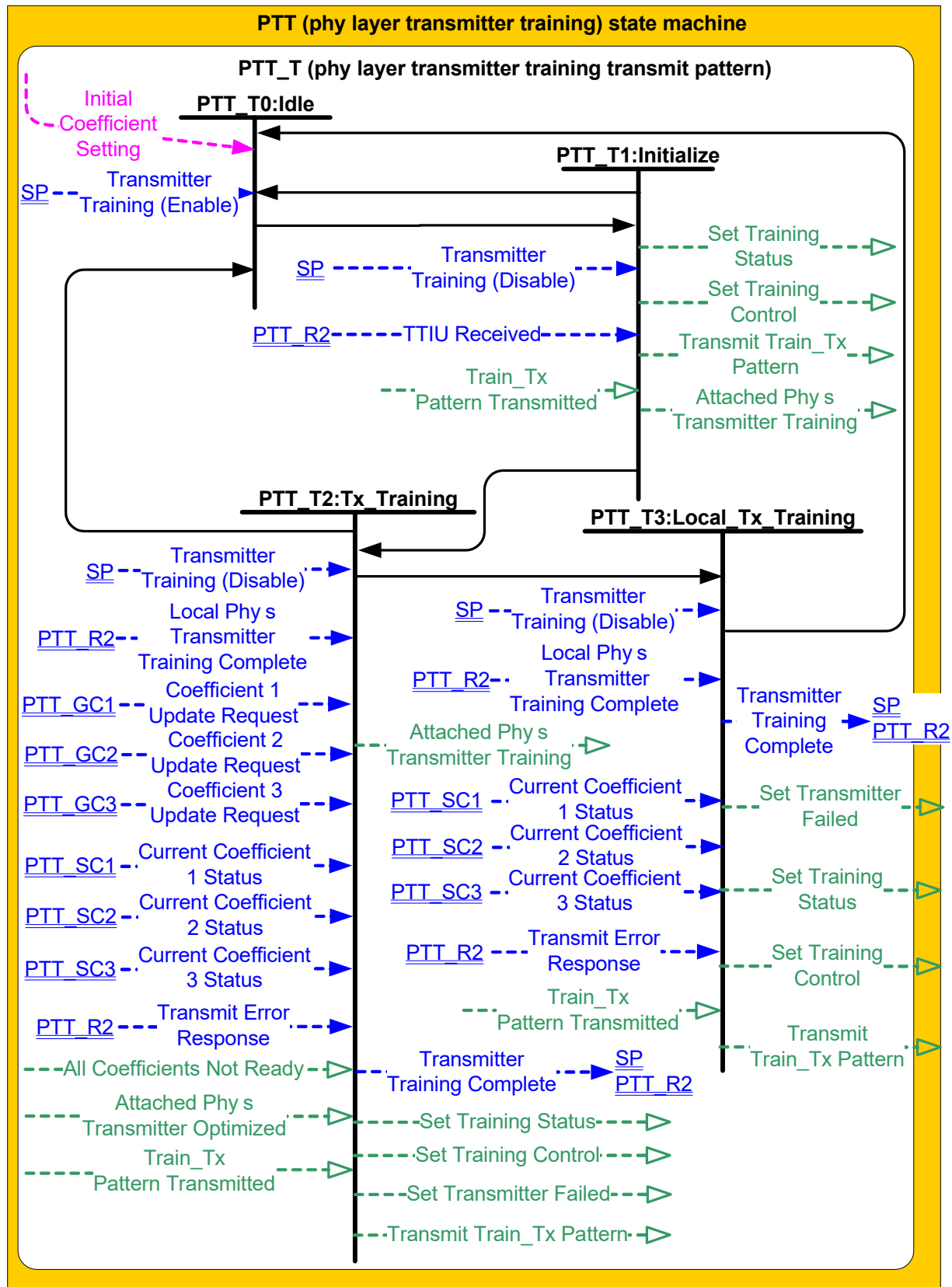
- a) Initial Coefficient Setting with an argument of Normal, No\_Equalization, Reference\_1, or Reference\_2.

This state machine sends the following message to the SP state machine:

- a) Transmitter Training Complete.



Figure 116 shows the PTT\_T state machine.



**Figure 116 – PTT\_T (phy layer transmitter training transmit pattern) state machine**

**5.18.4.2 PTT\_T0:Idle state****5.18.4.2.1 State description**

This is the initial state of this state machine.

This state waits for a request to start transmitter training.

**5.18.4.2.2 Transition PTT\_T0:Idle to PTT\_T1:Initialize**

This transition shall occur after receiving:

- a) a Transmitter Training (Enable) message; and
- b) an Initial Coefficient Setting request.

This transition shall include the coefficient setting as an argument (i.e., Normal, No\_Equalization, Reference\_1, or Reference\_2).

**5.18.4.3 PTT\_T1:Initialize state****5.18.4.3.1 State description**

This state:

- a) sets the initial values of the Training Status word and Training Control word (see 5.10.2);
- b) requests Train\_Tx patterns be transmitted; and
- c) waits for the receipt of a Control/Status TTIU.

Upon entry into this state, this state shall:

- 1) send an Attached Phy's Transmitter Training (Stop) message to the SP receiver;
- 2) set the bits and fields of the Training Status word (see 5.10.2) as follows:
  - A) set the TRAIN COMP bit to zero;
  - B) set the TX INIT bit to one;
  - C) set the COEFFICIENT 1 STATUS field to 00b (i.e., ready);
  - D) set the COEFFICIENT 2 STATUS field to 00b (i.e., ready); and
  - E) set the COEFFICIENT 3 STATUS field to 00b (i.e., ready);
- 3) set the fields of the Training Control word as follows:
  - A) set the PATTERN TYPE field set to 000b (i.e., Control/Status TTIU);
  - B) set the COEFFICIENT SETTINGS field to 00b (i.e., normal);
  - C) set the COEFFICIENT 1 REQUEST field to 00b (i.e., hold);
  - D) set the COEFFICIENT 2 REQUEST field to 00b (i.e., hold); and
  - E) set the COEFFICIENT 3 REQUEST field to 00b (i.e., hold);
- 4) send a Set Training Control message to the SP transmitter;
- 5) send a Set Training Status message to the SP transmitter;
- 6) send a Transmit Train\_Tx Pattern message to the SP transmitter;
- 7) wait for a Train\_Tx Pattern Transmitted message; and
- 8) repeat steps 6) and 7) until this state receives a TTIU Received message with an Attached Transmitter Initialized argument.

If this state receives a TTIU Received message with an Attached Transmitter Not Initialized argument, then this state shall:

- 1) set the TX INIT bit to zero; and
- 2) send a Set Training Status message to the SP transmitter.

**5.18.4.3.2 Transition PTT\_T1:Initialize to PTT\_T0:Idle**

This transition shall occur:

- a) after receiving a Transmitter Training (Disable) message.

**5.18.4.3.3 Transition PTT\_T1:Initialize to PTT\_T2:Tx\_Training**

This transition shall occur:

- a) after receiving a TTIU Received message with an Attached Transmitter Initialized argument.

This transition shall include the coefficient setting as an argument (i.e., Normal, No\_Equalization, Reference\_1, or Reference\_2).

**5.18.4.4 PTT\_T2:Tx\_Training state****5.18.4.4.1 State description**

This state:

- a) sets values of the Training Status word and Training Control word (see 5.10.2);
- b) requests Train\_Tx patterns be transmitted;
- c) waits for attached phy's transmitter training to complete; and
- d) transmits the local phy's transmitter status to the attached phy's receiver.

**5.18.4.4.2 Entry conditions**

Upon entry into this state, this state shall:

- 1) set the bits and fields of the Training Status word (see 5.10.2) as follows:
  - A) set the TRAIN COMP bit to zero;
  - B) set the TX INIT bit to zero;
  - C) set the COEFFICIENT 1 STATUS field to 00b (i.e., ready);
  - D) set the COEFFICIENT 2 STATUS field to 00b (i.e., ready); and
  - E) set the COEFFICIENT 3 STATUS field to 00b (i.e., ready);
- 2) if the Coefficient Settings argument is set to No\_Equalization, Reference\_1, or Reference\_2, then process the coefficient settings as defined in 5.18.4.4.5;
- 3) send an Attached Phy's Transmitter Training (Start) message to the SP receiver;
- 4) set the fields of the Training Control word (see table 78) as follows:
  - A) set the PATTERN TYPE field to 000b (i.e., Control/Status TTIU);
  - B) set the COEFFICIENT SETTINGS field to 00b (i.e., normal);
  - C) set the COEFFICIENT 1 REQUEST field to 00b (i.e., hold);
  - D) set the COEFFICIENT 2 REQUEST field to 00b (i.e., hold); and
  - E) set the COEFFICIENT 3 REQUEST field to 00b (i.e., hold);
- 5) send a Set Training Control message to the SP transmitter;
- 6) send a Set Training Status message to the SP transmitter;
- 7) send a Transmit Train\_Tx Pattern message to the SP transmitter;
- 8) wait for a Train\_Tx Pattern Transmitted message; and
- 9) repeat steps 7) and 8) while there is no other looping occurring within this state.

#### 5.18.4.4.3 Control word and status word mappings

Table 97 defines the mapping of the messages received from the PTT\_GC state machines to the Training Control word.

**Table 97 – Mapping messages to the Training Control word**

Message received	Training Control word field <sup>e</sup>
Coefficient 1 Update Request <sup>a b</sup>	COEFFICIENT 1 REQUEST
Coefficient 2 Update Request <sup>a c</sup>	COEFFICIENT 2 REQUEST
Coefficient 3 Update Request <sup>a d</sup>	COEFFICIENT 3 REQUEST
<sup>a</sup> This message contains a Decrement, Increment, Hold, No_Equalization, Reference_1, or Reference_2 argument. <sup>b</sup> This message is from the PTT_GC1 state machine. <sup>c</sup> This message is from the PTT_GC2 state machine. <sup>d</sup> This message is from the PTT_GC3 state machine. <sup>e</sup> If the message is received with an argument of Hold, Increment, or Decrement, then the Training Control word field shall be set to the argument received in the corresponding message (i.e., 00b (i.e., hold), 01b (i.e., increment), or 10b (i.e., decrement)). If the message is received with an argument of No_Equalization, Reference_1, or Reference_2, then the Training Control word field shall be set to 00b (i.e., hold).	

After this state sets the Training Control word to the argument received in the Coefficient 1 Update Request message, Coefficient 2 Update Request message, or Coefficient 3 Update Request message, this state shall send a Set Training Control message to the SP transmitter after receiving a Train\_Tx Pattern Transmitted message.

Table 98 defines the mapping of the messages received from the PTT\_SC1 state machine, PTT\_SC2 state machine, and PTT\_SC3 state machine to the Training Status word.

**Table 98 – Mapping messages from PTT\_SC1 state machine, PTT\_SC2 state machine, and PTT\_SC3 state machine to the Training Status word**

Messages received	Training Status word field <sup>e</sup>
Current Coefficient 1 Status <sup>a b</sup>	COEFFICIENT 1 STATUS
Current Coefficient 2 Status <sup>a c</sup>	COEFFICIENT 2 STATUS
Current Coefficient 3 Status <sup>a d</sup>	COEFFICIENT 3 STATUS
<sup>a</sup> This message contains a Maximum, Minimum, Update_Complete, or Ready argument. <sup>b</sup> This message is from the PTT_SC1 state machine. <sup>c</sup> This message is from the PTT_SC2 state machine. <sup>d</sup> This message is from the PTT_SC3 state machine. <sup>e</sup> On receipt of the message this state shall: <ol style="list-style-type: none"> <li>1) set the Training Status word field to the argument received in the corresponding message (i.e., 00b (i.e., ready), 01b (i.e., update complete), 10b (i.e., minimum), or 11b (i.e., maximum)); and</li> <li>2) send a Set Training Status message to the SP transmitter after receiving a Train_Tx Pattern Transmitted message.</li> </ol>	

**5.18.4.4.4 Error message handling**

If this state receives a Transmit Error Response message, then this state shall:

- 1) set the Error Response TTIU fields (see table 83) as follows:
  - A) set the PATTERN TYPE field to 111b (i.e., Error Response TTIU); and
  - B) set the Error Response TTIU fields to the arguments received in the Transmit Error Response message as defined in table 99;
- 2) send a Set Transmitter Failed message to the SP transmitter;
- 3) send a Transmit Train Tx Pattern message to the SP transmitter;
- 4) wait for a Train Tx Pattern Transmitted message;
- 5) send a Set Training Control message to the SP transmitter;
- 6) set the bits and fields of the Training Status word as follows:
  - A) set the PATTERN TYPE field set to 000b (i.e., Control/Status TTIU);
  - B) set the TRAIN COMP bit to zero;
  - C) set the TX INIT bit to zero;
  - D) set the COEFFICIENT 1 STATUS field to 00b (i.e., ready);
  - E) set the COEFFICIENT 2 STATUS field to 00b (i.e., ready); and
  - F) set the COEFFICIENT 3 STATUS field to 00b (i.e., ready);
- 7) send a Set Training Status message to the SP transmitter;
- 8) send a Transmit Train Tx Pattern message to the SP transmitter; and
- 9) wait for a Train Tx Pattern Transmitted message.

Table 99 defines that mapping of the Transmit Error Response message arguments to the Error Response TTIU fields.

**Table 99 – Mapping Transmit Error Response message arguments to Error Response TTIU fields**

Transmit Error Response message argument	Error Response TTIU field
Coefficient Settings	RECEIVED COEFFICIENT SETTINGS
Coefficient 1 Request	RECEIVED COEFFICIENT 1 REQUEST
Coefficient 2 Request	RECEIVED COEFFICIENT 2 REQUEST
Coefficient 3 Request	RECEIVED COEFFICIENT 3 REQUEST
Error	ERROR CODE

**5.18.4.4.5 Resetting attached phy's transmitter**

If this state:

- a) receives a Coefficient 1 Update Request message, Coefficient 2 Update Request message, or Coefficient 3 Update Request message with an argument of No\_Equalization, Reference\_1, or Reference\_2; or
- b) is entered with Coefficient Settings argument set to No\_Equalization, Reference\_1, or Reference\_2 (see 5.18.4.4.2),

then this state shall:

- 1) send an Attached Phy's Transmitter Training (Stop) message to the SP receiver;
- 2) if:
  - A) this state is entered with an argument of No\_Equalization or receives an argument of No\_Equalization, then this state shall set the fields of the Training Control word (see table 78) as follows:
    - a) set the PATTERN TYPE field to 000b (i.e., Control/Status TTIU);

- b) set the COEFFICIENT SETTINGS field to 11b (i.e., no equalization);
- c) set the COEFFICIENT 1 REQUEST field to 00b (i.e., hold);
- d) set the COEFFICIENT 2 REQUEST field to 00b (i.e., hold); and
- e) set the COEFFICIENT 3 REQUEST field to 00b (i.e., hold);
- B) this state is entered with an argument of Reference\_1 or receives an argument of Reference\_1, then this state shall set the fields of the Training Control word as follows:
  - a) set the PATTERN TYPE field to 000b (i.e., Control/Status TTIU);
  - b) set the COEFFICIENT SETTINGS field to 01b (i.e., reference 1);
  - c) set the COEFFICIENT 1 REQUEST field to 00b (i.e., hold);
  - d) set the COEFFICIENT 2 REQUEST field to 00b (i.e., hold); and
  - e) set the COEFFICIENT 3 REQUEST field to 00b (i.e., hold);
- or
- C) this state is entered with an argument of Reference\_2 or receives an argument of Reference\_2, then this state shall set the fields of the Training Control word as follows:
  - a) set the PATTERN TYPE field set to 000b (i.e., Control/Status TTIU);
  - b) set the COEFFICIENT SETTINGS field to 10b (i.e., reference 2);
  - c) set the COEFFICIENT 1 REQUEST field to 00b (i.e., hold);
  - d) set the COEFFICIENT 2 REQUEST field to 00b (i.e., hold); and
  - e) set the COEFFICIENT 3 REQUEST field to 00b (i.e., hold);
- 3) send a Set Training Control message to the SP transmitter;
- 4) send a Set Training Status message to the SP transmitter;
- 5) send a Transmit Train\_Tx Pattern message to the SP transmitter;
- 6) wait for a Train\_Tx Pattern Transmitted message;
- 7) repeat steps 5) and 6) when there is no other looping occurring within this state (i.e., if a Transmit Error Response message is received, then the error as defined in 5.18.4.5.5 shall be processed) until this state receives an All Coefficients Not Ready message;
- 8) send an Attached Phy's Transmitter Training (Start) message to the SP receiver; and
- 9) set the COEFFICIENT SETTINGS field to 00b (i.e., normal).

#### 5.18.4.4.6 Local phy's transmitter and attached phy's transmitter training completed

If this state receives:

- a) a Local Phy's Transmitter Training Complete message; and
- b) an Attached Phy's Transmitter Optimized message,

then this state shall:

- 1) set the TRAIN COMP bit of the Training Status word (see table 78) to one;
- 2) set the fields of the Training Control word as follows:
  - A) set the PATTERN TYPE field set to 000b (i.e., Control/Status TTIU);
  - B) set the COEFFICIENT SETTINGS field to 00b (i.e., normal);
  - C) set the COEFFICIENT 1 REQUEST field to 00b (i.e., hold);
  - D) set the COEFFICIENT 2 REQUEST field to 00b (i.e., hold); and
  - E) set the COEFFICIENT 3 REQUEST field to 00b (i.e., hold);
- 3) send a Set Training Control message to the SP transmitter;
- 4) send a Set Training Status message to the SP transmitter;
- 5) send a Transmit Train\_Tx Pattern message to the SP transmitter;
- 6) wait for a Train\_Tx Pattern Transmitted message;
- 7) repeat steps 5) and 6) five times; and
- 8) send a Transmitter Training Complete message to the SP34:SAS\_Train\_Tx state and the PTT\_R2:Receive\_Train\_Tx\_Pattern state.

#### 5.18.4.4.7 Transition PTT\_T2:Tx\_Training to PTT\_T0:Idle

This transition shall occur after:

- a) receiving a Transmitter Training (Disable) message; or

- b) sending the Transmitter Training Complete messages.

#### 5.18.4.4.8 Transition PTT\_T2:Tx\_Training to PTT\_T3:Local\_Tx\_Training

If this state has not received a Local Phy's Transmitter Training Complete message, then this transition shall occur:

- a) after receiving an Attached Phy's Transmitter Optimized message.

This transition shall include the following from the current Training Status word as arguments:

- a) Coefficient 1 Status;
- b) Coefficient 2 Status; and
- c) Coefficient 3 Status.

#### 5.18.4.5 PTT\_T3:Local\_Tx\_Training state

##### 5.18.4.5.1 State description

This state:

- a) sets values of the Training Status word and the Training Control word (see 5.10.2);
- b) requests Train\_Tx patterns be transmitted; and
- c) waits for the local phy's transmitter training to complete.

##### 5.18.4.5.2 Entry conditions

Upon entry into this state, this state shall:

- 1) set the bits and fields of the Training Status word (see table 78) as follows:
  - A) set the TRAIN COMP bit to one;
  - B) set the TX INIT bit to zero;
  - C) set the COEFFICIENT 1 STATUS field to the Coefficient 1 Status argument;
  - D) set the COEFFICIENT 2 STATUS field to the Coefficient 2 Status argument; and
  - E) set the COEFFICIENT 3 STATUS field to the Coefficient 3 Status argument;
- 2) set the fields of the Training Control word as follows:
  - A) set the PATTERN TYPE field to 000b (i.e., Control/Status TTIU);
  - B) set the COEFFICIENT SETTINGS field to 00b (i.e., normal);
  - C) set the COEFFICIENT 1 REQUEST field to 00b (i.e., hold);
  - D) set the COEFFICIENT 2 REQUEST field to 00b (i.e., hold); and
  - E) set the COEFFICIENT 3 REQUEST field to 00b (i.e., hold);
- 3) send a Set Training Control message to the SP transmitter;
- 4) send a Set Training Status message to the SP transmitter;
- 5) send a Transmit Train\_Tx Pattern message to the SP transmitter;
- 6) wait for a Train\_Tx Pattern Transmitted message; and
- 7) repeat steps 5) and 6) while there is no other looping occurring with in this state.

##### 5.18.4.5.3 Status word mappings

Table 98 defines the mapping of the messages received from the PTT\_SC1 state machine, PTT\_SC2 state machine, and PTT\_SC3 state machine to the Training Status word.

##### 5.18.4.5.4 Local phy's transmitter and attached phy's transmitter training completed

If this state receives a Local Phy's Transmitter Training Complete message and this state has received at least six Train\_Tx Pattern Transmitted messages, then this state shall send a Transmitter Training Complete message to the SP34:SAS\_Train\_Tx state and the PTT\_R2:Receive\_Train\_Tx\_Pattern state.

#### 5.18.4.5.5 Error message handling

If this state receives a Transmit Error Response message, then this state shall:

- 1) set the Error Response TTIU fields (see table 83) as follows:
  - A) set the PATTERN TYPE field to 111b (i.e., Error Response TTIU); and
  - B) set the Error Response TTIU fields to the arguments received in the Transmit Error Response message as defined in table 99;
- 2) send a Set Transmitter Failed message to the SP transmitter;
- 3) send a Transmit Train\_Tx Pattern message to the SP transmitter;
- 4) wait for a Train\_Tx Pattern Transmitted message;
- 5) send a Set Training Control message to the SP transmitter;
- 6) set the bits and fields of the Training Status word (see table 78) as follows:
  - A) set the PATTERN TYPE field set to 000b (i.e., Control/Status TTIU);
  - B) set the TRAIN COMP bit to one;
  - C) set the TX INIT bit to zero;
  - D) set the COEFFICIENT 1 STATUS field to 00b (i.e., ready);
  - E) set the COEFFICIENT 2 STATUS field to 00b (i.e., ready); and
  - F) set the COEFFICIENT 3 STATUS field to 00b (i.e., ready);
- 7) send a Set Training Status message to the SP transmitter;
- 8) send a Transmit Train Tx Pattern message to the SP transmitter; and
- 9) wait for a Train Tx Pattern Transmitted message.

#### 5.18.4.5.6 Transition PTT\_T3:Local\_Tx\_Training to PTT\_T0:Idle

This transition shall occur after:

- a) receiving a Transmitter Training (Disable) message; or
- b) sending the Transmitter Training Complete messages.

### 5.18.5 PTT\_R (phy layer transmitter training receive pattern) state machine

#### 5.18.5.1 PTT\_R state machine overview

The PTT\_R state machine's function is to:

- a) monitor the receipt of the attached phy's Train\_Tx pattern;
- b) start pattern lock synchronization;
- c) transfer adjustment requests received from the attached phy's receiver to the local phy's transmitter; and
- d) transfer the attached phy's transmitter coefficient status to the local phy's receiver.

This state machine consists of the following states:

- a) PTT\_R0:Idle (see 5.18.5.2) (initial state);
- b) PTT\_R1:Initialize (see 5.18.5.3); and
- c) PTT\_R2:Receive\_Train\_Tx\_Pattern (see 5.18.5.4).

This state machine receives the following messages from the SP state machine:

- a) Transmitter Training (Enable); and
- b) Transmitter Training (Disable).



Figure 117 shows the PTT\_R state machine.

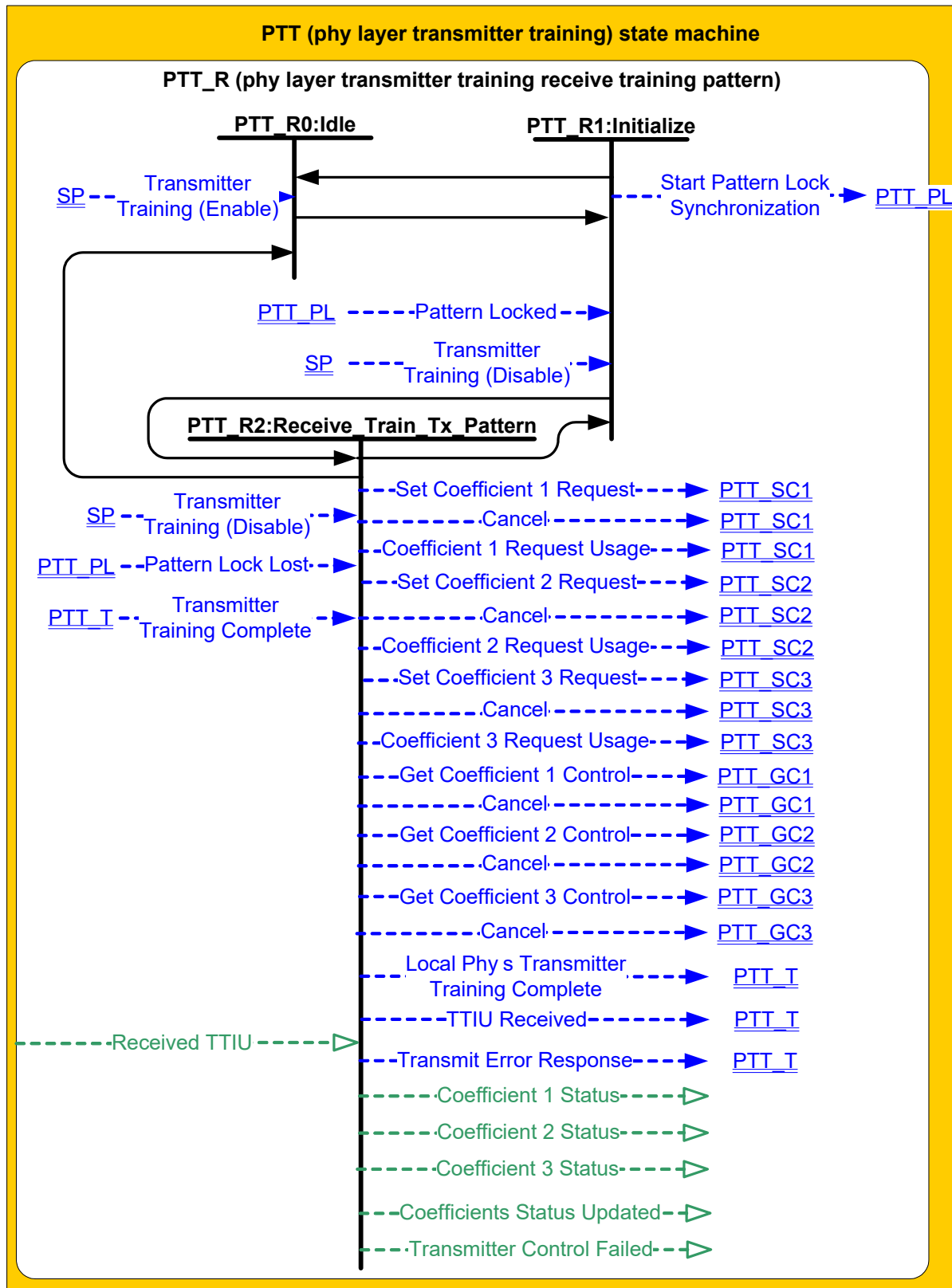


Figure 117 – PTT\_R (phy layer transmitter training receive pattern) state machine

**5.18.5.2 PTT\_R0:Idle state****5.18.5.2.1 State description**

This is the initial state of this state machine.

This state waits for a request to start transmitter training.

**5.18.5.2.2 Transition PTT\_R0:Idle to PTT\_R1:Initialize**

This transition shall occur:

- a) after receiving a Transmitter Training (Enable) message.

**5.18.5.3 PTT\_R1:Initialize state****5.18.5.3.1 State description**

This state waits for pattern lock synchronization.

Upon entry into this state, this state shall:

- a) send a Start Pattern Lock Synchronization message to the PTT\_PL phy layer state machine.

**5.18.5.3.2 Transition PTT\_R1:Initialize to PTT\_R0:Idle**

This transition shall occur:

- a) after receiving a Transmitter Training (Disable) message.

**5.18.5.3.3 Transition PTT\_R1:Initialize to PTT\_R2:Receive\_Train\_Tx\_Pattern**

This transition shall occur:

- a) after receiving a Pattern Locked message.

**5.18.5.4 PTT\_R2:Receive\_Train\_Tx\_Pattern state****5.18.5.4.1 State description**

This state:

- a) receives Training Control words and Training Status words;
- b) transfers adjustment requests received from the attached phy's receiver to the local phy's transmitter;
- c) checks the TTIU for errors; and
- d) transfers the attached phy's transmitter coefficient status to the local phy's receiver.

If this state receives a Received TTIU message with an odd number of bits set to zero in the TTIU, then this state shall discard the TTIU.

If this state receives a Received TTIU message and the TTIU is a Control/Status TTIU that contains:

- a) an unsupported value;
- b) a reserved value;
- c) a reserved bit; or
- d) a reserved combination in the Training Control word (see table 100),

then this state shall:

- 1) send a Transmit Error Response message to the PTT\_T state machine with the arguments defined in table 100;
- 2) send a Cancel message to the PTT\_SC1 state machine, PTT\_SC2 state machine, and PTT\_SC3 state machine; and



- b) a PATTERN TYPE field of the Training Control word set to 000b (i.e., Control/Status TTIU); and
- c) an even number of bits set to zero in the TTIU,

then:

- a) if the TX INIT bit is set to zero in the received Training Status word (see table 78), then this state shall:
  - 1) send:
    - 1) the messages and arguments shown in table 101 to the SP receiver; and
    - 2) a Coefficients Status Updated message to the SP receiver;
  - 2) send to the PTT\_GC1 state machine, PTT\_GC2 state machine, and PTT\_GC3 state machine the messages and arguments shown in table 102;
  - 3) if the TRAIN COMP bit in the received Training Status word:
    - a) is set to zero, then send to the PTT\_SC1 state machine, PTT\_SC2 state machine, and PTT\_SC3 state machine the messages and arguments shown in table 103 and table 104; or
    - b) is set to one, then send a Cancel message to the PTT\_SC1 state machine, PTT\_SC2 state machine, and PTT\_SC3 state machine;

and

  - 4) send a TTIU Received (Attached Transmitter Initialized) message to the PTT\_T state machine;

or

- b) if TX INIT bit is set to one in the received Training Status word, then this state shall send a TTIU Received (Attached Transmitter Not Initialized) message to the PTT\_T state machine.

Table 101 defines the mapping of transmitted status word fields to messages sent to the SP receiver.

**Table 101 – Mapping the Training Status word to SP receiver messages**

Training Status word field	Message to SP receiver
COEFFICIENT 1 STATUS	Coefficient 1 Status <sup>a</sup>
COEFFICIENT 2 STATUS	Coefficient 2 Status <sup>a</sup>
COEFFICIENT 3 STATUS	Coefficient 3 Status <sup>a</sup>
<sup>a</sup> This message's argument is set to the value (i.e., Maximum, Minimum, Update_Complete, or Ready) of the corresponding Training Status word field associated with the attached phy's transmitter.	

Table 102 defines the mapping of Training Status word fields to messages sent to the PTT\_GC1 state machine, PTT\_GC2 state machine, and PTT\_GC3 state machine.

**Table 102 – Mapping the Training Status word to PTT\_GC1 state machine messages, PTT\_GC2 state machine messages, and PTT\_GC3 state machine messages**

Training Status word			Message
Field name	Code	Name	
COEFFICIENT 1 STATUS	00b	ready	Get Coefficient 1 Control (Start) <sup>a</sup>
	01b	update complete	Get Coefficient 1 Control (Restart) <sup>a</sup>
	10b	minimum	
	11b	maximum	
COEFFICIENT 2 STATUS	00b	ready	Get Coefficient 2 Control (Start) <sup>b</sup>
	01b	update complete	Get Coefficient 2 Control (Restart) <sup>b</sup>
	10b	minimum	
	11b	maximum	
COEFFICIENT 3 STATUS	00b	ready	Get Coefficient 3 Control (Start) <sup>c</sup>
	01b	update complete	Get Coefficient 3 Control (Restart) <sup>c</sup>
	10b	minimum	
	11b	maximum	
<sup>a</sup> This message is sent to the PTT_GC1 state machine. <sup>b</sup> This message is sent to the PTT_GC2 state machine. <sup>c</sup> This message is sent to the PTT_GC3 state machine.			

Table 103 defines the mapping of transmitted control word fields to messages sent to the PTT\_SC1 state machine, PTT\_SC2 state machine, and PTT\_SC3 state machine.

**Table 103 – Mapping the Training Control word to PTT\_SC1 state machine messages, PTT\_SC2 state machine messages, and PTT\_SC3 state machine messages**

Training Control word fields			Message
COEFFICIENT SETTINGS field code			
Code	Name		
00b	normal	see table 104	see table 104
01b	reference_1 <sup>g</sup>	COEFFICIENT 1 REQUEST	Set Coefficient 1 Request <sup>a d</sup>
		COEFFICIENT 2 REQUEST	Set Coefficient 2 Request <sup>a e</sup>
		COEFFICIENT 3 REQUEST	Set Coefficient 3 Request <sup>a f</sup>
10b	reference_2 <sup>g</sup>	COEFFICIENT 1 REQUEST	Set Coefficient 1 Request <sup>b d</sup>
		COEFFICIENT 2 REQUEST	Set Coefficient 2 Request <sup>b e</sup>
		COEFFICIENT 3 REQUEST	Set Coefficient 3 Request <sup>b f</sup>
11b	no_equalization <sup>g</sup>	COEFFICIENT 1 REQUEST	Set Coefficient 1 Request <sup>c d</sup>
		COEFFICIENT 2 REQUEST	Set Coefficient 2 Request <sup>c e</sup>
		COEFFICIENT 3 REQUEST	Set Coefficient 3 Request <sup>c f</sup>
<sup>a</sup> This message argument is set to Set Reference_1 regardless of the values in the COEFFICIENT 1 REQUEST field, the COEFFICIENT 2 REQUEST field, and the COEFFICIENT 3 REQUEST field.			
<sup>b</sup> This message argument is set to Set Reference_2 regardless of the values in the COEFFICIENT 1 REQUEST field, the COEFFICIENT 2 REQUEST field, and the COEFFICIENT 3 REQUEST field.			
<sup>c</sup> This message argument is set to Set No_Equalization regardless of the values in the COEFFICIENT 1 REQUEST field, the COEFFICIENT 2 REQUEST field, and the COEFFICIENT 3 REQUEST field.			
<sup>d</sup> This message is sent to the PTT_SC1 state machine.			
<sup>e</sup> This message is sent to the PTT_SC2 state machine.			
<sup>f</sup> This message is sent to the PTT_SC3 state machine.			
<sup>g</sup> The Coefficient 1 Request Usage message (see table 104), the Coefficient 2 Request Usage message (see table 104), or the Coefficient 3 Request Usage message (see table 104) shall not be sent to the PTT_SC1 state machine, PTT_SC 2 state machine, or PTT_SC 3 state machine.			

Table 104 defines the mapping of the Coefficient Request byte to messages sent to the PTT\_SC3 state machine, PTT\_SC2 state machine, and PTT\_SC1 state machine.

**Table 104 – Mapping Coefficient Request byte to PTT\_SC3 state machine message, PTT\_SC2 state machine message, and PTT\_SC1 state machine messages**

<b>Coefficient Request byte (table 81)</b>	<b>PTT_SC3 state machine message</b>		<b>PTT_SC2 state machine message</b>		<b>PTT_SC1 state machine message</b>	
<b>Code</b>	<b>Set Coefficient 3 Request (argument)</b>	<b>Coefficient 3 Request Usage (argument) <sup>a</sup></b>	<b>Set Coefficient 2 Request (argument)</b>	<b>Coefficient 2 Request Usage (argument) <sup>a</sup></b>	<b>Set Coefficient 1 Request (argument)</b>	<b>Coefficient 1 Request Usage (argument) <sup>a</sup></b>
00h	Hold	message not sent	Hold	message not sent	Hold	message not sent
01h	Hold	message not sent	Hold	message not sent	Increment	Single
02h	Hold	message not sent	Hold	message not sent	Decrement	Single
04h	Hold	message not sent	Increment	Single	Hold	message not sent
05h	Hold	message not sent	Increment	Dual	Increment	Dual
08h	Hold	message not sent	Decrement	Single	Hold	message not sent
0Ah	Hold	message not sent	Decrement	Dual	Decrement	Dual
10h	Increment	Single	Hold	message not sent	Hold	message not sent
14h	Increment	Dual	Increment	Dual	Hold	message not sent
20h	Decrement	Single	Hold	message not sent	Hold	message not sent
28h	Decrement	Dual	Decrement	Dual	Hold	message not sent
<sup>a</sup> If a Coefficient 1 Request Usage message is sent to the PTT_SC1 state machine, a Coefficient 2 Request Usage message is sent to the PTT_SC 2 state machine, or a Coefficient 3 Request Usage message is sent to the PTT_SC 3 state machine, then PTT_R2 shall send: <ol style="list-style-type: none"> <li>1) the Coefficient 1 Request Usage message, Coefficient 2 Request Usage message, or Coefficient 3 Request Usage message; and</li> <li>2) the Set Coefficient 1 Request message, Set Coefficient 2 Request message, or Set Coefficient 3 Request message.</li> </ol>						

This state count of the number of consecutive received Training Status words in which the TRAIN COMP bit is set to one. If this number is three or greater, then this state shall send a Local Phy's Transmitter Training Complete message to the PTT\_T state machine.

#### **5.18.5.4.2 Transition PTT\_R2:Receive\_Train\_Tx\_Pattern to PTT\_R0:Idle**

This transition shall occur after receiving a:

- a) Transmitter Training (Disable) message; or
- b) Transmitter Training Complete message.

#### **5.18.5.4.3 Transition PTT\_R2:Receive\_Train\_Tx\_Pattern to PTT\_R1:Initialize**

This transition shall occur:

- a) after receiving a Pattern Lock Lost message.

### **5.18.6 PTT\_SC (phy layer transmitter training set transmitter coefficient) state machines**

#### **5.18.6.1 PTT\_SC (phy layer transmitter training set transmitter coefficient) state machines overview**

The PTT\_SC1 state machines' functions are to adjust the local phy's transmitter coefficients' and to manage those coefficients' status.



Figure 118 shows the PTT\_SC1 state machine, PTT\_SC2 state machine, and PTT\_SC3 state machine.

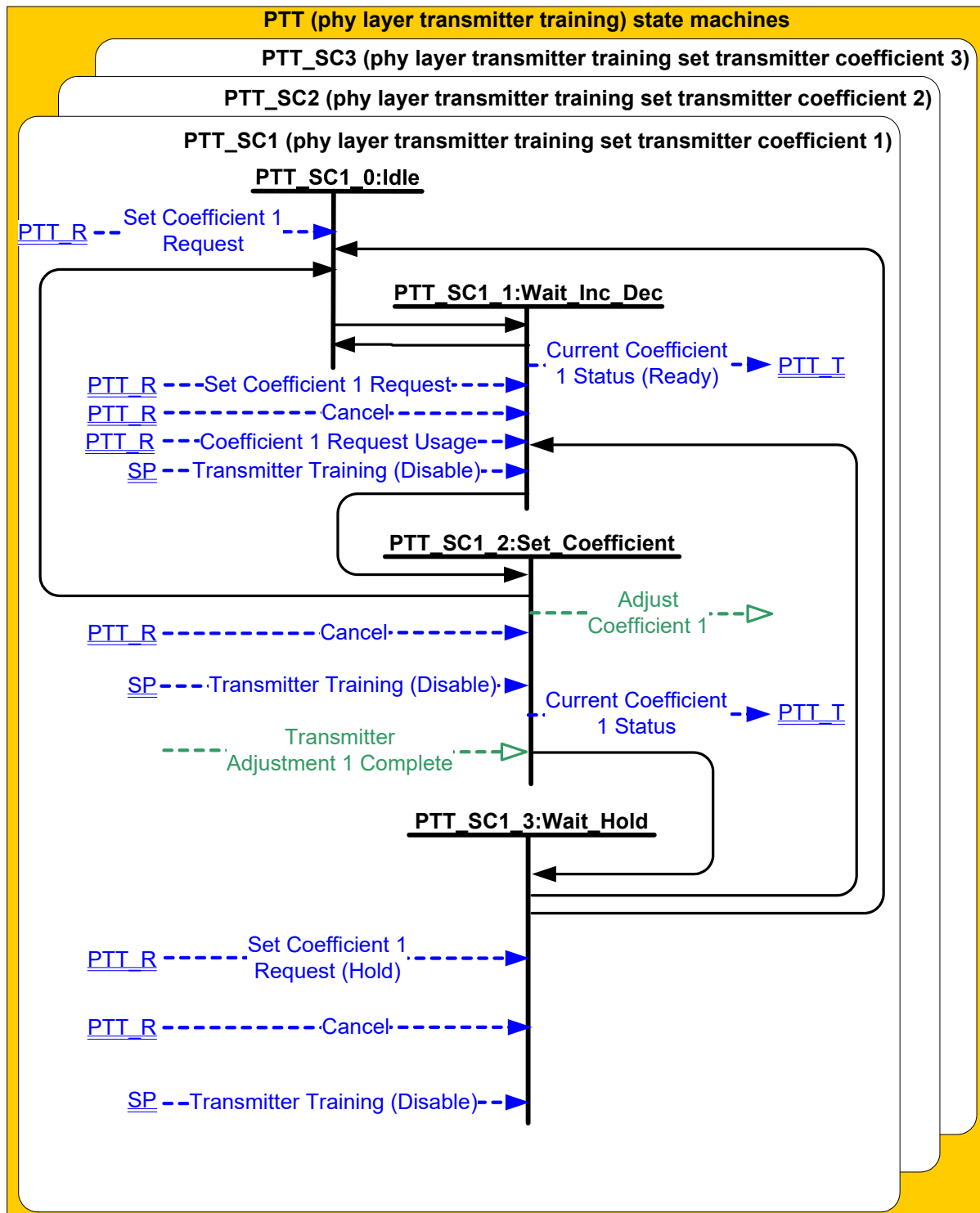


Figure 118 – PTT\_SC1, PTT\_SC2, and PTT\_SC3 (phy layer transmitter training set transmitter coefficient) state machines

### 5.18.6.2 PTT\_SC1 state machine overview

The PTT\_SC1 state machine's function is to adjust one of the local phy's transmitter coefficients and to manage that coefficient's status. This state machine consists of the following states:

- a) PTT\_SC1\_0:Idle (see 5.18.6.3) (initial state);
- b) PTT\_SC1\_1:Wait\_Inc\_Dec (see 5.18.6.4);
- c) PTT\_SC1\_2:Set\_Coefficient (see 5.18.6.5); and
- d) PTT\_SC1\_3:Wait\_Hold (see 5.18.6.6).

This state machine receives the following message from the SP state machine:

- a) Transmitter Training (Disable).

### 5.18.6.3 PTT\_SC1\_0:Idle state

#### 5.18.6.3.1 State description

This is the initial state of this state machine.

This state waits for a request to start setting the local phy's transmitter coefficient.

#### 5.18.6.3.2 Transition PTT\_SC1\_0:Idle to PTT\_SC1\_1:Wait\_Inc\_Dec

This transition shall occur:

- a) after receiving a Set Coefficient 1 Request message.

### 5.18.6.4 PTT\_SC1\_1:Wait\_Inc\_Dec state

#### 5.18.6.4.1 State description

This state waits for a request to:

- a) increment or decrement the local phy's transmitter coefficient; or
- b) set the local phy's transmitter coefficient to the no equalization value, reference\_1 value, or reference\_2 value.

Upon entry into this state, this state shall:

- a) send a Current Coefficient 1 Status (Ready) message to the PTT\_T state machine.

#### 5.18.6.4.2 Transition PTT\_SC1\_1:Wait\_Inc\_Dec to PTT\_SC1\_0:Idle

This transition shall occur after receiving a:

- a) Cancel message; or
- b) Transmitter Training (Disable) message.

#### 5.18.6.4.3 Transition PTT\_SC1\_1:Wait\_Inc\_Dec to PTT\_SC1\_2:Set\_Coefficient

This transition shall occur:

- a) after receiving a Set Coefficient 1 Request message with an argument of Decrement, Increment, Set No\_Equalization, Set Reference\_1, or Set Reference\_2.

This transition shall include the argument from the:

- a) Set Coefficient 1 Request message (i.e., Increment, Decrement, Set No\_Equalization, Set Reference\_1, or Set Reference\_2); and
- b) Coefficient 1 Request Usage message (i.e., Single or Dual), if any.

**5.18.6.5 PTT\_SC1\_2:Set\_Coefficient state****5.18.6.5.1 State description**

This state:

- a) requests the SP transmitter increment the coefficient, decrement the coefficient, set the coefficient to its no equalization value, set the coefficient to its reference\_1 value, or set the coefficient to its reference\_2 value;
- b) waits for the SP transmitter to complete the requested adjustment; and
- c) reports the status of the SP transmitter's coefficient to the PTT\_T state machine.

If this state was entered with an Increment argument and a Single argument, then this state shall send an Adjust Coefficient 1 message to the SP transmitter with the following arguments:

- a) Increment; and
- b) Single.

If this state was entered with an Increment argument and a Dual argument, then this state shall send an Adjust Coefficient 1 message to the SP transmitter with the following arguments:

- a) Increment; and
- b) Dual.

If this state was entered with a Decrement argument and a Single argument, then this state shall send an Adjust Coefficient 1 message to the SP transmitter with the following arguments:

- a) Decrement; and
- b) Single.

If this state was entered with a Decrement argument and a Dual argument, then this state shall send an Adjust Coefficient 1 message to the SP transmitter with the following arguments:

- a) Decrement; and
- b) Dual.

If this state was entered with a Set No\_Equalization argument, then this state shall send an Adjust Coefficient 1 (Set No\_Equalization) message to the SP transmitter.

If this state was entered with a Set Reference\_1 argument, then this state shall send an Adjust Coefficient 1 (Set Reference\_1) message to the SP transmitter.

If this state was entered with a Set Reference\_2 argument, then this state shall send an Adjust Coefficient 1 (Set Reference\_2) message to the SP transmitter.

This state shall respond to the Transmitter Adjustment 1 Complete message received from the SP transmitter with a message to the PTT\_T state machine as defined in table 105.

**Table 105 – Mapping messages to the PTT\_T state machine**

Message from SP transmitter	Message sent to PTT_T state machine
Transmitter Adjustment 1 Complete (Maximum)	Current Coefficient 1 Status (Maximum)
Transmitter Adjustment 1 Complete (Minimum)	Current Coefficient 1 Status (Minimum)
Transmitter Adjustment 1 Complete (Update_Complete)	Current Coefficient 1 Status (Update_Complete)

**5.18.6.5.2 Transition PTT\_SC1\_2:Set\_Coefficient to PTT\_SC1\_0:Idle**

This transition shall occur after receiving a:

- a) Cancel message; or
- b) Transmitter Training (Disable) message.

**5.18.6.5.3 Transition PTT\_SC1\_2:Set\_Coefficient to PTT\_SC1\_3:Wait\_Hold**

This transition shall occur after:

- a) receiving a Transmitter Adjustment 1 Complete message; and
- b) sending a Current Coefficient 1 Status message to the PTT\_T state machine.

**5.18.6.6 PTT\_SC1\_3:Wait\_Hold state****5.18.6.6.1 State description**

This state waits for the attached phy's transmitter to request a hold.

**5.18.6.6.2 Transition PTT\_SC1\_3:Wait\_Hold to PTT\_SC1\_0:Idle**

This transition shall occur after receiving a:

- a) Cancel message; or
- b) Transmitter Training (Disable) message.

**5.18.6.6.3 Transition PTT\_SC1\_3:Wait\_Hold to PTT\_SC1\_1:Wait\_Inc\_Dec**

This transition shall occur:

- a) after receiving a Set Coefficient 1 Request (Hold) message.

**5.18.7 PTT\_SC2 (phy layer transmitter training set transmitter coefficient 2) state machine**

This state machine is identical to the PTT\_SC1 state machine (see 5.18.6) except for the messages described in table 106.

**Table 106 – PTT\_SC1 messages to substitute for PTT\_SC2 messages**

<b>PTT_SC1 message</b>	<b>PTT_SC2 message</b>
Set Coefficient 1 Request	Set Coefficient 2 Request
Current Coefficient 1 Status	Current Coefficient 2 Status
Adjust Coefficient 1	Adjust Coefficient 2
Transmitter Adjustment 1 Complete	Transmitter Adjustment 2 Complete
Coefficient 1 Request Usage	Coefficient 2 Request Usage

**5.18.8 PTT\_SC3 (phy layer transmitter training set transmitter coefficient 3) state machine**

This state machine is identical to the PTT\_SC1 state machine (see 5.18.6) except for the messages described in table 107.

**Table 107 – PTT\_SC1 messages to substitute for PTT\_SC3 messages**

<b>PTT_SC1 message</b>	<b>PTT_SC3 message</b>
Set Coefficient 1 Request	Set Coefficient 3 Request
Current Coefficient 1 Status	Current Coefficient 3 Status
Adjust Coefficient 1	Adjust Coefficient 3
Transmitter Adjustment 1 Complete	Transmitter Adjustment 3 Complete
Coefficient 1 Request Usage	Coefficient 3 Request Usage

**5.18.9 PTT\_GC (phy layer transmitter training get transmitter coefficient) state machines****5.18.9.1 PTT\_GC (phy layer transmitter training get transmitter coefficient) state machines overview**

The PTT\_GC state machines' functions are to get coefficient adjustments being requested by the local phy's SP receiver for the attached phy's transmitter and to manage those coefficients' controls.

Figure 119 shows the PTT\_GC1 state machine, PTT\_GC2 state machine, and PTT\_GC3 state machine.

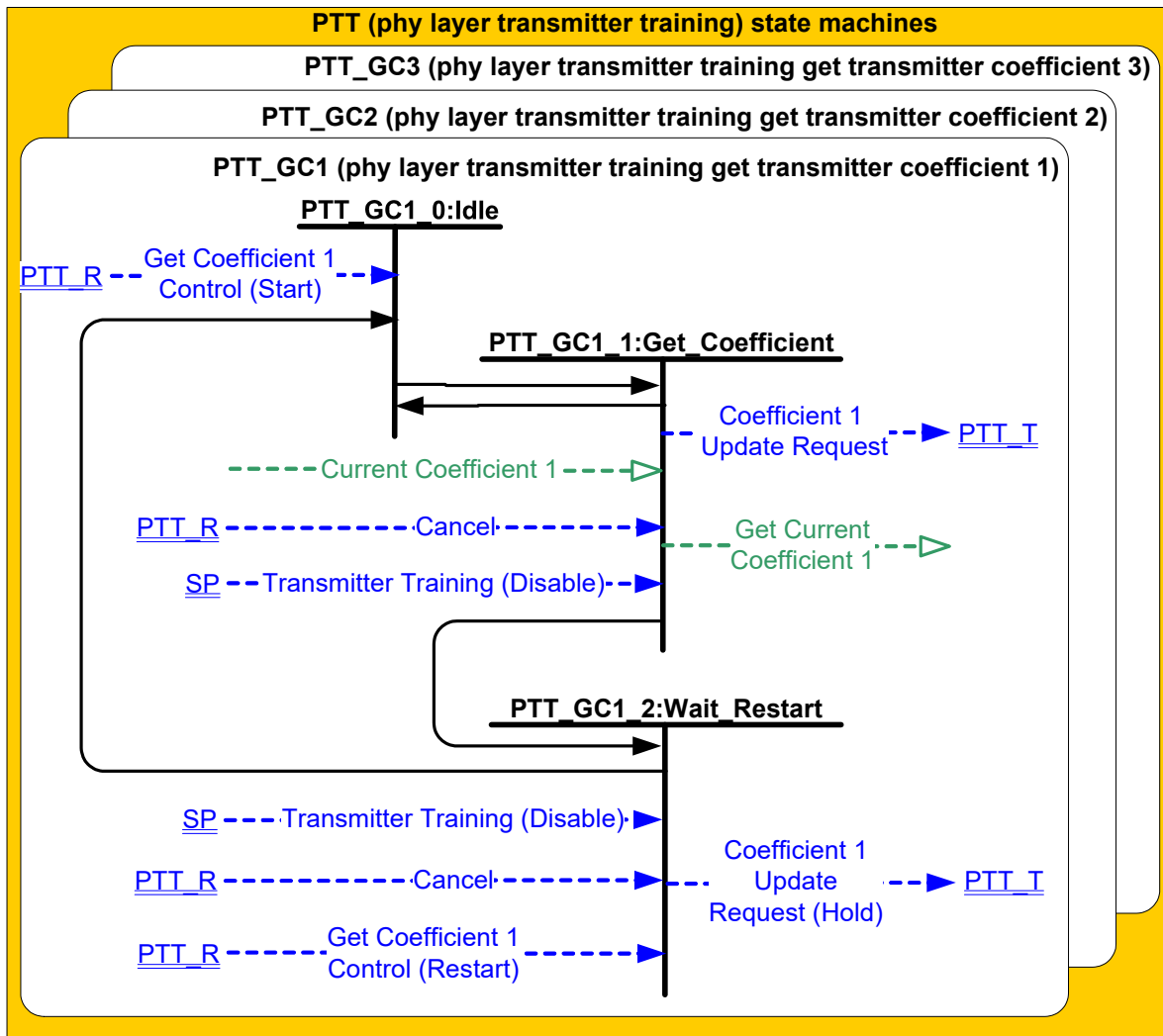


Figure 119 – PTT\_GC1, PTT\_GC2, and PTT\_GC3 (phy layer transmitter training get transmitter coefficient) state machines

#### 5.18.9.2 PTT\_GC1 state machine

The PTT\_GC1 state machine's function is to get the coefficient 1 adjustment being requested by the local phy's SP receiver for the attached phy's transmitter and to manage that coefficient's control. This state machine consists of the following states:

- PTT\_GC1\_0:Idle (see 5.18.9.3) (initial state);
- PTT\_GC1\_1:Get\_Coefficient (see 5.18.9.4); and
- PTT\_GC1\_2:Wait\_Restart (see 5.18.9.5).

This state machine receives the following messages from the SP state machine:

- Transmitter Training (Disable).

**5.18.9.3 PTT\_GC1\_0:Idle state****5.18.9.3.1 State description**

This is the initial state of this state machine.

This state waits for a request to get the coefficient.

**5.18.9.3.2 Transition PTT\_GC1\_0:Idle to PTT\_GC1\_1:Get\_Coefficient**

This transition shall occur:

- a) after receiving a Get Coefficient 1 Control (Start) message.

**5.18.9.4 PTT\_GC1\_1:Get\_Coefficient state****5.18.9.4.1 State description**

This state gets the coefficient that the local phy's receiver is requesting be sent to the attached phy's transmitter.

This state shall repeatedly:

- 1) send a Get Current Coefficient 1 message to the SP receiver;
- 2) wait for a Current Coefficient 1 message received from the SP receiver; and
- 3) respond to the Current Coefficient 1 message received from the SP receiver with a message to the PTT\_T state machine as defined in table 108.

**Table 108 – Mapping messages to the PTT\_T state machine**

Message from the SP receiver	Message sent to the PTT_T state machine
Current Coefficient 1 (Increment)	Coefficient 1 Update Request (Increment)
Current Coefficient 1 (Decrement)	Coefficient 1 Update Request (Decrement)
Current Coefficient 1 (Hold)	Coefficient 1 Update Request (Hold)
Current Coefficient 1 (No_Equalization)	Coefficient 1 Update Request (No_Equalization)
Current Coefficient 1 (Reference_1)	Coefficient 1 Update Request (Reference_1)
Current Coefficient 1 (Reference_2)	Coefficient 1 Update Request (Reference_2)

**5.18.9.4.2 Transition PTT\_GC1\_1:Get\_Coefficient to PTT\_GC1\_0:Idle**

This transition shall occur after receiving a:

- a) Transmitter Training (Disable) message; or
- b) Cancel message.

**5.18.9.4.3 Transition PTT\_GC1\_1:Get\_Coefficient to PTT\_GC1\_2:Wait\_Restart**

This transition shall occur after sending a:

- a) Coefficient 1 Update Request message with an argument of Increment, Decrement, No\_Equalization, Reference\_1, or Reference\_2.

**5.18.9.5 PTT\_GC1\_2:Wait\_Restart state****5.18.9.5.1 State description**

This state waits for the attached phy's transmitter to indicate that the requested coefficient update is complete.

If this state receives a Get Coefficient 1 Control (Restart) message, then this state shall send a Coefficient 1 Update Request (Hold) message to the PTT\_T state machine.

**5.18.9.5.2 Transition PTT\_GC1\_2:Wait\_Restart to PTT\_GC1\_0:Idle**

This transition shall occur after:

- a) receiving a Transmitter Training (Disable) message;
- b) receiving a Cancel message; or
- c) sending a Coefficient 1 Update Request (Hold) message.

**5.18.10 PTT\_GC2 (phy layer transmitter training get transmitter coefficient 2) state machine**

This state machine is identical to the PTT\_GC1 state machine (see 5.18.9) except for the messages described in table 109.

**Table 109 – PTT\_GC1 messages to substitute for PTT\_GC2 messages**

<b>PTT_GC1 message</b>	<b>PTT_GC2 message</b>
Get Coefficient 1 Control	Get Coefficient 2 Control
Coefficient 1 Update Request	Coefficient 2 Update Request
Current Coefficient 1	Current Coefficient 2
Get Current Coefficient 1	Get Current Coefficient 2

**5.18.11 PTT\_GC3 (phy layer transmitter training get transmitter coefficient 3) state machine**

This state machine is identical to the PTT\_GC1 state machine (see 5.18.9) except for the messages described in table 110.

**Table 110 – PTT\_GC1 messages to substitute for PTT\_GC2 messages**

<b>PTT_GC1 message</b>	<b>PTT_GC3 message</b>
Get Coefficient 1 Control	Get Coefficient 3 Control
Coefficient 1 Update Request	Coefficient 3 Update Request
Current Coefficient 1	Current Coefficient 3
Get Current Coefficient 1	Get Current Coefficient 3



## 5.18.12 PTT\_PL (phy layer transmitter training pattern lock) state machine

### 5.18.12.1 PTT\_PL state machine overview

The PTT\_PL state machine establishes the same transmitter training pattern lock at the local phy's receiver as was sent from the attached phy's transmitter by searching for the pattern marker (see 5.11.4.2.3.4.3). The SP receiver monitors and decodes the incoming data stream and forces the pattern marker to the first position of the Train\_Tx pattern (see 5.11.4.2.3.4.4) to perform pattern lock, when requested by the PTT\_PL state machine.

After pattern lock synchronization has been achieved, this state machine evaluates the pattern marker at the beginning of each Train\_Tx pattern that is received. If five consecutive invalid pattern markers (see 5.11.4.2.3.4.3) are detected, then pattern lock is considered lost. If pattern lock is lost, then receipt of two consecutive valid pattern markers (see 5.11.4.2.3.4.3) is required to reestablish pattern lock.

While pattern lock is lost, any Train\_Tx pattern received is invalid.

The state machine shall transition to the PTT\_PL0:Idle state from any other state after receiving a Transmitter Training (Disable) message from the SP state machine (see 5.14.4.13).

This state machine consists of the following states:

- a) PTT\_PL0:Idle (see 5.18.12.2) (initial state);
- b) PTT\_PL1:Acquire\_Lock (see 5.18.12.3);
- c) PTT\_PL2:Valid (see 5.18.12.4);
- d) PTT\_PL3:Lock\_Acquired (see 5.18.12.5);
- e) PTT\_PL4:Lost1 (see 5.18.12.6);
- f) PTT\_PL5:Lost2 (see 5.18.12.7);
- g) PTT\_PL6:Lost3 (see 5.18.12.8); and
- h) PTT\_PL7:Lost4 (see 5.18.12.9).

This state machine receives the following message from the SP state machine:

- a) Transmitter Training (Disable).

Figure 120 shows the PTT\_PL state machine.

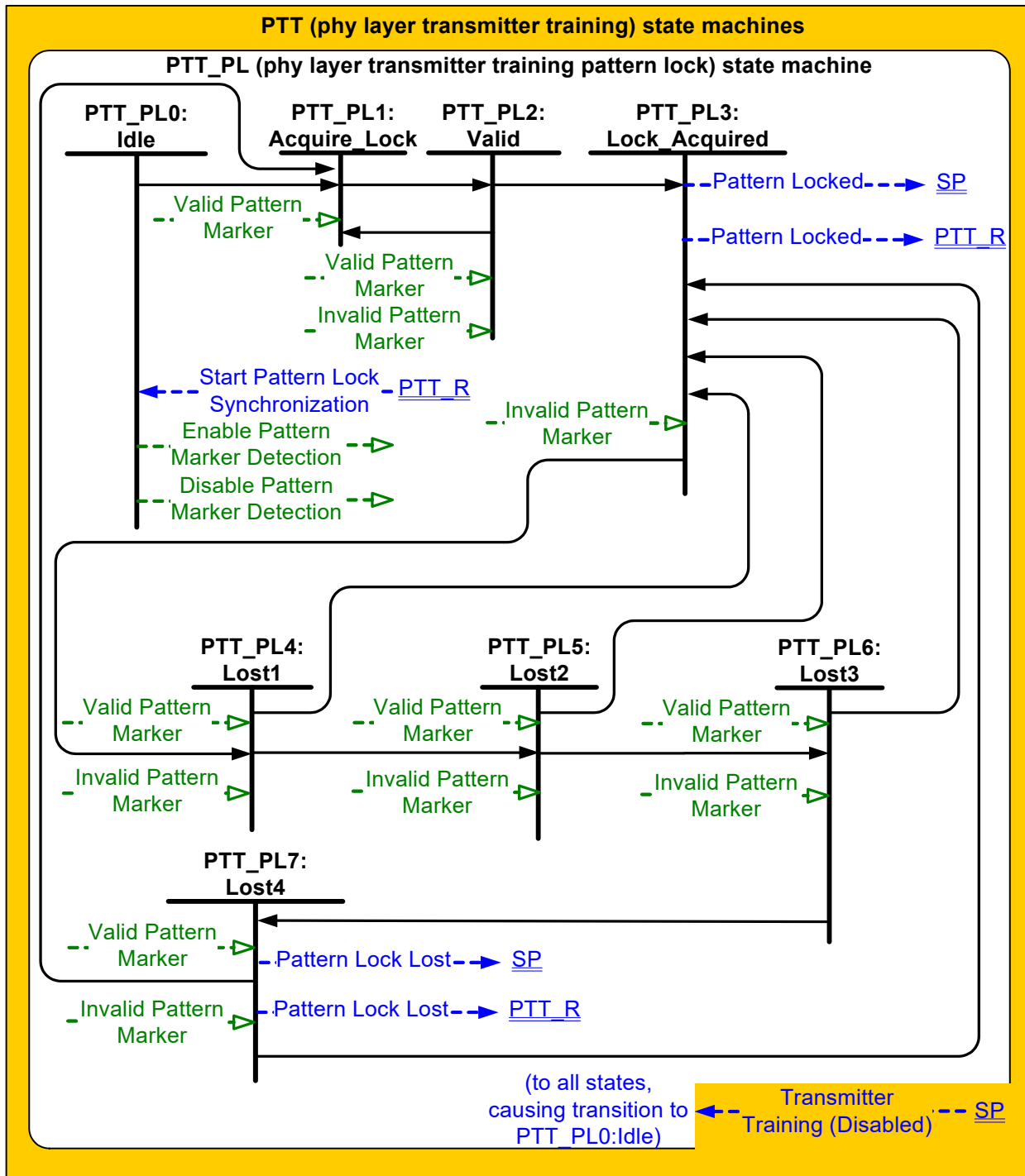


Figure 120 – PTT\_PL (phy layer transmitter training pattern lock) state machine

#### 5.18.12.2 PTT\_PL0:Idle state

##### 5.18.12.2.1 State description

This is the initial state of this state machine.

This state waits for a request to establish pattern lock.

Upon entry into this state, this state shall send a Disable Pattern Marker Detection message to the SP receiver.

If this state receives a Start Pattern Lock Synchronization message, then this state shall send an Enable Pattern Marker Detection message to the SP receiver.

#### **5.18.12.2.2 Transition PTT\_PL0:Idle to PTT\_PL1:Acquire\_Lock**

This transition shall occur:

- a) after sending an Enable Pattern Marker Detection message.

#### **5.18.12.3 PTT\_PL1:Acquire\_Lock state**

##### **5.18.12.3.1 State description**

This state waits for the SP receiver to receive a valid pattern marker.

##### **5.18.12.3.2 Transition PTT\_PL1:Acquire\_Lock to PTT\_PL2:Valid**

This transition shall occur:

- a) after receiving a Valid Pattern Marker message.

#### **5.18.12.4 PTT\_PL2:Valid state**

##### **5.18.12.4.1 State description**

This state waits for the SP receiver to receive the second pattern marker.

##### **5.18.12.4.2 Transition PTT\_PL2:Valid to PTT\_PL1:Acquire\_Lock**

This transition shall occur:

- a) after receiving an Invalid Pattern Marker message.

##### **5.18.12.4.3 Transition PTT\_PL2:Valid to PTT\_PL3:Lock\_Acquired**

This transition shall occur:

- a) after receiving a Valid Pattern Marker message.

#### **5.18.12.5 PTT\_PL3:Lock\_Acquired state**

##### **5.18.12.5.1 State description**

This state is reached after two valid pattern markers have been received without the receipt of an intervening invalid pattern marker.

If this state is entered from the PTT\_PL2:Valid state, then upon entry into this state, this state shall send a Pattern Locked message to the PTT\_R state machine (see 5.14) and the PTT\_R state machine (see 5.18.5).

This state waits for the SP receiver to receive an invalid pattern marker.

##### **5.18.12.5.2 Transition PTT\_PL3:Lock\_Acquired to PTT\_PL4:Lost1**

This transition shall occur:

- a) after receiving an Invalid Pattern Marker message.

**5.18.12.6 PTT\_PL4:Lost1 state****5.18.12.6.1 State description**

This state is reached after one invalid pattern marker has been received.

**5.18.12.6.2 Transition PTT\_PL4:Lost1 to PTT\_PL3:Lock\_Acquired**

This transition shall occur:

- a) after receiving a Valid Pattern Marker message.

**5.18.12.6.3 Transition PTT\_PL4:Lost1 to PTT\_PL5:Lost2**

This transition shall occur:

- a) after receiving an Invalid Pattern Marker message.

**5.18.12.7 PTT\_PL5:Lost2 state****5.18.12.7.1 State description**

This state is reached after two invalid pattern markers have been received.

**5.18.12.7.2 Transition PTT\_PL5:Lost2 to PTT\_PL3:Lock\_Acquired**

This transition shall occur:

- a) after receiving a Valid Pattern Marker message.

**5.18.12.7.3 Transition PTT\_PL5:Lost2 to PTT\_PL6:Lost3**

This transition shall occur:

- a) after receiving an Invalid Pattern Marker message.

**5.18.12.8 PTT\_PL6:Lost3 state****5.18.12.8.1 State description**

This state is reached after three invalid pattern markers have been received.

**5.18.12.8.2 Transition PTT\_PL6:Lost3 to PTT\_PL3:Lock\_Acquired**

This transition shall occur:

- a) after receiving a Valid Pattern Marker message.

**5.18.12.8.3 Transition PTT\_PL6:Lost3 to PTT\_PL7:Lost4**

This transition shall occur:

- a) after receiving an Invalid Pattern Marker message.

**5.18.12.9 PTT\_PL7:Lost4 state****5.18.12.9.1 State description**

This state is reached after four invalid pattern markers have been received.

If this state receives an Invalid Pattern Marker message, then this state shall send a Pattern Lock Lost message to the:

- a) PTT\_R state machine (see 5.18.5); and
- b) SP state machine (see 5.14).

#### **5.18.12.9.2 Transition PTT\_PL7:Lost4 to PTT\_PL3:Lock\_Acquired**

This transition shall occur:

- a) after receiving a Valid Pattern Marker message.

#### **5.18.12.9.3 Transition PTT\_PL7:Lost4 to PTT\_PL1:Acquire\_Lock**

This transition shall occur:

- a) after sending a Pattern Lock Lost message.

### **5.19 PAPTA (phy layer active phy transmitter adjustment) state machines**

#### **5.19.1 PAPTA state machines overview**

The phy layer contains PAPTA state machines that control the physical link during active SP transmitter adjustment. The PAPTA state machines are as follows:

- a) PAPTA\_A\_L (phy layer attached SP receiver adjusts the local SP transmitter coefficients) state machine (see 5.19.4);
- b) PAPTA\_L\_A (phy layer local SP receiver adjusts the attached SP transmitter coefficients) state machine (see 5.19.5); and
- c) PAPTA\_TC (phy layer SP receiver management of attached SP transmitter coefficient adjustments) state machine (see 5.19.6).

After an Enable APTA message is received the phy may:

- a) start adjustment of the attached phy; or
- b) be adjusted by the attached phy.

After a Disable APTA message or a Disable APTA request is received the phy shall stop:

- a) adjustment; and
- b) responding to adjustment requests.

After a Received APTA\_ADJUST message with an argument of Terminate is received the phy shall stop adjustment.

If the state machine consists of multiple states, then the initial state is as indicated in the state machine description in this subclause.

Any message, request, or confirmation received by a state that is not referred to in the description of that state shall be ignored.

#### **5.19.2 SP transmitter additions for APTA**

##### **5.19.2.1 SP transmitter additions for APTA overview**

The SP transmitter sends binary primitives in response to messages from the PAPTA state machines (see 5.14.2) that:

- a) specify changes to the attached SP transmitter coefficients (see 5.19.4); and
- b) indicate APTA status (see 5.19.6).

The SP transmitter relationship to other SP transmitters is defined in 4.3.2.

**5.19.2.2 SP transmitter sends APTA binary primitives messages when messages are received**

The SP transmitter sends APTA binary primitives when messages are received from the PAPTA state machines as indicated in table 111.

**Table 111 – SP transmitter binary primitive messages**

<b>Message received by the SP transmitter</b>	<b>Binary primitive sent by the SP transmitter</b>
Transmit APTA_ADJUST (Ready)	APTA_ADJUST (READY)
Transmit APTA_ADJUST (Start)	APTA_ADJUST (START)
Transmit APTA_ADJUST (Complete)	APTA_ADJUST (COMPLETE)
Transmit APTA_ADJUST (Terminate)	APTA_ADJUST (TERMINATE)
Transmit APTA_COEFFICIENT_1 (Updated)	APTA_COEFFICIENT_1 (UPDATED)
Transmit APTA_COEFFICIENT_1 (Maximum)	APTA_COEFFICIENT_1 (MAXIMUM)
Transmit APTA_COEFFICIENT_1 (Minimum)	APTA_COEFFICIENT_1 (MINIMUM)
Transmit APTA_COEFFICIENT_1 (Increment)	APTA_COEFFICIENT_1 (INCREMENT)
Transmit APTA_COEFFICIENT_1 (Decrement)	APTA_COEFFICIENT_1 (DECREMENT)
Transmit APTA_COEFFICIENT_2 (Updated)	APTA_COEFFICIENT_2 (UPDATED)
Transmit APTA_COEFFICIENT_2 (Maximum)	APTA_COEFFICIENT_2 (MAXIMUM)
Transmit APTA_COEFFICIENT_2 (Minimum)	APTA_COEFFICIENT_2 (MINIMUM)
Transmit APTA_COEFFICIENT_2 (Increment)	APTA_COEFFICIENT_2 (INCREMENT)
Transmit APTA_COEFFICIENT_2 (Decrement)	APTA_COEFFICIENT_2 (DECREMENT)
Transmit APTA_COEFFICIENT_3 (Updated)	APTA_COEFFICIENT_3 (UPDATED)
Transmit APTA_COEFFICIENT_3 (Maximum)	APTA_COEFFICIENT_3 (MAXIMUM)
Transmit APTA_COEFFICIENT_3 (Minimum)	APTA_COEFFICIENT_3 (MINIMUM)
Transmit APTA_COEFFICIENT_3 (Increment)	APTA_COEFFICIENT_3 (INCREMENT)
Transmit APTA_COEFFICIENT_3 (Decrement)	APTA_COEFFICIENT_3 (DECREMENT)
Transmit APTA_COEFFICIENT_1_2 (Updated)	APTA_COEFFICIENT_1_2 (UPDATED)
Transmit APTA_COEFFICIENT_1_2 (Maximum)	APTA_COEFFICIENT_1_2 (MAXIMUM)
Transmit APTA_COEFFICIENT_1_2 (Minimum)	APTA_COEFFICIENT_1_2 (MINIMUM)
Transmit APTA_COEFFICIENT_1_2 (Increment)	APTA_COEFFICIENT_1_2 (INCREMENT)
Transmit APTA_COEFFICIENT_1_2 (Decrement)	APTA_COEFFICIENT_1_2 (DECREMENT)
Transmit APTA_COEFFICIENT_2_3 (Updated)	APTA_COEFFICIENT_2_3 (UPDATED)
Transmit APTA_COEFFICIENT_2_3 (Maximum)	APTA_COEFFICIENT_2_3 (MAXIMUM)
Transmit APTA_COEFFICIENT_2_3 (Minimum)	APTA_COEFFICIENT_2_3 (MINIMUM)
Transmit APTA_COEFFICIENT_2_3 (Increment)	APTA_COEFFICIENT_2_3 (INCREMENT)
Transmit APTA_COEFFICIENT_2_3 (Decrement)	APTA_COEFFICIENT_2_3 (DECREMENT)

**5.19.2.3 APTA Coefficient limits****5.19.2.3.1 APTA Coefficient limits overview**

The individual coefficient limits and APTA status reporting definitions associated with relationships between coefficients specified in SAS-4 shall be maintained by the SP transmitter while processing coefficient adjustment requests.

**5.19.2.3.2 APTA Coefficient request result of updated****5.19.2.3.2.1 APTA Coefficient request processing**

In response to:

- a) a Received APTA\_COEFFICIENT\_1 with an argument of Increment or Decrement;
- b) a Received APTA\_COEFFICIENT\_2 with an argument of Increment or Decrement; or
- c) a Received APTA\_COEFFICIENT\_3 with an argument of Increment or Decrement,

the SP transmitter shall adjust the specified coefficient if that adjustment results in the specified coefficient being:

- a) less than a maximum value; and
- b) greater than a minimum value (see SAS-4).

In response to:

- a) a Received APTA\_COEFFICIENTS\_1\_2 with an argument of Increment or Decrement; or
- b) a Received APTA\_COEFFICIENTS\_2\_3 with an argument of Increment or Decrement,

the SP transmitter shall for each specified coefficient, adjust that coefficient if that adjustment results in:

- a) that coefficient being less than a maximum value and greater than a minimum value; and
- b) the other coefficient being less than a maximum value and greater than a minimum value.

**5.19.2.3.2.2 APTA Coefficient adjustment completes**

If the SP transmitter adjusts a specified coefficient to a value less than a maximum value and greater than a minimum value, then after the adjustment is complete the SP transmitter shall send to the APTA state machine state that requested the SP transmitter adjustment the message associated with the adjusted coefficient or coefficients (i.e., Transmit APTA\_COEFFICIENT\_1 (Updated) message, Transmit APTA\_COEFFICIENT\_2 (Updated) message, Transmit APTA\_COEFFICIENT\_3 (Updated) message, Transmit APTA\_COEFFICIENTS\_1\_2 (Updated) message, or Transmit APTA\_COEFFICIENTS\_2\_3 (Updated) message).

**5.19.2.3.3 APTA Coefficient request result of maximum****5.19.2.3.3.1 APTA Coefficient request processing**

In response to:

- a) a Received APTA\_COEFFICIENT\_1 (Increment);
- b) a Received APTA\_COEFFICIENT\_2 (Increment); or
- c) a Received APTA\_COEFFICIENT\_3 (Increment),

the SP transmitter shall adjust the specified coefficient if that adjustment results in the specified coefficient being equal to a maximum value (see SAS-4).

In response to:

- a) a Received APTA\_COEFFICIENTS\_1\_2 (Increment); or
- b) a Received APTA\_COEFFICIENTS\_2\_3 (Increment),

the SP transmitter shall, for each specified coefficient, adjust that coefficient if that adjustment results in:

- a) that coefficient being less than or equal to a maximum value; and
- b) no coefficient being greater than a maximum value.

#### **5.19.2.3.3.2 APTA Coefficient adjustment completes**

If the SP transmitter adjusts a specified coefficient to a maximum value, then after the adjustment is complete the SP transmitter shall send to the APTA state machine state that requested the SP transmitter adjustment the message associated with the adjusted coefficient or coefficients (i.e., Transmit APTA\_COEFFICIENT\_1 (Maximum) message, Transmit APTA\_COEFFICIENT\_2 (Maximum) message, Transmit APTA\_COEFFICIENT\_3 (Maximum) message, Transmit APTA\_COEFFICIENTS\_1\_2 (Maximum) message, or Transmit APTA\_COEFFICIENTS\_2\_3 (Maximum) message).

#### **5.19.2.3.3.3 APTA No coefficient adjustment**

If the processing of the requested SP transmitter adjustment results in an adjustment that is greater than a coefficient's maximum value, then:

- a) no adjustment shall be made to any coefficient; and
- b) the SP transmitter shall send to the APTA state machine state that requested the SP transmitter adjustment the message associated with the specified coefficient or coefficients (i.e., Transmit APTA\_COEFFICIENT\_1 (Maximum) message, Transmit APTA\_COEFFICIENT\_2 (Maximum) message, Transmit APTA\_COEFFICIENT\_3 (Maximum) message, Transmit APTA\_COEFFICIENTS\_1\_2 (Maximum) message, or Transmit APTA\_COEFFICIENTS\_2\_3 (Maximum) message).

#### **5.19.2.3.4 APTA Coefficient request result of minimum**

##### **5.19.2.3.4.1 APTA Coefficient request processing**

In response to:

- a) a Received APTA\_COEFFICIENT\_1 (Decrement);
- b) a Received APTA\_COEFFICIENT\_2 (Decrement); or
- c) a Received APTA\_COEFFICIENT\_3 (Decrement),

the SP transmitter shall adjust the specified coefficient if that adjustment results in the specified coefficient being equal to a minimum value (see SAS-4).

In response to:

- a) a Received APTA\_COEFFICIENTS\_1\_2 (Decrement); or
- b) a Received APTA\_COEFFICIENTS\_2\_3 (Decrement),

the SP transmitter shall for each specified coefficient, adjust that coefficient if that adjustment results in:

- a) that coefficient being greater than or equal to a minimum value; and
- b) no coefficient being less than a minimum value.

#### **5.19.2.3.4.2 APTA Coefficient adjustment completes**

If the SP transmitter adjusts a specified coefficient to a minimum value, then after the adjustment is complete the SP transmitter shall send to the APTA state machine state that requested the SP transmitter adjustment the message associated with the adjusted coefficient or coefficients (i.e., Transmit APTA\_COEFFICIENT\_1 (Minimum) message, Transmit APTA\_COEFFICIENT\_2 (Minimum) message, Transmit APTA\_COEFFICIENT\_3 (Minimum) message, Transmit APTA\_COEFFICIENTS\_1\_2 (Minimum) message, or Transmit APTA\_COEFFICIENTS\_2\_3 (Minimum) message).



#### 5.19.2.3.4.3 APTA No coefficient adjustment

If the processing of the requested SP transmitter adjustment results in an adjustment that is less than a coefficient's minimum value, then:

- a) no adjustment shall be made to any coefficient; and
- b) the SP transmitter shall send to the APTA state machine state that requested the SP transmitter adjustment the message associated with the specified coefficient or coefficients that is equal to or less than a minimum value (i.e., Transmit APTA\_COEFFICIENT\_1 (Minimum) message, Transmit APTA\_COEFFICIENT\_2 (Minimum) message, Transmit APTA\_COEFFICIENT\_3 (Minimum) message, Transmit APTA\_COEFFICIENTS\_1\_2 (Minimum) message, or Transmit APTA\_COEFFICIENTS\_2\_3 (Minimum) message).

### 5.19.3 SP receiver additions for APTA

#### 5.19.3.1 SP receiver additions for APTA overview

The SP receiver:

- a) manages APTA using an algorithm that is beyond the scope of this standard;
- b) sends messages to the PAPT state;
- c) in response to binary primitives received by the SP receiver (see 5.14.2) that specify changes to the local SP transmitter coefficients (see 5.19.4); and
- d) requests coefficient changes by sending messages to the PAPT\_L\_A state machine (see 5.19.5).

The SP receiver relationship to other SP receivers is defined in 4.3.3.

#### 5.19.3.2 SP receiver messages when APTA binary primitives are received

The SP receiver sends messages to the PAPT state machines as indicated in table 112 when binary primitives are received.

**Table 112 – SP receiver binary primitive messages** (part 1 of 2)

Binary primitive received by the SP receiver	Message sent to the PAPT state machines
APTA_ADJUST (READY)	Received APTA_ADJUST (Ready)
APTA_ADJUST (START)	Received APTA_ADJUST (Start)
APTA_ADJUST (COMPLETE)	Received APTA_ADJUST (Complete)
APTA_ADJUST (TERMINATE)	Received APTA_ADJUST (Terminate)
APTA_ADJUST (RESERVED 1)	None
APTA_ADJUST (RESERVED 2)	None
APTA_ADJUST (RESERVED 3)	None
APTA_COEFFICIENT_1 (UPDATED)	Received APTA_COEFFICIENT_1 (Updated)
APTA_COEFFICIENT_1 (MAXIMUM)	Received APTA_COEFFICIENT_1 (Maximum)
APTA_COEFFICIENT_1 (MINIMUM)	Received APTA_COEFFICIENT_1 (Minimum)
APTA_COEFFICIENT_1 (INCREMENT)	Received APTA_COEFFICIENT_1 (Increment)
APTA_COEFFICIENT_1 (DECREMENT)	Received APTA_COEFFICIENT_1 (Decrement)
APTA_COEFFICIENT_1 (RESERVED 1)	None
APTA_COEFFICIENT_2 (UPDATED)	Received APTA_COEFFICIENT_2 (Updated)

Table 112 – SP receiver binary primitive messages (part 2 of 2)

Binary primitive received by the SP receiver	Message sent to the PAPT state machines
APTA_COEFFICIENT_2 (MAXIMUM)	Received APTA_COEFFICIENT_2 (Maximum)
APTA_COEFFICIENT_2 (MINIMUM)	Received APTA_COEFFICIENT_2 (Minimum)
APTA_COEFFICIENT_2 (INCREMENT)	Received APTA_COEFFICIENT_2 (Increment)
APTA_COEFFICIENT_2 (DECREMENT)	Received APTA_COEFFICIENT_2 (Decrement)
APTA_COEFFICIENT_2 (RESERVED 1)	None
APTA_COEFFICIENT_3 (UPDATED)	Received APTA_COEFFICIENT_3 (Updated)
APTA_COEFFICIENT_3 (MAXIMUM)	Received APTA_COEFFICIENT_3 (Maximum)
APTA_COEFFICIENT_3 (MINIMUM)	Received APTA_COEFFICIENT_3 (Minimum)
APTA_COEFFICIENT_3 (INCREMENT)	Received APTA_COEFFICIENT_3 (Increment)
APTA_COEFFICIENT_3 (DECREMENT)	Received APTA_COEFFICIENT_3 (Decrement)
APTA_COEFFICIENT_3 (RESERVED 1)	None
APTA_COEFFICIENT_1_2 (UPDATED)	Received APTA_COEFFICIENT_1_2 (Updated)
APTA_COEFFICIENT_1_2 (MAXIMUM)	Received APTA_COEFFICIENT_1_2 (Maximum)
APTA_COEFFICIENT_1_2 (MINIMUM)	Received APTA_COEFFICIENT_1_2 (Minimum)
APTA_COEFFICIENT_1_2 (INCREMENT)	Received APTA_COEFFICIENT_1_2 (Increment)
APTA_COEFFICIENT_1_2 (DECREMENT)	Received APTA_COEFFICIENT_1_2 (Decrement)
APTA_COEFFICIENT_1_2 (RESERVED 1)	None
APTA_COEFFICIENT_2_3 (UPDATED)	Received APTA_COEFFICIENT_2_3 (Updated)
APTA_COEFFICIENT_2_3 (MAXIMUM)	Received APTA_COEFFICIENT_2_3 (Maximum)
APTA_COEFFICIENT_2_3 (MINIMUM)	Received APTA_COEFFICIENT_2_3 (Minimum)
APTA_COEFFICIENT_2_3 (INCREMENT)	Received APTA_COEFFICIENT_2_3 (Increment)
APTA_COEFFICIENT_2_3 (DECREMENT)	Received APTA_COEFFICIENT_2_3 (Decrement)
APTA_COEFFICIENT_2_3 (RESERVED 1)	None

### 5.19.3.3 SP receiver messages that tune the attached SP transmitter

The SP receiver receives the following messages from the PAPT<sub>L\_A</sub> state machine (see 5.19.5):

- a) Get Next Coefficient.

In response to a Get Next Coefficient message, the SP receiver sends one of the following messages to the PAPT<sub>L\_A</sub> state machine (see 5.19.5):

- a) Next Coefficient with an argument of:
  - A) Increment Coefficient 1;
  - B) Decrement Coefficient 1;
  - C) Increment Coefficient 2;
  - D) Decrement Coefficient 2;
  - E) Increment Coefficient 3;
  - F) Decrement Coefficient 3;
  - G) Increment Coefficients 1\_2;
  - H) Decrement Coefficients 1\_2;
  - I) Increment Coefficients 2\_3;
  - J) Decrement Coefficients 2\_3; or

K) Adjustment Complete.

#### **5.19.4 PAPTA\_A\_L (phy layer attached SP receiver adjusts the local SP transmitter coefficients) state machine**

##### **5.19.4.1 PAPTA\_A\_L state machine overview**

The PAPTA\_A\_L state machine adjust the local SP transmitter.

This state machine:

- a) waits to receive a request to start adjustment from the attached phy;
- b) receives change request APTA binary primitives from the attached phy; and
- c) requests the SP transmitter to:
  - A) adjust its coefficient values;
  - B) send APTA binary primitives that report the status of the SP transmitter; and
  - C) terminate APTA.

This state machine consists of the following states:

- a) PAPTA\_A\_L0:Idle (see 5.19.4.2) (initial state);
- b) PAPTA\_A\_L1:Wait For Start (see 5.19.4.3); and
- c) PAPTA\_A\_L2:Adjust Local Transmitter (see 5.19.4.4).

This state machine receives the following request from the management application layer:

- a) Terminate APTA.

This state machine receives the following request from the link layer:

- a) Disable APTA.

This state machine sends the following confirmations to the management application layer:

- a) Attached Phy Terminated APTA.

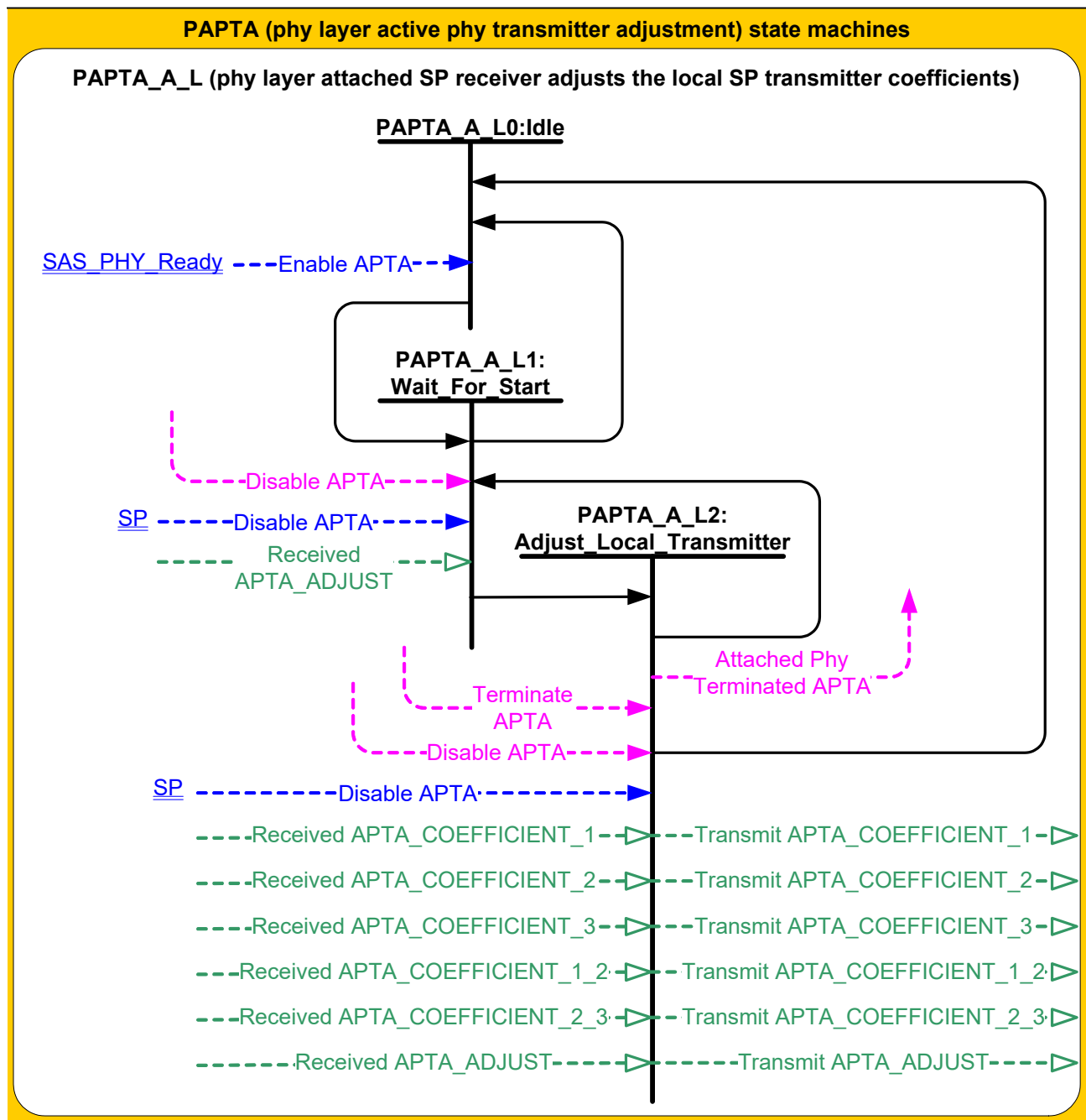
This state machine receives the following messages from the SP receiver:

- a) Received APTA\_ADJUST with an argument of Start, Complete, or Terminate;
- b) Received APTA\_COEFFICIENT\_1 with an argument of Increment or Decrement;
- c) Received APTA\_COEFFICIENT\_2 with an argument of Increment or Decrement;
- d) Received APTA\_COEFFICIENT\_3 with an argument of Increment or Decrement;
- e) Received APTA\_COEFFICIENTS\_1\_2 with an argument of Increment or Decrement; and
- f) Received APTA\_COEFFICIENTS\_2\_3 with an argument of Increment or Decrement.

This state machine sends the following messages to the SP state machines:

- a) Transmit APTA\_ADJUST with an argument of Ready or Terminate;
- b) Transmit APTA\_COEFFICIENT\_1 with an argument of Updated, Maximum, or Minimum;
- c) Transmit APTA\_COEFFICIENT\_2 with an argument of Updated, Maximum, or Minimum;
- d) Transmit APTA\_COEFFICIENT\_3 with an argument of Updated, Maximum, or Minimum;
- e) Transmit APTA\_COEFFICIENTS\_1\_2 with an argument of Updated, Maximum, or Minimum; and
- f) Transmit APTA\_COEFFICIENTS\_2\_3 with an argument of Updated, Maximum, or Minimum.

Figure 121 shows the PAPT<sub>A</sub>\_L state machine.



**Figure 121 – PAPTA\_A\_L (phy layer attached SP receiver adjusts the local SP transmitter coefficients) state machine**

#### 5.19.4.2 PAPTA\_A\_L0:Idle state

#### 5.19.4.2.1 State description

This is the initial state of this state machine.

This state waits for an Enable APTA message.

**5.19.4.2.2 Transition PAPTA\_A\_L0:Idle to PAPTA\_A\_L1:Wait\_for\_Start**

This transition shall occur:

- a) after receiving an Enable APTA message.

**5.19.4.3 PAPTA\_A\_L1:Wait\_For\_Start state****5.19.4.3.1 State description**

This state waits for a Received APTA\_ADJUST message with an argument of Start.

**5.19.4.3.2 Transition PAPTA\_A\_L1:Wait\_For\_Start to PAPTA\_A\_L0:Idle**

This transition shall occur after receiving:

- a) a Disable APTA message; or
- b) a Disable APTA request from the link layer.

**5.19.4.3.3 Transition PAPTA\_A\_L1:Wait\_For\_Start to PAPTA\_A\_L2:Adjust\_Local\_Transmitter**

This transition shall occur:

- a) after receiving a Received APTA\_ADJUST message with an argument of Start.

**5.19.4.4 PAPTA\_A\_L2:Adjust\_Local\_Transmitter state****5.19.4.4.1 State description**

Upon entry into this state, this state shall send:

- a) a Transmit APTA\_ADJUST message with an argument of Ready to the SP transmitter.

If this state receives a Received APTA\_ADJUST message with an argument of Terminate, then this state shall send:

- a) an Attached Phy Terminated APTA confirmation to the management application layer.

If this state receives:

- a) a Terminate APTA request from the management application layer;
- b) a Disable APTA request from the link layer; or
- c) a Disable APTA message,

then this state shall send:

- a) a Transmit APTA\_ADJUST message with an argument of Terminate to the SP transmitter.

**5.19.4.4.2 Transition PAPTA\_A\_L2:Adjust\_Local\_Transmitter to PAPTA\_A\_L0:Idle**

This transition shall occur after sending:

- a) an Attached Phy Terminated APTA confirmation to the management application layer; or
- b) a Transmit APTA\_ADJUST message with an argument of Terminate to the SP transmitter.

**5.19.4.4.3 Transition PAPTA\_A\_L2:Adjust\_Local\_Transmitter to PAPTA\_A\_L1:Wait\_For\_Start**

This transition shall occur:

- a) after receiving a Received APTA\_ADJUST message with an argument of Complete.

### **5.19.5 PAPTA\_L\_A (phy layer local SP receiver adjusts the attached SP transmitter coefficients) state machine**

#### **5.19.5.1 PAPTA\_L\_A state machine overview**

The PAPTA\_L\_A state machine adjust the attached SP transmitter coefficients.

The PAPTA\_L\_A state machine:

- a) starts APTA;
- b) sends messages to adjust the coefficients of the attached SP transmitter;
- c) receives coefficient adjustment status messages from the attached SP transmitter; and
- d) completes APTA or terminates APTA.

This state machine consists of the following states:

- a) PAPTA\_L\_A0:Initialize (see 5.19.5.2) (initial state);
- b) PAPTA\_L\_A1:Start (see 5.19.5.3); and
- c) PAPTA\_L\_A2:Adjust\_Attached\_Transmitter (see 5.19.5.4).

This state machine receives the following requests from the management application layer:

- a) Adjust Attached Transmitter; and
- b) Terminate APTA.

This state machine receives the following request from the link layer:

- a) Disable APTA.

This state machine sends the following confirmations to the management application layer:

- a) Attached Phy Terminated APTA; and
- b) Adjustment Complete.

This state machine receives the following message from the SP state machine:

- a) Disable APTA.

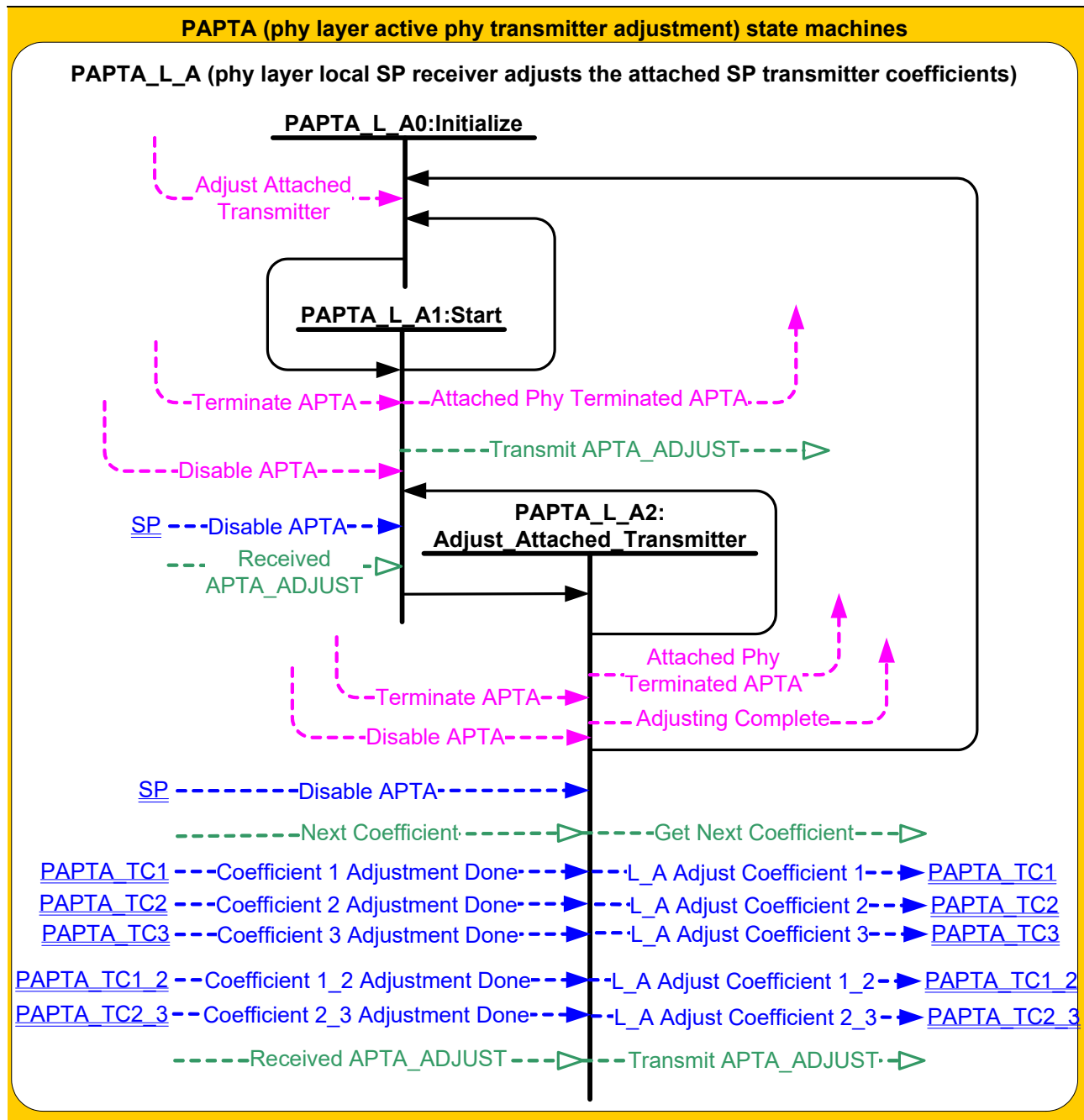
This state machine sends the following messages to the SP receiver:

- a) Get Next Coefficient.

This state machine receives the following messages from the SP receiver:

- a) Next Coefficient;
- b) Received APTA\_ADJUST with an argument of Ready or Terminate;
- c) Received APTA\_COEFFICIENT\_1;
- d) Received APTA\_COEFFICIENT\_2;
- e) Received APTA\_COEFFICIENT\_3;
- f) Received APTA\_COEFFICIENTS\_1\_2; and
- g) Received APTA\_COEFFICIENTS\_2\_3.

Figure 122 shows the PAPTA\_L\_A state machine.



**Figure 122 – PAPTA\_L\_A (phy layer local SP receiver adjusts the attached SP transmitter coefficients) state machine**

#### 5.19.5.2 PAPTA\_L\_A0:Initialize state

##### 5.19.5.2.1 State description

This is the initial state of this state machine.

This state receives:

- an Adjust Attached Transmitter request from the management application layer.

**5.19.5.2.2 Transition from PAPTA\_L\_A0:Initialize to PAPTA\_L\_A1:Start**

This transition shall occur:

- a) after receiving an Adjust Attached Transmitter request from the management application layer.

**5.19.5.3 PAPTA\_L\_A1:Start state****5.19.5.3.1 State description**

This state sends a start APTA adjustment to the attached phy and waits for a response.

Upon entry into this state, this state shall send:

- a) a Transmit APTA\_ADJUST message with an argument of Start to the SP transmitter.

This state waits for:

- a) a Received APTA\_ADJUST message with an argument of Ready.

If this state receives a Received APTA\_ADJUST message with an argument of Terminate, then this state shall send:

- a) an Attached Phy Terminated APTA confirmation to the management application layer.

If this state receives:

- a) a Terminate APTA request from the management application layer;
- b) a Disable APTA request from the link layer; or
- c) a Disable APTA message,

then this state shall send:

- a) a Transmit APTA\_ADJUST message with an argument of Terminate to the SP transmitter.

**5.19.5.3.2 Transition from PAPTA\_L\_A1:Start to PAPTA\_L\_A0:Initialize**

This transition shall occur after sending:

- a) an Attached Phy Terminated APTA confirmation to the management application layer; or
- b) a Transmit APTA\_ADJUST message with an argument of Terminate to the SP transmitter.

**5.19.5.3.3 Transition from PAPTA\_L\_A1:Start to PAPTA\_L\_A2:Adjust\_Attached\_Transmitter**

This transition shall occur:

- a) after receiving a Received APTA\_ADJUST message with an argument of Ready.

**5.19.5.4 PAPTA\_L\_A2:Adjust\_Attached\_Transmitter state****5.19.5.4.1 State description**

This state:

- a) gets the coefficient that the local phy's SP receiver is requesting be sent to the attached phy's SP transmitter;
- b) sends messages to the PAPTA\_TC state machines;
- c) reports that the adjustment is complete; and
- d) waits for a PAPTA\_TC state machine to report completion of each coefficient change request.

Upon entry into this state, this state shall send:

- a) a Get Next Coefficient message to the SP receiver.



This state receives a Next Coefficient message from the SP receiver and then sends a message to the PAPTA\_TC state machine (see 5.19.6) as defined in table 113.

**Table 113 – Mapping messages to the PAPTA\_TC state machine**

<b>Argument of the Next Coefficient message from the SP receiver</b>	<b>Message sent to the PAPTA_TC state machine</b>
Increment Coefficient 1	L_A Adjust Coefficient 1 (Increment)
Decrement Coefficient 1	L_A Adjust Coefficient 1 (Decrement)
Increment Coefficient 2	L_A Adjust Coefficient 2 (Increment)
Decrement Coefficient 2	L_A Adjust Coefficient 2 (Decrement)
Increment Coefficient 3	L_A Adjust Coefficient 3 (Increment)
Decrement Coefficient 3	L_A Adjust Coefficient 3 (Decrement)
Increment Coefficients 1_2	L_A Adjust Coefficients 1_2 (Increment)
Decrement Coefficients 1_2	L_A Adjust Coefficients 1_2 (Decrement)
Increment Coefficients 2_3	L_A Adjust Coefficients 2_3 (Increment)
Decrement Coefficients 2_3	L_A Adjust Coefficients 2_3 (Decrement)

If this state receives a Next Coefficient message with an argument of Adjustment Complete, then this state shall send:

- a) a Transmit APTA\_ADJUST message with an argument of Complete to the SP transmitter; and
- b) an Adjustment Complete confirmation to the management application layer.

After receiving:

- a) a Coefficient 1 Adjustment Done message;
- b) a Coefficient 2 Adjustment Done message;
- c) a Coefficient 3 Adjustment Done message;
- d) a Coefficients 1\_2 Adjustment Done message; or
- e) a Coefficients 2\_3 Adjustment Done message,

then this state shall send a Get Next Coefficient message to the SP receiver.

If this state receives a Received APTA\_ADJUST message with an argument of Terminate, then this state shall send:

- a) an Attached Phy Terminated APTA confirmation to the management application layer.

If this state receives:

- a) a Terminate APTA request from the management application layer;
- b) a Disable APTA request from the link layer; or
- c) a Disable APTA message,

then this state shall send:

- a) a Transmit APTA\_ADJUST message with an argument of Terminate to the SP transmitter.

#### 5.19.5.4.2 Transition from PAPTA\_L\_A2:Adjust\_Attached\_Transmitter to PAPTA\_L\_A0:Initialize

This transition shall occur after sending:

- a) an Attached Phy Terminated APTA confirmation to the management application layer; or
- b) a Transmit APTA\_ADJUST message with an argument of Terminate to the SP transmitter.

#### 5.19.6 PAPTA\_TC (phy layer SP receiver management of attached SP transmitter coefficient adjustments) state machines

##### 5.19.6.1 PAPTA\_TC state machines overview

The PAPTA\_TC state machines are as follows:

- a) PAPTA\_TC1 (local receiver requests adjustment to SP transmitter coefficient 1);
- b) PAPTA\_TC2 (local receiver requests adjustment to SP transmitter coefficient 2);
- c) PAPTA\_TC3 (local receiver requests adjustment to SP transmitter coefficient 3);
- d) PAPTA\_TC1\_2 (local receiver requests adjustment to SP transmitter coefficients 1\_2); and
- e) PAPTA\_TC2\_3 (local receiver requests adjustment to SP transmitter coefficients 2\_3).

The function of the PAPTA\_TC state machines is to:

- a) send adjustment request messages to the SP transmitter;
- b) wait for a status response messages from the SP receiver;
- c) wait for at least a Receiver Adaptation time (see table 114); and
- d) report completion of the adjustment request.

These state machines receive the following requests from the management application layer:

- a) Terminate APTA.

This state machine receives the following request from the link layer:

- a) Disable APTA.

These state machines sends the following confirmations to the management application layer:

- a) Attached Phy Terminated APTA.

These state machines receives the following message from the SP state machine:

- a) Disable APTA.

This state machine shall maintain the timer listed in table 114.

**Table 114 – PAPTA\_TC state machine timers**

Timer	Initial value
Receiver Adaptation timer	1 ms

Figure 123 shows the PAPTA\_TC state machines.

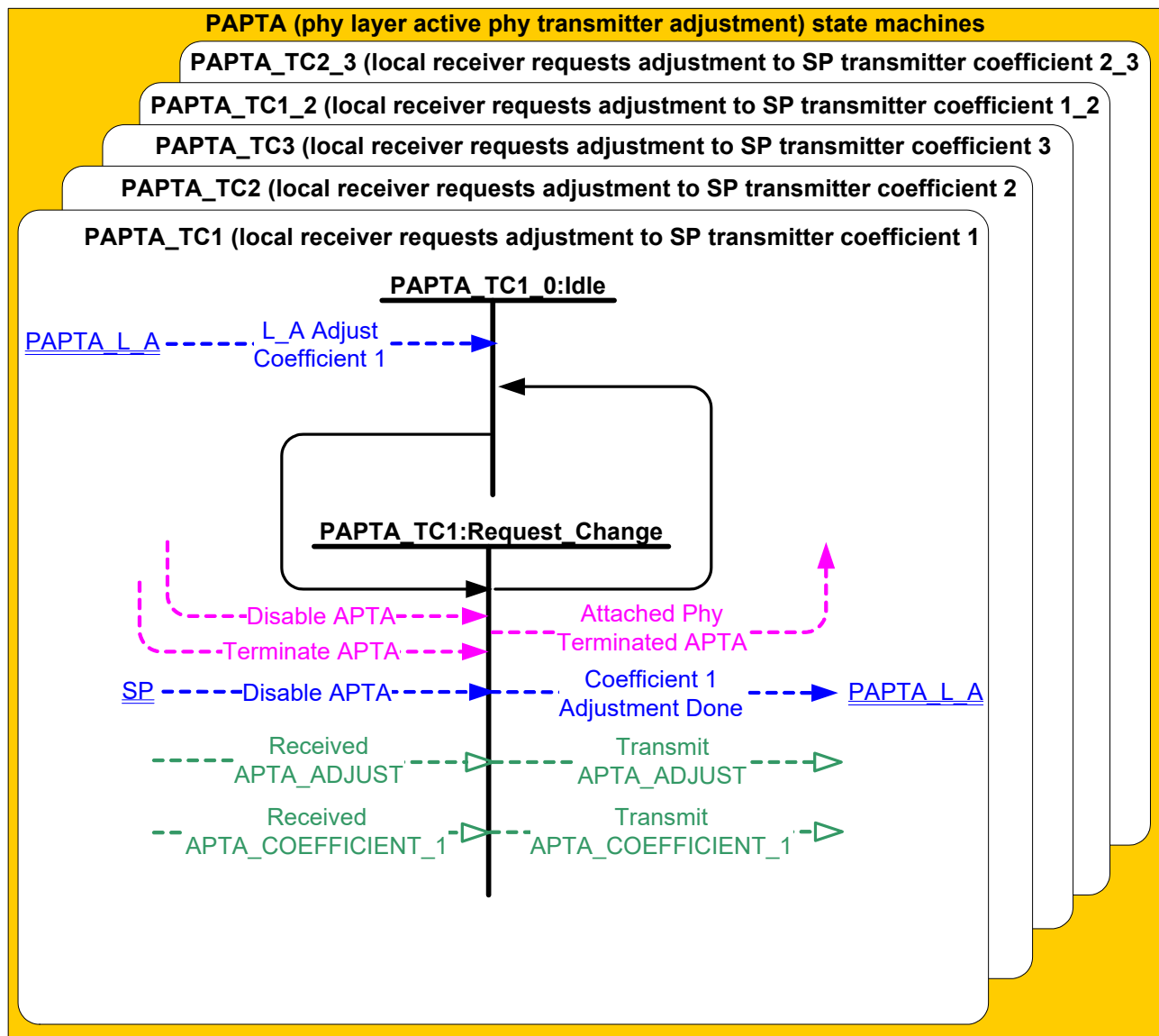


Figure 123 – PAPTA\_TC (phy layer SP receiver management of attached SP transmitter coefficient adjustments) state machines

#### 5.19.6.2 PAPTA\_TC1 state machine

##### 5.19.6.2.1 PAPTA\_TC1 state machine overview

This state machine requests changes to the value of coefficient 1 (see SAS-4) of the attached SP transmitter.

This state machine consists of the following states:

- PAPTA\_TC1\_0:Idle (see 5.19.6.2.2) (initial state); and
- PAPTA\_TC1\_1:Request\_Change (see 5.19.6.2.3).

**5.19.6.2.2 PAPTA\_TC1\_0:Idle state****5.19.6.2.2.1 State description**

This is the initial state of this state machine.

This state waits for:

- a) a L\_A Adjust Coefficient 1 message.

**5.19.6.2.2.2 Transition PAPTA\_TC1\_0:Idle to PAPTA\_TC1\_1:Request\_Change**

This transition shall occur after:

- a) an PAPTA\_L\_A:L\_A Adjust Coefficient 1 message is received.

**5.19.6.2.3 PAPTA\_TC1\_1:Request\_Change state****5.19.6.2.3.1 State description**

This state:

- a) sends requests to the SP transmitter to send APTA binary primitives to adjust the attached phys SP transmitter coefficient values;
- b) waits for APTA status messages from the SP receiver; and
- c) manages the Receiver Adaptation timer.

If this state was entered with an Increment argument, then this state shall send:

- a) a Transmit APTA\_COEFFICIENT\_1 message with an argument of Increment to the SP transmitter.

If this state was entered with a Decrement argument, then this state shall send:

- a) a Transmit APTA\_COEFFICIENT\_1 message with an argument of Decrement to the SP transmitter.

On receiving a Received APTT\_COEFFICIENT\_1 message with an argument of Updated, Maximum, or Minimum, this state shall initialize and start the Receiver Adaptation timer.

When the Receiver Adaptation timer expires this state shall:

- a) send a Coefficient 1 Adjustment Done message to the PAPTA\_L\_A state machine; and
- b) stop the Receiver Adaptation timer.

If this state receives a Received APTA\_ADJUST message with an argument of Terminate, then this state shall send:

- a) an Attached Phy Terminated APTA confirmation to the management application layer.

If this state receives:

- a) a Terminate APTA request from the management application layer;
- b) a Disable APTA request from the link layer; or
- c) a Disable APTA message,

then this state shall send:

- a) a Transmit APTA\_ADJUST message with an argument of Terminate to the SP transmitter.

**5.19.6.2.3.2 Transition PAPTA\_TC1\_1:Request\_Change to PAPTA\_TC0:Idle**

This transition shall occur after sending:

- a) a Coefficient 1 Adjustment Done message to the PAPTA\_L\_A state machine;
- b) an Attached Phy Terminated APTT confirmation to the management application layer; or
- c) a Transmit APTA\_ADJUST message with an argument of Terminate to the SP transmitter.

This transition shall stop the Receiver Adaptation timer.

#### 5.19.6.3 PAPTA\_TC2 state machine

This state machine is identical to PAPTA\_TC1 (see 5.19.6.2) except for the messages described in table 115.

**Table 115 – PAPTA\_TC1 messages to substitute for PAPTA\_TC2 messages**

<b>PAPTA_TC1 message</b>	<b>PAPTA_TC2 message</b>
L_A Adjust Coefficient 1	L_A Adjust Coefficient 2
Received APTA_COEFFICIENT_1	Received APTA_COEFFICIENT_2
Coefficient 1 Adjustment Done	Coefficient 2 Adjustment Done

#### 5.19.6.4 PAPTA\_TC3 state machine

This state machine is identical to PAPTA\_TC1 (see 5.19.6.2) except for the messages described in table 116.

**Table 116 – PAPTA\_TC1 messages to substitute for PAPTA\_TC3 messages**

<b>PAPTA_TC1 message</b>	<b>PAPTA_TC3 message</b>
L_A Adjust Coefficient 1	L_A Adjust Coefficient 3
Received APTA_COEFFICIENT_1	Received APTA_COEFFICIENT_3
Coefficient 1 Adjustment Done	Coefficient 3 Adjustment Done

#### 5.19.6.5 PAPTA\_TC1\_2 state machine

This state machine is identical to PAPTA\_TC1 (see 5.19.6.2) except for the messages described in table 117.

**Table 117 – PAPTA\_TC1 messages to substitute for PAPTA\_TC1\_2 messages**

<b>PAPTA_TC1 message</b>	<b>PAPTA_TC1_2 message</b>
L_A Adjust Coefficient 1	L_A Adjust Coefficient 1_2
Received APTA_COEFFICIENT_1	Received APTA_COEFFICIENT_1_2
Coefficient 1 Adjustment Done	Coefficient 1_2 Adjustment Done

### 5.19.6.6 PAPTA\_TC2\_3 state machine

This state machine is identical to PAPTA\_TC1 (see 5.19.6.2) except for the messages described in table 118.

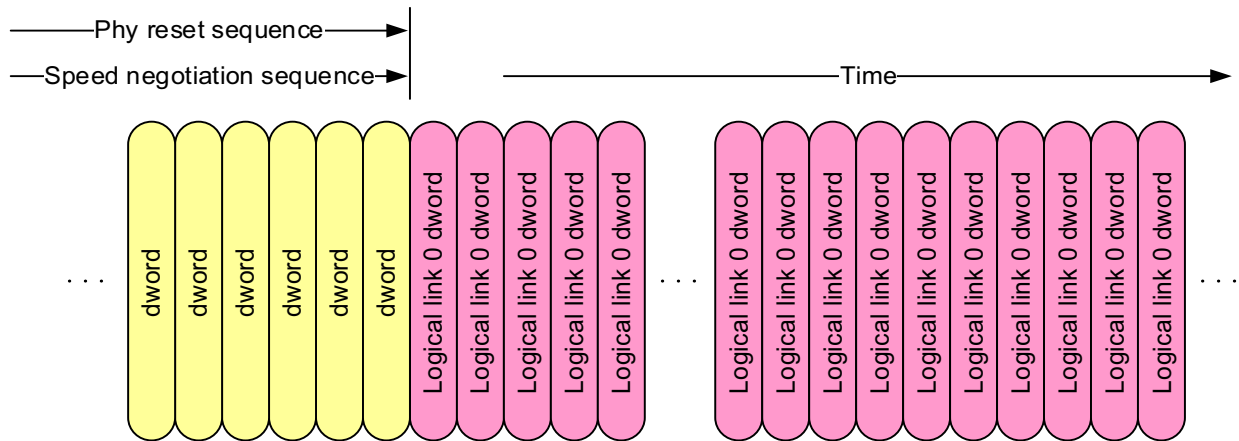
**Table 118 – PAPTA\_TC1 messages to substitute for PAPTA\_TC2\_3 messages**

PAPTA_TC1 message	PAPTA_TC2_3 message
L_A Adjust Coefficient 1	L_A Adjust Coefficient 2_3
Received APTA_COEFFICIENT_1	Received APTA_COEFFICIENT_2_3
Coefficient 1 Adjustment Done	Coefficient 2_3 Adjustment Done

## 5.20 Multiplexing

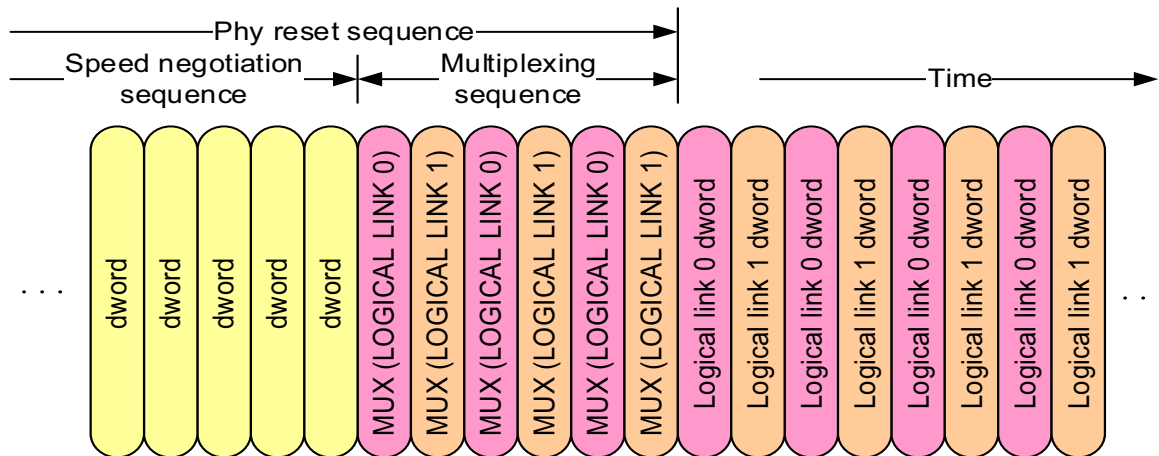
If SNW-3 indicates multiplexing is enabled (see table 72), then the phy shall begin multiplexing immediately after the multiplexing sequence (see 5.11.4.3).

Figure 124 shows multiplexing disabled (i.e., one logical link).



**Figure 124 – Multiplexing disabled**

Figure 125 shows multiplexing enabled (i.e., two logical links).



**Figure 125 – Multiplexing enabled**

After the multiplexing sequence completes, each logical phy shall honor the deletable primitive insertion requirements for physical link rate tolerance management defined in 6.5. The logical phys shall ignore MUXs.

If a phy with multiplexing enabled ever loses dword synchronization, then it shall restart the link reset sequence rather than attempt to reestablish dword synchronization.

Once the multiplexing sequence is complete, the phy shall not perform another multiplexing sequence until a new link reset sequence.

Once the multiplexing sequence is complete:

- a) a logical phy originating dwords shall transmit MUX as a deletable primitive (e.g., substituted in place of an ALIGN) at least once every millisecond; and
- b) a logical phy forwarding dwords should transmit MUX as a deletable primitive at least once every millisecond to confirm the logical link numbers.

Transmitting NOTIFY has higher priority than transmitting MUX.

NOTE 14 - Periodic MUX transmission is for the convenience of logic analyzers and to provide additional assurance that the receiving phy is in agreement with the transmitting phy.

If a phy ever receives a MUX that does not match the MUX expected in that position (i.e., it receives MUX (LOGICAL LINK 1) on logical link 0 or receives MUX (LOGICAL LINK 0) on logical link 1), then it shall perform a link reset sequence.

## 5.21 Spinup

If a SAS target device receives COMSAS during the OOB sequence, then it shall not temporarily consume additional power (e.g., to spin up rotating media) until allowed by the SA\_PC state machine (see 9.2.10).

Expander devices that detect an attached SATA phy may halt the automatic phy reset sequence after the COMSAS Detect Timeout (see 5.14) to delay temporary consumption of additional power. The resulting SATA spinup hold is reported in the NEGOTIATED PHYSICAL LINK RATE field in the SMP DISCOVER response (see 9.4.3.10) and is released with the SMP PHY CONTROL function (see 9.4.3.28).

NOTE 15 - Some enclosures supporting both SATA devices and SAS target devices sequence power to each attached device to avoid excessive power consumption during power on, since some SATA devices

temporarily consume additional power automatically after power on if staggered spin-up is not implemented (see SATA).



## 6 Link layer

### 6.1 Link layer overview

The link layer defines primitives, address frames, and connections. Link layer state machines interface to the port layer and the phy layer and perform the identification and hard reset sequences, connection management, and SSP, STP, and SMP specific frame transmission and reception.

### 6.2 Primitives

#### 6.2.1 Primitives overview

Primitives are dwords whose first character is a K28.3 or K28.5. Primitives are not considered big-endian or little-endian. Primitives are interpreted as first, second, third, and last characters. Table 119 defines the 00primitive format.

**Table 119 – Primitive format**

Character	Description
First	K28.5 control character (for primitives defined in this standard) or K28.3 control character (for primitives defined by SATA).
Second	Constant data character.
Third	Constant data character.
Last	Constant data character.

## 6.2.2 Primitive summary

Table 120 defines the deletable primitives.

**Table 120 – Deletable primitives**

Primitive	Use <sup>a</sup>	From <sup>b</sup>			To <sup>b</sup>			Primitive sequence type <sup>c</sup>
		I	E	T	I	E	T	
ALIGN (0)	All, SpNeg							Single
ALIGN (1)	SAS, SpNeg	I	E	T	I	E	T	
ALIGN (2)	SAS							
ALIGN (3)								
MUX (LOGICAL LINK 0)	SAS	I	E	T	I	E	T	Single
MUX (LOGICAL LINK 1)								
NOTIFY (ENABLE SPINUP)	SAS	I	E				T	Single
NOTIFY (POWER LOSS EXPECTED)		I	E				T	
NOTIFY (RESERVED 1)					I	E	T	
OOB_IDLE	SpNeg	I	E	T	I	E	T	Single
<p>Key:</p> <p>All = SAS logical links and SATA physical links</p> <p>SAS = SAS logical links, both outside connections and inside any type of connection</p> <p>NoConn = SAS logical links, outside connections</p> <p>Conn = SAS logical links, inside connections</p> <p>STP = SAS logical links, inside STP connections</p> <p>SpNeg = SAS physical links, during speed negotiation</p> <p>I = SAS initiator ports</p> <p>E = expander ports</p> <p>T = SAS target ports</p> <p><sup>a</sup> The Use column indicates when the primitive is used.</p> <p><sup>b</sup> The From and To columns indicate the type of ports that originate each primitive or are the intended destinations of each primitive. Expander ports are not considered originators of primitives that are being forwarded from expander port to expander port within an expander.</p> <p><sup>c</sup> The Primitive sequence type columns indicate whether the primitive is a single primitive sequence, a repeated primitive sequence, a continued primitive sequence, a triple primitive sequence, or a redundant primitive sequence (see 6.2.4).</p>								

Table 121 defines the primitives not specific to the type of connection.

**Table 121 – Primitives not specific to type of connection (part 1 of 4)**

Primitive	Use <sup>a</sup>	From <sup>b</sup>			To <sup>b</sup>			Primitive sequence type <sup>c</sup>
		I	E	T	I	E	T	
AIP (NORMAL)	NoConn		E					Extended
AIP (RESERVED 0)								
AIP (RESERVED 1)								
AIP (RESERVED 2)					I	E	T	
AIP (RESERVED WAITING ON PARTIAL)								
AIP (WAITING ON CONNECTION)			E					
AIP (WAITING ON DEVICE)			E					
AIP (WAITING ON PARTIAL)			E					
BREAK	SAS	I	E	T	I	E	T	Redundant
BREAK_REPLY	SAS	I	E	T	I	E	T	Redundant
BROADCAST (CHANGE)	NoConn	I	E		I			Redundant
BROADCAST (SES)				T	I			
BROADCAST (EXPANDER)			E		I			
BROADCAST (ASYNCHRONOUS EVENT)				T	I			
BROADCAST (RESERVED 3)								
BROADCAST (RESERVED 4)								
BROADCAST (RESERVED CHANGE 0)					I			
BROADCAST (RESERVED CHANGE 1)					I			
Key:  All = SAS logical links and SATA physical links SAS = SAS logical links, both outside connections and inside any type of connection NoConn = SAS logical links, outside connections Conn = SAS logical links, inside connections STP = SAS logical links, inside STP connections SpNeg = SAS physical links, during speed negotiation I = SAS initiator ports E = expander ports T = SAS target ports								
<sup>a</sup> The Use column indicates when the primitive is used. <sup>b</sup> The From and To columns indicate the type of ports that originate each primitive or are the intended destinations of each primitive. Expander ports are not considered originators of primitives that are being forwarded from expander port to expander port within an expander. <sup>c</sup> The Primitive sequence type columns indicate whether the primitive is a single primitive sequence, a repeated primitive sequence, a continued primitive sequence, an extended primitive sequence, a triple primitive sequence, or a redundant primitive sequence (see 6.2.4).								

Table 121 – Primitives not specific to type of connection (part 2 of 4)

Primitive	Use <sup>a</sup>	From <sup>b</sup>			To <sup>b</sup>			Primitive sequence type <sup>c</sup>
		I	E	T	I	E	T	
CLOSE (CLEAR AFFILIATION)	STP	I					T	Triple
CLOSE (NORMAL)	Conn	I		T				
CLOSE (RESERVED 0)					I		T	
CLOSE (RESERVED 1)								
EOAF	NoConn	I	E	T	I	E	T	Single
ERROR	SAS		E		I	E	T	Single
HARD_RESET	NoConn	I	E		I	E	T	Redundant
OPEN_ACCEPT	NoConn	I		T	I		T	Single
<p>Key:</p> <p>All = SAS logical links and SATA physical links</p> <p>SAS = SAS logical links, both outside connections and inside any type of connection</p> <p>NoConn = SAS logical links, outside connections</p> <p>Conn = SAS logical links, inside connections</p> <p>STP = SAS logical links, inside STP connections</p> <p>SpNeg = SAS physical links, during speed negotiation</p> <p>I = SAS initiator ports</p> <p>E = expander ports</p> <p>T = SAS target ports</p> <p><sup>a</sup> The Use column indicates when the primitive is used.</p> <p><sup>b</sup> The From and To columns indicate the type of ports that originate each primitive or are the intended destinations of each primitive. Expander ports are not considered originators of primitives that are being forwarded from expander port to expander port within an expander.</p> <p><sup>c</sup> The Primitive sequence type columns indicate whether the primitive is a single primitive sequence, a repeated primitive sequence, a continued primitive sequence, an extended primitive sequence, a triple primitive sequence, or a redundant primitive sequence (see 6.2.4).</p>								

Table 121 – Primitives not specific to type of connection (part 3 of 4)

Primitive	Use <sup>a</sup>	From <sup>b</sup>			To <sup>b</sup>			Primitive sequence type <sup>c</sup>
		I	E	T	I	E	T	
OPEN_REJECT (BAD DESTINATION)	NoConn		E					Single
OPEN_REJECT (CONNECTION RATE NOT SUPPORTED)		I	E	T				
OPEN_REJECT (NO DESTINATION)			E					
OPEN_REJECT (PATHWAY BLOCKED)			E					
OPEN_REJECT (PROTOCOL NOT SUPPORTED)		I	E	T				
OPEN_REJECT (RESERVED ABANDON 1)								
OPEN_REJECT (RESERVED ABANDON 2)								
OPEN_REJECT (RESERVED ABANDON 3)					I		T	
OPEN_REJECT (RESERVED CONTINUE 0)								
OPEN_REJECT (RESERVED CONTINUE 1)								
OPEN_REJECT (RESERVED INITIALIZE 0)								
OPEN_REJECT (RESERVED INITIALIZE 1)								
OPEN_REJECT (RESERVED STOP 0)								
OPEN_REJECT (RESERVED STOP 1)								
OPEN_REJECT (RETRY)		I	E	T				
OPEN_REJECT (STP RESOURCES BUSY)			E	T	I			
OPEN_REJECT (WRONG DESTINATION)		I		T			T	
OPEN_REJECT (ZONE VIOLATION)			E		I			
PS_ACK	NoConn	I	E	T	I	E	T	Redundant
PS_NAK	NoConn	I	E	T	I	E	T	Redundant
PS_REQ (PARTIAL)	NoConn	I	E	T	I	E	T	Redundant
PS_REQ (SLUMBER)	NoConn	I	E	T	I	E	T	Redundant
<p>Key:</p> <p>All = SAS logical links and SATA physical links</p> <p>SAS = SAS logical links, both outside connections and inside any type of connection</p> <p>NoConn = SAS logical links, outside connections</p> <p>Conn = SAS logical links, inside connections</p> <p>STP = SAS logical links, inside STP connections</p> <p>SpNeg = SAS physical links, during speed negotiation</p> <p>I = SAS initiator ports</p> <p>E = expander ports</p> <p>T = SAS target ports</p> <p><sup>a</sup> The Use column indicates when the primitive is used.</p> <p><sup>b</sup> The From and To columns indicate the type of ports that originate each primitive or are the intended destinations of each primitive. Expander ports are not considered originators of primitives that are being forwarded from expander port to expander port within an expander.</p> <p><sup>c</sup> The Primitive sequence type columns indicate whether the primitive is a single primitive sequence, a repeated primitive sequence, a continued primitive sequence, an extended primitive sequence, a triple primitive sequence, or a redundant primitive sequence (see 6.2.4).</p>								

Table 121 – Primitives not specific to type of connection (part 4 of 4)

Primitive	Use <sup>a</sup>	From <sup>b</sup>			To <sup>b</sup>			Primitive sequence type <sup>c</sup>
		I	E	T	I	E	T	
PWR_ACK	NoConn	I	E	T	I	E	T	Redundant
PWR_DONE	NoConn			T	I	E		Redundant
PWR_GRANT	NoConn	I	E				T	Redundant
PWR_REQ	NoConn			T	I	E		Redundant
SOAF	NoConn	I	E	T	I	E	T	Single
TRAIN	SpNeg	I	E	T	I	E	T	Redundant
TRAIN_DONE								
Key:  All = SAS logical links and SATA physical links SAS = SAS logical links, both outside connections and inside any type of connection NoConn = SAS logical links, outside connections Conn = SAS logical links, inside connections STP = SAS logical links, inside STP connections SpNeg = SAS physical links, during speed negotiation I = SAS initiator ports E = expander ports T = SAS target ports								
<sup>a</sup> The Use column indicates when the primitive is used. <sup>b</sup> The From and To columns indicate the type of ports that originate each primitive or are the intended destinations of each primitive. Expander ports are not considered originators of primitives that are being forwarded from expander port to expander port within an expander. <sup>c</sup> The Primitive sequence type columns indicate whether the primitive is a single primitive sequence, a repeated primitive sequence, a continued primitive sequence, an extended primitive sequence, a triple primitive sequence, or a redundant primitive sequence (see 6.2.4).								

Table 122 defines the primitives used only inside SSP and SMP connections.

**Table 122 – Primitives used inside SSP and SMP connections** (part 1 of 2)

Primitive	Use <sup>a</sup>	From <sup>b</sup>			To <sup>b</sup>			Primitive sequence type <sup>c</sup>
		I	E	T	I	E	T	
ACK	SSP	I		T	I		T	Single
CREDIT_BLOCKED	SSP	I		T	I		T	Single
DONE (ACK/NAK TIMEOUT)	SSP	I		T	I		T	Single
DONE (CREDIT TIMEOUT)		I		T				
DONE (NORMAL)		I		T				
DONE (RESERVED 0)								
DONE (CLOSE)		I		T				
DONE (RESERVED TIMEOUT 0)								
DONE (RESERVED TIMEOUT 1)								
EOF	SSP, SMP	I		T	I		T	Single
EXTEND_CONNECTION (NORMAL)	SSP	I		T	I		T	Single
EXTEND_CONNECTION (CLOSE)			E					
Key:  SSP = SAS logical links, inside SSP connections SMP = SAS logical links, inside SMP connections I = SSP initiator ports and SMP initiator ports E = expander ports T = SSP target ports and SMP target ports								
<div><div><sup>a</sup> The Use column indicates when the primitive is used.</div><div><sup>b</sup> The From and To columns indicate the type of ports that originate each primitive or are the intended destinations of each primitive. Expander ports are not considered originators of primitives that are being forwarded from expander port to expander port within an expander.</div><div><sup>c</sup> The Primitive sequence type columns indicate whether the primitive is a single primitive sequence, a repeated primitive sequence, a continued primitive sequence, an extended primitive sequence, a triple primitive sequence, or a redundant primitive sequence (see 6.2.4).</div></div>								

Table 122 – Primitives used inside SSP and SMP connections (part 2 of 2)

Primitive	Use <sup>a</sup>	From <sup>b</sup>			To <sup>b</sup>			Primitive sequence type <sup>c</sup>
		I	E	T	I	E	T	
NAK (CRC ERROR)	SSP	I		T	I		T	Single
NAK (RESERVED 0)								
NAK (RESERVED 1)								
NAK (RESERVED 2)								
RRDY (NORMAL)	SSP	I		T	I		T	Single
RRDY (RESERVED 0)								
RRDY (CLOSE)		I	E	T				
SOF	SSP, SMP	I		T	I		T	Single
Key: SSP = SAS logical links, inside SSP connections SMP = SAS logical links, inside SMP connections I = SSP initiator ports and SMP initiator ports E = expander ports T = SSP target ports and SMP target ports								
<sup>a</sup> The Use column indicates when the primitive is used. <sup>b</sup> The From and To columns indicate the type of ports that originate each primitive or are the intended destinations of each primitive. Expander ports are not considered originators of primitives that are being forwarded from expander port to expander port within an expander. <sup>c</sup> The Primitive sequence type columns indicate whether the primitive is a single primitive sequence, a repeated primitive sequence, a continued primitive sequence, an extended primitive sequence, a triple primitive sequence, or a redundant primitive sequence (see 6.2.4).								



Table 123 lists the primitives used only inside STP connections and on SATA physical links.

**Table 123 – Primitives used inside STP connections and on SATA physical links (part 1 of 2)**

Primitive	Use <sup>a</sup>	From <sup>b</sup>			To <sup>b</sup>			Primitive sequence type <sup>c</sup>
		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 <sup>d</sup>	SATA		E				T	Single
SATA_HOLD	STP, SATA	I		T	I		T	Continued
SATA_HOLDA	STP, SATA	I		T	I		T	Continued
SATA_PMACK	SATA	I	E					Repeated
	STP							
SATA_PMNAK	STP, SATA	I	E				T	Repeated
SATA_PMREQ_P	SATA				I	E		Continued
	STP							
Key:  STP = SAS logical links, inside STP connections SATA = SATA physical links I = STP initiator ports and SATA host ports E = expander ports T = STP target ports and SATA device ports								
<sup>a</sup> The Use column indicates when the primitive is used. <sup>b</sup> The From and To columns indicate the type of ports that originate each primitive or are the intended destinations of each primitive. Expander ports are not considered originators of primitives that are being forwarded from expander port to expander port within an expander. <sup>c</sup> The Primitive sequence type columns indicate whether the primitive is a single primitive sequence, a repeated primitive sequence, a continued primitive sequence, an extended primitive sequence, a triple primitive sequence, or a redundant primitive sequence (see 6.2.4). <sup>d</sup> Although included in this table, SATA_ERROR is not a primitive since it starts with K28.6. It does not appear inside STP connections. It is an invalid dword, used by expander devices forwarding an error onto a SATA physical link (see 6.2.8.1).								

Table 123 – Primitives used inside STP connections and on SATA physical links (part 2 of 2)

Primitive	Use <sup>a</sup>	From <sup>b</sup>			To <sup>b</sup>			Primitive sequence type <sup>c</sup>
		I	E	T	I	E	T	
SATA_PMREQ_S	SATA				I	E		Continued
	STP							
SATA_R_ERR	STP, SATA	I		T	I		T	Continued
SATA_R_IP	STP, SATA	I		T	I		T	Continued
SATA_R_OK	STP, SATA	I		T	I		T	Continued
SATA_R_RDY	STP, SATA	I		T	I		T	Continued
SATA_SOF	STP, SATA	I		T	I		T	Single
SATA_SYNC	STP, SATA	I		T	I		T	Continued
SATA_WTRM	STP, SATA	I		T	I		T	Continued
SATA_X_RDY	STP, SATA	I		T	I		T	Continued
Key: STP = SAS logical links, inside STP connections SATA = SATA physical links I = STP initiator ports and SATA host ports E = expander ports T = STP target ports and SATA device ports								
<sup>a</sup> The Use column indicates when the primitive is used. <sup>b</sup> The From and To columns indicate the type of ports that originate each primitive or are the intended destinations of each primitive. Expander ports are not considered originators of primitives that are being forwarded from expander port to expander port within an expander. <sup>c</sup> The Primitive sequence type columns indicate whether the primitive is a single primitive sequence, a repeated primitive sequence, a continued primitive sequence, an extended primitive sequence, a triple primitive sequence, or a redundant primitive sequence (see 6.2.4). <sup>d</sup> Although included in this table, SATA_ERROR is not a primitive since it starts with K28.6. It does not appear inside STP connections. It is an invalid dword, used by expander devices forwarding an error onto a SATA physical link (see 6.2.8.1).								

### 6.2.3 Primitive encodings

Table 124 defines the primitive encoding for deletable primitives.

**Table 124 – Primitive encoding for deletable primitives**

Primitive	Character				Hexadecimal
	1 <sup>st</sup>	2 <sup>nd</sup>	3 <sup>rd</sup>	4 <sup>th</sup> (last)	
ALIGN (0)	K28.5	D10.2	D10.2	D27.3	BC4A4A7Bh
ALIGN (1)	K28.5	D07.0	D07.0	D07.0	BC070707h
ALIGN (2)	K28.5	D01.3	D01.3	D01.3	BC616161h
ALIGN (3)	K28.5	D27.3	D27.3	D27.3	BC7B7B7Bh
MUX (LOGICAL LINK 0)	K28.5	D02.0	D16.7	D31.4	BC02F09Fh
MUX (LOGICAL LINK 1)	K28.5	D04.7	D31.4	D27.4	BCE49F9Bh
NOTIFY (ENABLE SPINUP)	K28.5	D31.3	D31.3	D31.3	BC7F7F7Fh
NOTIFY (POWER LOSS EXPECTED)	K28.5	D31.3	D07.0	D01.3	BC7F0761h
NOTIFY (RESERVED 1)	K28.5	D31.3	D01.3	D07.0	BC7F6107h
Obsolete	K28.5	D31.3	D10.2	D10.2	BC7F4A4Ah
OOB_IDLE	K28.5	D07.0	D03.4	D13.4	BC07838Dh

Table 125 defines the primitive encoding for primitives not specific to type of connection.

**Table 125 – Primitive encoding for primitives not specific to type of connection** (part 1 of 2)

Primitive	Character				Hexadecimal
	1 <sup>st</sup>	2 <sup>nd</sup>	3 <sup>rd</sup>	4 <sup>th</sup> (last)	
AIP (NORMAL)	K28.5	D27.4	D27.4	D27.4	BC9B9B9Bh
AIP (RESERVED 0)	K28.5	D27.4	D31.4	D16.7	BC9B9FF0h
AIP (RESERVED 1)	K28.5	D27.4	D16.7	D30.0	BC9BF01Eh
AIP (RESERVED 2)	K28.5	D27.4	D29.7	D01.4	BC9BFD81h
AIP (RESERVED WAITING ON PARTIAL)	K28.5	D27.4	D01.4	D07.3	BC9B8167h
AIP (WAITING ON CONNECTION)	K28.5	D27.4	D07.3	D24.0	BC9B6718h
AIP (WAITING ON DEVICE)	K28.5	D27.4	D30.0	D29.7	BC9B1EFDh
AIP (WAITING ON PARTIAL)	K28.5	D27.4	D24.0	D04.7	BC9B18E4h
BREAK	K28.5	D02.0	D24.0	D07.3	BC021867h
BREAK_REPLY	K28.5	D02.0	D29.7	D16.7	BC02FDF0h
BROADCAST (CHANGE)	K28.5	D04.7	D02.0	D01.4	BCE40281h
BROADCAST (SES)	K28.5	D04.7	D07.3	D29.7	BCE467FDh
BROADCAST (EXPANDER)	K28.5	D04.7	D01.4	D24.0	BCE48118h
BROADCAST (ASYNCHRONOUS EVENT)	K28.5	D04.7	D04.7	D04.7	BCE4E4E4h
BROADCAST (RESERVED 3)	K28.5	D04.7	D16.7	D02.0	BCE4F002h
BROADCAST (RESERVED 4)	K28.5	D04.7	D29.7	D30.0	BCE4FD1Eh
BROADCAST (RESERVED CHANGE 0)	K28.5	D04.7	D24.0	D31.4	BCE4189Fh
BROADCAST (RESERVED CHANGE 1)	K28.5	D04.7	D27.4	D07.3	BCE49B67h
CLOSE (CLEAR AFFILIATION)	K28.5	D02.0	D07.3	D04.7	BC0267E4h
CLOSE (NORMAL)	K28.5	D02.0	D30.0	D27.4	BC021E9Bh
CLOSE (RESERVED 0)	K28.5	D02.0	D31.4	D30.0	BC029F1Eh
CLOSE (RESERVED 1)	K28.5	D02.0	D04.7	D01.4	BC02E481h
EOAF	K28.5	D24.0	D07.3	D31.4	BC18679Fh
ERROR	K28.5	D02.0	D01.4	D29.7	BC0281FDh
HARD_RESET	K28.5	D02.0	D02.0	D02.0	BC020202h
OPEN_ACCEPT	K28.5	D16.7	D16.7	D16.7	BCF0F0F0h
OPEN_REJECT (BAD DESTINATION)	K28.5	D31.4	D31.4	D31.4	BC9F9F9Fh
OPEN_REJECT (CONNECTION RATE NOT SUPPORTED)	K28.5	D31.4	D04.7	D29.7	BC9FE4FDh
OPEN_REJECT (NO DESTINATION)	K28.5	D29.7	D29.7	D29.7	BCFDFDFDh
OPEN_REJECT (PATHWAY BLOCKED)	K28.5	D29.7	D16.7	D04.7	BCFDF0E4h

Table 125 – Primitive encoding for primitives not specific to type of connection (part 2 of 2)

Primitive	Character				Hexadecimal
	1 <sup>st</sup>	2 <sup>nd</sup>	3 <sup>rd</sup>	4 <sup>th</sup> (last)	
OPEN_REJECT (PROTOCOL NOT SUPPORTED)	K28.5	D31.4	D29.7	D07.3	BC9FFD67h
OPEN_REJECT (RESERVED ABANDON 1)	K28.5	D31.4	D30.0	D16.7	BC9F1EF0h
OPEN_REJECT (RESERVED ABANDON 2)	K28.5	D31.4	D07.3	D02.0	BC9F6702h
OPEN_REJECT (RESERVED ABANDON 3)	K28.5	D31.4	D01.4	D30.0	BC9F811Eh
OPEN_REJECT (RESERVED CONTINUE 0)	K28.5	D29.7	D02.0	D30.0	BCFD021Eh
OPEN_REJECT (RESERVED CONTINUE 1)	K28.5	D29.7	D24.0	D01.4	BCFD1881h
OPEN_REJECT (RESERVED INITIALIZE 0)	K28.5	D29.7	D30.0	D31.4	BCFD1E9Fh
OPEN_REJECT (RESERVED INITIALIZE 1)	K28.5	D29.7	D07.3	D16.7	BCFD67F0h
OPEN_REJECT (RESERVED STOP 0)	K28.5	D29.7	D31.4	D07.3	BCFD9F67h
OPEN_REJECT (RESERVED STOP 1)	K28.5	D29.7	D04.7	D27.4	BCFDE49Bh
OPEN_REJECT (RETRY)	K28.5	D29.7	D27.4	D24.0	BCFD9B18h
OPEN_REJECT (STP RESOURCES BUSY)	K28.5	D31.4	D27.4	D01.4	BC9F9B81h
OPEN_REJECT (WRONG DESTINATION)	K28.5	D31.4	D16.7	D24.0	BC9FF018h
OPEN_REJECT (ZONE VIOLATION)	K28.5	D31.4	D02.0	D27.4	BC9F029Bh
PS_ACK	K28.5	D16.7	D27.4	D30.0	BCF09B1Eh
PS_NAK	K28.5	D24.0	D27.4	D02.0	BC189B02h
PS_REQ (PARTIAL)	K28.5	D07.3	D02.0	D04.7	BC6702E4h
PS_REQ (SLUMBER)	K28.5	D30.0	D24.0	D02.0	BC1E1802h
PWR_ACK	K28.5	D07.3	D07.3	D07.3	BC676767h
PWR_DONE	K28.5	D07.3	D24.0	D29.7	BC6718FDh
PWR_GRANT	K28.5	D07.3	D27.4	D16.7	BC679BF0h
PWR_REQ	K28.5	D07.3	D29.7	D27.4	BC67FD9Bh
SOAF	K28.5	D24.0	D30.0	D01.4	BC181E81h
TRAIN	K28.5	D30.3	D30.3	D30.3	BC7E7E7Eh
TRAIN_DONE	K28.5	D30.3	D30.3	D10.2	BC7E7E4Ah

Table 126 defines the primitive encodings for primitives used only inside SSP and SMP connections.

**Table 126 – Primitive encoding for primitives used only inside SSP and SMP connections**

Primitive	Character				Hexadecimal
	1 <sup>st</sup>	2 <sup>nd</sup>	3 <sup>rd</sup>	4 <sup>th</sup> (last)	
ACK	K28.5	D01.4	D01.4	D01.4	BC818181h
CREDIT_BLOCKED	K28.5	D01.4	D07.3	D30.0	BC81671Eh
DONE (ACK/NAK TIMEOUT)	K28.5	D30.0	D01.4	D04.7	BC1E81E4h
DONE (CREDIT TIMEOUT)	K28.5	D30.0	D07.3	D27.4	BC1E679Bh
DONE (NORMAL)	K28.5	D30.0	D30.0	D30.0	BC1E1E1Eh
DONE (RESERVED 0)	K28.5	D30.0	D16.7	D01.4	BC1EF081h
DONE (CLOSE)	K28.5	D30.0	D29.7	D31.4	BC1EFD9Fh
DONE (RESERVED TIMEOUT 0)	K28.5	D30.0	D27.4	D29.7	BC1E9BFDh
DONE (RESERVED TIMEOUT 1)	K28.5	D30.0	D31.4	D24.0	BC1E9F18h
EOF	K28.5	D24.0	D16.7	D27.4	BC18F09Bh
EXTEND_CONNECTION (NORMAL)	K28.5	D24.0	D29.7	D04.7	BC18FDE4h
EXTEND_CONNECTION (CLOSE)	K28.5	D24.0	D01.4	D16.7	BC1881F0h
NAK (CRC ERROR)	K28.5	D01.4	D27.4	D04.7	BC819BE4h
NAK (RESERVED 0)	K28.5	D01.4	D31.4	D29.7	BC819FFDh
NAK (RESERVED 1)	K28.5	D01.4	D04.7	D24.0	BC81E418h
NAK (RESERVED 2)	K28.5	D01.4	D16.7	D07.3	BC81F067h
RRDY (NORMAL)	K28.5	D01.4	D24.0	D16.7	BC8118F0h
RRDY (RESERVED 0)	K28.5	D01.4	D02.0	D31.4	BC81029Fh
RRDY (CLOSE)	K28.5	D01.4	D30.0	D02.0	BC811E02h
SOF	K28.5	D24.0	D04.7	D07.3	BC18E467h

Table 127 lists the primitive encodings for primitives used only inside STP connections and on SATA physical links.

**Table 127 – Primitive encoding for primitives used only inside STP connections and only on SATA physical links**

Primitive	Character				Hexadecimal
	1 <sup>st</sup>	2 <sup>nd</sup>	3 <sup>rd</sup>	4 <sup>th</sup> (last)	
SATA_CONT	K28.3	D10.5	D25.4	D25.4	7CAA9999h
SATA_DMAT	K28.3	D21.5	D22.1	D22.1	7CB53636h
SATA_EOF	K28.3	D21.5	D21.6	D21.6	7CB5D5D5h
SATA_ERROR <sup>a b</sup>	K28.6	D02.0	D01.4	D29.7	DC0281FDh
SATA_HOLD	K28.3	D10.5	D21.6	D21.6	7CAAD5D5h
SATA_HOLDA	K28.3	D10.5	D21.4	D21.4	7CAA9595h
SATA_PMACK	K28.3	D21.4	D21.4	D21.4	7C959595h
SATA_PMNAK	K28.3	D21.4	D21.7	D21.7	7C95F5F5h
SATA_PMREQ_P	K28.3	D21.5	D23.0	D23.0	7CB51717h
SATA_PMREQ_S	K28.3	D21.4	D21.3	D21.3	7C957575h
SATA_R_ERR	K28.3	D21.5	D22.2	D22.2	7CB55656h
SATA_R_IP	K28.3	D21.5	D21.2	D21.2	7CB55555h
SATA_R_OK	K28.3	D21.5	D21.1	D21.1	7CB53535h
SATA_R_RDY	K28.3	D21.4	D10.2	D10.2	7C954A4Ah
SATA_SOF	K28.3	D21.5	D23.1	D23.1	7CB53737h
SATA_SYNC	K28.3	D21.4	D21.5	D21.5	7C95B5B5h
SATA_WTRM	K28.3	D21.5	D24.2	D24.2	7CB55858h
SATA_X_RDY	K28.3	D21.5	D23.2	D23.2	7CB55757h
<sup>a</sup> Except for SATA_ERROR, all values are defined by SATA. <sup>b</sup> Although included in this table, SATA_ERROR is not a primitive since it starts with K28.6. It does not appear inside STP connections. It is an invalid dword, used by expander devices forwarding an error onto a SATA physical link (see 6.2.8.1).					

## 6.2.4 Primitive sequences

### 6.2.4.1 Primitive sequences overview

Table 128 summarizes the types of primitive sequences.

**Table 128 – Primitive sequences**

Primitive sequence type	Transmit <sup>a</sup>	Receive <sup>b</sup>	Length of primitive parameter <sup>c</sup> (dwords)	Reference
Single	1	1	0 to 3	6.2.4.2
Repeated	1 or more	1	0	6.2.4.3
Continued	2 followed by SATA_CONT	1	0	6.2.4.4
Extended	3	1	0 to 3	6.2.4.5
Triple	3	3	0 to 3	6.2.4.6
Redundant	6	3	0 or 1	6.2.4.7
<sup>a</sup> Number of times the transmitter transmits the primitive to transmit the primitive sequence. <sup>b</sup> Number of times the receiver receives the primitive to detect the primitive sequence. <sup>c</sup> If the phy is in the SAS packet mode, then the length in dwords of the primitive parameter that may occur after a primitive sequence within a primitive segment.				

If the phy is in the SAS dword mode, then:

- a) any number of deletable primitives may be transmitted inside primitive sequences without affecting the count or breaking the consecutiveness requirements unless prohibited by the primitives description (e.g., TRAIN, TRAIN\_DONE); and
- b) rate matching deletable primitives shall be transmitted inside primitive sequences inside of connections if rate matching is enabled (see 6.17.2).

If the phy is in the SAS packet mode, then:

- a) any number of SPL packet payloads containing scrambled idle segments or deletable extended binary primitives may be transmitted between the first primitive segment and the second primitive segment of a primitive sequence (see 5.5.4) without affecting the count or breaking the consecutiveness requirements unless prohibited by the primitive's description; and
- b) rate matching SPL packet payloads containing scrambled idle segments shall be transmitted between the first primitive segment and the second primitive segment of a primitive sequence inside of connections if rate matching is enabled (see 6.17.3).

#### 6.2.4.2 Single primitive sequence

Primitives labeled as single primitive sequences (e.g., RRDY, SATA\_SOF) shall be transmitted one time to form a single primitive sequence.

Receivers count each primitive received that is labeled as a single primitive sequence as a distinct single primitive sequence.

ALIGNs, NOTIFYs, and MUXs are deletable primitives (see 6.2.5 and 6.5).

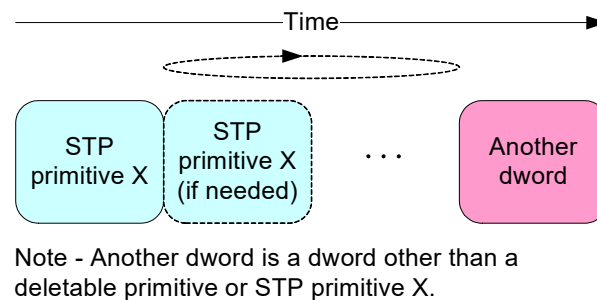


If the phy is in the SAS packet mode, then a primitive parameter with a length of one to three dwords may follow a single primitive sequence. The primitive parameter that follows a single primitive sequence shall be contained within a single primitive segment (i.e., the single primitive sequence plus the associated primitive parameter, if any, shall be contained within a single SPL packet).

#### 6.2.4.3 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. Any number of deletable primitives may be transmitted inside repeated primitive sequences as described in 6.2.4.1.

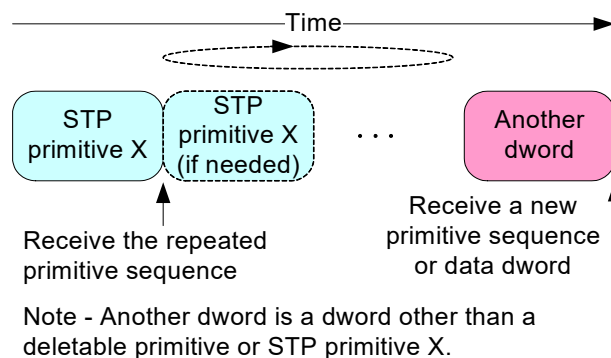
Figure 126 shows an example of transmitting a repeated primitive sequence.



**Figure 126 –Transmitting a repeated primitive sequence**

Receivers do not count the number of times a repeated primitive is received. An expander device forwarding a repeated primitive sequence may transmit more repeated primitives than it receives (i.e., expand) or transmit fewer repeated primitives than it receives (i.e., contract). While transmitting a repeated primitive sequence, the expander device is considered to be originating (see 6.5.2) rather than forwarding (see 6.5.4) for purposes of deletable primitive insertion.

Figure 127 shows an example of receiving a repeated primitive sequence.



**Figure 127 –Receiving a repeated primitive sequence**

#### 6.2.4.4 Continued primitive sequence

Primitives that form continued primitive sequences (e.g., SATA\_HOLD) shall be transmitted as specified in 6.21.5. Any number of deletable primitives may be transmitted inside continued primitive sequences as described in 6.2.4.1.

#### 6.2.4.5 Extended primitive sequence

Primitives that form extended primitive sequences (e.g., AIP) shall be transmitted three times consecutively. Any number of deletable primitives may be transmitted inside extended primitive sequences or between primitive segments as described in 6.2.4.1.

A receiver shall detect an extended primitive sequence after the primitive is received one time. The receiver shall process an extended primitive sequence the same as a single primitive sequence (see 6.2.4.2).

If the phy is in the SAS packet mode, then a primitive parameter with a length of one to three dwords may follow an extended primitive sequence. The primitive parameter that follows an extended primitive sequence shall be contained within the second primitive segment of the extended primitive sequence (i.e., the extended primitive sequence plus the associated primitive parameter, if any, shall be contained within two SPL packets where the second SPL packet contains the associated primitive parameter, if any).

Figure 128 shows examples of extended primitive sequences while a phy is in the SAS dword mode.

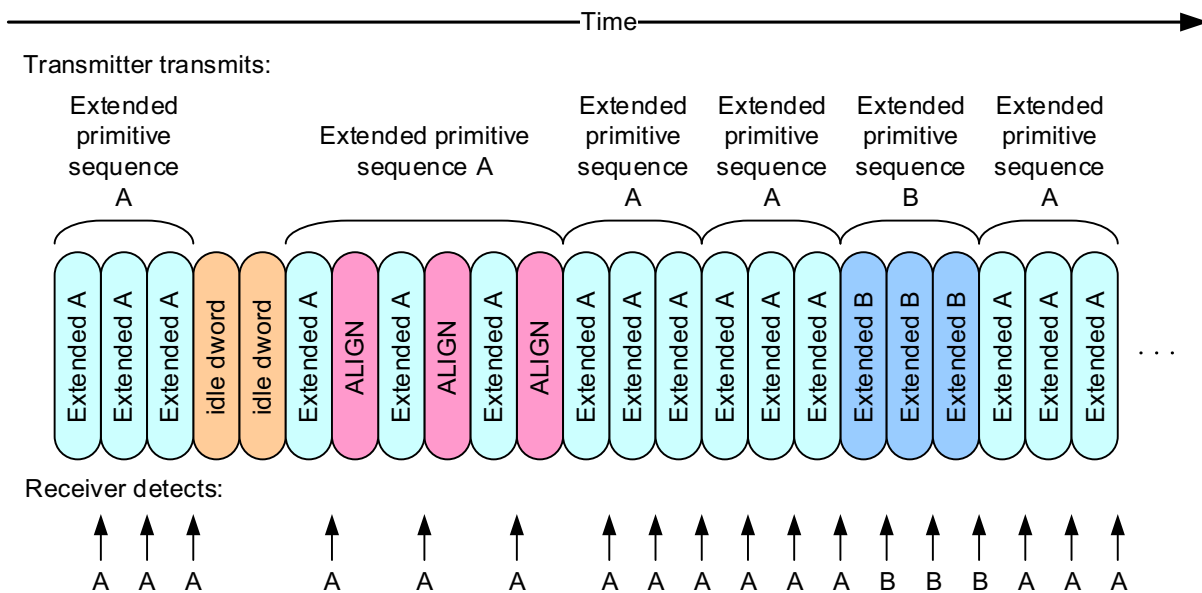
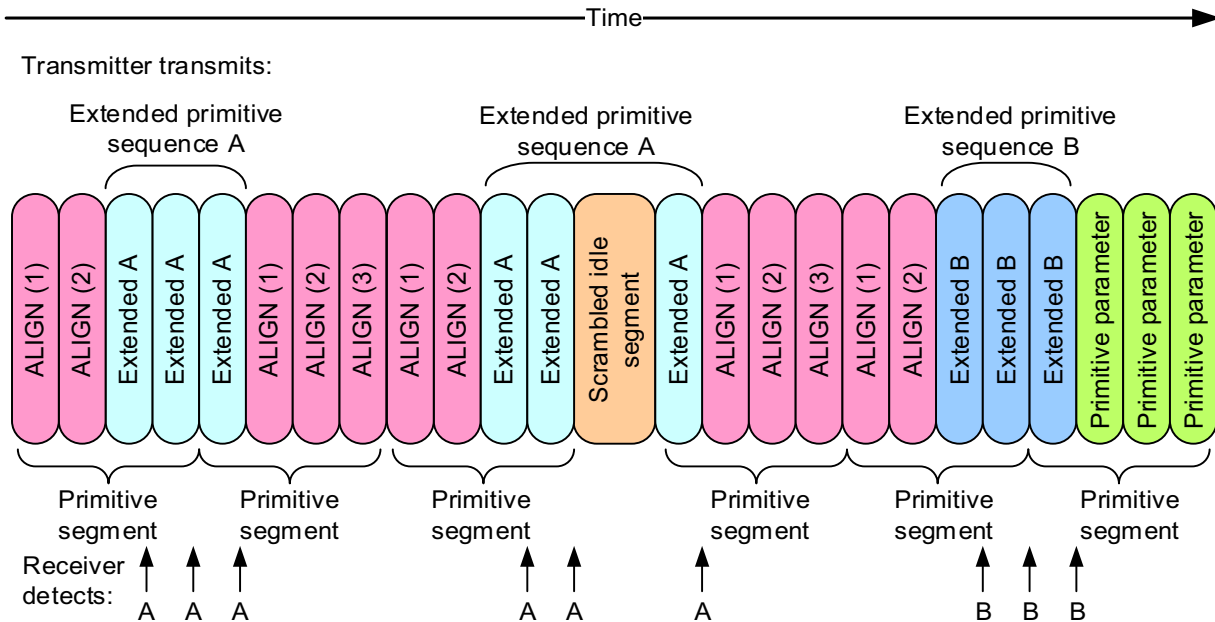


Figure 128 –Extended primitive sequences while in the SAS dword mode

Figure 129 shows examples of extended primitive sequences while a phy is in the SAS packet mode.



**Figure 129 –Extended primitive sequences while in the SAS packet mode**

#### 6.2.4.6 Triple primitive sequence

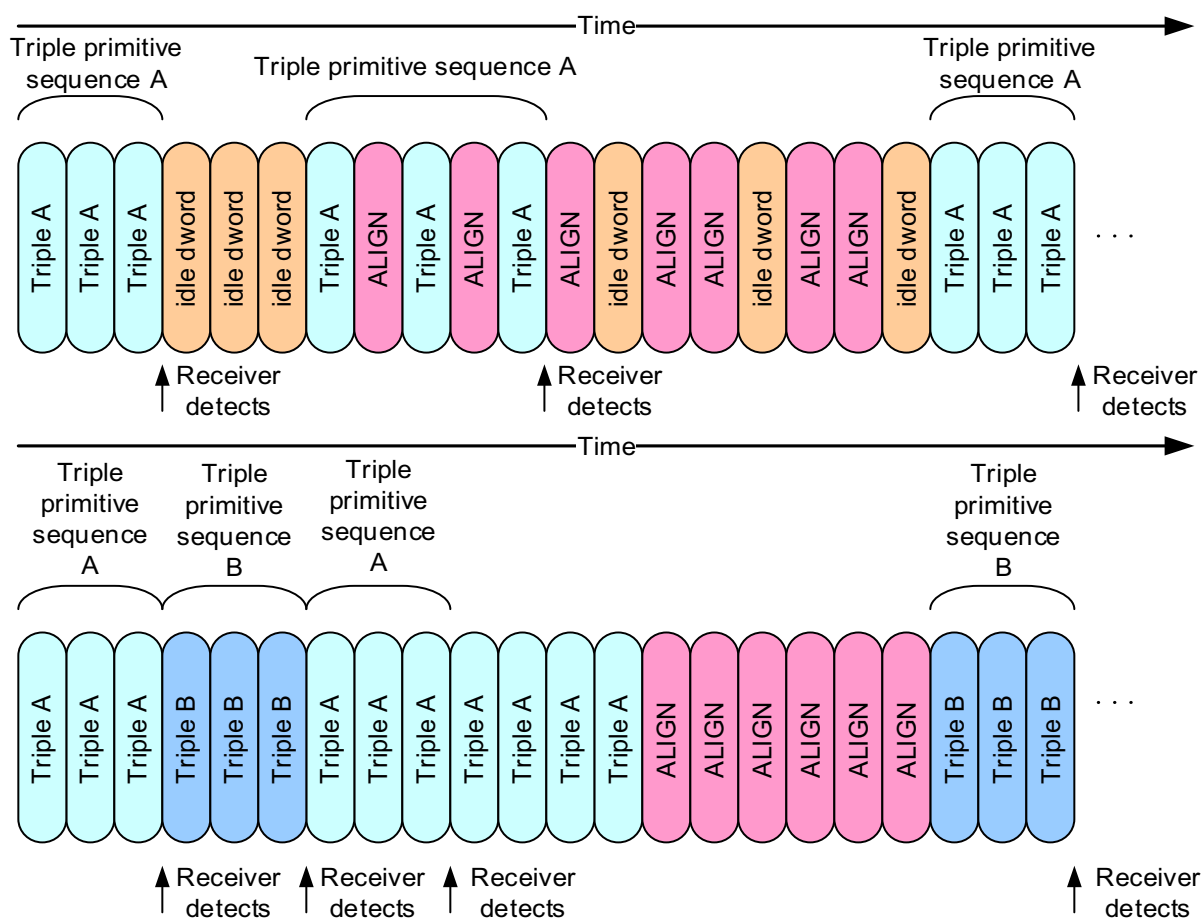
Primitives that form triple primitive sequences (e.g., CLOSE (NORMAL)) shall be transmitted three times consecutively. Any number of deletable primitives may be transmitted inside triple primitive sequences or between primitive segments as described in 6.2.4.1.

Receivers shall detect a triple primitive sequence after the identical primitive is received in three consecutive dwords. After detecting a triple primitive sequence, a receiver shall not detect a second instance of the same triple primitive sequence until it has received three consecutive dwords that are not any of the following:

- a) the original primitive; or
- b) a deletable primitive.

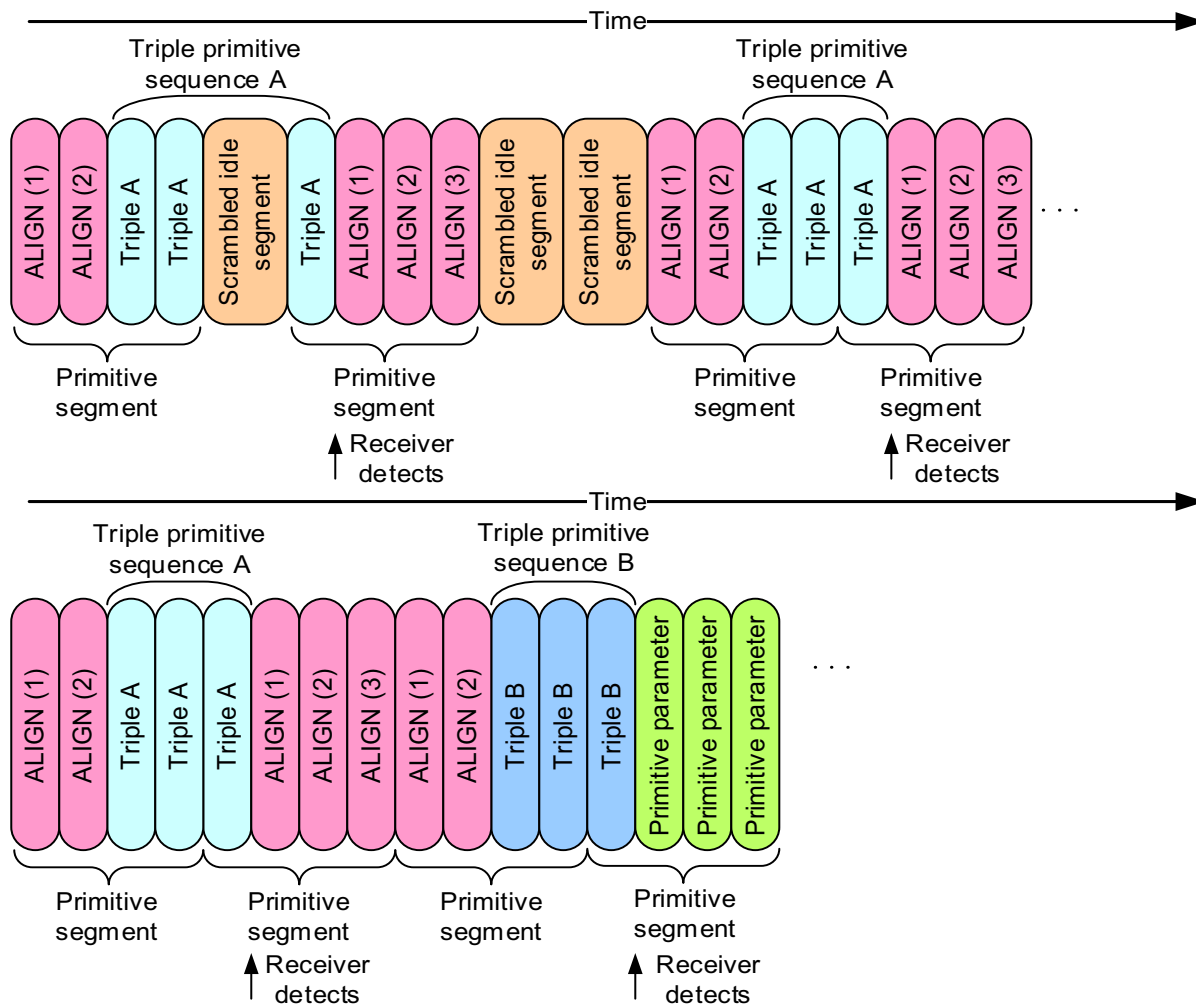
If the phy is in the SAS packet mode, then a primitive parameter with a length of one to three dwords may follow a triple primitive sequence. The primitive parameter that follows a triple primitive sequence shall be contained within the second primitive segment of the triple primitive sequence (i.e., the triple primitive sequence plus the associated primitive parameter, if any, shall be contained within two SPL packets where the second SPL packet contains the associated primitive parameter, if any).

Figure 130 shows examples of triple primitive sequences while a phy is in the SAS dword mode.



**Figure 130 –Triple primitive sequences while in the SAS dword mode**

Figure 131 shows examples of triple primitive sequences while a phy is in the SAS packet mode.



**Figure 131 –Triple primitive sequences while in the SAS packet mode**

#### 6.2.4.7 Redundant primitive sequence

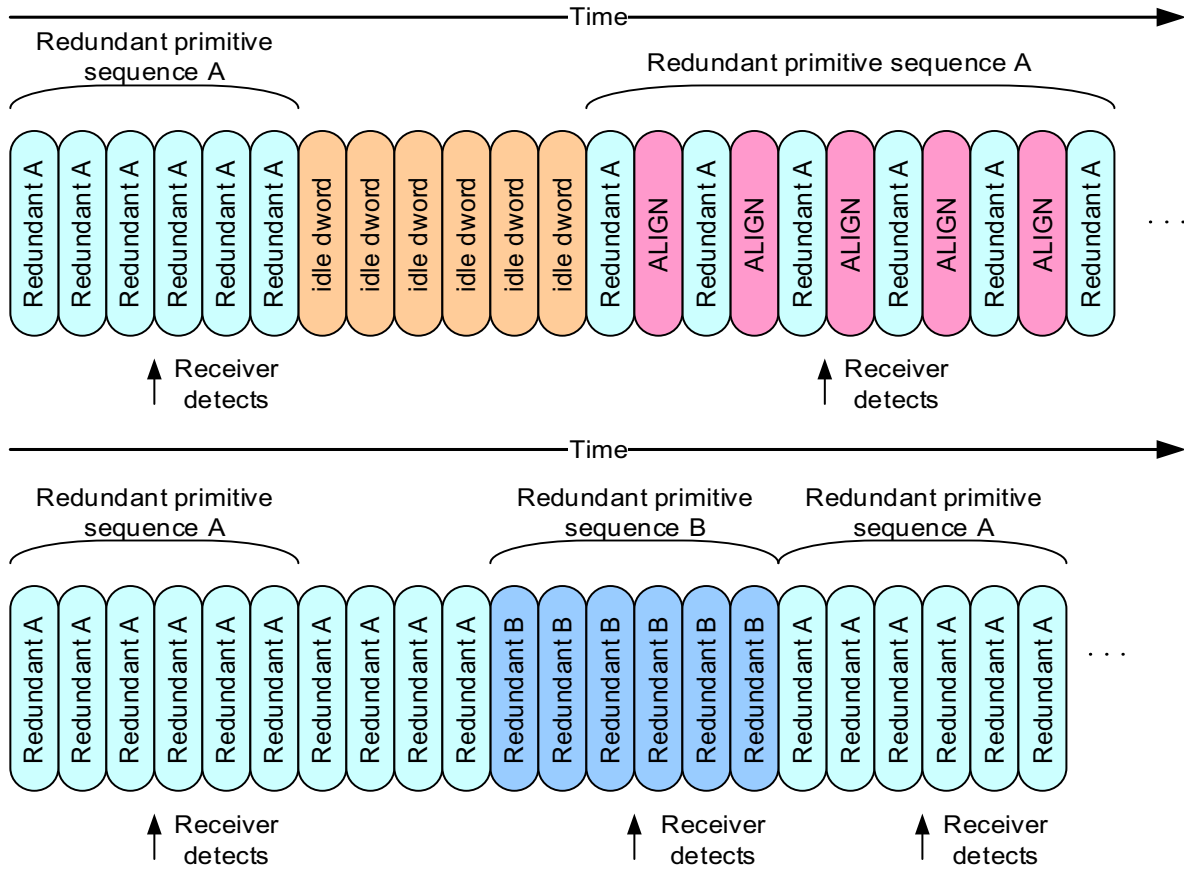
Primitives that form redundant primitive sequences (e.g., BROADCAST (CHANGE)) shall be transmitted six times consecutively. Any number of deletable primitives may be transmitted inside redundant primitive sequences or between primitive segments as described in 6.2.4.1 unless prohibited by the primitives description (e.g., TRAIN, TRAIN\_DONE).

A receiver shall detect a redundant primitive sequence after the identical primitive is received in any three of six consecutive dwords. After detecting a redundant primitive sequence, a receiver shall not detect a second instance of the same redundant primitive sequence until it has received six consecutive dwords that are not any of the following:

- a) the original primitive; or
- b) a deletable primitive.

If the phy is in the SAS packet mode, then a primitive parameter with a length of one dword may follow a redundant primitive sequence. The primitive parameter that follows a redundant primitive sequence shall be contained within the second primitive segment of the redundant primitive sequence (i.e., the redundant primitive sequence plus the associated primitive parameter, if any, shall be contained within two SPL packets where the second SPL packet contains the associated primitive parameter, if any).

Figure 132 shows examples of redundant primitive sequences while a phy is in the SAS dword mode.



**Figure 132 –Redundant primitive sequences while in the SAS dword mode**

The figure consists of two horizontal timelines labeled 'Time' at the top. The top timeline shows a sequence of segments: a pink 'ALIGN (1)' segment, followed by five light blue 'Redundant A' segments, a pink 'ALIGN (2)' segment, an orange 'Scrambled idle segment', another pink 'ALIGN (1)' segment, three light blue 'Redundant A' segments, two orange 'Scrambled idle segment's, three more light blue 'Redundant A' segments, and a final pink 'ALIGN (2)' segment. Brackets group the first six segments as 'Redundant primitive sequence A' and the last six as 'Redundant primitive sequence A'. Vertical arrows labeled 'Receiver detects' point to the first and fourth 'ALIGN (1)' segments. The bottom timeline shows: a pink 'ALIGN (1)' segment, five light blue 'Redundant A' segments, a pink 'ALIGN (2)' segment, a pink 'ALIGN (1)' segment, five light blue 'Redundant B' segments, a green 'Primitive parameter' segment, a pink 'ALIGN (1)' segment, five light blue 'Redundant A' segments, and a pink 'ALIGN (2)' segment. Brackets group the first six segments as 'Redundant primitive sequence A', the next six as 'Redundant primitive sequence B', and the last six as 'Redundant primitive sequence A'. Vertical arrows labeled 'Receiver detects' point to the first, fourth, and seventh 'ALIGN (1)' segments.

**Figure 133 –Redundant primitive sequences while in the SAS packet mode**

### 6.2.5.1 ALIGN

- a) OOB signals (see 5.7);
- b) character and dword alignment during the speed negotiation sequence (see 5.11.4.2);
- c) physical link rate tolerance management after the phy reset sequence (see 6.5); and
- d) rate matching during connections (see 6.17).

ALIGNs are deletable primitives (see 6.5).

Table 129 defines the different versions of ALIGN primitives.

**Table 129 – ALIGN primitives**

Primitive	Description
ALIGN (0)	Used for OOB signals while D.C. mode is enabled and for the speed negotiation sequence, physical link rate tolerance management, phy power condition sequence, and rate matching.
ALIGN (1)	Used for: a) the speed negotiation sequence; b) physical link rate tolerance management; c) phy power condition sequence; d) padding within a primitive segment; and e) rate matching.
ALIGN (2)	Used for: a) physical link rate tolerance management; b) padding within a primitive segment; and c) rate matching.
ALIGN (3)	Used for: a) OOB signals while optical mode is enabled; b) padding within a primitive segment; and c) physical link rate tolerance management, and rate matching.

If D.C. mode is enabled, then phys may use ALIGN (0) to construct OOB signals (see 5.7 and SAS-4). If optical mode is enabled, then phys use ALIGN (3) to construct OOB signals (see 5.7 and SAS-4). Phys use ALIGN (0) and ALIGN (1) during the:

- a) speed negotiation sequence (see 5.11.4.2); and
- b) phy power condition sequence (see 5.13).

If a phy is in SAS dword mode, then that phy shall rotate through transmitting ALIGN (0), ALIGN (1), ALIGN (2), and ALIGN (3) for all ALIGNs transmitted after the phy reset sequence (e.g., if the first ALIGN transmitted after the phy reset sequence is ALIGN (0), then the second transmitted is ALIGN (1), the third transmitted is ALIGN (2), the fourth transmitted is ALIGN (3), the fifth transmitted is ALIGN (0), etc.).

NOTE 16 - ALIGN rotation is performed on a physical phy basis and is used to reduce radiated emissions.

Phys receiving ALIGNs after the phy reset sequence shall not verify the rotation and shall accept any of the ALIGNs at any time.

Phys shall only detect an ALIGN after decoding all four characters in the primitive.

NOTE 17 - SATA devices are allowed to decode every dword starting with a K28.5 as an ALIGN, since ALIGN is the only primitive defined starting with K28.5.

For physical link rate tolerance management and rate matching, ALIGNs may be replaced by NOTIFYs (see 6.2.5.3) or MUXs (see 6.2.5.2). ALIGNs shall not be replaced by NOTIFYs or MUXs during OOB signals or speed negotiation.

#### 6.2.5.2 MUX (Multiplex)

MUX is used if multiplexing (see 5.20) is enabled (see table 72) as follows:

- a) transmitted during the multiplexing sequence (see 5.11.4.3); and



- b) substituted in place of an ALIGN (see 6.2.5.1) being transmitted for physical link rate tolerance management (see 6.5) or rate matching (see 6.17) to confirm the logical link number as defined in 5.20.

Substitution of a MUX for an ALIGN may or may not affect the ALIGN rotation (i.e., the MUX may take the place of one of the ALIGNs in the rotation through ALIGN (0), ALIGN (1), ALIGN (2), and ALIGN (3), or may delay the rotation).

MUXs are deletable primitives (see 6.5). A phy supporting multiplexing shall process MUX primitives in logic running off the received clock without using an elasticity buffer rather than logic after the elasticity buffer, because they are not accompanied by additional deletable primitives (e.g., ALIGNs and/or NOTIFYs).

The versions of MUX are defined in table 130.

**Table 130 – MUX primitives**

Primitive	Description
MUX (LOGICAL LINK 0)	Establishes the position of dwords in logical link 0.
MUX (LOGICAL LINK 1)	Establishes the position of dwords in logical link 1.

See 5.20 for details on multiplexing.

### 6.2.5.3 NOTIFY

#### 6.2.5.3.1 NOTIFY overview

NOTIFY may be substituted in place of any ALIGN (see 6.2.5.1) being transmitted for physical link rate tolerance management (see 6.5) or rate matching (see 6.17). Substitution of a NOTIFY in place of an ALIGN may or may not affect the ALIGN rotation (i.e., the NOTIFY may take the place of one of the ALIGNs in the rotation through ALIGN (0), ALIGN (1), ALIGN (2), and ALIGN (3), or may delay the rotation). A specific NOTIFY shall not be transmitted in more than three consecutive dwords until at least three other dwords have been transmitted.

NOTIFYs are deletable primitives (see 6.5). If a phy supports a specific NOTIFY primitive, then the phy should decode that NOTIFY in logic running off the received clock without using an elasticity buffer rather than logic after the elasticity buffer to avoid missing detection of important information.

NOTIFY shall not be forwarded through expander devices. Expander devices shall substitute an ALIGN for a NOTIFY if necessary.

SAS target devices are not required to detect every transmitted NOTIFY.

The versions of NOTIFY representing different reasons are defined in table 131.

**Table 131 – NOTIFY primitives**

Primitive	Description	Reference
NOTIFY (ENABLE SPINUP)	Specify to a SAS target device that it may temporarily consume additional power (see 9.2.10).	6.2.5.3.2
NOTIFY (POWER LOSS EXPECTED)	Specify to a SAS target device that power loss may occur within the time specified by the POWER LOSS TIMEOUT field in the Shared Port Control mode page (see 9.2.7.6).	6.2.5.3.3
NOTIFY (RESERVED 1)	Reserved.	

NOTIFY (RESERVED 1) shall be ignored by all devices.

#### **6.2.5.3.2 NOTIFY (ENABLE SPINUP)**

NOTIFY (ENABLE SPINUP) is transmitted by a SAS initiator port or expander port and is used to specify to a SAS target device that it may temporarily consume additional power (e.g., to spin up rotating media) (see 9.2.10). The length of time the SAS target device consumes additional power and the amount of additional power is vendor specific. NOTIFY (ENABLE SPINUP) shall interact with the SCSI target device's power condition state transitions, controlled by the Power Conditions mode page (see SPC-4) and/or the START STOP UNIT command (see SBC-3), as described in 9.2.10.

If power control is not enabled, then:

- SAS initiator devices and expander devices shall use NOTIFY (ENABLE SPINUP) while attached to SCSI target devices (i.e., SCSI target devices that report SSP target port support in their IDENTIFY address frames);
- SAS initiator ports and expander ports shall transmit one NOTIFY (ENABLE SPINUP) after power on when the enclosure is ready for initial temporary consumption of additional power; and
- after the initial NOTIFY (ENABLE SPINUP), SAS initiator ports and expander ports shall transmit NOTIFY (ENABLE SPINUP) periodically.

The selection of when and how often to transmit NOTIFY (ENABLE SPINUP) is outside the scope of this standard.

NOTE 18 - The SAS initiator device or expander device uses NOTIFY (ENABLE SPINUP) to avoid exceeding enclosure power supply capabilities during temporary consumption of additional power by multiple SAS target devices.

NOTIFY (ENABLE SPINUP) should be transmitted as frequently as possible to avoid incurring SCSI application layer timeouts.

A SAS target device with multiple SSP target ports shall process a NOTIFY (ENABLE SPINUP) received by any of its SSP target ports (e.g., if a dual-port SAS target device that powers on in the Stopped state receives a START STOP UNIT command with the START bit set to one through one SSP target port followed by a receipt of a NOTIFY (ENABLE SPINUP) on the other SSP target port, then the SAS target device may temporarily consume additional power (see 9.2.10)).

#### **6.2.5.3.3 NOTIFY (POWER LOSS EXPECTED)**

NOTIFY (POWER LOSS EXPECTED) is transmitted by a SAS initiator port or expander port and is used to specify to a SAS target device that power loss may occur within the time specified in the POWER LOSS TIMEOUT field in the Shared Port Control mode page (see 9.2.7.6).

At least three NOTIFY (POWER LOSS EXPECTED) primitive sequences shall be transmitted by the SAS initiator port or expander port.

If a SAS target device supports NOTIFY (POWER LOSS EXPECTED) and receives NOTIFY (POWER LOSS EXPECTED) on an SSP target port, then:

- a) each logical unit to which the SSP target port has access shall:
  - 1) stop writing data to the medium as soon as possible without creating read errors for future reads (e.g., on a direct-access block device, when a physical block boundary is reached); and
  - 2) perform the actions for a power loss expected condition as defined in SAM-5;
 and
- b) the SAS target device shall:
  - A) on each phy that receives NOTIFY (POWER LOSS EXPECTED), if there is an SSP connection, then transmit a BREAK primitive sequence on that connection; and
  - B) on each phy that does not receive NOTIFY (POWER LOSS EXPECTED), if there is an SSP connection, then transmit a BREAK primitive sequence or a CLOSE primitive sequence on that connection.

If the SAS target device receives any frames after receiving NOTIFY (POWER LOSS EXPECTED) before a connection is closed, then it should discard the received frames.

The SCSI application layer that receives a Power Loss Expected event shall:

- a) start the power loss timer;
- b) send an Accept\_Reject OPENs (Reject SSP) request to all ST\_T state machines (i.e., all SSP connection requests result in OPEN\_REJECT (RETRY));
- c) if a SCSI Command Received transport protocol service indication is received, then the SCSI device server shall abort that command and send an Accept\_Reject OPENs (Reject SSP) request to the ST\_T state machine on which the SCSI Command Received transport protocol service indication was received; and
- d) if the power loss timeout timer expires, then the SCSI application layer shall send an Accept\_Reject OPENs (Accept SSP) request to all ST\_T state machines.

After power on, the power loss timeout timer shall be initialized and stopped until a NOTIFY (POWER LOSS EXPECTED) is received.

#### 6.2.5.4 OOB\_IDLE

OOB\_IDLE (see 5.14.4.2 and SAS-4) is used while a phy is in optical mode for:

- a) OOB signals; and
- b) speed negotiation.

### 6.2.6 Primitives not specific to type of connections

#### 6.2.6.1 AIP (Arbitration in progress)

AIP is transmitted by an expander device after a connection request to specify that the connection request is being processed and specify the status of the connection request.

The versions of AIP representing different statuses are defined in table 132.

**Table 132 – AIP primitives**

Primitive	Description
AIP (NORMAL)	Expander device has accepted the connection request. This primitive may be transmitted multiple times (see 6.16.5.3).
AIP (RESERVED 0)	Reserved. Processed the same as AIP (NORMAL).
AIP (RESERVED 1)	
AIP (RESERVED 2)	
AIP (WAITING ON CONNECTION)	Expander device has determined the routing for the connection request, but either the destination phys are all being used for connections or there are insufficient routing resources to complete the connection request. This may be transmitted multiple times (see 6.16.5.3).
AIP (WAITING ON DEVICE)	Expander device has determined the routing for the connection request and forwarded the request to the output physical link. This is transmitted one time (see 6.16.5.3).
AIP (WAITING ON PARTIAL)	Expander device has determined the routing for the connection request, but the destination phys are all busy with other partial pathways. This primitive may be transmitted multiple times (see 6.16.5.3).
AIP (RESERVED WAITING ON PARTIAL)	Reserved. Processed the same as AIP (WAITING ON PARTIAL).

See 6.16 for details on connections.

#### **6.2.6.2 BREAK**

BREAK is used to abort a connection request or break a connection.

See 6.16.7 and 6.16.11 for details on breaking connections.

#### **6.2.6.3 BREAK\_REPLY**

BREAK\_REPLY is used to confirm the receipt of a BREAK.

See 6.16.7 and 6.16.11 for details on breaking connections.

#### **6.2.6.4 BROADCAST**

BROADCASTs are used to notify SAS ports and expander devices in a SAS domain about certain events.

The versions of BROADCAST representing different Broadcasts (see table 15 in 4.1.15) are defined in table 133.

**Table 133 – BROADCAST primitives**

Primitive	Description
BROADCAST (CHANGE)	Broadcast (Change)
BROADCAST (RESERVED CHANGE 0)	Broadcast (Reserved Change 0)
BROADCAST (RESERVED CHANGE 1)	Broadcast (Reserved Change 1)
BROADCAST (SES)	Broadcast (SES)
BROADCAST (EXPANDER)	Broadcast (Expander)
BROADCAST (ASYNCHRONOUS EVENT)	Broadcast (Asynchronous Event)
BROADCAST (RESERVED 3)	Broadcast (Reserved 3)
BROADCAST (RESERVED 4)	Broadcast (Reserved 4)

A phy that has not completed the link reset sequence or a phy inside a connection shall:

- a) not transmit a BROADCAST; and
- b) ignore any received BROADCAST.

#### 6.2.6.5 CLOSE

##### 6.2.6.5.1 CLOSE overview

CLOSE is used to close a connection.

The versions of CLOSE representing different reasons are defined in table 134.

**Table 134 – CLOSE primitives**

Primitive	Description
CLOSE (CLEAR AFFILIATION)	Close an open STP connection and clear the affiliation (see 6.21.6). Processed the same as CLOSE (NORMAL) if: <ul style="list-style-type: none"> <li>a) the connection is not an STP connection;</li> <li>b) the connection is an STP connection, but affiliations are not implemented by the STP target port; or</li> <li>c) the connection is an STP connection, but an affiliation is not present.</li> </ul>
CLOSE (NORMAL)	Close a connection.
CLOSE (RESERVED 0)	Reserved. Processed the same as CLOSE (NORMAL).
CLOSE (RESERVED 1)	

See 6.16.9 for details on closing connections.

### 6.2.6.5.2 CLOSE primitive parameter

The CLOSE primitive parameter may be used by an expander device to send the extended fairness priority (see 6.19.10) to an attached expander device.

The contents of the CLOSE primitive parameter shall be:

- a) associated with a single OPEN address frame for which an expander device is arbitrating for access to an outgoing expander port; and
- b) set based on which of the following has the higher priority (see 6.19.10) of:
  - A) a received CLOSE primitive parameter (see 6.2.6.5.4); and
  - B) an OPEN address frame that an expander device is arbitrating for access to an outgoing expander port (see 6.2.6.5.3).

Expander devices that do not support CLOSE primitive parameters and end devices shall ignore the contents of the primitive segment's dwords after the CLOSE (i.e., byte four to byte 15 of the primitive segment).

Table 135 defines the primitive parameter format for the CLOSE.

**Table 135 – CLOSE primitive parameter format**

Byte\ Bit	7	6	5	4	3	2	1	0
n	HOP COUNT				PARAMETER LENGTH (11b)		CONTROL 1 (10b)	
n+1	HIGH PRIORITY	SMP OPEN PRIORITY	Reserved		OPEN CONNECTION RATE			
n+2	(MSB)							
n+3	OPEN ARBITRATION WAIT TIME							
n+4	(MSB)							
...	OPEN DESTINATION SAS ADDRESS							
n+11	(LSB)							

### 6.2.6.5.3 CLOSE primitive parameter fields when being set from an open address frame

The CONTROL 1 field and the PARAMETER LENGTH field are defined in 5.5.4 and shall be set as shown in table 135 for the CLOSE primitive parameter.

The HOP COUNT field is a count of the number of expander devices the CLOSE has passed through without any change to the CLOSE primitive parameter fairness information (see 6.19.10) and is a saturating counter.

The OPEN CONNECTION RATE field is set to contents of the CONNECTION RATE field (see 6.10.3) of the OPEN address frame received by an expander device that the expander device is prevented from transmitting as a result of resource limitations (e.g., all phys have active connections) (see 6.16.5.3).

The SMP OPEN PRIORITY bit is set to one when the SAS PROTOCOL field (see 6.10.3) of the OPEN address frame received by an expander device is set to SMP. The SMP OPEN PRIORITY bit is set to zero when the SAS PROTOCOL field is not set to SMP.

The HIGH PRIORITY bit is set to the contents of the High Priority argument as described in (see 6.16.5.1)

The OPEN ARBITRATION WAIT TIME field is set to the contents of the Arbitration Wait Time state machine variable as described in (see 6.16.5.2.1).

The OPEN DESTINATION SAS ADDRESS field is set to the contents of the DESTINATION SAS ADDRESS field (see 6.10.3) of the OPEN address frame received by an expander device that the expander device is prevented from transmitting as a result of resource limitations (e.g., all phys have active connections) (see 6.16.5.2).

#### **6.2.6.5.4 CLOSE primitive parameter fields when being set from a received CLOSE with a primitive parameter**

The CONTROL 1 field and the PARAMETER LENGTH field are defined in 5.5.4 and shall be set as shown in table 135 for the CLOSE primitive parameter.

The HOP COUNT field is a count of the number of expander devices the CLOSE has passed through without any change to the CLOSE primitive parameter fairness information (see 6.19.10) and is a saturating counter.

The OPEN CONNECTION RATE field, SMP OPEN PRIORITY bit, HIGH PRIORITY bit, OPEN ARBITRATION WAIT TIME field, and OPEN DESTINATION SAS ADDRESS field are set to the contents of the corresponding field of the received CLOSE primitive parameter (see 6.2.6.5.3).

#### **6.2.6.6 EOAF (End of address frame)**

EOAF specifies the end of an address frame.

See 6.10 for details on address frames.

#### **6.2.6.7 ERROR**

ERROR should be transmitted by an expander device while the expander device is forwarding dwords from a SAS physical link or SATA physical link to a SAS physical link and receives an invalid dword or an ERROR.

Since an 8b10b coding error in one dword is sometimes not detected until the next dword (see table 52 in 5.3.9), expander devices should avoid deleting invalid dwords or ERRORS unless necessary (e.g., if the elasticity buffer is full) to avoid hiding evidence that an error has occurred.

See 6.19 for details on error handling by expander devices.

#### **6.2.6.8 HARD\_RESET**

HARD\_RESET is used to force a phy to generate a hard reset (see 4.4.2) to its port. This primitive is only valid after the phy reset sequence without an intervening identification sequence (see 4.4) and shall be ignored at other times.

#### **6.2.6.9 OPEN\_ACCEPT**

OPEN\_ACCEPT specifies the acceptance of a connection request.

See 6.16 for details on connection requests.

#### **6.2.6.10 OPEN\_REJECT**

##### **6.2.6.10.1 OPEN\_REJECT overview**

OPEN\_REJECT specifies that a connection request has been rejected and specifies the reason for the rejection. The result of some OPEN\_REJECTs is to abandon (i.e., not retry) the connection request; and the result of other OPEN\_REJECTs is to retry the connection request.

All of the OPEN\_REJECT versions defined in table 136 shall result in the originating port abandoning the connection request.

**Table 136 – Abandon-class OPEN\_REJECT primitives**

Primitive	Originator	Description
OPEN_REJECT (BAD DESTINATION)	Expander phy	A connection request routes to a destination expander phy in the same expander port as the source expander phy, and the expander port is using the direct routing method (see 4.5.7.1).
OPEN_REJECT (CONNECTION RATE NOT SUPPORTED)	Any phy	<p>The requested connection rate is not supported on some physical link on the pathway between the source phy and destination phy. If a SAS initiator phy is directly attached to a SAS target phy, then the requested connection rate is not supported by the destination phy.</p> <p>If the connection rate is 1.5 Gbit/s, then this shall be considered an abandon-class OPEN_REJECT.</p> <p>If the connection rate is greater than 1.5 Gbit/s, then the connection request shall be modified and reattempted as described in 6.10.3.</p>
OPEN_REJECT (PROTOCOL NOT SUPPORTED)	Any phy	Phy with destination SAS address exists, but the destination phy does not support the requested initiator role, target role, protocol, initiator connection tag, or features (i.e., the values in the INITIATOR PORT bit, the SAS PROTOCOL field, the INITIATOR CONNECTION TAG field, and/or the FEATURES field in the OPEN address frame are not supported).
OPEN_REJECT (RESERVED ABANDON 1)	Unknown	Reserved. Process the same as OPEN_REJECT (WRONG DESTINATION).
OPEN_REJECT (RESERVED ABANDON 2)		
OPEN_REJECT (RESERVED ABANDON 3)		
OPEN_REJECT (STP RESOURCES BUSY)	Destination phy	STP target port with destination SAS address exists, but the STP target port supports affiliations and is not able to establish an affiliation with this STP initiator port (e.g., because it has reached its maximum number of affiliations), or the STP target port does not support affiliations and all of the available affiliation contexts have been allocated to other STP initiator ports (see 6.21.6). Process the same as OPEN_REJECT (WRONG DESTINATION) for non-STP connection requests.
OPEN_REJECT (WRONG DESTINATION)	Destination phy	The destination SAS address does not match the SAS address of the SAS port to which the connection request was delivered.
OPEN_REJECT (ZONE VIOLATION)	Zoning expander phy	The connection request is from a zone group that does not have permission to access the zone group that contains the destination phy according to the zone permission table of an unlocked zoning expander device.



All of the OPEN\_REJECT versions defined in table 137 shall result in the originating port retrying the connection request.

**Table 137 – Retry-class OPEN\_REJECT primitives**

Primitive	Originator	Description
OPEN_REJECT (NO DESTINATION) <sup>a</sup>	Expander phy	An expander device in the pathway is not configuring (see 4.6.4) and determines that: a) there is no such destination phy; b) the connection request routes to a destination expander phy in the same expander port as the source expander phy and the expander port is using the subtractive routing method; or c) the SAS address is valid for an STP target port in an STP SATA bridge, but the initial Register - Device to Host FIS has been received with an error (see 9.4.3.12).
OPEN_REJECT (PATHWAY BLOCKED) <sup>b</sup>	Expander phy	An expander device determined the pathway was blocked by higher priority connection requests.
OPEN_REJECT (RESERVED CONTINUE 0) <sup>c</sup>	Unknown	Reserved. Process the same as OPEN_REJECT (RETRY).
OPEN_REJECT (RESERVED CONTINUE 1) <sup>c</sup>		
OPEN_REJECT (RESERVED INITIALIZE 0) <sup>a</sup>	Unknown	Reserved. Process the same as OPEN_REJECT (NO DESTINATION).
OPEN_REJECT (RESERVED INITIALIZE 1) <sup>a</sup>		
OPEN_REJECT (RESERVED STOP 0) <sup>b</sup>	Unknown	Reserved. Process the same as OPEN_REJECT (PATHWAY BLOCKED).
OPEN_REJECT (RESERVED STOP 1) <sup>b</sup>		
OPEN_REJECT (RETRY) <sup>c</sup>	Destination phy or zoning expander phy	Either: a) a phy with the destination SAS address exists but is temporarily not able to accept connections (see 6.20.1, 6.21.7, and 6.22.4); b) an expander device in the pathway is configuring (see 4.6.4) and detects a connection that results in an OPEN_REJECT (NO DESTINATION) if the condition is not resolved (see 4.6.4 and 6.16.5.2.5); c) an expander device in the pathway is locked and detects a connection that results in an OPEN_REJECT (ZONE VIOLATION) if the condition is not resolved (see 4.8.3.5 and 6.16.5.2.5); d) an expander device in the pathway has reduced functionality (see 4.5.8 and 6.16.5.2.5); or e) a phy with the destination SAS address exists but is in the slumber phy power condition (see 4.10.1.6).
<sup>a</sup> If the I_T Nexus Loss timer (see 7.2.2) is already running, then it continues running. If it is not already running, then it is initialized and started. Stop retrying the connection request if the I_T Nexus Loss timer expires. <sup>b</sup> If the I_T Nexus Loss timer is already running, then it continues running. Stop retrying the connection request if the I_T Nexus Loss timer expires. <sup>c</sup> If the I_T Nexus Loss timer is already running, then it is stopped.		

NOTE 19 - Some SAS logical phys compliant with earlier versions of this standard also transmit OPEN\_REJECT (RETRY) if they receive an OPEN address frame while their SL\_CC state machines are in the SL\_CC5:BreakWait state (see 6.18.4.7).

When a SAS logical phy detects more than one reason to transmit an OPEN\_REJECT, the SL\_CC state machine determines the priority in the SL\_CC2:Selected state (see 6.18.4.4).

When an expander logical phy detects more than one reason to transmit an OPEN\_REJECT, the ECM determines the priority (see 6.16.5).

See 6.16 for details on connection requests.

#### 6.2.6.10.2 OPEN\_REJECT retry-class primitive parameter

OPEN\_REJECT retry-class primitives (see table 137) may have an associated OPEN\_REJECT retry-class primitive parameter.

The OPEN\_REJECT retry-class primitive parameter may be used by an end device or an expander device to request the originator of a rejected OPEN address frame delay at least the time specified by the OPEN\_REJECT retry-class primitive parameter before initiating another OPEN address frame to the same SAS destination address.

End devices that do not support OPEN\_REJECT retry-class primitive parameters shall ignore the contents of the next primitive segment's dword after the OPEN\_REJECT (see 5.5.4) if that dword's:

- a) CONTROL1 field is set to 10b (i.e., byte four to byte seven of the primitive segment);
- b) CONTROL2 field is set to 10b (i.e., byte eight to byte 11 of the primitive segment); or
- c) CONTROL3 field is set to 10b (i.e., byte 12 to byte 15 of the primitive segment).

Table 138 defines the primitive parameter format for the OPEN\_REJECT retry-class primitives.

**Table 138 – OPEN\_REJECT retry-class primitive parameter format**

Byte\ Bit	7	6	5	4	3	2	1	0
n	Reserved		TIME SCALE		PARAMETER LENGTH (01b)		CONTROL1 (10b) CONTROL2 (10b) CONTROL3 (10b)	
n+1	Reserved							
n+2	(MSB)							
n+3	OPEN RETRY DELAY (LSB)							

The CONTROL1 field, CONTROL2 field, or CONTROL3 field, and the PARAMETER LENGTH field are defined in 5.5.4 and shall be set as shown in table 138 for the OPEN\_REJECT retry-class primitive parameter.

The TIME SCALE field (see table 139) indicates the time units for the contents of the OPEN RETRY DELAY field.

**Table 139 – TIME SCALE field**

Code	Description
00b	100 ns
01b	10 $\mu$ s
10b	1 000 $\mu$ s
11b	Reserved

The OPEN\_RETRY\_DELAY field indicates the minimum time, in units indicated by the TIME\_SCALE field, that a SAS port that receives an OPEN\_REJECT should wait to establish a connection request with an associated SAS port on the I\_T nexus that received the OPEN\_REJECT. An OPEN\_RETRY\_DELAY field set to 0000h indicates that the Reject To Open Limit timer is set as described in table 202.

This Reject To Open Limit time is enforced by the port layer (see 7.2.2).

#### 6.2.6.11 PS\_ACK

PS\_ACK specifies the positive acknowledgement of a PS\_REQ. See 4.10, 6.13, 6.18, and 6.19 for details on phy power conditions.

#### 6.2.6.12 PS\_NAK

PS\_NAK specifies the negative acknowledgement of a PS\_REQ. See 4.10, 6.13, 6.18, and 6.19 for details on phy power conditions.

A PS\_NAK occurs for the following reasons:

- a) a PS\_REQ is received and the requested low phy power condition is supported and disabled; or
- b) a PS\_REQ is received during an attempt to establish a connection.

#### 6.2.6.13 PS\_REQ

PS\_REQ is used to request a transition to a specific low phy power condition (see 4.10.1).

The versions of PS\_REQ representing different low phy power conditions are defined in table 140.

**Table 140 – PS\_REQ primitives**

Primitive	Description
PS_REQ (PARTIAL)	Requests a transition into the partial phy power condition (see 4.10.1.3).
PS_REQ (SLUMBER)	Requests a transition into the slumber phy power condition (see 4.10.1.4).

#### 6.2.6.14 PWR\_ACK

PWR\_ACK specifies the positive acknowledgement of PWR\_DONE, PWR\_GRANT, or PWR\_REQ (see 6.14).

#### 6.2.6.15 PWR\_DONE

PWR\_DONE is used by an end device to indicate it has completed consumption of the additional power requested as a result of a PWR\_GRANT (see 6.14).

#### **6.2.6.16 PWR\_GRANT**

PWR\_GRANT is used to specify that an end device may consume additional power (see 6.14).

#### **6.2.6.17 PWR\_REQ**

PWR\_REQ is used by an end device to request the consumption of additional power (see 6.14).

#### **6.2.6.18 SOAF (Start of address frame)**

SOAF specifies the start of an address frame.

See 6.10 for details on address frames.

#### **6.2.6.19 TRAIN**

TRAIN is used during Train\_Rx-SNW during speed negotiation. Deletable primitives shall not be transmitted inside a TRAIN primitive sequence.

See 5.11.4.2.3.5 for details on Train\_Rx-SNW.

#### **6.2.6.20 TRAIN\_DONE**

TRAIN\_DONE is used during Train\_Rx-SNW during speed negotiation. Deletable primitives shall not be transmitted inside a TRAIN\_DONE primitive sequence.

See 5.11.4.2.3.5 for details on Train\_Rx-SNW.

### **6.2.7 Primitives used only inside SSP and SMP connections**

#### **6.2.7.1 ACK (Acknowledge)**

ACK specifies the positive acknowledgement of an SSP frame.

See 6.20.3 for details on SSP frame transmission.

ACK is not used during an SMP connection.

#### **6.2.7.2 CREDIT\_BLOCKED**

CREDIT\_BLOCKED specifies that no more RRDYs are going to be transmitted during this SSP connection (i.e., transmit SSP frame credit is not going to be increased).

See 6.20.4 for details on SSP flow control.

CREDIT\_BLOCKED is not used during an SMP connection.

#### **6.2.7.3 DONE**

DONE is used to start closing an SSP connection and specify a reason for doing so.

The versions of DONE representing different reasons are defined in table 141. The SSP state machines describe when these are used (see 6.20.9).

**Table 141 – DONE primitives**

Primitive	Description
DONE (ACK/NAK TIMEOUT)	The SSP state machines (see 6.20.9) timed out waiting for an ACK or NAK, and the phy is going to transmit BREAK primitive sequence in 1 ms unless DONE is received within 1 ms of transmitting the DONE (ACK/NAK TIMEOUT).
DONE (RESERVED TIMEOUT 0)	Reserved. Processed the same as DONE (ACK/NAK TIMEOUT).
DONE (RESERVED TIMEOUT 1)	
DONE (NORMAL)	Finished transmitting all frames.
DONE (CLOSE)	The SSP state machine (see 6.20.9) has received an RRDY (CLOSE).
DONE (RESERVED 0)	Reserved. Processed the same as DONE (NORMAL).
DONE (CREDIT TIMEOUT)	The SSP state machines (see 6.20.9) timed out waiting for an RRDY or received a CREDIT BLOCKED, and the phy is going to transmit BREAK primitive sequence unless the phy receives a frame or a DONE within 1 ms of transmitting the DONE (CREDIT TIMEOUT).

See 6.20.8 for details on closing SSP connections.

#### 6.2.7.4 EOF (End of frame)

If the phy is in the SAS dword mode, then EOF specifies the end of an SSP or SMP frame.

See 6.20.3 for details on SSP frame transmission and 6.22.1 for details on SMP frame transmission.

#### 6.2.7.5 EXTEND\_CONNECTION

EXTEND\_CONNECTION (NORMAL) is used to:

- a) establish a persistent connection (see 4.1.13);
- b) continue a persistent connection; and
- c) indicate that an SSP port has no SSP frames to send.

EXTEND\_CONNECTION (NORMAL) is not used during an SMP connection.

EXTEND\_CONNECTION (CLOSE) may be originated by an expander port to initiate the closing of a persistent connection. EXTEND\_CONNECTION (CLOSE) is not used during an SMP connection.

#### 6.2.7.6 NAK (Negative acknowledgement)

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

The versions of NAK representing different reasons are defined in table 142.

**Table 142 – NAK primitives**

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

See 6.20.3 for details on SSP frame transmission.

NAK is not used during an SMP connection.

#### 6.2.7.7 RRDY (Receiver ready)

RRDY is used to increase SSP frame credit.

The versions of RRDY representing different reasons are defined in table 143.

**Table 143 – RRDY primitives**

Primitive	Description
RRDY (NORMAL)	Increase transmit SSP frame credit by one.
RRDY (CLOSE)	Increase transmit SSP frame credit by one and transmit a DONE (CLOSE).
RRDY (RESERVED 0)	Reserved. Processed the same as RRDY (NORMAL).

A phy shall not transmit RRDY after transmitting CREDIT\_BLOCKED in a connection. See 6.20.4 for details on SSP flow control.

RRDY is not used during an SMP connection.

#### 6.2.7.8 SOF (Start of frame)

SOF specifies the start of an SSP or SMP frame.

See 6.20.3 for details on SSP frame transmission and 6.22.1 for details on SMP frame transmission.

### 6.2.8 Primitives used only inside STP connections and on SATA physical links

#### 6.2.8.1 SATA\_ERROR

SATA\_ERROR should be transmitted by an expander device when it is forwarding dwords from a SAS logical link to a SATA physical link and it receives an invalid dword or an ERROR.

Since an 8b10b coding error in one dword is sometimes not detected until the next dword (see table 52 in 5.3.9), expander devices should avoid deleting invalid dwords or ERRORS unless necessary (e.g., if the elasticity buffer is full) to avoid hiding evidence that an error has occurred.

See 5.15 for details on error handling by expander devices.

Although included in this subclause, SATA\_ERROR is not a primitive since it starts with K28.6. SATA\_ERROR does not appear inside STP connections. SATA\_ERROR is an invalid dword.

#### **6.2.8.2 SATA\_PMACK, SATA\_PMNAK, SATA\_PMREQ\_P, and SATA\_PMREQ\_S (Power management acknowledgements and requests)**

SATA\_PMREQ\_P and SATA\_PMREQ\_S request entry into the partial interface power management sequence and slumber interface power management sequence (see SATA). SATA\_PMACK is used to accept a power management request. SATA\_PMNAK is used to reject an interface power management request.

#### **6.2.8.3 SATA\_HOLD and SATA\_HOLD\_A (Hold and hold acknowledge)**

See 6.21.4 for rules on STP flow control, which uses SATA\_HOLD and SATA\_HOLD\_A.

#### **6.2.8.4 SATA\_R\_RDY and SATA\_X\_RDY (Receiver ready and transmitter ready)**

When a SATA port has a frame to transmit, it transmits SATA\_X\_RDY and waits for SATA\_R\_RDY before transmitting the frame.

#### **6.2.8.5 SATA\_EOF (End of frame)**

If the phy is in the SAS packet mode, then the expander device substitutes B\_EOF for SATA\_EOF to specify the end of an STP frame as described in 6.21.3.

#### **6.2.8.6 Other primitives used inside STP connections and on SATA physical links**

Other primitives used in STP connections and on SATA physical links are defined in SATA.

## 6.3 Binary primitives

### 6.3.1 Binary primitives overview

Table 144 defines the deletable binary primitives.

**Table 144 – Deletable binary primitives** (part 1 of 2)

Binary primitive	Use <sup>a</sup>	From <sup>b</sup>			To <sup>b</sup>			Binary primitive sequence type <sup>c</sup>
		I	E	T	I	E	T	
APTA_ADJUST (COMPLETE)	NoConn	I	E	T	I	E	T	Single
APTA_ADJUST (READY)								
APTA_ADJUST (START)								
APTA_ADJUST (TERMINATE)								
APTA_ADJUST (RESERVED 1)								
APTA_ADJUST (RESERVED 2)								
APTA_ADJUST (RESERVED 3)								
APTA_COEFFICIENT_1 (DECREMENT)	NoConn	I	E	T	I	E	T	Single
APTA_COEFFICIENT_1 (INCREMENT)								
APTA_COEFFICIENT_1 (MAXIMUM)								
APTA_COEFFICIENT_1 (MINIMUM)								
APTA_COEFFICIENT_1 (UPDATED)								
APTA_COEFFICIENT_1 (RESERVED 1)								
APTA_COEFFICIENT_2 (DECREMENT)								
APTA_COEFFICIENT_2 (INCREMENT)								
APTA_COEFFICIENT_2 (MAXIMUM)								
APTA_COEFFICIENT_2 (MINIMUM)								
APTA_COEFFICIENT_2 (UPDATED)								
APTA_COEFFICIENT_2 (RESERVED 1)								

Key:

NoConn = SAS logical links, outside connections

I = SAS initiator ports

E = expander ports

T = SAS target ports

<sup>a</sup> The Use column indicates when the binary primitive is used.

<sup>b</sup> The From and To columns indicate the type of ports that originate each binary primitive or are the intended destinations of each binary primitive.  
Expander ports are not considered originators of binary primitives that are being forwarded from expander port to expander port within an expander.

<sup>c</sup> The Binary primitive sequence type columns indicate whether the binary primitive is a single binary primitive sequence, a triple binary primitive sequence, or a redundant binary primitive sequence (see 6.3.3).



Table 144 – Deletable binary primitives (part 2 of 2)

Binary primitive	Use <sup>a</sup>	From <sup>b</sup>			To <sup>b</sup>			Binary primitive sequence type <sup>c</sup>
		I	E	T	I	E	T	
APTA_COEFFICIENT_3 (DECREMENT)	NoConn	I	E	T	I	E	T	Single
APTA_COEFFICIENT_3 (INCREMENT)								
APTA_COEFFICIENT_3 (MAXIMUM)								
APTA_COEFFICIENT_3 (MINIMUM)								
APTA_COEFFICIENT_3 (UPDATED)								
APTA_COEFFICIENT_3 (RESERVED 1)								
APTA_COEFFICIENT_1_2 (DECREMENT)	NoConn	I	E	T	I	E	T	Single
APTA_COEFFICIENT_1_2 (INCREMENT)								
APTA_COEFFICIENT_1_2 (MAXIMUM)								
APTA_COEFFICIENT_1_2 (MINIMUM)								
APTA_COEFFICIENT_1_2 (UPDATED)								
APTA_COEFFICIENT_1_2 (RESERVED 1)								
APTA_COEFFICIENT_1_2 (DECREMENT)	NoConn	I	E	T	I	E	T	Single
APTA_COEFFICIENT_2_3 (INCREMENT)								
APTA_COEFFICIENT_2_3 (MAXIMUM)								
APTA_COEFFICIENT_2_3 (MINIMUM)								
APTA_COEFFICIENT_2_3 (UPDATED)								
APTA_COEFFICIENT_2_3 (RESERVED 1)								
Key:								
NoConn = SAS logical links, outside connections								
I = SAS initiator ports								
E = expander ports								
T = SAS target ports								
<sup>a</sup> The Use column indicates when the binary primitive is used.								
<sup>b</sup> The From and To columns indicate the type of ports that originate each binary primitive or are the intended destinations of each binary primitive. Expander ports are not considered originators of binary primitives that are being forwarded from expander port to expander port within an expander.								
<sup>c</sup> The Binary primitive sequence type columns indicate whether the binary primitive is a single binary primitive sequence, a triple binary primitive sequence, or a redundant binary primitive sequence (see 6.3.3).								

Table 145 defines the binary primitives used only inside SSP connections and STP connections.

**Table 145 – Binary primitives used inside SSP connections and STP connections**

Binary primitive	Use <sup>a</sup>	From <sup>b</sup>			To <sup>b</sup>			Binary primitive sequence type <sup>c</sup>
		I	E	T	I	E	T	
B_EOF (0)	SSP, STP	I	E	T	I	E	T	Single
B_EOF (0) (RESERVED 1)								
B_EOF (1)								
B_EOF (1) (RESERVED 1)								
B_EOF (1) (RESERVED 2)								
B_EOF (2)								
B_EOF (2) (RESERVED 1)								
B_EOF (2) (RESERVED 2)								
B_EOF (3)								
B_EOF (3) (RESERVED 1)								
B_EOF (3) (RESERVED 2)								
Key:								
SAS = SAS logical links, both outside connections <del>or</del> and inside any type of connection								
I = SAS initiator ports								
E = expander ports								
T = SAS target ports								
<sup>a</sup> The Use column indicates when the binary primitive is used.								
<sup>b</sup> The From and To columns indicate the type of ports that originate each binary primitive or are the intended destinations of each binary primitive. Expander ports are not considered originators of binary primitives that are being forwarded from expander port to expander port within an expander.								
<sup>c</sup> The Binary primitive sequence type columns indicate whether the binary primitive is a single binary primitive sequence, a triple binary primitive sequence, or a redundant binary primitive sequence (see 6.3.3).								

## 6.3.2 Binary primitive codes

Table 146 defines the binary primitive codes for deletable binary primitives.

Table 146 – Binary primitive codes for deletable binary primitives

Binary primitive	Byte				Hexadecimal
	0 (first)	1	2	3 (last)	
APTA_ADJUST (COMPLETE)	0Dh	F1h	55h	65h	0DF15565h
APTA_ADJUST (READY)	0Dh	F1h	59h	99h	0DF15999h
APTA_ADJUST (START)	0Dh	F1h	A5h	59h	0DF1A559h
APTA_ADJUST (TERMINATE)	0Dh	F1h	A9h	A5h	0DF1A9A5h
APTA_ADJUST (RESERVED 1)	0Dh	FDh	01h	3Dh	0DFD013Dh
APTA_ADJUST (RESERVED 2)	0Dh	FDh	0Dh	C1h	0DFD0DC1h
APTA_ADJUST (RESERVED 3)	0Dh	FDh	F1h	01h	0DFDF101h
APTA_COEFFICIENT_1 (DECREMENT)	01h	F1h	3Dh	3Dh	01F13D3Dh
APTA_COEFFICIENT_1 (INCREMENT)	01h	F1h	C1h	FDh	01F1C1FDh
APTA_COEFFICIENT_1 (MAXIMUM)	01h	FDh	65h	99h	01FD6599h
APTA_COEFFICIENT_1 (MINIMUM)	01h	FDh	69h	65h	01FD6965h
APTA_COEFFICIENT_1 (UPDATED)	01h	FDh	95h	A5h	01FD95A5h
APTA_COEFFICIENT_1 (RESERVED 1)	01h	FDh	99h	59h	01FD9959h
APTA_COEFFICIENT_2 (DECREMENT)	09h	B1h	6Dh	D5h	09B16DD5h
APTA_COEFFICIENT_2 (INCREMENT)	09h	B1h	9Dh	E9h	09B19DE9h
APTA_COEFFICIENT_2 (MAXIMUM)	09h	BDh	35h	4Dh	09BD354Dh
APTA_COEFFICIENT_2 (MINIMUM)	09h	BDh	39h	B1h	09BD39B1h
APTA_COEFFICIENT_2 (UPDATED)	09h	BDh	C5h	71h	09BDC571h
APTA_COEFFICIENT_2 (RESERVED 1)	09h	BDh	C9h	8Dh	09BDC98Dh
APTA_COEFFICIENT_3 (DECREMENT)	0Dh	C1h	0Dh	FDh	0DC10DFDh
APTA_COEFFICIENT_3 (INCREMENT)	0Dh	C1h	F1h	3Dh	0DC1F13Dh
APTA_COEFFICIENT_3 (MAXIMUM)	0Dh	C1h	FDh	C1h	0DC1FDC1h
APTA_COEFFICIENT_3 (MINIMUM)	0Dh	CDh	55h	59h	0DCD5559h
APTA_COEFFICIENT_3 (UPDATED)	0Dh	CDh	59h	A5h	0DCD59A5h
APTA_COEFFICIENT_3 (RESERVED 1)	0Dh	CDh	A5h	65h	0DCDA565h
APTA_COEFFICIENT_1_2 (DECREMENT)	0Dh	D5h	C5h	A9h	0DD5C5A9h
APTA_COEFFICIENT_1_2 (INCREMENT)	0Dh	D5h	C9h	55h	0DD5C955h
APTA_COEFFICIENT_1_2 (MAXIMUM)	0Dh	D9h	61h	CDh	0DD961CDh
APTA_COEFFICIENT_1_2 (MINIMUM)	0Dh	D9h	6Dh	31h	0DD96D31h
APTA_COEFFICIENT_1_2 (UPDATED)	0Dh	D9h	91h	F1h	0DD991F1h
APTA_COEFFICIENT_1_2 (RESERVED 1)	0Dh	D9h	9Dh	0Dh	0DD99D0Dh
APTA_COEFFICIENT_2_3 (DECREMENT)	0Dh	E5h	61h	F1h	0DE561F1h
APTA_COEFFICIENT_2_3 (INCREMENT)	0Dh	E5h	6Dh	0Dh	0DE56D0Dh
APTA_COEFFICIENT_2_3 (MAXIMUM)	0Dh	E5h	91h	CDh	0DE591CDh
APTA_COEFFICIENT_2_3 (MINIMUM)	0Dh	E5h	9Dh	31h	0DE59D31h
APTA_COEFFICIENT_2_3 (UPDATED)	0Dh	E9h	C5h	95h	0DE9C595h
APTA_COEFFICIENT_2_3 (RESERVED 1)	0Dh	E9h	C9h	69h	0DE9C969h

Table 147 defines the binary primitive codes for binary primitives used only inside SSP and STP connections.

**Table 147 – Binary primitive codes for binary primitives only used inside SSP and STP connections-**

Binary primitive	Byte				Hexadecimal
	0 (first)	1	2	3 (last)	
B_EOF (0)	31h	19h	71h	DDh	311971DDh
B_EOF (0) (RESERVED 1)	31h	1Dh	CDh	55h	311DCD55h
B_EOF (1)	31h	45h	D5h	D9h	3145D5D9h
B_EOF (1) (RESERVED 1)	31h	49h	A5h	BDh	3149A5BDh
B_EOF (1) (RESERVED 2)	31h	4Dh	79h	2Dh	314D792Dh
B_EOF (2)	31h	E1h	B1h	E5h	31E1B1E5h
B_EOF (2) (RESERVED 1)	31h	EDh	2Dh	A1h	31ED2DA1h
B_EOF (2) (RESERVED 2)	31h	EDh	D1h	31h	31EDD131h
B_EOF (3)	35h	51h	49h	F5h	355149F5h
B_EOF (3) (RESERVED 1)	35h	5Dh	25h	C9h	355D25C9h
B_EOF (3) (RESERVED 2)	35h	5Dh	B9h	11h	355DB911h

### 6.3.3 Binary primitive sequences

#### 6.3.3.1 Binary primitive sequences overview

Table 148 summarizes the types of binary primitive sequences.

**Table 148 – Binary primitive sequences**

Binary primitive sequence type	Transmit <sup>a</sup>	Receive <sup>b</sup>	Length of primitive parameter <sup>c</sup> (dword)	Reference
Single	1	1	0 to 3	6.3.3.2
<sup>a</sup> Number of times the transmitter transmits the binary primitive to transmit the binary primitive sequence. <sup>b</sup> Number of times the receiver receives the binary primitive to detect the binary primitive sequence. <sup>c</sup> The length of a primitive parameter that may occur after primitive sequence within a primitive segment.				

#### 6.3.3.2 Single binary primitive sequence

Binary primitives labeled as single binary primitive sequences (e.g., APTA\_COEFFICIENT\_1, APTA\_ADJUST) shall be transmitted one time to form a single binary primitive sequence.

Receivers count each binary primitive received that is labeled as a single binary primitive sequence as a distinct single binary primitive sequence.

A primitive parameter with a length of one to three dwords may follow a single binary primitive sequence. The primitive parameter that follows a single binary primitive sequence shall be contained within a single primitive segment (i.e., the single binary primitive sequence plus the associated primitive parameter, if any, shall be contained within a single SPL packet).

### 6.3.4 Deletable binary primitives

#### 6.3.4.1 APTA\_ADJUST

APTA\_ADJUSTs are deletable binary primitives (see 6.5).

Table 149 defines the different versions of APTA\_ADJUST binary primitives.

**Table 149 – APTA\_ADJUST binary primitives**

Binary primitive	Description
APTA_ADJUST (COMPLETE)	Indicates the SP receiver has completed adjusting the SP transmitter coefficients and APTA is finished.
APTA_ADJUST (READY)	Indicates the SP receiver is ready to receive APTA change request primitives.
APTA_ADJUST (START)	Requests the start of APTA.
APTA_ADJUST (TERMINATE)	Requests the adjustment be terminated, or indicates that the attached phy has terminated adjustment.
APTA_ADJUST (RESERVED 1)	Reserved
APTA_ADJUST (RESERVED 2)	Reserved
APTA_ADJUST (RESERVED 3)	Reserved

#### 6.3.4.2 APTA\_COEFFICIENT\_1

APTA\_COEFFICIENT\_1s are deletable binary primitives (see 6.5).

Table 150 defines the different versions of APTA\_COEFFICIENT\_1 binary primitives.

**Table 150 – APTA\_COEFFICIENT\_1 binary primitives**

Binary primitive	Description
APTA_COEFFICIENT_1 (DECREMENT)	Request the attached SP transmitter to decrement C1.
APTA_COEFFICIENT_1 (INCREMENT)	Request the attached SP transmitter to increment C1.
APTA_COEFFICIENT_1 (MAXIMUM)	Coefficient 1 has reached a maximum value or a value where coefficient 1 is not able to be incremented as a result of a combination of coefficient 2 limit and coefficient 3 limit (see SAS-4).
APTA_COEFFICIENT_1 (MINIMUM)	Coefficient 1 has reached a minimum value or a value where coefficient 1 is not able to be decremented as a result of a combination of coefficient 2 limit and coefficient 3 limit (see SAS-4).
APTA_COEFFICIENT_1 (UPDATED)	Coefficient 1 has completed the requested adjustment.
APTA_COEFFICIENT_1 (RESERVED 1)	Reserved

#### 6.3.4.3 APTA\_COEFFICIENT\_2

APTA\_COEFFICIENT\_2s are deletable binary primitives (see 6.5).

Table 151 defines the different versions of APTA\_COEFFICIENT\_2 binary primitives.

**Table 151 – APTA\_COEFFICIENT\_2 binary primitives**

Binary primitive	Description
APTA_COEFFICIENT_2 (DECREMENT)	Request the attached SP transmitter to decrement C2.
APTA_COEFFICIENT_2 (INCREMENT)	Request the attached SP transmitter to increment C2.
APTA_COEFFICIENT_2 (MAXIMUM)	Coefficient 2 has reached a maximum value or a value where coefficient 2 is not able to be incremented as a result of a combination of coefficient 1 limit and coefficient 2 limit or as a result of a combination of coefficient 2 limit and coefficient 3 limit (see SAS-4).
APTA_COEFFICIENT_2 (MINIMUM)	Coefficient 2 has reached a minimum value or a value where coefficient 2 is not able to be decremented as a result of a combination of coefficient 1 limit and coefficient 2 limit or as a result of a combination of coefficient 2 limit and coefficient 3 limit (see SAS-4).
APTA_COEFFICIENT_2 (UPDATED)	Coefficient 2 has completed the requested adjustment.
APTA_COEFFICIENT_2 (RESERVED 1)	Reserved

#### 6.3.4.4 APTA\_COEFFICIENT\_3

APTA\_COEFFICIENT\_3s are deletable binary primitives (see 6.5).

Table 152 defines the different versions of APTA\_COEFFICIENT\_3 binary primitives.

**Table 152 – APTA\_COEFFICIENT\_3 binary primitives**

Binary primitive	Description
APTA_COEFFICIENT_3 (DECREMENT)	Request the attached SP transmitter to decrement C3.
APTA_COEFFICIENT_3 (INCREMENT)	Request the attached SP transmitter to increment C3.
APTA_COEFFICIENT_3 (MAXIMUM)	Coefficient 3 has reached a maximum value or a value where coefficient 3 is not able to be incremented as a result of a combination of coefficient 1 limit and coefficient 2 limit (see SAS-4).
APTA_COEFFICIENT_3 (MINIMUM)	Coefficient 3 has reached a minimum value or a value where coefficient 3 is not able to be decremented as a result of a combination of coefficient 1 limit and coefficient 2 limit (see SAS-4).
APTA_COEFFICIENT_3 (UPDATED)	Coefficient 3 has completed the requested adjustment.
APTA_COEFFICIENT_3 (RESERVED 1)	Reserved

#### 6.3.4.5 APTA\_COEFFICIENT\_1\_2

APTA\_COEFFICIENT\_1\_2s are deletable binary primitives (see 6.5).

Table 153 defines the different versions of APTA\_COEFFICIENT\_1\_2 binary primitives.

**Table 153 – APTA\_COEFFICIENT\_1\_2 binary primitives**

Binary primitive	Description
APTA_COEFFICIENT_1_2 (DECREMENT)	Request the attached SP transmitter to decrement C1 and decrement C2.
APTA_COEFFICIENT_1_2 (INCREMENT)	Request the attached SP transmitter to increment C1 and increment C2.
APTA_COEFFICIENT_1_2 (MAXIMUM)	Coefficient 1 or coefficient 2 has reached a maximum value or a value where coefficient 1 or coefficient 2 is not able to be incremented (see SAS-4).
APTA_COEFFICIENT_1_2 (MINIMUM)	Coefficient 1 or coefficient 2 has reached a minimum value or a value where coefficient 1 or coefficient 2 is not able to be decremented (see SAS-4).
APTA_COEFFICIENT_1_2 (UPDATED)	Coefficient 1 and coefficient 2 have both completed the requested adjustment.
APTA_COEFFICIENT_1_2 (RESERVED 1)	Reserved

#### 6.3.4.6 APTA\_COEFFICIENT\_2\_3

APTA\_COEFFICIENT\_2\_3s are deletable binary primitives (see 6.5).

Table 154 defines the different versions of APTA\_COEFFICIENT\_2\_3 binary primitives.

**Table 154 – APTA\_COEFFICIENT\_2\_3 binary primitives**

Binary primitive	Description
APTA_COEFFICIENT_2_3 (DECREMENT)	Request the attached SP transmitter to decrement C2 and decrement C3.
APTA_COEFFICIENT_2_3 (INCREMENT)	Request the attached SP transmitter to increment C2 and increment C3.
APTA_COEFFICIENT_2_3 (MAXIMUM)	Coefficient 2 or coefficient 3 has reached a maximum value or a value where coefficient 2 or coefficient 3 is not able to be incremented (see SAS-4).
APTA_COEFFICIENT_2_3 (MINIMUM)	Coefficient 2 or coefficient 3 has reached a minimum value or a value where coefficient 2 or coefficient 3 is not able to be decremented (see SAS-4).
APTA_COEFFICIENT_2_3 (UPDATED)	Coefficient 2 and coefficient 3 have both completed the requested adjustment.
APTA_COEFFICIENT_2_3 (RESERVED 1)	Reserved

### 6.3.5 Binary primitives used only inside SSP and STP connections

#### 6.3.5.1 B\_EOF (binary end of frame)

B\_EOF specifies the end of an SSP or STP frame.

See 6.20.3.3 for details on SSP frame transmission and 6.21.3 for details on STP frame transmission.



Table 155 defines the different versions of B\_EOF binary primitives.

**Table 155 – B\_EOF binary primitives**

Binary primitive	Description
B_EOF (0)	No pad dwords in the last SPL frame segment or STP frame segment.
B_EOF (0) (RESERVED 1)	Reserved
B_EOF (1)	One pad dword in the last SPL frame segment or STP frame segment.
B_EOF (1) (RESERVED 1)	Reserved
B_EOF (1) (RESERVED 2)	Reserved
B_EOF (2)	Two pad dwords in the last SPL frame segment or STP frame segment.
B_EOF (2) (RESERVED 1)	Reserved
B_EOF (2) (RESERVED 2)	Reserved
B_EOF (3)	Three pad dwords in the last SPL frame segment or STP frame segment.
B_EOF (3) (RESERVED 1)	Reserved
B_EOF (3) (RESERVED 2)	Reserved

## 6.4 Extended binary primitives

### 6.4.1 Deletable extended binary primitives

Table 156 defines the deletable extended binary primitives.

**Table 156 – Deletable extended binary primitives**

Extended binary primitive	Use <sup>a</sup>	From <sup>b</sup>			To <sup>b</sup>			Extended binary primitive sequence type <sup>c</sup>
		I	E	T	I	E	T	
PACKET_SYNC	SAS	I	E	T	I	E	T	Single
PACKET_SYNC_LOST								
LINK_RATE_MANAGEMENT	SAS	I	E	T	I	E	T	Single
<p>Key:</p> <p>SAS = SAS logical links, both outside connections <del>or</del> and inside any type of connection</p> <p>I = SAS initiator ports</p> <p>E = expander ports</p> <p>T = SAS target ports</p> <p><sup>a</sup> The Use column indicates when the extended binary primitive is used.</p> <p><sup>b</sup> The From and To columns indicate the type of ports that originate each extended binary primitive or are the intended destinations of each extended binary primitive. Expander ports are not considered originators of extended binary primitives that are being forwarded from expander port to expander port within an expander.</p> <p><sup>c</sup> The Extended binary primitive sequence type column indicate whether the extended binary primitive is a single extended binary primitive sequence, a triple extended binary primitive sequence, or a redundant extended binary primitive sequence (see 6.4.3).</p>								

Table 157 defines the extended binary primitives not specific to the type of connection.

**Table 157 – Extended binary primitives not specific to type of connection**

Extended binary primitive	Use <sup>a</sup>	From <sup>b</sup>			To <sup>b</sup>			Extended binary primitive sequence type <sup>c</sup>
		I	E	T	I	E	T	
END_TRAIN	SpNeg	I	E	T	I	E	T	Single
<p>Key:</p> <p>SpNeg = SAS physical links, during speed negotiation</p> <p>I = SAS initiator ports</p> <p>E = expander ports</p> <p>T = SAS target ports</p> <p><sup>a</sup> The Use column indicates when the extended binary primitive is used.</p> <p><sup>b</sup> The From and To columns indicate the type of ports that originate each extended binary primitive or are the intended destinations of each extended binary primitive. Expander ports are not considered originators of extended binary primitives that are being forwarded from expander port to expander port within an expander.</p> <p><sup>c</sup> The Extended binary primitive sequence type column indicate whether the extended binary primitive is a single extended binary primitive sequence, a triple extended binary primitive sequence, or a redundant extended binary primitive sequence (see 6.4.3).</p>								

### 6.4.2 Extended binary primitive codes

Table 158 defines the extended binary primitive codes for deletable extended binary primitives.

**Table 158 – Extended binary primitive codes for deletable extended binary primitives**

Extended binary primitive	Dword	Byte				Hexadecimal
		0 (first)	1	2	3 (last)	
PACKET_SYNC	0	E2h <sup>a</sup>	89h	E6h	CAh	E289E6CA 178E646B 6F2CDD99 EA0F0443h
	1	17h	8Eh	64h	6Bh	
	2	6Fh	2Ch	DDh	99H	
	3	EAh	0Fh	04h	43h	
PACKET_SYNC_LOST	0	A6h <sup>a</sup>	FAh	03h	C1h	A6FA03C1 503CD15C F9C2916C ED8DA53Bh
	1	50h	3Ch	D1h	5Ch	
	2	F9h	C2h	91h	6Ch	
	3	EDh	8Dh	A5h	3Bh	
LINK_RATE_MANAGEMENT	0	AEh <sup>a</sup>	7Ch	E1h	48h	AE7CE148 B6F6C6D2 9DA9FE40 30144F34h
	1	B6h	F6h	C6h	D2h	
	2	9Dh	A9h	FEh	40h	
	3	30h	14h	4Fh	34h	
<sup>a</sup> The PRIMITIVE SYNCHRONIZE SELECT field (see table 63) is set to 10b.						

Table 159 defines the primitive codes for extended binary primitives not specific to type of connection.

**Table 159 – Extended binary primitive codes for extended binary primitives not specific to any type of connection**

Extended binary primitive	Dword	Byte				Hexadecimal
		0 (first)	1	2	3 (last)	
END_TRAIN	0	0Eh <sup>a</sup>	F2h	AAh	A6h	0EF2AAA6 0EFEF202 0EFE0EC2 0EFE023Eh
	1	0Eh	FEh	F2h	02h	
	2	0Eh	FEh	0Eh	C2h	
	3	0Eh	FEh	02h	3Eh	
<sup>a</sup> The PRIMITIVE SYNCHRONIZE SELECT field (see table 63) is set to 10b.						

### 6.4.3 Extended binary primitive sequences

#### 6.4.3.1 Extended binary primitive sequences overview

Table 160 summarizes the types of extended binary primitive sequences.

**Table 160 – Extended binary primitive sequences**

Extended binary primitive sequence type	Transmit <sup>a</sup>	Receive <sup>b</sup>	Reference
Single	1	1	6.4.3.2
<sup>a</sup> Number of times the transmitter transmits the extended binary primitive to transmit the extended binary primitive sequence. <sup>b</sup> Number of times the receiver receives the extended binary primitive to detect the extended binary primitive sequence.			

#### 6.4.3.2 Single extended binary primitive sequence

Extended binary primitives labeled as single extended binary primitive sequences (e.g., PACKET\_SYNC, PACKET\_SYNC\_LOST, and LINK\_RATE\_MANAGEMENT) shall be transmitted one time to form a single extended binary primitive sequence.

Receivers count each extended binary primitive received that is labeled as a single extended binary primitive sequence as a distinct single extended binary primitive sequence.

### 6.4.4 Deletable extended binary primitives

#### 6.4.4.1 PACKET\_SYNC

PACKET\_SYNC results in the scrambler being initialized as described in 6.8.3.

PACKET\_SYNC is used for:

- periodic initialization of the scrambler (see 5.14.4.10)
- establishing synchronization during Train\_Rx-SNW (see 5.11.4.2.3.6); and
- in combination with PACKET\_SYNC\_LOST to reestablish synchronization if a loss of synchronization occurs.

PACKET\_SYNC is a deletable extended binary primitive (see 6.5).

#### 6.4.4.2 PACKET\_SYNC\_LOST

PACKET\_SYNC\_LOST requests the attached phy send a PACKET\_SYNC.

PACKET\_SYNC\_LOST is used for:

- recovery of scrambling phase;
- establishing synchronization during Train\_Rx-SNW (see 5.11.4.2.3.6); and
- in combination with PACKET\_SYNC to reestablish synchronization if a loss of synchronization occurs.

PACKET\_SYNC\_LOST is a deletable extended binary primitive (see 6.5).

#### 6.4.4.3 LINK\_RATE\_MANAGEMENT

LINK\_RATE\_MANAGEMENT is used for physical link rate tolerance management after a phy reset sequence (see 6.5).

LINK\_RATE\_MANAGEMENT is a deletable extended binary primitive (see 6.5).

#### 6.4.5 Extended binary primitives not specific to type of connections

##### 6.4.5.1 END\_TRAIN

END\_TRAIN is used during Train\_Tx-SNW during speed negotiation to indicate the end of a Train\_Tx pattern.

See 5.11.4.2.3.4 for details on Train\_Tx-SNW.

### 6.5 Physical link rate tolerance management

#### 6.5.1 Physical link rate tolerance management overview

A phy may have three clocks:

- a) an internal clock (e.g., based on a PLL clock generator);
- b) a transmit clock (e.g., based on a PLL clock generator with SSC, if SSC is enabled). Used when transmitting dwords on the physical link; and
- c) a receive clock, derived from the input bit stream. Used when receiving dwords or SPL packets from the physical link.

Although the receive clock has the same nominal fixed frequency as the internal clock, the receive clock may differ from the internal clock frequency up to the physical link rate tolerance (see SAS-4). Over time:

- a) if the receive clock is faster than the internal clock, then an overrun occurs if the phy's receiver receives a dword or an SPL packets and is not able to forward it to an internal receive buffer; or
- b) if the receive clock is slower than the internal clock, then an underrun occurs if the phy's receiver is not able to obtain a dword or an SPL packets from an internal transmit buffer when needed.

To avoid overruns and underruns, the phy's transmitters insert deletable primitives (see 6.2.5), deletable binary primitives (see 6.4.4), or deletable extended binary primitives (see 6.3.4) in the dword stream. The phy's receivers may pass deletable primitives, deletable binary primitives, or deletable extended binary primitives through to their internal buffers, or may strip them out when an overrun occurs. The phy's receivers add deletable primitives, deletable binary primitives, or deletable extended binary primitives when an underrun occurs. The internal logic shall ignore all deletable primitives or deletable extended binary primitives that arrive in the internal buffers.

Circuitry (e.g., an elasticity buffer) is required to absorb the slight differences in frequencies between the phys. The frequency tolerance for a phy is specified in SAS-4. The depth of the elasticity buffer is vendor specific but shall accommodate while in:

- a) the SAS dword mode, the physical link rate tolerance management deletable primitive insertion requirements in table 161 (see 6.5.2); or
- b) the SAS packet mode, the physical link rate tolerance management deletable extended binary primitive insertion requirements in 6.5.3.

Figure 134 shows an example of an elasticity buffer if the phy is in the SAS dword mode.

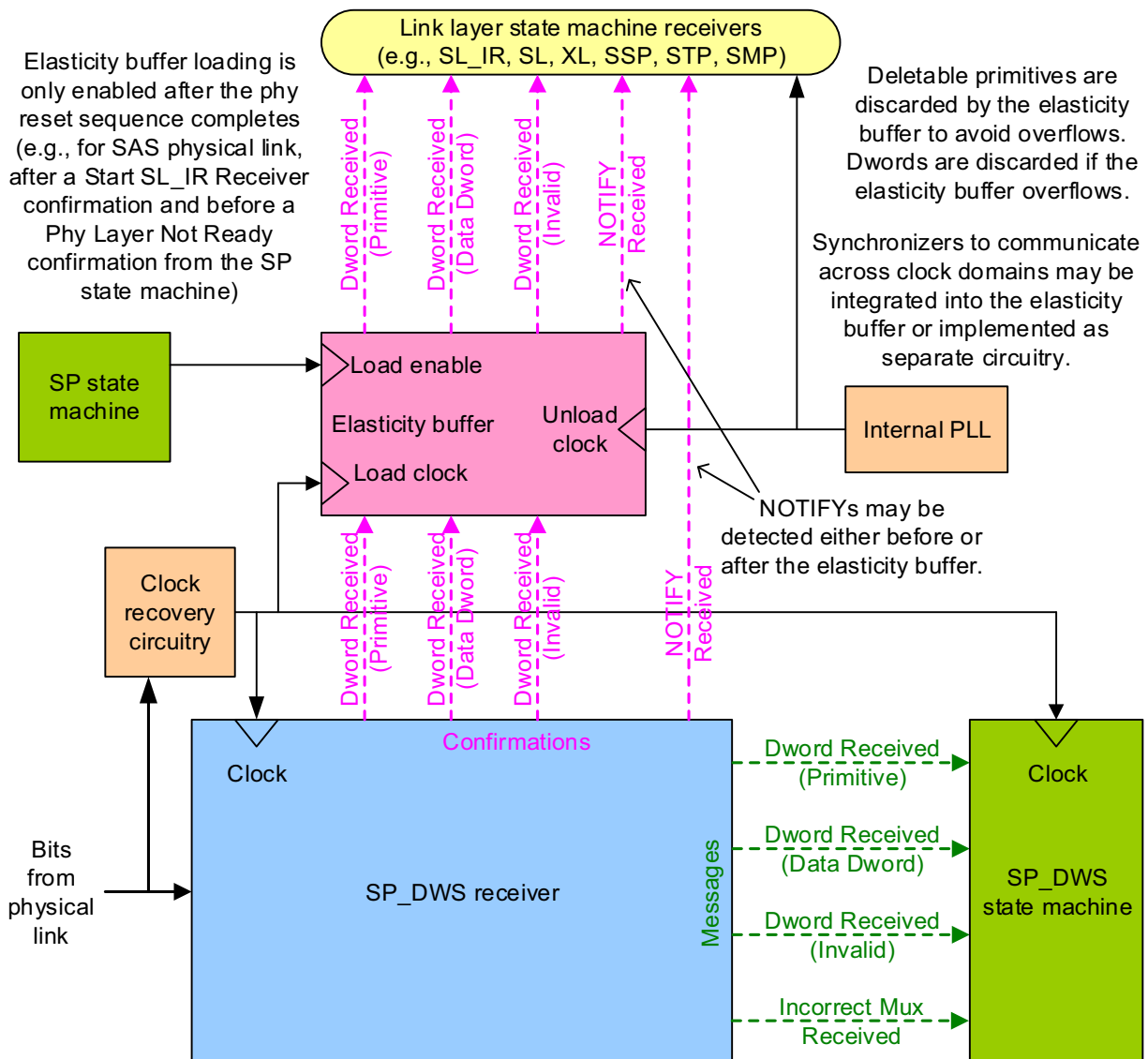


Figure 134 –Elasticity buffer with phys in the SAS dword mode

Figure 135 shows an example of an elasticity buffer if the phy is in the SAS packet mode.

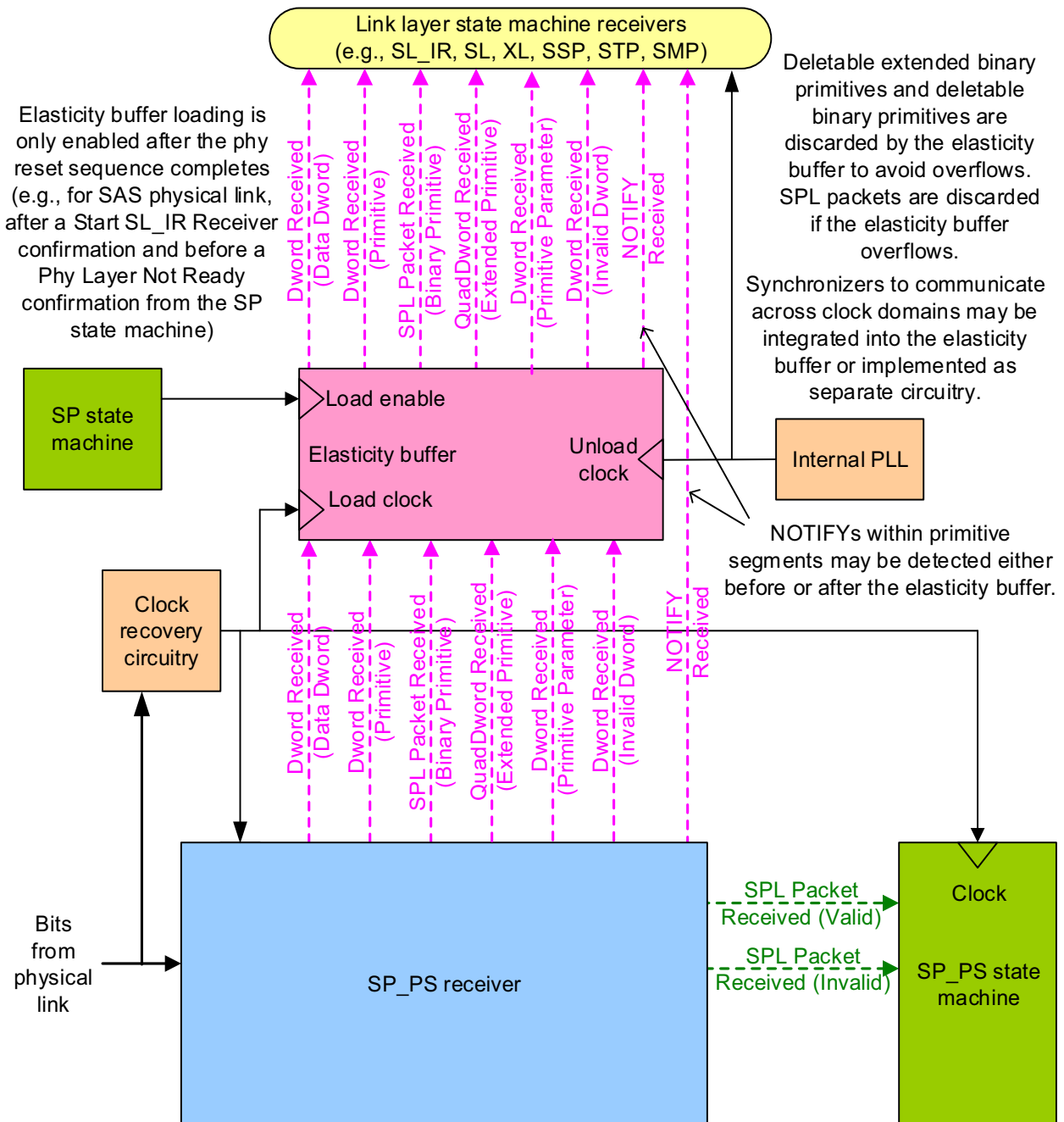


Figure 135 –Elasticity buffer with phys in the SAS packet mode



### 6.5.2 Phys originating dwords while in the SAS dword mode

A logical phy in the SAS dword mode that is originating dwords (i.e., a logical phy in the SAS dword mode that is not an expander logical phy forwarding dwords from another expander logical phy) shall only insert deletable primitives for physical link rate tolerance management after the phy reset sequence (see 5.11) completes as described in table 161.

**Table 161 – Physical link rate tolerance management deletable primitive insertion requirement**

Physical link rate	Requirement
1.5 Gbit/s	One deletable primitive within every 128 dwords <sup>a b c</sup>
3 Gbit/s	Two deletable primitives within every 256 dwords <sup>a b d</sup>
6 Gbit/s	Four deletable primitives within every 512 dwords <sup>a b</sup>
12 Gbit/s	Eight deletable primitives within every 1 024 dwords <sup>e b</sup>
<p><sup>a</sup> These numbers account for the worst case clock frequency differences between the fastest phy transmitter and the slowest phy receiver (e.g., a center-spreading expander phy originating dwords in an STP connection at +2 400 ppm that are forwarded to a down-spreading SATA device with an internal clock at -5 350 ppm). The difference of 7 750 ppm (i.e., 0.775 % or 1/129) is less than the deletable primitive insertion rate of 1/128 (i.e., 7 813 ppm or 0.781 25 %), ensuring there are enough deletable primitives for the phy's receiver to delete without having to buffer dwords.</p> <p><sup>b</sup> 128 dwords at 1.5 Gbit/s, 256 dwords at 3 Gbit/s, 512 dwords at 6 Gbit/s, and 1 024 dwords at 12 Gbit/s are each nominally 3 413.3 ns.</p> <p><sup>c</sup> Phys compliant with SAS-1.1 were required to insert one deletable primitive within every 2 048 dwords at 1.5 Gbit/s.</p> <p><sup>d</sup> Phys compliant with SAS-1.1 were required to insert two deletable primitives within every 4 096 dwords at 3 Gbit/s.</p> <p><sup>e</sup> These numbers account for the worst case clock frequency differences between the fastest phy transmitter and the slowest phy receiver (e.g., a center-spreading expander phy originating dwords in an STP connection at +1 100 ppm that are forwarded to a down-spreading SATA device with an internal clock at -5 350 ppm). The difference of 6 450 ppm (i.e., 0.645 % or 1/155) is less than the deletable primitive insertion rate of 1/128 (i.e., 7 813 ppm or 0.781 25 %), ensuring there are enough deletable primitives for the phy's receiver to delete without having to buffer dwords.</p>	

Deletable primitives inserted for physical link rate tolerance management are in addition to deletable primitives inserted for rate matching (see 6.17). See Annex H for a summary of their combined requirements.

See 6.2.5.1 for details on rotating through ALIGN (0), ALIGN (1), ALIGN (2), and ALIGN (3). NOTIFYs may also be transmitted in place of ALIGNs (see 6.2.5.3) on SAS logical links. MUXs may also be transmitted in place of ALIGNs on multiplexed SAS physical links.

### 6.5.3 Phys originating SPL packets while in the SAS packet mode

A logical phy in the SAS packet mode that is originating SPL packets (i.e., a logical phy in the SAS packet mode that is not an expander logical phy forwarding SPL packets from another expander logical phy) shall only insert deletable extended binary primitives for physical link rate tolerance management after the phy reset sequence (see 5.11) completes as described in table 162.

**Table 162 – Physical link rate tolerance management for deletable extended binary primitive insertion requirement**

Physical link rate	Requirement
22.5 Gbit/s	Four primitive segments containing: a) deletable extended binary primitives (see 6.3.4); b) deletable binary primitives (see 6.3.3); or c) deletable primitives (see 6.2.5), within every 512 SPL packets <sup>a b</sup>
<sup>a</sup> These numbers account for the worst case clock frequency differences between the fastest phy transmitter and the slowest phy receiver (e.g., a center-spreading expander phy originating dwords in an STP connection at +600 ppm that are forwarded to a down-spreading SATA device with an internal clock at -5 350 ppm). The difference of 5 950 ppm (i.e., 0.595 % or 1/168) is less than the deletable extended binary primitive insertion rate of 1/128 (i.e., 7 813 ppm or 0.781 25 %), ensuring there are enough deletable extended binary primitives for the phy's receiver to delete without having to buffer dwords. <sup>b</sup> 512 SPL packets is nominally 3 413.3 ns.	

### 6.5.4 Expander phys forwarding dwords and deletable extended binary primitives

An expander device that is forwarding dwords (i.e., is not originating dwords) is allowed to insert or delete as many deletable primitives or deletable extended binary primitives as required to match the transmit and receive connection rates. An expander device is not required to transmit the number of deletable primitives or deletable extended binary primitives for physical link rate tolerance management described in table 161 and table 162 when forwarding dwords to a SAS logical link. An expander device shall increase or reduce the number of deletable primitives, deletable binary primitives, or deletable extended binary primitives based on clock frequency differences between the expander device's receiving phy and the expander device's transmitting phy (e.g., if receiving at -100 ppm and transmitting at +100 ppm, then it transmits more deletable primitives than it receives).

The expander device is also required to insert deletable primitives or scrambled idle segments for rate matching (see 6.17). During an STP connection, the expander device shall:

- preserve the incoming rate of any additional deletable primitives or deletable extended binary primitives that it receives that are not discarded because of physical link rate tolerance management or rate matching (e.g., the 1/128 deletable primitives received from an originating STP initiator phy compliant with SAS-1.1 for STP initiator phy throttling); or
- transmit one deletable primitive within every 128 dwords, without discarding any data dwords or primitives.

The expander device may reduce the length of repeated primitive sequences (i.e., primitive, SATA\_CONT, and data dword sequences).

NOTE 20 - One possible implementation for expander devices forwarding dwords is for the expander device to delete all deletable primitives received and to insert deletable primitives at the transmit phy whenever its elasticity buffer is empty.

The STP target port of an STP SATA bridge is allowed to insert or delete as many deletable primitives as required to match the transmit and receive connection rates. It is not required to transmit any particular number of deletable primitives for physical link rate tolerance management when forwarding to a SAS logical link and is not required to ensure that any deletable primitives it transmits are in pairs.

Due to physical link rate tolerance management deletable primitive removal, the STP target port may not receive a pair of deletable primitives every 256 dwords, even if the STP initiator port transmitted them in pairs. However, the rate of the dword stream allows for deletable primitive insertion by the STP SATA bridge.

EXAMPLE - The STP SATA bridge deletes all deletable primitives received by the STP target port and inserts two consecutive ALIGNs at the SATA host port when its elasticity buffer is empty or when 254 non-ALIGN dwords have been transmitted. This meets the SATA host port requirement to buffer up to two dwords concurrently while they are being received by the STP target port.

An expander device supporting the SSC modulation type of center-spreading also includes a center-spreading tolerance buffer (see SAS-4).

## 6.6 Idle physical links

If the phy is in the SAS dword mode, then idle dwords are vendor specific data dwords that are scrambled (see 6.8.2).

If the phy is in the SAS packet mode, then Idle dwords are vendor specific data dwords that are scrambled and placed into idle dword segments.

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

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

SATA\_SYNC is a continued primitive sequence that may contain vendor specific data dwords (see 6.2.4.4) that are scrambled (see 6.8) during an STP connection.

## 6.7 CRC

### 6.7.1 CRC overview

All frames include cyclic redundancy check (CRC) values to help detect transmission errors.

Frames transmitted in an STP connection shall include a CRC as defined by SATA. Address frames, SSP frames, and SMP frames shall include a CRC as defined by this standard.

Annex C contains information on CRC generation/checker implementation.

Table 163 defines notation used in the following text describing CRC calculation. Arithmetic is modulo 2.

**Table 163 – CRC notation and definitions**

Notation	Definition
$T(i)$	A transformation over the non-negative integers (i.e., $Z^+$ ): $T(i) = i + 7 - 2 \times (i \bmod 8)$ , $i \geq 0$ , $i \in Z^+$
$F(x)$	A polynomial representing the bits covered by the CRC: $F(x) = b_0x^{(k-1)} + b_1x^{(k-2)} + \dots + b_{(k-2)}x + b_{(k-1)}$ where: k is the number of bits; and $b_i$ describes a bit, where the bit index i denotes that bit $b_i$ is more significant than bit $b_{(i+1)}$ .  For example, if the frame, except for the CRC field, contains one data dword set to 516F3019h, then: $F(x) = x^{30} + x^{28} + x^{24} + x^{22} + x^{21} + x^{19} + x^{18} + x^{17} + x^{16} + x^{13} + x^{12} + x^4 + x^3 + 1$ (i.e., in finite field notation $F(x) = 516F3019h$ )
$F_t(x)$	$F(x)$ with the bit positions of each byte transposed (i.e., bit 7 is bit 0, bit 6 is bit 1, etc.): $F_t(x) = b_{T(0)}x^{(k-1)} + b_{T(1)}x^{(k-2)} + \dots + b_{T(k-2)}x + b_{T(k-1)}$  For example, if the frame, except for the CRC field, contains one data dword set to 516F3019h, then: $F_t(x) = x^{31} + x^{27} + x^{25} + x^{23} + x^{22} + x^{21} + x^{20} + x^{18} + x^{17} + x^{11} + x^{10} + x^7 + x^4 + x^3$ (i.e., in finite field notation $F_t(x) = 8AF60C98h$ )
$L(x)$	The identity polynomial of degree 31 (i.e., a polynomial with all of the coefficients set to one): $L(x) = x^{31} + x^{30} + \dots + x + 1$ (i.e., in finite field notation $L(x) = FFFFFFFFh$ )
$G(x)$	The CRC generator polynomial (i.e., the divisor polynomial): $G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$ (i.e., in finite field notation $G(x) = 1\_04C11DB7h$ )
$R(x)$	The remainder polynomial, which is of degree less than 32.
$R_t(x)$	$R(x)$ with the bit positions of each byte transposed.
$Q(x)$	A quotient polynomial resulting from CRC calculation by the transmitter. This value is discarded.
$Q'(x)$	A quotient polynomial resulting from CRC calculation by the receiver. This value is discarded.
$M(x)$	A polynomial representing the transmitted frame including the CRC field, which is of degree $k+31$ .
$M'(x)$	A polynomial representing the received frame including the received CRC field. If the received frame has no errors, then $M'(x) = M(x)$ and $M'(x)$ is of degree $k+31$ .
$M'_t(x)$	$M'(x)$ with the bit positions of each byte transposed.
$R'(x)$	The result of finding the remainder of an error-free reception of $M(x)$ and also the remainder of $x^{32}L(x) / G(x)$ , which is a unique constant polynomial: $R'(x) = x^{31} + x^{30} + x^{26} + x^{25} + x^{24} + x^{18} + x^{15} + x^{14} + x^{12} + x^{11} + x^{10} + x^8 + x^6 + x^5 + x^4 + x^3 + x + 1$ (i.e., $R'(x) = C704DD7Bh$ )
$R'_t(x)$	$R'(x)$ with the bit positions of each byte transposed (i.e., bit 7 is bit 0, bit 6 is bit 1, etc.): $R'_t(x) = x^{31} + x^{30} + x^{29} + x^{25} + x^{24} + x^{21} + x^{15} + x^{13} + x^{12} + x^{11} + x^9 + x^8 + x^7 + x^6 + x^4 + x^3 + x^2 + x$ (i.e., in finite field notation $R'_t(x) = E320BBDEh$ )

### 6.7.2 CRC generation

The CRC is calculated from  $F(x)$  as follows:

$$x^k \times L(x) + x^{32} \times F_t(x) = Q(x) \times G(x) + R(x)$$

That is:

- 1) the frame  $F(x)$ , not including the CRC field, is transposed into  $F_t(x)$ ;
- 2) the first 32 bits of the transposed frame are inverted (i.e.,  $x^k \times L(x)$  is added);
- 3) 32 bits of zero are appended to the end (i.e.,  $F_t(x)$  is multiplied by  $x^{32}$ ); and
- 4) this result is divided by the generator polynomial  $G(x)$  to find the remainder  $R(x)$ .

The transmitter shall present  $M(x)$  to the 8b10b encoder:

$$M(x) = x^{32} \times F(x) + L(x) + R_t(x)$$

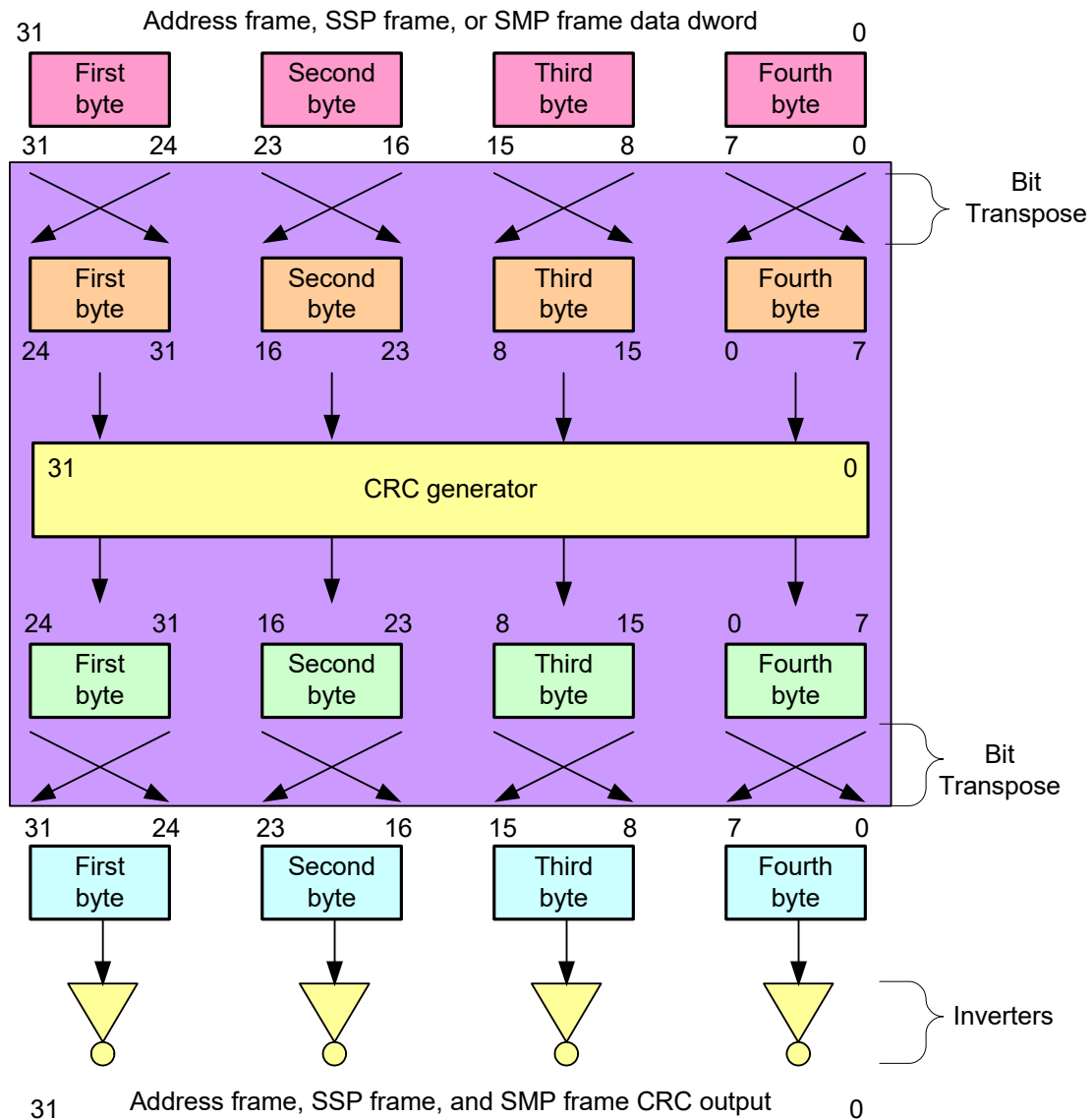
That is, the inverted transposed remainder is appended to the end of the frame, then this result (i.e.,  $M(x)$ ) is presented to the 8b10b encoder for transmission.

For the purposes of CRC computation, inverting the first 32 bits of a frame may be performed by one of the following methods:

- a) inverting the first 32 bits of the frame  $F(x)$  and seeding the CRC remainder register with 00000000h;
- b) seeding the CRC remainder register with FFFFFFFFh; or
- c) prepending the constant 62F52692h to  $F(x)$  and seeding the CRC remainder register with 00000000h.

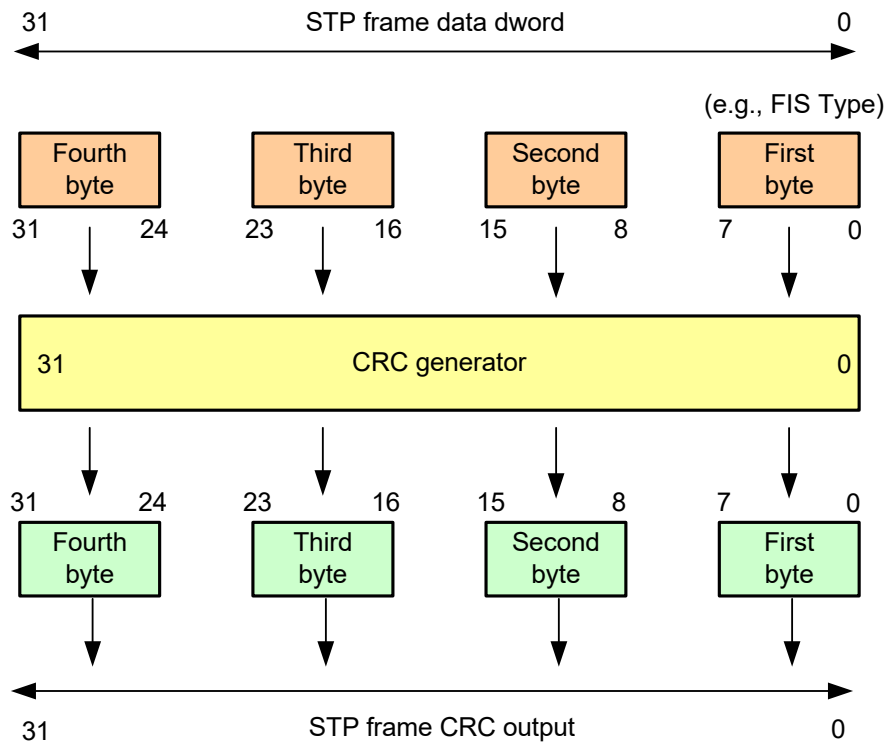
NOTE 21 - The bit order of  $F(x)$  used to calculate the CRC is the same order as the bit transmission order (i.e., the bits within each byte encoded into a data dword are transposed to match the implicit transposition in the 8b10b encoding process).

Figure 136 shows the CRC process for an address frame, an SSP frame and SMP frame.



**Figure 136 –Address frame, SSP frame, and SMP frame CRC bit ordering**

Dwords within STP frames use little-endian. Those dwords are fed into the STP CRC generator without swapping bits within each byte and without inverting the output like the SAS CRC generator. Figure 137 shows the STP CRC bit ordering.



**Figure 137 –STP frame CRC bit ordering**

Since STP uses little-endian byte ordering, the first byte of a dword is in bits 7:0 rather than 31:24 as in SSP and SMP. As a result, the first byte contains the least significant bit. In SSP and SMP, the first byte contains the most significant bit.

See 6.9 for details on how the CRC generator fits into the dword flow along with the scrambler.

### 6.7.3 CRC checking

The CRC of received frame is calculated by the receiver in the same manner that it is generated by the transmitter.

That is:

- 1) the received frame  $M'(x)$ , including the CRC field, is transposed into  $M_t'(x)$ ;
- 2) the first 32 bits of the received transposed frame are inverted;
- 3) 32 bits of zero are appended to the end; and
- 4) this result is divided by the generator polynomial  $G(x)$  to find the remainder.

A received frame that has not incurred any CRC detectable errors during transmission generates a remainder equal to  $R'(x)$ .

If there were no transmission errors, then the received frame  $M'(x)$  equals  $M(x)$ :

$$\begin{aligned} M'(x) &= M(x) \\ &= x^{32} \times F(x) + L(x) + R_t(x) \end{aligned}$$

The CRC  $R'(x)$  is derived as follows:

$$\begin{aligned}
 x^{(k+32)} \times L(x) + x^{32} \times M_t'(x) &= x^{(k+32)} \times L(x) + x^{32} \times (x^{32} \times F_t(x) + L(x) + R(x)) \\
 &= x^{32} \times Q(x) \times G(x) + x^{32} \times L(x)
 \end{aligned}$$

However,  $G(x)$  divides  $x^{32}L(x)$ :

$$x^{32} \times L(x) = Q'(x) \times G(x) + R'(x)$$

Since  $L(x)$  and  $G(x)$  are known and constant,  $R'(x)$  is known and constant and is expected by the receiver after calculating the CRC of a received frame.

From the previous two results:

$$x^{(k+32)} \times L(x) + x^{32} \times (x^{32} \times F_t(x) + L(x) + R(x)) = (x^{32} \times Q(x) + Q'(x)) \times G(x) + R'(x)$$

$R'(x)$  is then transposed and inverted, in the same manner as is done by the transmitter, to obtain:

$$R_t'(x) + L(x) = 1CDF4421h$$

As an alternative to this process, the receiver may check the CRC validity of the frame by stripping off the last 32 bits, leaving  $F(x)$ , and calculating the CRC as defined in 6.7.2. The frame has a valid CRC if the result  $L(x) + R_t(x)$  equals the last 32 bits of the frame that were stripped.

See 6.9 for details on where the CRC checker fits into the dword flow along with the descrambler.

## 6.8 Scrambling

### 6.8.1 Scrambling overview

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



### 6.8.2 Scrambling while in the SAS dword mode

If the phy is in the SAS dword mode, then all data dwords are scrambled. Table 164 lists the scrambling for different types of data dwords.

**Table 164 – Scrambling for different data dword types while in the SAS dword mode**

Connection state	Data dword type	Description of scrambling
Outside connections	SAS idle dword	While 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	While 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	While 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.
	Continued primitive	After a SATA_CONT, vendor specific scrambled data dwords shall be transmitted until a primitive other than a deletable primitive is transmitted.

Data dwords being transmitted in the SAS dword mode shall be XORed with a defined pattern to produce a scrambled value encoded and transmitted on the physical link. Received data dwords shall be XORed with the same pattern after decoding to produce the original data dword value, provided there are no transmission errors.

The pattern that is XORed with the data dwords is defined by the output of a linear feedback shift register implemented with the following polynomial:

$$G(x) = x^{16} + x^{15} + x^{13} + x^4 + 1$$

The output of the pattern generator is 16 bits wide. For each data dword, two outputs of the pattern generator are used as follows:

- 1) the first output of the generator is applied to the lower 16 bits (i.e., bits 15 to 0) of the 32-bit data dword being transmitted or received; and
- 2) the second output of the generator is applied to the upper 16 bits (i.e., bits 31 to 16).

NOTE 22 - Scrambling is not based on data feedback, so the sequence of values XORed with the data being transmitted is constant.

The value of the linear feedback shift register shall be initialized at each SOF and SOAF to FFFFh.

For detailed requirements about scrambling of data dwords following SATA\_SOF and SATA\_CONT, see SATA.

NOTE 23 - STP scrambling uses two linear feedback shift registers, since continued primitive sequences are able to occur inside STP frames and the STP frame and the continued primitive sequence have independent scrambling patterns.

Annex F.1 contains information on scrambling implementations.

### 6.8.3 Scrambling while in the SAS packet mode

If the phy is in the SAS packet mode, then:

- a) the content of the SPL PACKET HEADER field shall not be scrambled and shall not advance the scrambler;
- b) all SPL frame segments (see 5.5.5), idle dword segments, and scrambled idle segments shall be scrambled and the scrambler shall be advanced;
- c) primitives, binary primitives, primitive parameters, and extended binary primitives shall not be scrambled and the scrambler shall be advanced; and
- d) the forward error correction information shall not be scrambled and shall not advance the scrambler.

The content of an SPL packet payload containing an SPL frame segment, an idle dword segment, or a scrambled idle segment while in the SAS packet mode shall be XORed with a defined pattern to produce a scrambled value that is transmitted on the physical link. The content of a received SPL frame segment, an idle dword segment, or a scrambled idle segment shall be XORed with the same pattern to produce the original SPL frame segment content, idle dword segment content, or scrambled idle segment content, provided there are no transmission errors.

For scrambled idle segments, transmitters shall set:

- a) the scrambled idle segment content to zero prior to scrambling; and
- b) the SPL PACKET HEADER field (see table 54):
  - A) to 00b if the last bit of the previous SPL packet to be transmitted (i.e., byte 15, bit seven of the packet payload) is set to one; or
  - B) to 11b if the last bit of the previous SPL packet to be transmitted (i.e., byte 15, bit seven of the packet payload) is set to zero.

Receivers shall not decode the content of a scrambled idle segment.

The pattern that is XORed with the content of an SPL frame segment, idle dword segment, or a scrambled idle segment is defined by the output of a linear feedback shift register implemented with the following polynomial:

$$G(x) = x^{23} + x^{21} + x^{16} + x^8 + x^5 + x^2 + 1$$

The output of the pattern generator is eight bits wide. For each SPL frame segment, idle dword segment, and scrambled idle segment, 16 outputs of the pattern generator are used as follows:

- 1) the first output of the pattern generator is applied to the first byte of the SPL frame segment, idle dword segment, or scrambled idle segment being transmitted or received;
- 2) the second output of the pattern generator is applied to the second data byte of the SPL frame segment, idle dword segment, or scrambled idle segment;
- 3) the next output of the pattern generator is applied to the next data byte of the SPL frame segment, idle dword segment, or scrambled idle segment; and
- 4) repeat 3) until the 16<sup>th</sup> output of the pattern generator is applied to the 16<sup>th</sup> byte of the SPL frame segment, idle dword segment, or scrambled idle segment.

NOTE 24 - Scrambling while in the SAS packet mode is not based on data feedback, so the sequence of values XORed with the data being transmitted is constant.

The value of the linear feedback shift register shall be initialized after each PACKET\_SYNC to 1D BFBCh (i.e., the linear feedback shift register starts with 1D BFBCh on the first byte sent or received after PACKET\_SYNC resulting in the associated pattern generator producing an output pattern based on that initial value for the next SPL packet payload following the PACKET\_SYNC).

See F.2 for the values of the first 128 8-bit outputs of the linear feedback shift register and arrangement of linear feedback shift register stages that correspond to the initialization value described in this subclause.

## 6.9 Bit order of CRC and scrambler

### 6.9.1 Bit order of CRC and scrambler while in the SAS dword mode

If the phy is in the SAS dword mode, then figure 138 shows how data dwords and primitives are routed to the bit transmission logic in figure 57 (see 5.4). Data dwords go through the CRC generator and scrambler. SAS primitives go through the CRC generator and scrambler.

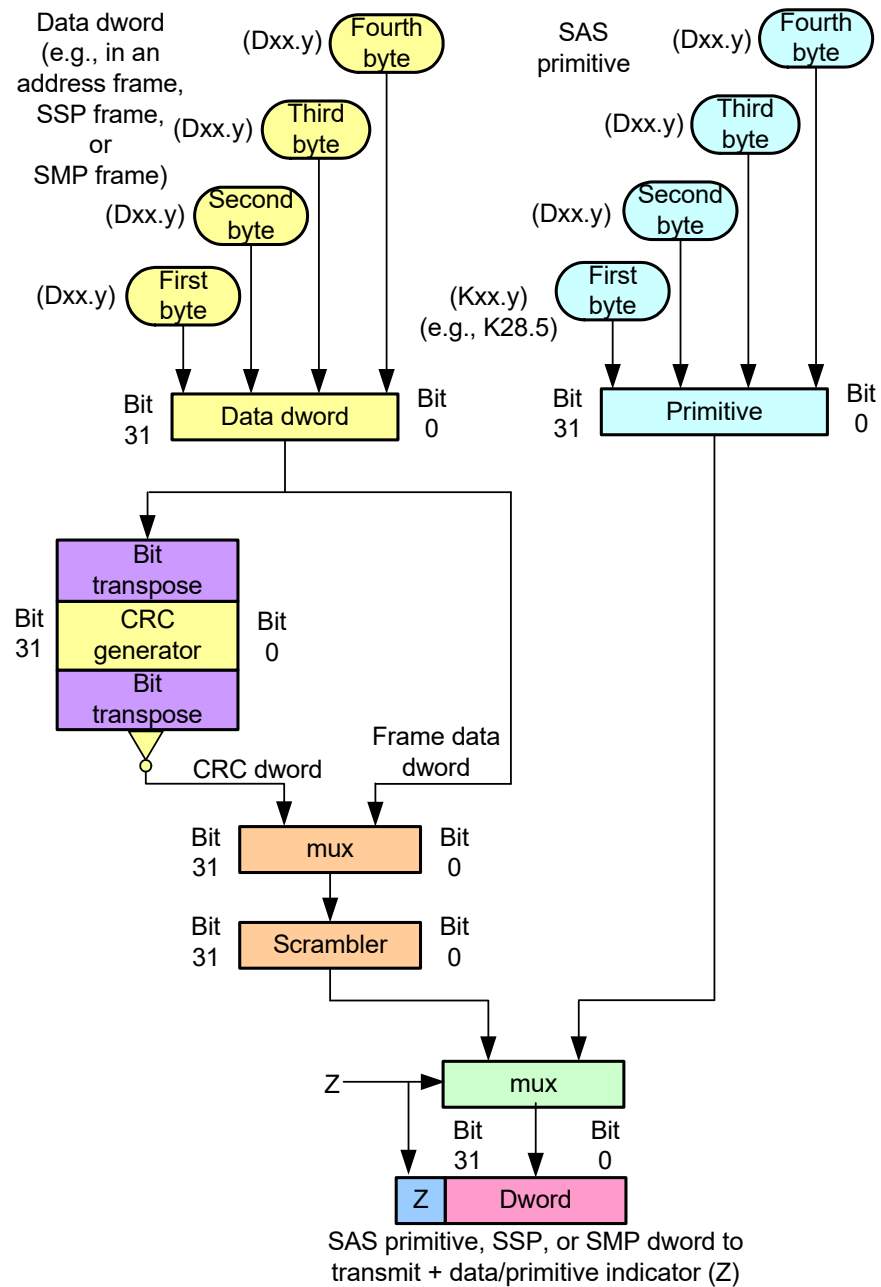
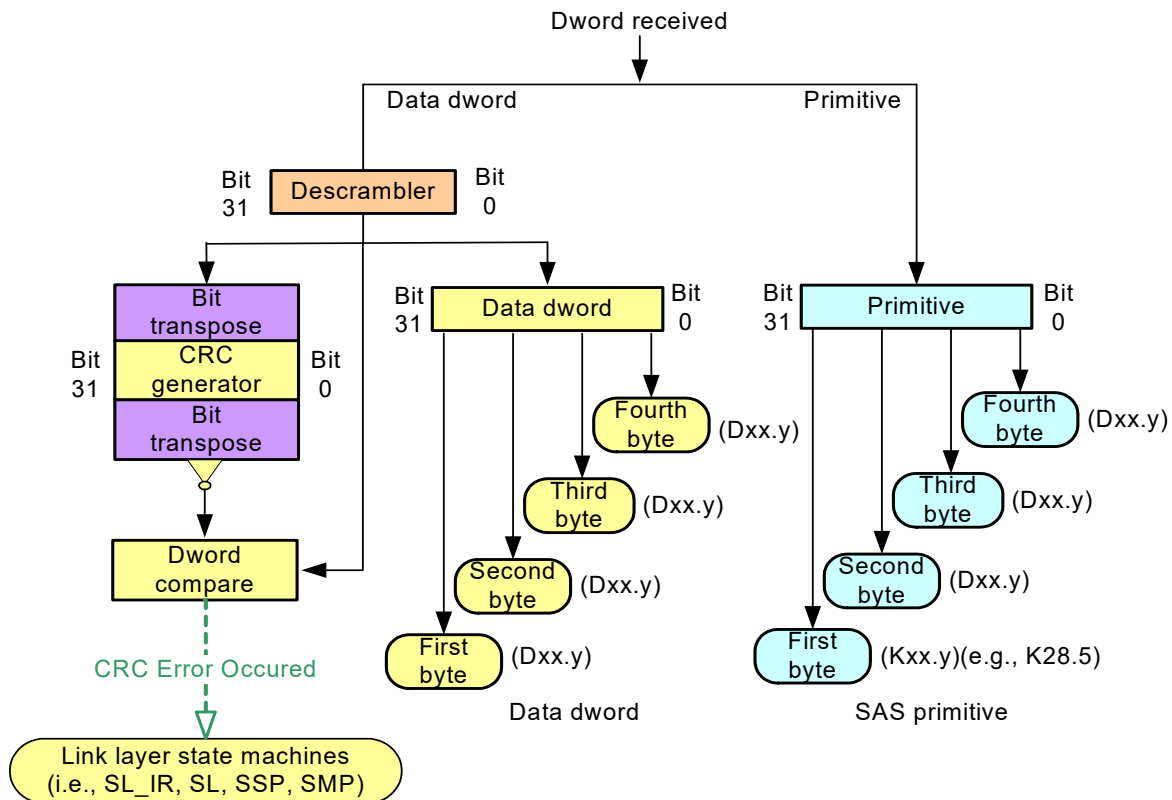


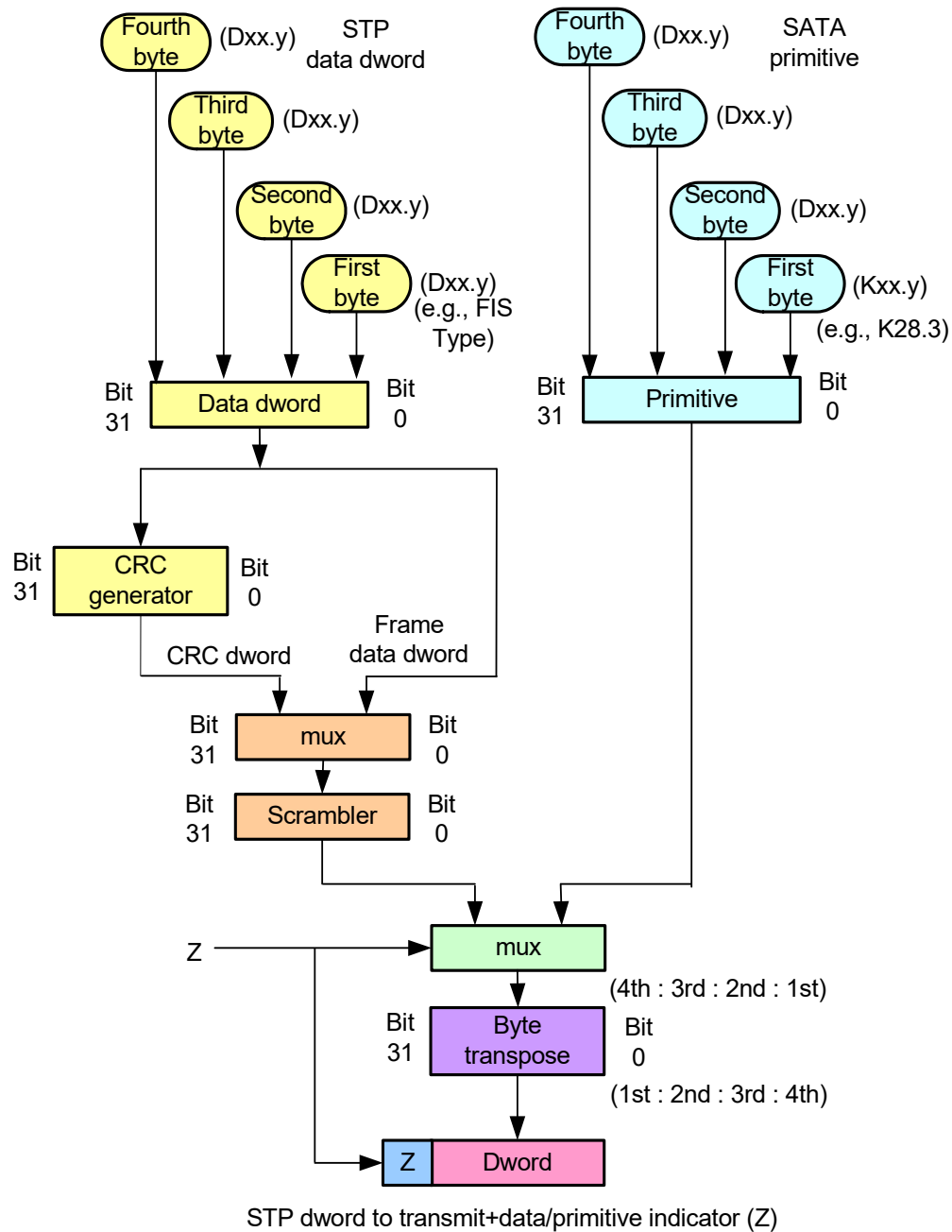
Figure 138 –Transmit path bit ordering while in the SAS dword mode

If the phy is in the SAS dword mode, then figure 139 shows the routing of dwords received from the bit reception logic in figure 58 (see 5.4). The CRC Error Occurred message is sent to the SL state machine (see 6.18), SL\_IR state machine (see 6.12), SSP state machine (see 6.20), and SMP state machine (see 6.22) to indicate that a CRC error occurred on the received frame.



**Figure 139 –Receive path bit ordering while in the SAS dword mode**

Figure 140 shows the STP transmit path bit ordering. The CRC Error Occurred message is sent to the STP state machine (see 6.21) to indicate that a CRC error occurred on the received frame.



**Figure 140 –STP transmit path bit ordering**

Figure 141 shows the STP receive path bit ordering.

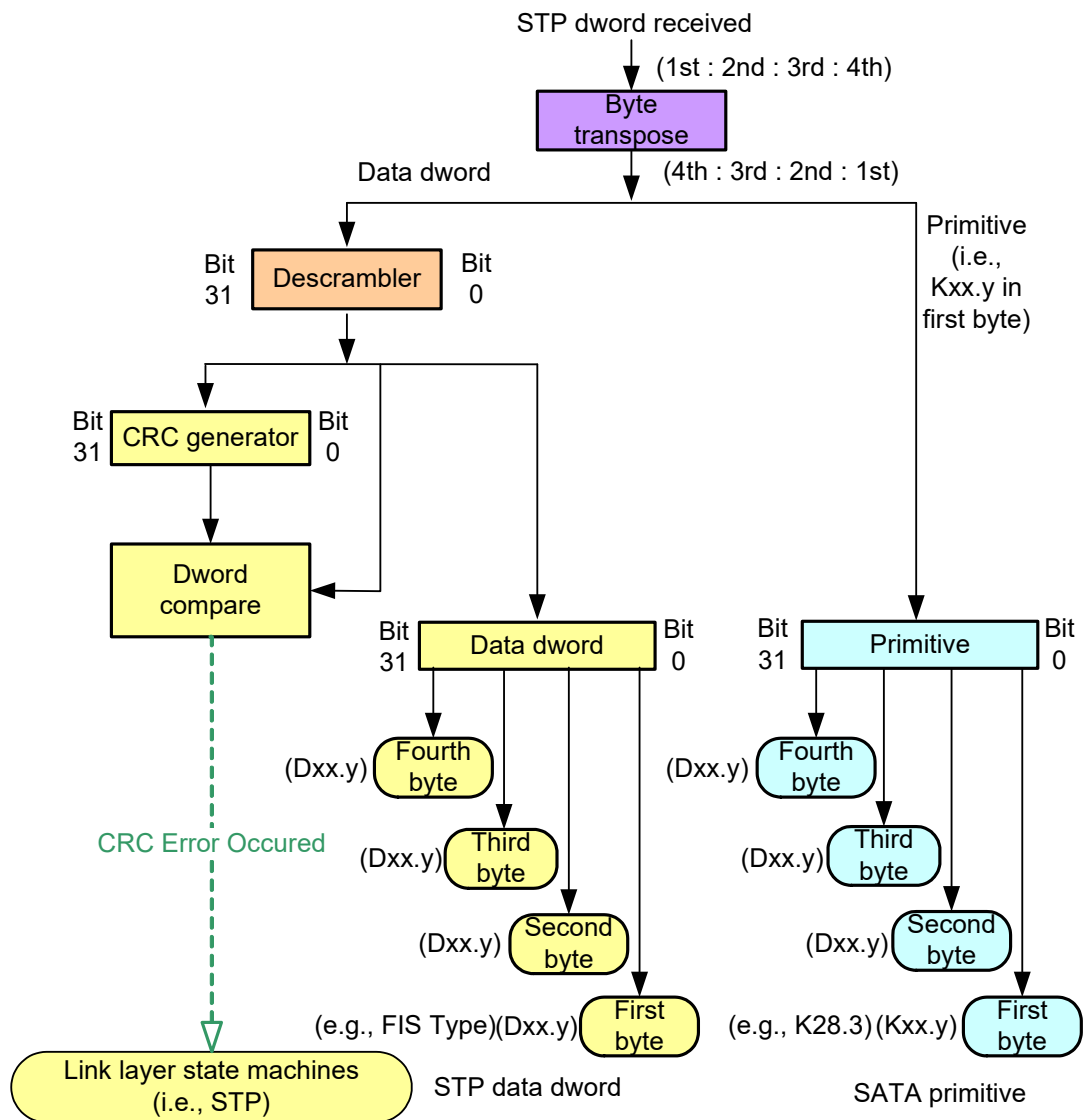


Figure 141 –STP receive path bit ordering

### 6.9.2 Bit order of CRC and scrambler while in the SAS packet mode

If the phy is in the SAS packet mode, then figure 142 shows how SPL packets are routed to the bit transmission logic in figure 61. SPL frame segments go through the CRC generator and then through the scrambler. Scrambled idle segments and idle dword segments go through the scrambler.

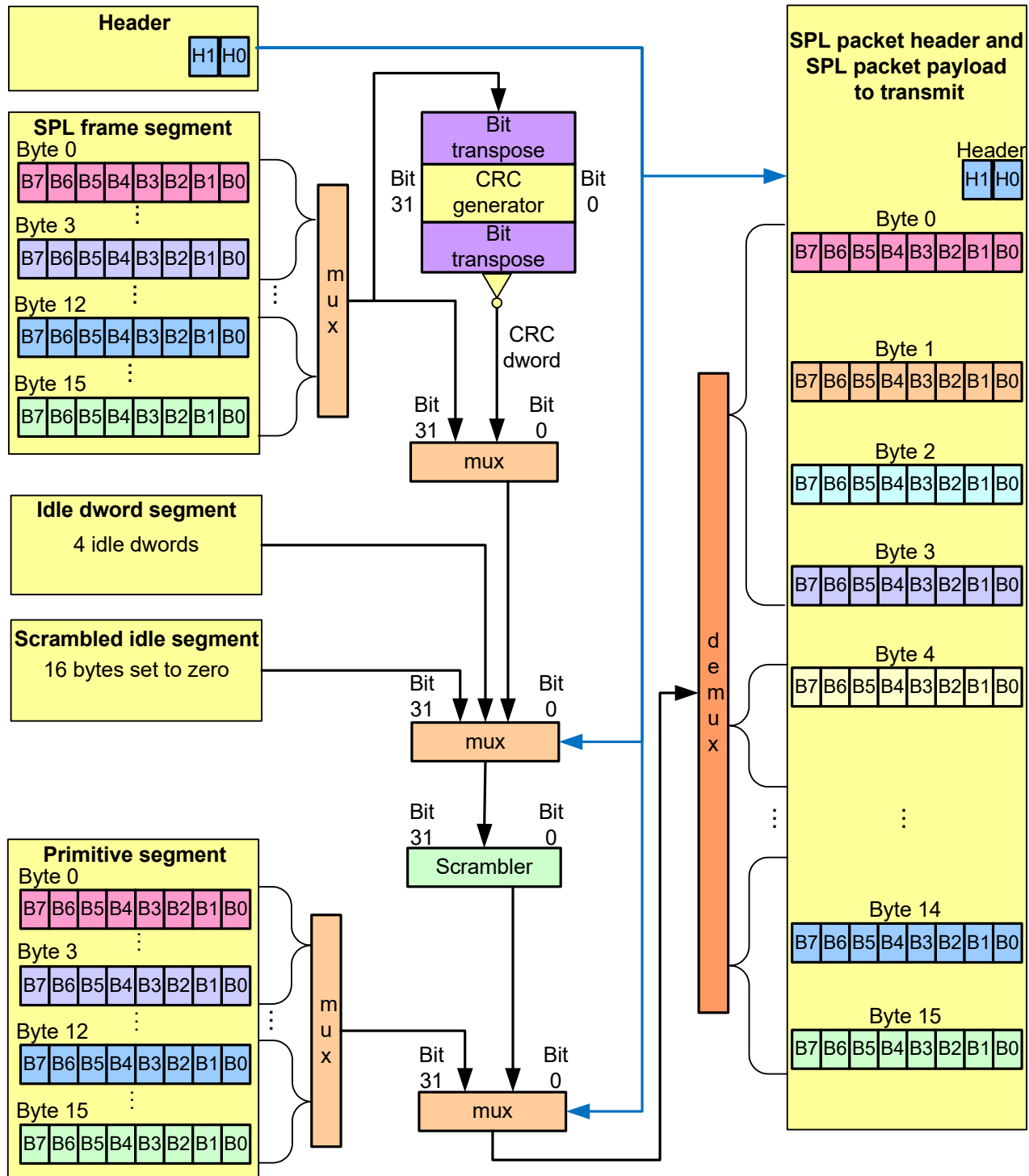


Figure 142 –Transmit path bit ordering while in the SAS packet mode

If the phy is in the SAS packet mode, then figure 143 shows the routing of unpacked data dwords contained in received SPL packets (see figure 114). Data dwords go through:

- a) the descrambler; and
- b) the CRC generator.

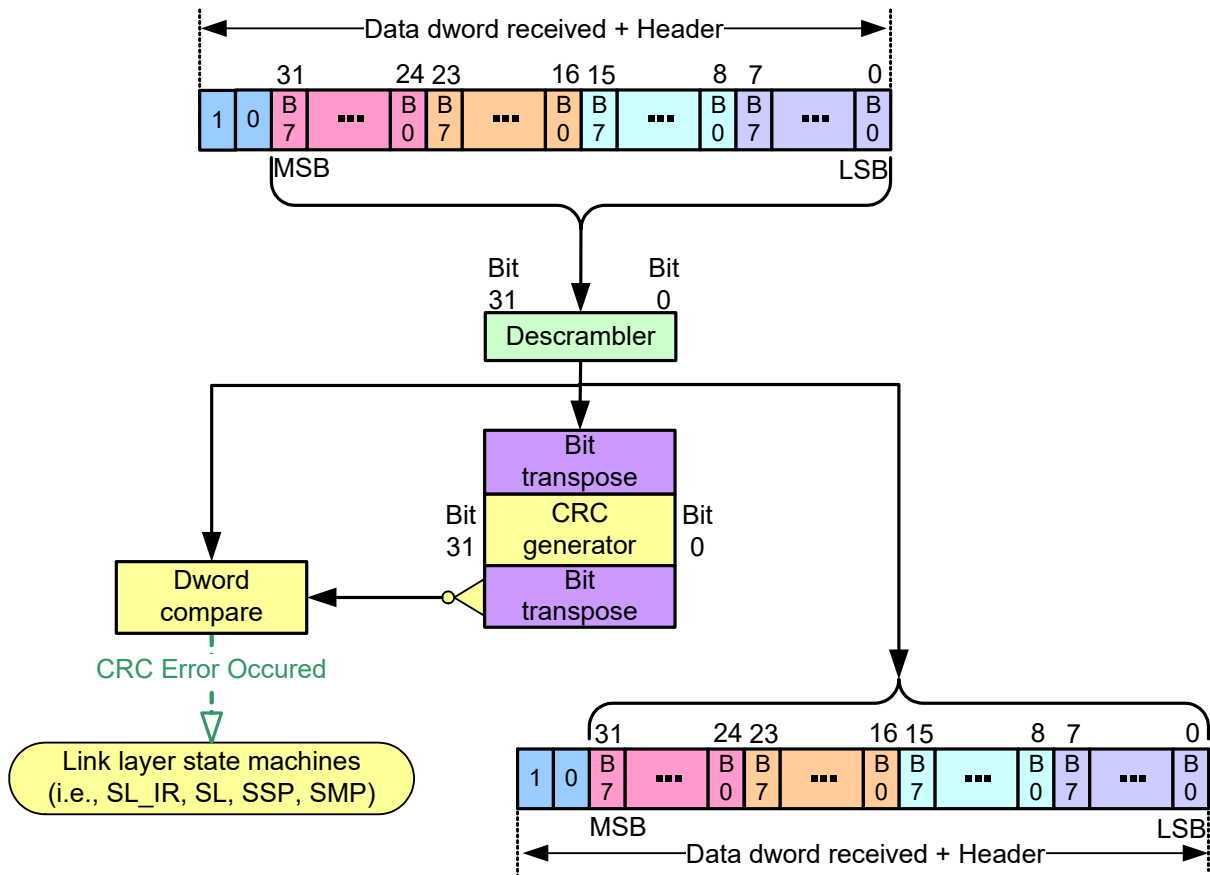


Figure 143 –Receive path bit ordering while in the SAS packet mode

## 6.10 Address frames

### 6.10.1 Address frames overview

Address frames are used for the identification sequence (see 6.11) and for connection requests (see 6.16). Address frames are preceded by SOAF and followed by EOAF as shown in figure 144.

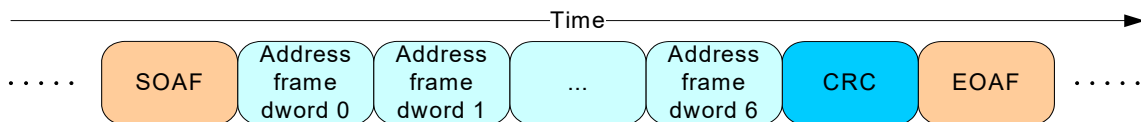


Figure 144 –Address frame transmission



Address frames shall only be transmitted outside connections. Partial address frames (i.e., not containing the number of data dwords defined for the frame) shall not be transmitted. All data dwords in an address frame shall be scrambled.

Table 165 defines the address frame format.

**Table 165 – Address frame format**

Byte\Bit	7	6	5	4	3	2	1	0
0					ADDRESS FRAME TYPE			
1	Frame type dependent bytes							
...								
27								
28	(MSB)	CRC						
...								
31								(LSB)

The ADDRESS FRAME TYPE field indicates the type of address frame and is defined in table 166. This field determines the definition of the frame type dependent bytes.

**Table 166 – ADDRESS FRAME TYPE field**

Code	Address frame type	Description
0h	IDENTIFY	Identification sequence
1h	OPEN	Connection request
All others	Reserved	

The CRC field contains a CRC value (see 6.7) that is computed over the entire address frame prior to the CRC field.

Address frames with unknown address frame types, incorrect lengths, or CRC errors shall be ignored by the recipient.

## 6.10.2 IDENTIFY address frame

Table 167 defines the IDENTIFY address frame format used for the identification sequence. The IDENTIFY address frame is transmitted by each logical phy after the phy reset sequence completes if the physical link is a SAS physical link. The IDENTIFY address frame transmitted by each logical phy in a physical phy shall be identical.

Table 167 – IDENTIFY address frame format

Byte\Bit	7	6	5	4	3	2	1	0
0	Reserved	SAS DEVICE TYPE			ADDRESS FRAME TYPE (0h)			
1	Reserved				REASON			
2	Reserved				SSP INITIATOR PORT	STP INITIATOR PORT	SMP INITIATOR PORT	Restricted (for OPEN address frame)
3	Reserved				SSP TARGET PORT	STP TARGET PORT	SMP TARGET PORT	Restricted (for OPEN address frame)
4	DEVICE NAME							
...								
11								
12	SAS ADDRESS							
...								
19								
20	PHY IDENTIFIER							
21	PERSISTENT CAPABLE	POWER CAPABLE	SLUMBER CAPABLE	PARTIAL CAPABLE	INSIDE ZPSDS PERSISTENT	REQUESTED INSIDE ZPSDS	BREAK_REPLY CAPABLE	
22	Reserved					APTA CAPABLE	SMP PRIORITY CAPABLE	PWR_DIS CAPABLE
23	Reserved							
...								
27								
28	(MSB)							
...	CRC							
31								

The SAS DEVICE TYPE field indicates the type of SAS device type that contains the phy and is defined in table 168.

**Table 168 – SAS DEVICE TYPE field**

Code	Description
001b	End device
010b	Expander device
011b	Obsolete
All others	Reserved

The ADDRESS FRAME TYPE field shall be set as shown in table 167 for the IDENTIFY address frame format.

The REASON field indicates the reason for the link reset sequence and is defined in table 169.

**Table 169 – REASON field**

Code	Description
0h	Unknown reason
1h	Power on
2h	Hard reset (e.g., the port containing this phy received a HARD_RESET primitive sequence during the hard reset sequence) (see 4.4.2) or SMP PHY CONTROL function HARD RESET phy operation (see 9.4.3.28)
3h	SMP PHY CONTROL function LINK RESET phy operation or TRANSMIT SATA PORT SELECTION SIGNAL phy operation (see 9.4.3.28)
4h	Loss of dword synchronization (see 5.15)
5h	After the multiplexing sequence completes, MUX (LOGICAL LINK 0) received in logical link 1 or MUX (LOGICAL LINK 1) received in logical link 0 (see 5.20)
6h	I_T nexus loss timer expired in the STP target port of an STP SATA bridge when the phy was attached to a SATA device (see 4.4.3).
7h	Break Timeout Timer expired (see 6.16.11)
8h	Phy test function stopped (see 9.4.3.29)
9h	Expander device reduced functionality (see 4.5.8)
Ah to Fh	Reserved

An SSP INITIATOR PORT bit set to one indicates that an SSP initiator port is present. An SSP INITIATOR PORT bit set to zero indicates that an SSP initiator port is not present. Expander devices shall set the SSP INITIATOR PORT bit to zero.

An STP INITIATOR PORT bit set to one indicates that an STP initiator port is present. An STP INITIATOR PORT bit set to zero indicates that an STP initiator port is not present. Expander devices shall set the STP INITIATOR PORT bit to zero.

An SMP INITIATOR PORT bit set to one indicates that an SMP initiator port is present. An SMP INITIATOR PORT bit set to zero indicates that an SMP initiator port is not present. Expander devices may set the SMP INITIATOR PORT bit to one.

An SSP TARGET PORT bit set to one indicates that an SSP target port is present. An SSP TARGET PORT bit set to zero indicates that an SSP target port is not present. Expander devices shall set the SSP TARGET PORT bit to zero.

An STP TARGET PORT bit set to one indicates that an STP target port is present. An STP TARGET PORT bit set to zero indicates that an STP target port is not present. Expander devices shall set the STP TARGET PORT bit to zero.

An SMP TARGET PORT bit set to one indicates that an SMP target port is present. An SMP TARGET PORT bit set to zero indicates that an SMP target port is not present. Expander devices shall set the SMP TARGET PORT bit to one.

The DEVICE NAME field indicates the device name (see 4.2.6) of the SAS device or expander device transmitting the IDENTIFY address frame. A DEVICE NAME field set to 00000000 00000000h indicates the device name is not provided in this field.

NOTE 25 - In expander devices, the DEVICE NAME field, if not set to 00000000 00000000h, contains the same value as the SAS ADDRESS field.

For SAS ports, the SAS ADDRESS field indicates the port identifier (see 4.2.9) of the SAS port transmitting the IDENTIFY address frame. For expander ports, the SAS ADDRESS field indicates the device name (see 4.2.6) of the expander device transmitting the IDENTIFY address frame.

The PHY IDENTIFIER field indicates the phy identifier (see 4.2.10) of the phy transmitting the IDENTIFY address frame.

The BREAK\_REPLY CAPABLE bit indicates that the phy is capable of responding to received BREAK primitive sequences with a BREAK\_REPLY primitive sequence (see 6.16.6).

The REQUESTED INSIDE ZPSDS bit indicates the value of the REQUESTED INSIDE ZPSDS bit in the zone phy information (see 4.8.3.1) at the time the IDENTIFY address frame is transmitted. If the phy transmitting the IDENTIFY address frame is contained in an end device, a non-zoning expander device, or a zoning expander device with zoning disabled, then the REQUESTED INSIDE ZPSDS bit shall be set to zero.

The INSIDE ZPSDS PERSISTENT bit indicates the value of the INSIDE ZPSDS PERSISTENT bit in the zone phy information (see 4.8.3.1) at the time the IDENTIFY address frame is transmitted. If the phy transmitting the IDENTIFY address frame is contained in an end device, a non-zoning expander device, or a zoning expander device with zoning disabled, then the INSIDE ZPSDS PERSISTENT bit shall be set to zero.

See 4.1.4 for additional requirements concerning the SAS DEVICE TYPE field, the BREAK\_REPLY CAPABLE bit, the SSP INITIATOR PORT bit, the STP INITIATOR PORT bit, the SMP INITIATOR PORT bit, the SSP TARGET PORT bit, the STP TARGET PORT bit, the SMP TARGET PORT bit, and the SAS ADDRESS field.

A PARTIAL CAPABLE bit set to one indicates that the phy is capable of supporting the partial phy power condition (see 4.10.1.3). A PARTIAL CAPABLE bit set to zero indicates that the phy is not capable of supporting the partial phy power condition.

A SLUMBER CAPABLE bit set to one indicates that the phy is capable of supporting the slumber phy power condition (see 4.10.1.4). A SLUMBER CAPABLE bit set to zero indicates that the phy is not capable of supporting the slumber phy power condition.

If multiplexing is enabled (see 5.20) or optical mode is enabled, then the PARTIAL CAPABLE bit and the SLUMBER CAPABLE bit shall be set to zero.

The POWER CAPABLE field is defined in table 170.

**Table 170 – POWER CAPABLE field**

Code	Description
00b	The device containing the phy: a) does not respond to PWR_GRANT with PWR_ACK, PWR_REQ with PWR_ACK, or PWR_DONE with PWR_ACK; and b) does not issue PWR_REQ or PWR_DONE.
01b	The device containing the phy is capable of allowing the management of additional consumption of power (see 6.14) by: a) issuing PWR_REQ and PWR_DONE; and b) responding to PWR_GRANT with PWR_ACK.
10b	The device containing the phy is capable of managing the additional consumption of power (see 6.14) by responding to: a) PWR_REQ with PWR_ACK; b) PWR_REQ with PWR_GRANT; and c) PWR_DONE with PWR_ACK.
11b	Reserved

A PERSISTENT CAPABLE bit set to one indicates that the phy is capable of establishing and maintaining a persistent connection (see 4.1.13). A PERSISTENT CAPABLE bit set to zero indicates that the phy is not capable of establishing or maintaining a persistent connection. If the phy transmitting the IDENTIFY address frame is contained in an expander device, then the PERSISTENT CAPABLE bit shall be set to zero.

A power disable capable (PWR\_DIS CAPABLE) bit set to one indicates that the SAS target device is a power consumer device (see 6.14.2) that is capable of using the POWER DISABLE signal (see SAS-4). A power source device (see 6.14.1) shall not set the PWR\_DIS CAPABLE bit to one. A PWR\_DIS CAPABLE bit set to zero indicates that the consumer device is not capable of using the POWER DISABLE signal.

An SMP PRIORITY CAPABLE bit set to one indicates that the phy is capable of determining SMP frame priority (see 6.16.3). An SMP PRIORITY CAPABLE bit set to zero indicates that the phy is not capable of determining SMP frame priority.

An APTA CAPABLE bit set to one indicates that the phy's transmitter supports SP transmitter coefficient adjustments using APTA (see 5.12). An APTA CAPABLE bit set to zero indicates that the phy's transmitter does not support SP transmitter coefficient adjustments using APTA.

The CRC field is defined in 6.10.1.

## 6.10.3 OPEN address frame

Table 171 defines the OPEN address frame format used for connection requests.

Table 171 – OPEN address frame format

Byte\Bit	7	6	5	4	3	2	1	0	
0	INITIATOR PORT	SAS PROTOCOL			ADDRESS FRAME TYPE (1h)				
1	FEATURES				CONNECTION RATE				
2	(MSB)	INITIATOR CONNECTION TAG							
3									(LSB)
4	DESTINATION SAS ADDRESS								
...									
11									
12	SOURCE SAS ADDRESS								
...									
19									
20	SOURCE ZONE GROUP								
21	PATHWAY BLOCKED COUNT								
22	(MSB)	ARBITRATION WAIT TIME							
23									(LSB)
24	COMPATIBLE FEATURES (000000b)						CREDIT ADVANCE	SEND EXTEND	
25	MORE COMPATIBLE FEATURES (000000h)								
...									
27									
28	(MSB)	CRC							
...									
31		(LSB)							

An INITIATOR PORT bit set to one specifies that the source port is acting as a SAS initiator port. An INITIATOR PORT bit set to zero specifies that the source port is acting as a SAS target port. If a SAS port sets the INITIATOR PORT bit to one, then the SAS phy shall operate only in its initiator role during the connection. If a SAS port sets the INITIATOR PORT bit to zero, then the SAS port shall operate only in its target role during the connection.

If a SAS port accepts an OPEN address frame with the INITIATOR PORT bit set to one, then the SAS port shall operate only in its target role during the connection. If a SAS port accepts an OPEN address frame with the INITIATOR PORT bit set to zero, then the SAS port shall operate only in its initiator role during the connection.

The SAS PROTOCOL field specifies the protocol for the connection being requested and is defined in table 172.

**Table 172 – SAS PROTOCOL field**

Code	Description
000b	SMP
001b	SSP
010b	STP
All others	Reserved

The ADDRESS FRAME TYPE field shall be set as shown in table 171 for the OPEN address frame format.

The FEATURES field specifies any additional features that are incompatible with SPL-2 and is defined in table 173.

**Table 173 – FEATURES field**

Code	Description
0h	No additional features
All others	Reserved

The CONNECTION RATE field specifies the connection rate (see 4.1.12) being requested between the source and destination, and is defined in table 174.

**Table 174 – CONNECTION RATE field**

Code	Description
8h	1.5 Gbit/s
9h	3 Gbit/s
Ah	6 Gbit/s
Bh	12 Gbit/s
Ch	22.5 Gbit/s
All others	Reserved

A SAS initiator port shall set the initial CONNECTION RATE field to:

- a) the highest supported connection rate supported by a potential pathway as determined during the discover process (e.g., based on the logical link rates of each logical link reported in the SMP DISCOVER responses); or
- b) the logical link rate of the logical phy used to transmit the OPEN address frame.

If a SAS initiator port selected a connection rate based on discover process information, but the connection request results in OPEN\_REJECT (CONNECTION RATE NOT SUPPORTED), then the discover process information is no longer current and the discover process should be run again.

A SAS target port shall set the initial CONNECTION RATE field to:

- a) the last known good connection rate established with the SAS initiator port; or
- b) for the first frame that it intends to transmit in the connection, the connection rate that was used by the SAS initiator port to deliver the command or task management function for that frame.

Each time that a connection request with a connection rate greater than 1.5 Gbit/s results in OPEN\_REJECT (CONNECTION RATE NOT SUPPORTED), the SAS port shall reattempt the connection request with a lower connection rate (e.g., drop from 6 Gbit/s to 3 Gbit/s or 1.5 Gbit/s) and send the same frames in the resulting connection that the SAS port intended to send at the initial connection rate.

The INITIATOR CONNECTION TAG field is used for SSP and STP connection requests to provide a SAS initiator port an alternative to using the SAS target port's SAS address for context lookup when the SAS target port originates a connection request. An SSP initiator port or STP initiator port shall set the INITIATOR CONNECTION TAG field to FFFFh if the SSP initiator port or STP initiator port does not require that this field be provided by the SAS target port. If an SSP initiator port or STP initiator port does require the field to be provided, then the SSP initiator port or STP initiator port should set the INITIATOR CONNECTION TAG field to a unique value per SAS target port. When requesting a connection to a SAS initiator port, a SAS target port shall set the INITIATOR CONNECTION TAG field to the most recent value received or the value received in one of the connection requests for one of the outstanding commands or task management functions from the SAS initiator port. A SAS initiator port shall:

- a) use the same INITIATOR CONNECTION TAG field value for all connection requests to the same SAS target port; and
- b) only change the INITIATOR CONNECTION TAG field value when it has no commands or task management functions outstanding to that SAS target port.

SAS target ports are not required to check consistency of the INITIATOR CONNECTION TAG field in different connection requests from the same SAS initiator port. SMP initiator ports shall set the INITIATOR CONNECTION TAG field to FFFFh for SMP connection requests.

The DESTINATION SAS ADDRESS field specifies the port identifier (see 4.2.9) of the SAS port to which a connection is being requested.

The SOURCE SAS ADDRESS field specifies the port identifier (see 4.2.9) of the SAS port that originated the OPEN address frame.

The SOURCE ZONE GROUP field identifies the zone group of the phy making the connection request. The SOURCE ZONE GROUP field shall be:

- a) set to 00h when transmitted by an end device;
- b) set to 00h when transmitted by an expander device on a phy with the INSIDE ZPSDS bit set to zero;
- c) set to the source zone group for the outgoing connection request as described in table 41 (see 4.8.3.5) when transmitted by an expander device on a phy with the INSIDE ZPSDS bit set to one;
- d) ignored when received by an end device;
- e) ignored when received by an expander device on a phy with the INSIDE ZPSDS bit set to zero; or
- f) used to determine the source zone group for the incoming connection request as described in table 41 (see 4.8.3.5) when received by an expander device on a phy with the INSIDE ZPSDS bit set to one.



The PATHWAY BLOCKED COUNT field specifies the number of times the port has retried this connection request due to receiving OPEN\_REJECT (PATHWAY BLOCKED), OPEN\_REJECT (RESERVED STOP 0), or OPEN\_REJECT (RESERVED STOP 1). The port shall not increment the PATHWAY BLOCKED COUNT value past FFh. If the port changes connection requests, then the port shall set the PATHWAY BLOCKED COUNT field to 00h.

The ARBITRATION WAIT TIME field specifies how long the port transmitting the OPEN address frame has been waiting for a connection request to be accepted or rejected. This time is maintained by the port layer in an Arbitration Wait Time timer (see 7.2.2). For values from 0000h to 7FFFh, the Arbitration Wait Time timer increments in one microsecond steps. For values from 8000h to FFFFh, the Arbitration Wait Time timer increments in one millisecond steps. The maximum value represents 32 767 ms + 32 768  $\mu$ s. Table 175 describes several values of the ARBITRATION WAIT TIME field. See 6.16.4 for details on arbitration fairness.

**Table 175 – ARBITRATION WAIT TIME field**

Code	Description
0000h	0 $\mu$ s
0001h	1 $\mu$ s
...	...
7FFFh	32 767 $\mu$ s
8000h	0 ms + 32 768 $\mu$ s
8001h	1 ms + 32 768 $\mu$ s
...	...
FFFFh	32 767 ms + 32 768 $\mu$ s

If the end device originating the OPEN address frame:

- a) is an initiator port (i.e., the INITIATOR PORT bit is set to one);
- b) supports persistent connections (see 4.1.13); and
- c) the SAS PROTOCOL field is set to 001b (i.e., SSP),

then a SEND EXTEND bit set to:

- a) one specifies that the source phy is requesting the establishment of a persistent connection with the destination phy (see 6.18); or
- b) zero specifies that the source phy is requesting that a persistent connection not be established (see 6.18).

If the end device transmitting the OPEN address frame is a target port (i.e., the INITIATOR PORT bit is set to zero), then the SEND EXTEND bit shall be set to zero.

If the end device receiving the OPEN address frame:

- a) does not support persistent connections; or
- b) the SAS PROTOCOL field is set to a value other than 001b,

then the SEND EXTEND bit shall be ignored.

If the SAS PROTOCOL field is set to 001b (i.e., SSP), then a CREDIT ADVANCE bit:

- a) set to one specifies that the destination SSP phy that implements credit advance, advances credit as defined in 4.1.14; or
- b) set to zero specifies that the destination SSP phy shall not advance credit (see 4.1.14).

If the SAS PROTOCOL field is not set to 001b (i.e., SSP), then the CREDIT ADVANCE bit shall be ignored.

The COMPATIBLE FEATURES field and the MORE COMPATIBLE FEATURES field shall be set as shown in table 171 for the OPEN address frame format. A phy receiving an OPEN address frame shall ignore the COMPATIBLE FEATURES field and the MORE COMPATIBLE FEATURES field.

The CRC field is defined in 6.10.1.

## 6.11 Link reset sequence

### 6.11.1 Link reset sequence overview

For SATA, a link reset sequence is a phy reset sequence (see 5.11).

For SAS, a link reset sequence is either:

- a) the following sequence:
  - 1) a phy reset sequence indicating that the physical link is using SAS rather than SATA; and
  - 2) an identification sequence;
- or
- b) the following sequence:
  - 1) a phy reset sequence indicating that the physical link is using SAS rather than SATA;
  - 2) a hard reset sequence;
  - 3) another phy reset sequence indicating that the physical link is using SAS rather than SATA; and
  - 4) an identification sequence.

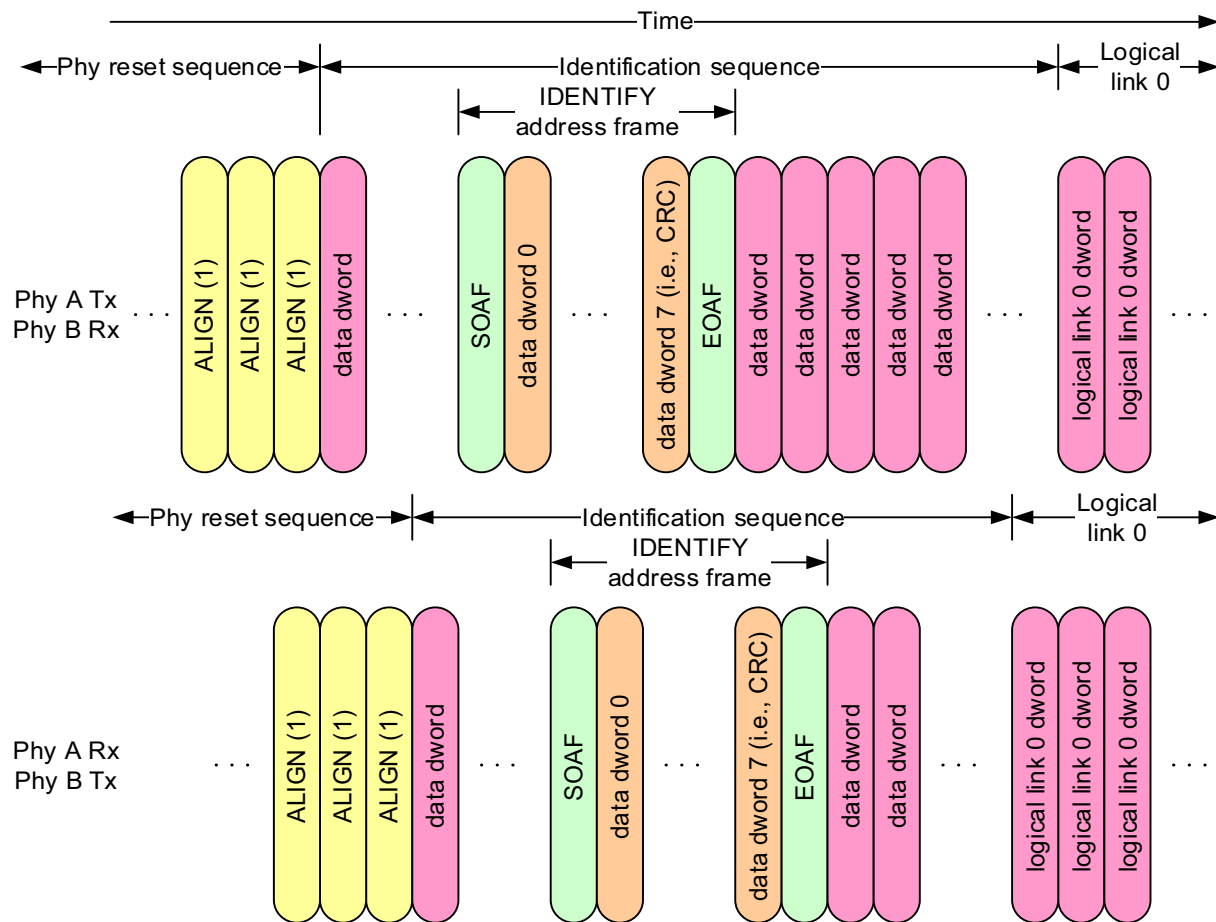
An identification sequence occurs when a logical phy:

- a) transmits one or three IDENTIFY address frames (see 6.10.2); and
- b) does not receive a HARD\_RESET primitive sequence.

A hard reset sequence occurs when, after the phy reset sequence, a logical phy:

- a) transmits a HARD\_RESET primitive sequence (see 6.2.6.8); or
- b) receives a HARD\_RESET primitive sequence.

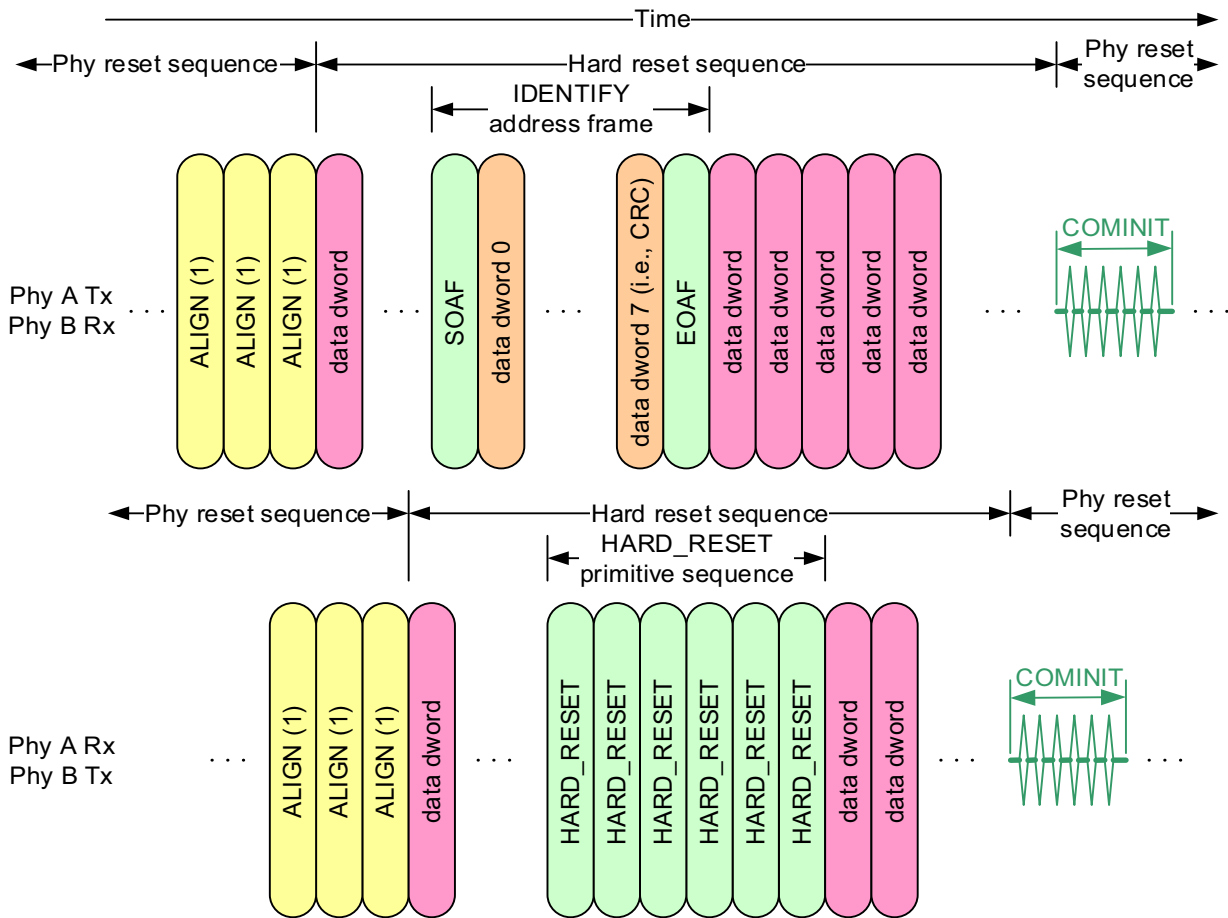
Figure 145 shows two phys with multiplexing disabled performing the identification sequence. Only one IDENTIFY address frame is shown in this example.



Note - Phys transmit deletable primitives for physical link rate tolerance management after the phy reset sequence.

**Figure 145 –Identification sequence**

Figure 146 shows phy A attempting to perform the identification sequence and phy B performing the hard reset sequence. Because phy A receives a HARD\_RESET primitive sequence, a hard reset sequence occurs. Multiplexing is disabled and only one IDENTIFY address frame is shown in this example.



Note - Phys transmit deletable primitives for physical link rate tolerance management after the phy reset sequence.

**Figure 146 –Hard reset sequence**

Each logical phy receives an IDENTIFY address frame or a HARD\_RESET primitive sequence from the logical phy to which it is attached.

If a logical phy receives a valid IDENTIFY address frame (see 6.12.4.3.1) within 1 ms of phy reset sequence completion, then the SAS address in the outgoing IDENTIFY address frames and the SAS address in the incoming IDENTIFY address frame determine the port to which the logical phy belongs (see 4.1.4). The logical phy ignores subsequent IDENTIFY address frames and HARD\_RESETs until another phy reset sequence occurs.

If a logical phy receives a HARD\_RESET primitive sequence within 1 ms of phy reset sequence completion, then the logical phy shall consider this to be a reset event and the port containing the logical phy shall process a hard reset (see 4.4.2).

If a logical phy does not receive a HARD\_RESET primitive sequence or a valid IDENTIFY address frame within 1 ms of phy reset sequence completion, then the physical phy containing the logical phy shall restart the phy reset sequence.

### 6.11.2 Expander device handling of link reset sequences

After completing the link reset sequence on a phy and completing internal initialization, the ECM within an expander device shall be capable of routing connection requests through that phy. The expander device may return OPEN\_REJECT (NO DESTINATION) until it is ready to process connection requests.

The ECM of an externally configurable expander device is dependent on the completion of the discover process (see 4.6) for routing connection requests using the table routing method.

## 6.12 SL\_IR (link layer identification and hard reset) state machines

### 6.12.1 SL\_IR state machines overview

The SL\_IR (link layer identification and hard reset) state machines control the flow of dwords on the physical link that are associated with the identification and hard reset sequences. The state machines are as follows:

- a) SL\_IR\_TIR (transmit IDENTIFY or HARD\_RESET primitive sequence) state machine (see 6.12.3);
- b) SL\_IR\_RIF (receive IDENTIFY address frame) state machine (see 6.12.4); and
- c) SL\_IR\_IRC (identification and hard reset control) state machine (see 6.12.5).

The SL\_IR state machines send the following messages to the SL state machines (see 6.18) in SAS devices or the XL (see 6.19) state machine in expander devices:

- a) Enable Disable SAS Link (Enable); and
- b) Enable Disable SAS Link (Disable).

This state machine shall maintain the timers listed in table 176.

**Table 176 – SL\_IR\_IRC state machine timers**

Timer	Initial value
Receive Identify Timeout timer	1 ms

Figure 147 shows the SL\_IR state machines.

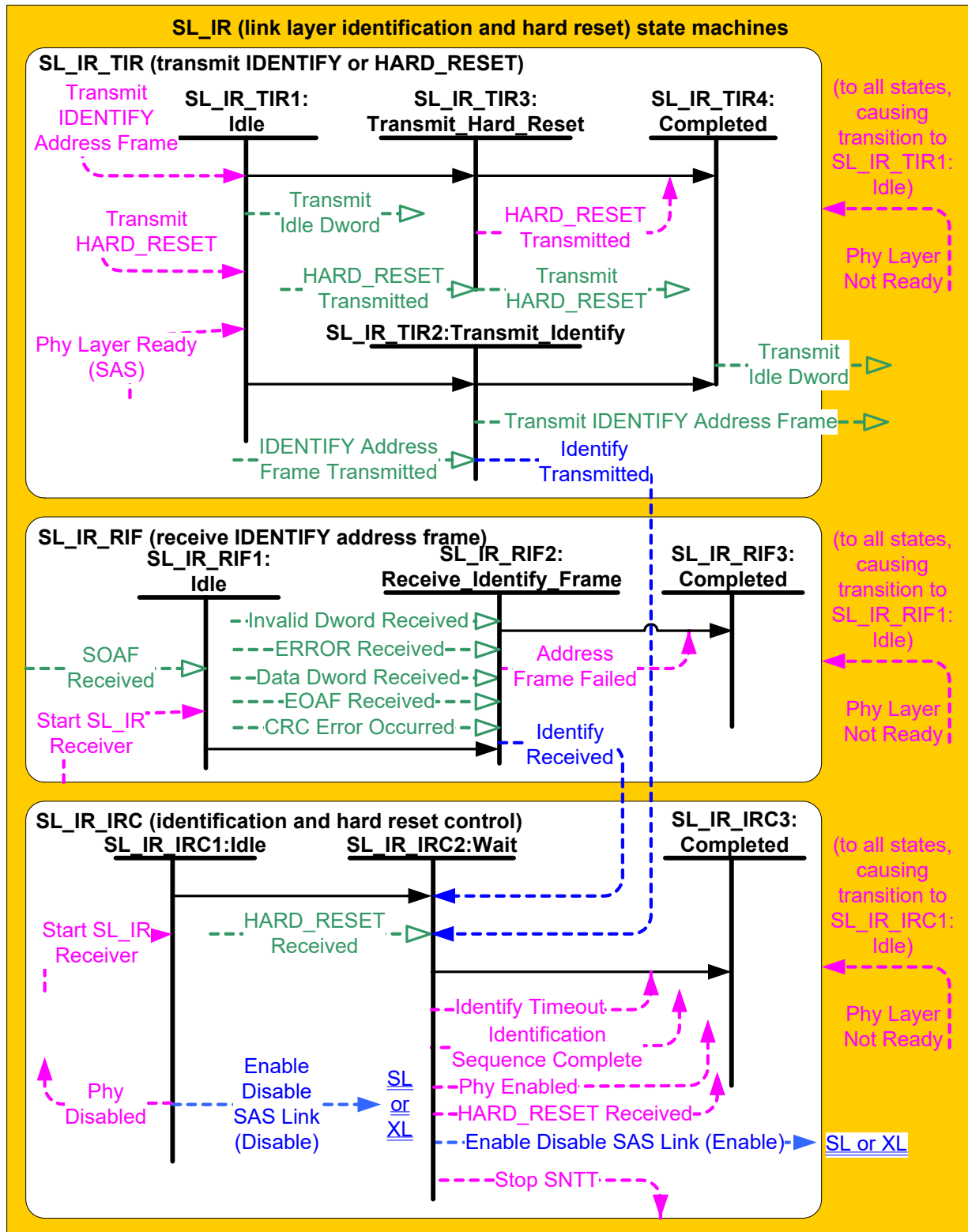


Figure 147 –SL\_IR (link layer identification and hard reset) state machines

### 6.12.2 SL\_IR transmitter and receiver

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

- a) Transmit IDENTIFY Address Frame;
- b) Transmit HARD\_RESET; and
- c) Transmit Idle Dword.

Upon receiving a Transmit IDENTIFY Address Frame message, the SL\_IR transmitter shall transmit:

- 1) SOAF;
- 2) data dwords;
- 3) EOAF; and
- 4) at least 3 idle dwords.

NOTE 26 - Phys compliant with SAS-1.1 were not required to transmit idle dwords after EOAF.

The SL\_IR transmitter sends the following messages to the SL\_IR state machines:

- a) HARD\_RESET Transmitted; and
- b) IDENTIFY Address Frame Transmitted.

The SL\_IR receiver sends the following messages to the SL\_IR state machines indicating primitive sequences and dwords received from the SP\_DWS receiver (see 5.15.2) and the SP\_PS receiver (see 5.16.2):

- a) SOAF Received;
- b) Data Dword Received;
- c) EOAF Received;
- d) ERROR Received;
- e) Invalid Dword Received; and
- f) HARD\_RESET Received.

The SL\_IR receiver shall not require reception of any idle dwords after an IDENTIFY address frame.

The SL\_IR receiver shall ignore all other dwords.

The SL\_IR transmitter relationship to other transmitters is defined in 4.3.2. The SL\_IR receiver relationship to other receivers is defined in 4.3.3.

### 6.12.3 SL\_IR\_TIR (transmit IDENTIFY or HARD\_RESET) state machine

#### 6.12.3.1 SL\_IR\_TIR state machine overview

The SL\_IR\_TIR state machine's function is to transmit one or three IDENTIFY address frames or a HARD\_RESET primitive sequence after the phy layer enables the link layer. This state machine consists of the following states:

- a) SL\_IR\_TIR1:Idle (see 6.12.3.2) (initial state);
- b) SL\_IR\_TIR2:Transmit\_Identify (see 6.12.3.3);
- c) SL\_IR\_TIR3:Transmit\_Hard\_Reset (see 6.12.3.4); and
- d) SL\_IR\_TIR4:Completed (see 6.12.3.5).

This state machine receives the following requests from the management application layer:

- a) Transmit IDENTIFY Address Frame; and
- b) Transmit HARD\_RESET.

This state machine shall start in the SL\_IR\_TIR1:Idle state. This state machine shall transition to the SL\_IR\_TIR1:Idle state from any other state after receiving a Phy Layer Not Ready confirmation.

**6.12.3.2 SL\_IR\_TIR1:Idle state****6.12.3.2.1 State description**

This state shall request that idle dwords be transmitted by repeatedly sending Transmit Idle Dword messages to the SL\_IR transmitter.

**6.12.3.2.2 Transition SL\_IR\_TIR1:Idle to SL\_IR\_TIR2:Transmit\_Identify**

This transition shall occur after:

- a) a Phy Layer Ready (SAS) confirmation is received; and
- b) a Transmit IDENTIFY Address Frame request is received.

**6.12.3.2.3 Transition SL\_IR\_TIR1:Idle to SL\_IR\_TIR3:Transmit\_Hard\_Reset**

This transition shall occur after:

- a) a Phy Layer Ready (SAS) confirmation is received; and
- b) a Transmit HARD\_RESET request is received.

**6.12.3.3 SL\_IR\_TIR2:Transmit\_Identify state****6.12.3.3.1 State description**

Upon entry into this state, this state shall send either one or three Transmit IDENTIFY Address Frame messages to the SL\_IR transmitter.

NOTE 27 - Phys compliant with SAS-1.1 only transmitted one Transmit IDENTIFY Address Frame message.

After this state receives an IDENTIFY Address Frame Transmitted message in response to its first Transmit IDENTIFY Address Frame message, this state shall send an Identify Transmitted message to the SL\_IR\_IRC state machine.

**6.12.3.3.2 Transition SL\_IR\_TIR2:Transmit\_Identify to SL\_IR\_TIR4:Completed**

If this state sends one Transmit IDENTIFY Address Frame message, then this transition shall occur:

- a) after sending an Identify Transmitted message to the SL\_IR\_IRC state machine.

If this state sends three Transmit IDENTIFY Address Frame messages, then this transition shall occur:

- a) after receiving three Identify Transmitted messages.

**6.12.3.4 SL\_IR\_TIR3:Transmit\_Hard\_Reset state****6.12.3.4.1 State description**

Upon entry into this state, this state shall send a Transmit HARD\_RESET message to the SL\_IR transmitter.

After this state receives a HARD\_RESET Transmitted message, this state shall send a HARD\_RESET Transmitted confirmation to the management application layer.

**6.12.3.4.2 Transition SL\_IR\_TIR3:Transmit\_Hard\_Reset to SL\_IR\_TIR4:Completed**

This transition shall occur:

- a) after sending a HARD\_RESET Transmitted confirmation to the management application layer.



**6.12.3.5 SL\_IR\_TIR4:Completed state**

This state shall request that idle dwords be transmitted by repeatedly sending Transmit Idle Dword messages to the SL\_IR transmitter.

**6.12.4 SL\_IR\_RIF (receive IDENTIFY address frame) state machine****6.12.4.1 SL\_IR\_RIF state machine overview**

The SL\_IR\_RIF state machine receives an IDENTIFY address frame and checks the IDENTIFY address frame to determine if the frame should be accepted or discarded by the link layer.

This state machine consists of the following states:

- a) SL\_IR\_RIF1:Idle (see 6.12.4.2) (initial state);
- b) SL\_IR\_RIF2:Receive\_Identify\_Frame (see 6.12.4.3); and
- c) SL\_IR\_RIF3:Completed (see 6.12.4.4).

This state machine shall start in the SL\_IR\_RIF1:Idle state. This state machine shall transition to the SL\_IR\_RIF1:Idle state from any other state after receiving a Phy Layer Not Ready confirmation.

**6.12.4.2 SL\_IR\_RIF1:Idle state****6.12.4.2.1 State description**

This state waits for an SOAF to be received from the physical link, indicating an address frame is arriving.

**6.12.4.2.2 Transition SL\_IR\_RIF1:Idle to SL\_IR\_RIF2:Receive\_Identify\_Frame**

This transition shall occur after:

- a) a Start SL\_IR Receiver confirmation is received; and
- b) an SOAF Received message is received.

**6.12.4.3 SL\_IR\_RIF2:Receive\_Identify\_Frame state****6.12.4.3.1 State description**

This state receives the dwords of an address frame and the EOAF.

If this state receives an SOAF Received message, then this state shall discard the address frame in progress, send an Address Frame Failed confirmation to the management application layer to indicate that an invalid address frame was received, and start receiving the new address frame.

If this state receives more than eight Data Dword Received messages (i.e., 32 bytes) after an SOAF Received message and before an EOAF Received message, then this state shall discard the address frame and send an Address Frame Failed confirmation to the management application layer to indicate that an invalid address frame was received.

If this state receives an Invalid Dword Received message or an ERROR Received message after an SOAF Received message and before an EOAF Received message, then this state shall either:

- a) ignore the invalid dword or ERROR; or
- b) discard the address frame in progress and send an Address Frame Failed confirmation to the management application layer to indicate that an invalid address frame was received.

After receiving an EOAF Received message, this state shall check if the received frame is a valid IDENTIFY address frame.

This state shall accept an IDENTIFY address frame and send an Identify Received message to the SL\_IR\_IRC state machine if:

- a) the ADDRESS FRAME TYPE field is set to 0h (i.e., IDENTIFY);
- b) the number of bytes between the SOAF and EOAF is 32; and
- c) no CRC Error Occurred message was received for this IDENTIFY address frame,

otherwise this state shall discard the address frame and send an Address Frame Failed confirmation to the management application layer to indicate that an invalid address frame was received.

#### **6.12.4.3.2 Transition SL\_IR\_RIF2:Receive\_Identify\_Frame to SL\_IR\_RIF3:Completed**

This transition shall occur:

- a) after sending an Identify Received message.

#### **6.12.4.4 SL\_IR\_RIF3:Completed state**

This state waits for a Phy Layer Not Ready confirmation.

### **6.12.5 SL\_IR\_IRC (identification and hard reset control) state machine**

#### **6.12.5.1 SL\_IR\_IRC state machine overview**

The SL\_IR\_IRC state machine ensures that IDENTIFY address frames have been both received and transmitted before enabling the rest of the link layer, and notifies the link layer if a HARD\_RESET primitive sequence is received before an IDENTIFY address frame has been received.

This state machine consists of the following states:

- a) SL\_IR\_IRC1:Idle (see 6.12.5.2) (initial state);
- b) SL\_IR\_IRC2:Wait (see 6.12.5.3); and
- c) SL\_IR\_IRC3:Completed (see 6.12.5.4).

This state machine shall start in the SL\_IR\_IRC1:Idle state. This state machine shall transition to the SL\_IR\_IRC1:Idle state from any other state after receiving a Phy Layer Not Ready confirmation.

#### **6.12.5.2 SL\_IR\_IRC1:Idle state**

##### **6.12.5.2.1 State description**

This state waits for the link layer to be enabled. Upon entry into this state, this state shall:

- a) send an Enable Disable SAS Link (Disable) message to the SL state machines (see 6.18) or XL state machine (see 6.19) halting any link layer activity; and
- b) send a Phy Disabled confirmation to the SL\_P\_C state machine, port layer, and the management application layer indicating that the phy is not ready for use.

##### **6.12.5.2.2 Transition SL\_IR\_IRC1:Idle to SL\_IR\_IRC2:Wait**

This transition shall occur:

- a) after a Start SL\_IR Receiver confirmation is received.

**6.12.5.3 SL\_IR\_IRC2:Wait state****6.12.5.3.1 State description**

This state ensures that an IDENTIFY address frame has been received by the SL\_IR\_RIF state machine and that an IDENTIFY address frame has been transmitted by the SL\_IR\_TIR state machine before enabling the rest of the link layer. The IDENTIFY address frames may be transmitted and received on the physical link in any order.

After this state receives an Identify Received message, this state shall send a Stop SNTT request to the phy layer.

NOTE 28 - If multiplexing is enabled, then each SL\_IR\_IRC state machine sends a Stop SNTT request to the phy layer. The phy layer honors the first request and ignores the second request.

After this state receives an Identify Transmitted message, this state shall initialize and start the Receive Identify Timeout timer. If an Identify Received message is received before the Receive Identify Timeout timer expires, then this state shall:

- a) send an Identification Sequence Complete confirmation to the management application layer, with arguments carrying the contents of the incoming IDENTIFY address frame;
- b) send an Enable Disable SAS Link (Enable) message to the SL state machines (see 6.18) in a SAS logical phy or the XL state machine (see 6.19) in an expander logical phy indicating that the rest of the link layer may start operation; and
- c) send a Phy Enabled confirmation to the SL\_P\_C state machine (see 6.14.5), port layer, and the management application layer indicating that the phy is ready for use.

If the Receive Identify Timeout timer expires before an Identify Received message is received, then this state shall send an Identify Timeout confirmation to the management application layer to indicate that an identify timeout occurred.

If this state receives a HARD\_RESET Received message before an Identify Received message is received, then this state shall send a HARD\_RESET Received confirmation to the port layer and the management application layer and a Stop SNTT request to the phy layer.

If this state receives a HARD\_RESET Received message after an Identify Received message is received, then the HARD\_RESET Received message shall be ignored.

**6.12.5.3.2 Transition SL\_IR\_IRC2:Wait to SL\_IR\_IRC3:Completed**

This transition shall occur:

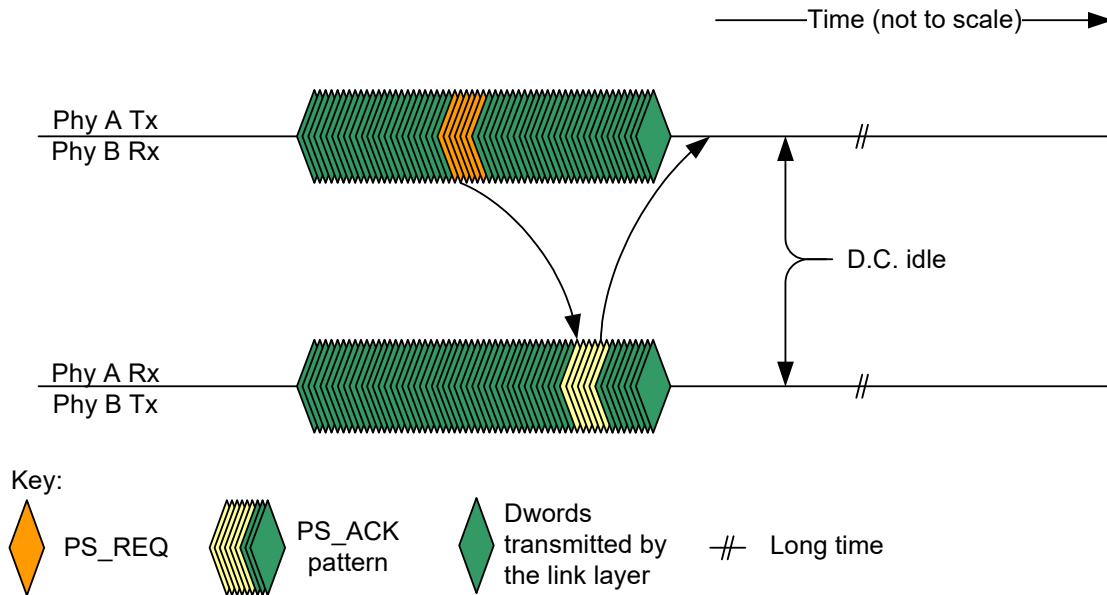
- a) after sending a HARD\_RESET Received confirmation, Identify Timeout confirmation, or an Identification Sequence Complete and a Phy Enabled confirmation.

**6.12.5.4 SL\_IR\_IRC3:Completed state**

This state waits for a Phy Layer Not Ready confirmation.

## 6.13 Entering a low phy power condition

Figure 148 shows the sequence to transition from the active phy power condition to a low phy power condition.



**Figure 148 –Transitioning from the active phy power condition to a low phy power condition**

The transition to a low phy power condition is acknowledged using a PS\_ACK pattern as defined in table 177.

**Table 177 – PS\_ACK pattern**

Pattern	Description
PS_ACK pattern	Sequence of: 1) PS_ACK primitive sequence (see 6.2.6.11); and 2) three idle dwords.

After sending a PS\_ACK pattern the transmitter may continue to transmit a vendor specific number of idle dwords.

After sending a PS\_REQ primitive sequence, if no PS\_ACK primitive sequence is received before the Power Condition Request Timeout timer (see table 189 and table 192) expires, then the transition to the low phy power condition is aborted and the phy remains in the active phy power condition.

## 6.14 Power control and SL\_P (link layer power control) state machines

### 6.14.1 Power source device

An expander device or SAS initiator device that is capable of processing requests for additional consumption of power (i.e., a power source device):

- indicates support for processing requests for additional consumption of power by setting the POWER CAPABLE field to 10b in the IDENTIFY address frame (see 6.10.2);

- b) notifies the management application layer when a power consumer device is requesting the consumption of power beyond the typical peak power used while in the active power condition;
- c) manages power consumption grants sent to a power consumer device; and
- d) processes power consumption requests received from a power consumer device.

The processing of requests for additional consumption of power is enabled in a power source device that:

- a) sets the POWER CAPABLE field to 10b in the IDENTIFY address frame; and
- b) receives an IDENTIFY address frame with the POWER CAPABLE field set to 01b.

A power source device that sets the POWER CAPABLE field to 10b in the IDENTIFY address frame on any phy shall set the POWER CAPABLE field to 10b in the IDENTIFY address frame for all phys within that power source device.

A power source device uses PWR\_REQ, PWR\_GRANT, PWR\_DONE, and PWR\_ACK.

A power source device shall on a phy:

- a) only make requests to a SL\_P\_S state machine if the phy is enabled;
- b) exit any low phy power condition (see 4.10.1) before requesting a grant be sent to a power consumer device; and
- c) disable any enabled low phy power condition until the additional consumption of power is complete and then re-enable any low phy power condition that was disabled.

#### 6.14.2 Power consumer device

A SAS target device that is capable of requesting additional consumption of power (i.e., a power consumer device):

- a) indicates support for requesting additional consumption of power by setting the POWER CAPABLE field to 01b in the IDENTIFY address frame (see 6.10.2);
- b) receives requests from the management application layer when consumption of power beyond the typical peak power used while in the active power condition is required;
- c) manages power consumption requests sent to a power source device; and
- d) processes power consumption grants received from a power source device.

The requesting of additional consumption of power is enabled in a power consumer device that:

- a) sets the POWER CAPABLE field to 01b in the IDENTIFY address frame; and
- b) receives an IDENTIFY address frame with the POWER CAPABLE field set to 10b.

A power consumer device that sets the POWER CAPABLE field to 01b in the IDENTIFY address frame on any phy shall set the POWER CAPABLE field to 01b in the IDENTIFY address frame for all phys within that power consumer device.

A power consumer device shall only request the additional consumption of power on one phy at a time.

A power consumer device uses PWR\_REQ, PWR\_GRANT, PWR\_DONE, and PWR\_ACK.

A power consumer device shall on a phy:

- a) exit any low phy power condition (see 4.10.1) before requesting consumption of power beyond the typical peak power; and
- b) disable any enabled low phy power conditions until consumption of power beyond the typical peak power is complete and then re-enable any low phy power condition that was disabled.

#### 6.14.3 NOTIFY (ENABLE SPINUP) usage

A power source device may use NOTIFY (ENABLE SPINUP) to manage power on a SAS target device (see 6.2.5.3.2).

A power source device shall use NOTIFY (ENABLE SPINUP) to manage power on a SAS target device (see 6.2.5.3.2) if that power source device:

- a) is not capable of managing requests for additional consumption of power; or

- b) receives an IDENTIFY address frame with the POWER CAPABLE field set to 00b.

#### 6.14.4 SL\_P\_S (link layer power source device) state machine

##### 6.14.4.1 SL\_P\_S state machine overview

The SL\_P\_S state machine consists of the following states:

- a) SL\_P\_S\_1:Idle (see 6.14.4.3) (initial state);
- b) SL\_P\_S\_2:Wait\_Grant (see 6.14.4.4); and
- c) SL\_P\_S\_3:Wait\_Done (see 6.14.4.5).

This state machine shall start in the SL\_P\_S\_1:Idle state after power on.

This state machine receives the following requests from the management application layer:

- a) Transmit NOTIFY;
- b) Cancel; and
- c) Power Use Granted.

This state machine sends the following confirmations to the management application layer:

- a) Additional Power Request;
- b) Power Done Timeout; and
- c) Power Use Complete.

This state machine receives the following messages from the SL\_CC link layer state machine and the XL link layer state machine:

- a) Idle State Condition (Active); and
- b) Idle State Condition (Inactive).

This state machine sends the following messages to the SL\_CC link layer state machine and the XL link layer state machine:

- a) Transmit Power Request (PWR\_ACK); and
- b) Transmit Power Request (PWR\_GRANT).

Any message, request, or confirmation received by a state that is not referred to in the description of that state shall be ignored.

This state machine shall maintain the timers listed in table 178.

**Table 178 – SL\_P\_S state machine timers**

Timer	Initial value
ACK Timeout timer	1 ms
Power Done timer	Depending on the protocol used by the port: <ul style="list-style-type: none"> <li>a) for expander ports, the value in the POWER DONE TIMEOUT field in the SMP REPORT GENERAL function (see 9.4.3.4); and</li> <li>b) for SSP initiator ports, a vendor specific value.</li> </ul>

Figure 149 shows the SL\_P\_S state machine.

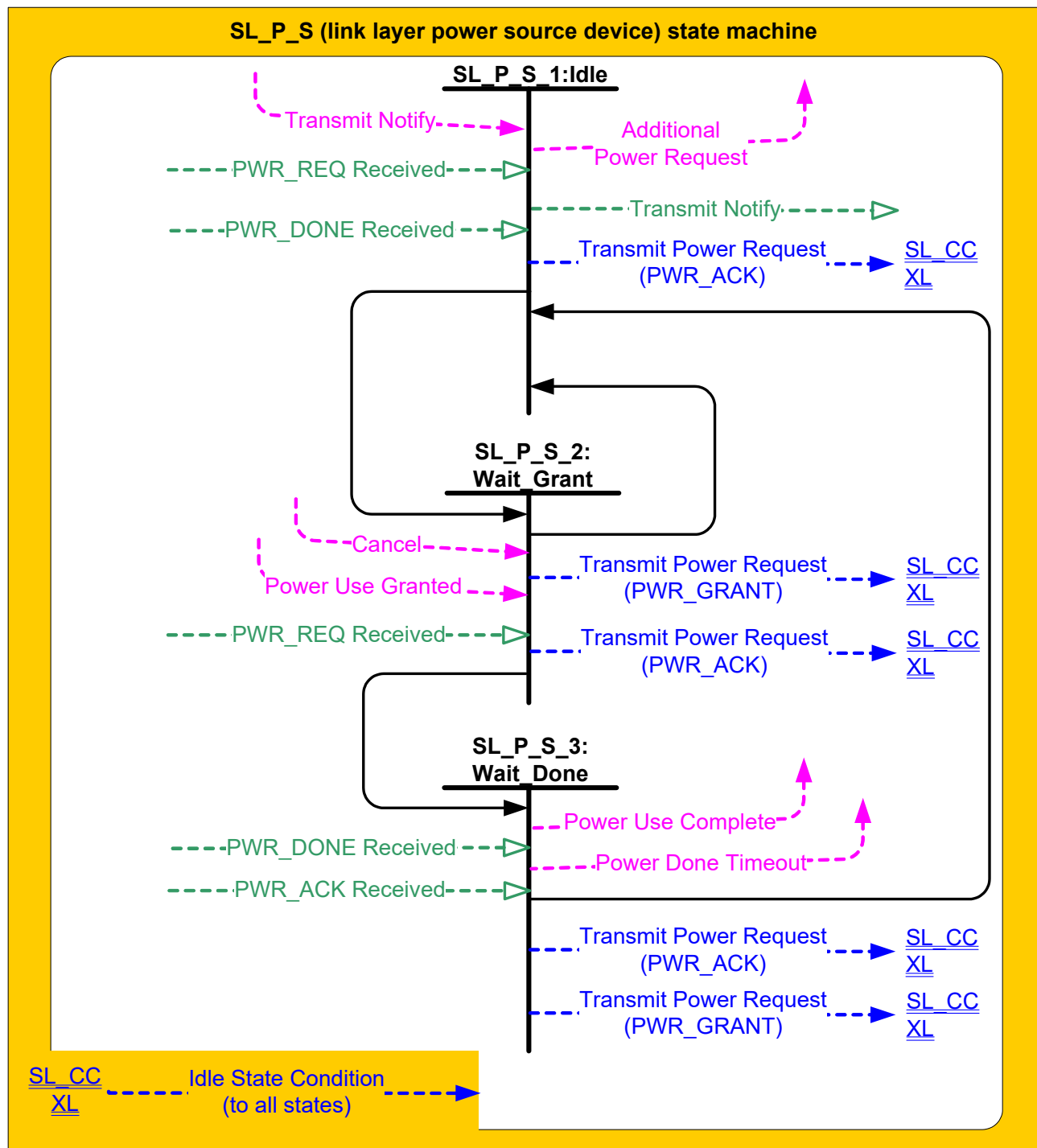


Figure 149 –SL\_P\_S (link layer power source device) state machine

#### 6.14.4.2 SL\_P\_S transmitter and SL\_P\_S receiver

The SL\_P\_S transmitter receives the following message from the SL\_P\_S state machine specifying primitive sequences to transmit:

- a) Transmit NOTIFY (Enable Spinup).

The SL\_P\_S receiver sends the following messages to the SL\_P\_S state machine indicating the primitive sequence received from the SP\_DWS receiver (see 5.15.2) and the SP\_PS receiver (see 5.16.2):

- a) PWR\_REQ Received;
- b) PWR\_DONE Received; and
- c) PWR\_ACK Received.

The SL\_P\_S receiver shall ignore all other dwords.

The SL\_P\_S transmitter relationship to other transmitters is defined in 4.3.2. The SL\_P\_S receiver relationship to other receivers is defined in 4.3.3.

#### **6.14.4.3 SL\_P\_S\_1:Idle state**

##### **6.14.4.3.1 State description**

If this state receives a Transmit NOTIFY request, then this state shall send a Transmit NOTIFY (Enable Spinup) message to the SL\_P\_S transmitter.

If this state receives a PWR\_REQ Received message and the last Idle State Condition message received contained an Active argument, then this state shall send:

- a) a Transmit Power Request (PWR\_ACK) message to the SL\_CC0:Idle state or XL0:Idle state; and
- b) an Additional Power Request confirmation to the management application layer.

If this state receives a PWR\_REQ Received message and the last Idle State Condition message received contained an Inactive argument, then this state shall:

- 1) wait until receiving an Idle State Condition (Active) message; and
- 2) send:
  - A) a Transmit Power Request (PWR\_ACK) message to the SL\_CC0:Idle state or XL0:Idle state; and
  - B) an Additional Power Request confirmation to the management application layer.

If this state receives a PWR\_DONE Received message and the last Idle State Condition message received contained an Active argument, then this state shall send, then this state should send a Transmit Power Request (PWR\_ACK) message to the SL\_CC0:Idle state or XL0:Idle state.

If this state receives a PWR\_DONE Received message and the last Idle State Condition message received contained an Inactive argument, then this state shall:

- 1) wait until receiving an Idle State Condition (Active) message; and
- 2) send a Transmit Power Request (PWR\_ACK) message to the SL\_CC0:Idle state or XL0:Idle state.

Implementations compliant with SPL-3 may not send a Transmit Power Request (PWR\_ACK) message in response to a PWR\_DONE Received message while in this state.

##### **6.14.4.3.2 Transition SL\_P\_S\_1:Idle to SL\_P\_S\_2:Wait\_Grant**

This transition shall occur after sending:

- a) a Transmit Power Request (PWR\_ACK) message to the SL\_CC0:Idle state or XL0:Idle state; and
- b) an Additional Power Request confirmation to the management application layer.

#### **6.14.4.4 SL\_P\_S\_2:Wait\_Grant state**

##### **6.14.4.4.1 State description**

This state waits for a Power Use Granted request from the management application layer.

If this state receives a Power Use Granted request and the last Idle State Condition message received contained an Active argument, then this state shall send a Transmit Power Request (PWR\_GRANT) message to the SL\_CC0:Idle state or XL0:Idle state.



If this state receives a Power Use Granted request and the last Idle State Condition message received contained an Inactive argument, then this state shall:

- 1) wait until receiving an Idle State Condition (Active) message; and
- 2) send a Transmit Power Request (PWR\_GRANT) message to the SL\_CC0:Idle state or XL0:Idle state.

If this state receives a PWR\_REQ Received message and the last Idle State Condition message received contained an Active argument, then this state shall send a Transmit Power Request (PWR\_ACK) message to the SL\_CC0:Idle state or XL0:Idle state.

If this state receives a PWR\_REQ Received message and the last Idle State Condition message received contained an Inactive argument, then this state shall:

- 1) wait until receiving an Idle State Condition (Active) message; and
- 2) send a Transmit Power Request (PWR\_ACK) message to the SL\_CC0:Idle state or XL0:Idle state.

#### **6.14.4.4.2 Transition SL\_P\_S\_2:Wait\_Grant to SL\_P\_S\_1:Idle**

This transition shall occur:

- a) after receiving a Cancel request.

#### **6.14.4.4.3 Transition SL\_P\_S\_2:Wait\_Grant to SL\_P\_S\_3:Wait\_Done**

This transition shall occur:

- a) after sending a Transmit Power Request (PWR\_GRANT) message to the SL\_CC0:Idle state or XL0:Idle state.

#### **6.14.4.5 SL\_P\_S\_3:Wait\_Done state**

##### **6.14.4.5.1 State description**

This state waits for the power consumer device to indicate the use of additional power consumption is complete.

On entry this state shall:

- a) initialize and start the Power Done timer; and
- b) initialize and start the ACK Timeout timer.

If this state receives a PWR\_ACK Received message, then this state shall stop the ACK Timeout timer.

If the ACK Timeout timer expires and the last Idle State Condition message received contained an Active argument, then this state shall:

- a) send a Transmit Power Request (PWR\_GRANT) message to the SL\_CC0:Idle state or XL0:Idle state; and
- b) initialize and start the ACK Timeout timer.

If the ACK Timeout timer expires and the last Idle State Condition message received contained an Inactive argument, then this state shall:

- 1) wait until receiving an Idle State Condition (Active) message;
- 2) send a Transmit Power Request (PWR\_GRANT) message to the SL\_CC0:Idle state or XL0:Idle state; and
- 3) initialize and start the ACK Timeout timer.

If this state receives a PWR\_DONE Received message and the last Idle State Condition message received contained an Active argument, then this state shall:

- a) send a Transmit Power Request (PWR\_ACK) message to the SL\_CC0:Idle state or XL0:Idle state; and
- b) send a Power Use Complete confirmation to the management application layer.

If this state receives a PWR\_DONE Received message and the last Idle State Condition message received contained an Inactive argument, then this state shall:

- 1) wait until receiving an Idle State Condition (Active) message; and
- 2) send:
  - A) a Transmit Power Request (PWR\_ACK) message to the SL\_CC0:Idle state or XL0:Idle state; and
  - B) a Power Use Complete confirmation to the management application layer.

If the Power Done timer expires, then this state shall send a Power Done Timeout confirmation to the management application layer.

#### 6.14.4.5.2 Transition SL\_P\_S\_3:Wait\_Done to SL\_P\_S\_1:Idle

This transition shall occur after sending:

- a) a Power Done Timeout confirmation to the management application layer; or
- b) a Transmit Power Request (PWR\_ACK) message to the SL\_CC0:Idle state or XL0:Idle state and sending a Power Use Complete confirmation to the management application layer.

### 6.14.5 SL\_P\_C (link layer power consumer device) state machine

#### 6.14.5.1 SL\_P\_C state machine overview

The SL\_P\_C state machine consists of the following states:

- a) SL\_P\_C\_1:Idle (see 6.14.5.3) (initial state);
- b) SL\_P\_C\_2:Request\_Power (see 6.14.5.4);
- c) SL\_P\_C\_3:Wait\_Grant (see 6.14.5.5); and
- d) SL\_P\_C\_4:Wait\_Done (see 6.14.5.6).

This state machine shall start in the SL\_P\_C\_1:Idle state after power on.

This state machine receives the following requests from the SA\_PC SCSI application layer state machine:

- a) Request Additional Power; and
- b) Power Use Complete.

This state machine sends the following confirmations to the SA\_PC SCSI application layer state machine:

- a) Power Use Granted; and
- b) Power Request Failed.

This state machine receives the following confirmations from the SL\_IR\_IRC link layer state machine:

- a) Phy Enabled; and
- b) Phy Disabled.

This state machine receives the following messages from the SL\_CC link layer state machine:

- a) Idle State Condition (Active); and
- b) Idle State Condition (Inactive).

This state machine sends the following messages to the SL\_CC link layer state machine:

- a) Transmit Power Request (PWR\_ACK);
- b) Transmit Power Request (PWR\_REQ); and
- c) Transmit Power Request (PWR\_DONE).

Any message, request, or confirmation received by a state that is not referred to in the description of that state shall be ignored.

This state machine shall maintain the timers listed in table 179.

**Table 179 – SL\_P\_C state machine timers**

Timer	Initial value
ACK Timeout timer	1 ms
Power Grant timer	The value in the POWER GRANT TIMEOUT field in the Shared Port Control mode page (see 9.2.7.6)

Figure 150 shows the SL\_P\_C state machine.

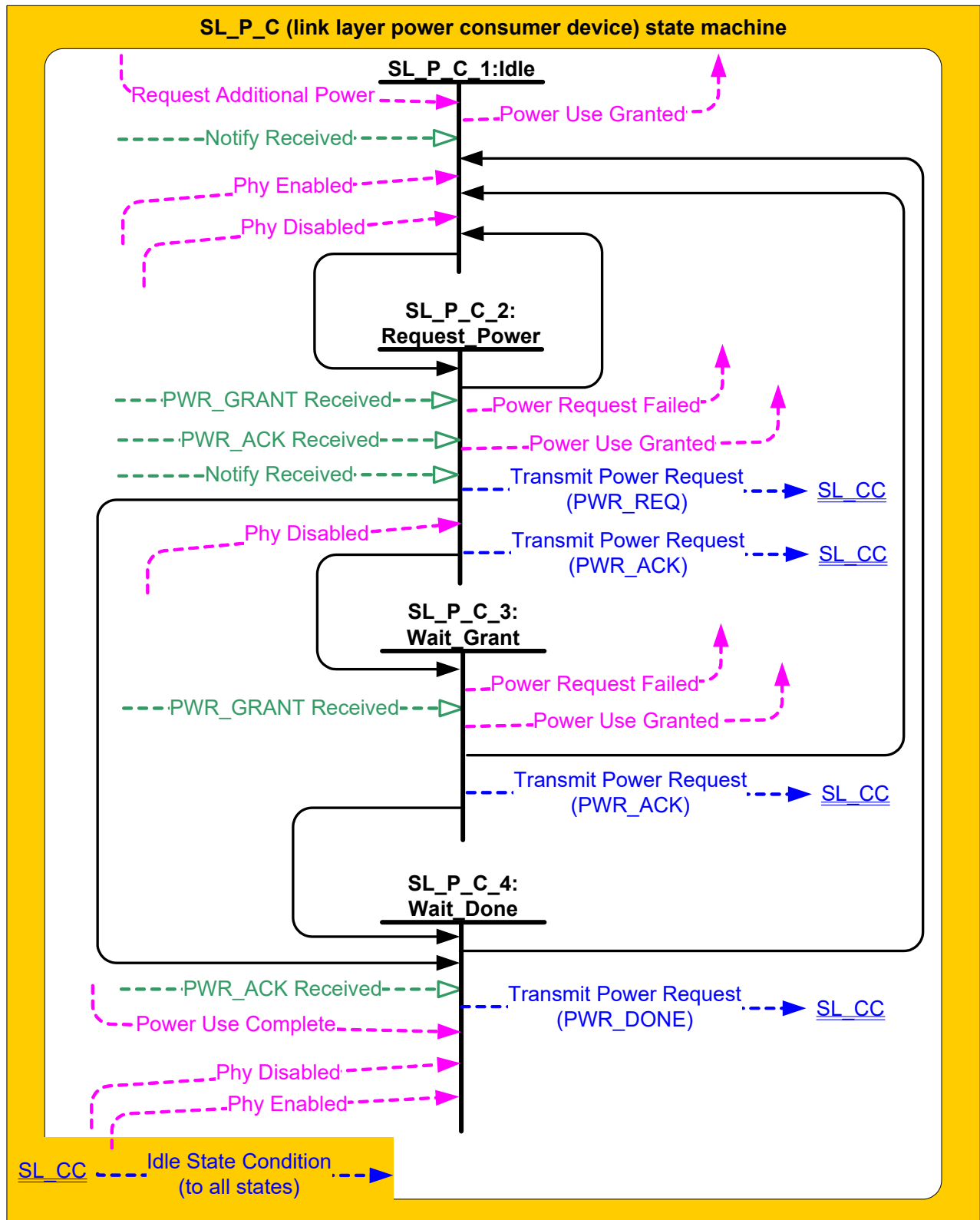


Figure 150 –SL\_P\_C (link layer power consumer device) state machine

**6.14.5.2 SL\_P\_C receiver**

The SL\_P\_C receiver sends the following messages to the SL\_P\_C state machine indicating the primitive sequence received from the SP\_DWS receiver (see 5.15.2) and the SP\_PS receiver (see 5.16.2):

- a) NOTIFY Received (Enable Spinup);
- b) PWR\_GRANT Received; and
- c) PWR\_ACK Received.

The SL\_P\_C receiver shall ignore all other dwords.

The SL\_P\_C receiver relationship to other receivers is defined in 4.3.3.

**6.14.5.3 SL\_P\_C\_1:Idle state****6.14.5.3.1 State description**

If this state receives a NOTIFY Received (Enable Spinup) message, then this state shall send a Power Use Granted confirmation to the SA\_PC state machine (see 9.2.10.2).

In this state, the phy is enabled:

- a) if no Phy Disabled confirmation has been received since this state is entered; or
- b) if a Phy Enable confirmation has been received after the last Phy Disabled confirmation was received.

In this state, the phy is disabled:

- a) if no Phy Enabled confirmation has been received since this state is entered with a Phy Disabled argument; or
- b) a Phy Disabled confirmation is received without a subsequent Phy Enabled confirmation.

In this state, requesting power is enabled when:

- a) the phy is enabled; and
- b) the last Idle State Condition message received contained an Active argument.

**6.14.5.3.2 Transition SL\_P\_C\_1:Idle to SL\_P\_C\_2:Request\_Power**

If requesting power is enabled, then this transition shall occur:

- a) after receiving a Request Additional Power request.

If requesting power is disabled, then this transition shall occur after:

- a) requesting power is enabled; and
- b) receiving a Request Additional Power request.

**6.14.5.4 SL\_P\_C\_2:Request\_Power state****6.14.5.4.1 State description**

On entry this state shall:

- a) send a Transmit Power Request (PWR\_REQ) message to the SL\_CC0:Idle state; and
- b) initialize and start the ACK Timeout timer.

If this state receives a NOTIFY Received (Enable Spinup) message, then this state shall send a Power Use Granted confirmation to the SA\_PC state machine (see 9.2.10.2).

If this state receives a PWR\_GRANT Received message and the last Idle State Condition message received contained an Active argument, then this state shall send:

- a) a Transmit Power Request (PWR\_ACK) message to the SL\_CC0:Idle state; and
- b) a Power Use Granted confirmation to the SA\_PC state machine (see 9.2.10.2).

If this state receives a PWR\_GRANT Received message and the last Idle State Condition message received contained an Inactive argument, then this state shall:

- 1) wait until receiving an Idle State Condition (Active) message; and
- 2) send:
  - A) a Transmit Power Request (PWR\_ACK) message to the SL\_CC0:Idle state; and
  - B) a Power Use Granted confirmation to the SA\_PC state machine (see 9.2.10.2).

If the ACK Timeout timer expires, then this state shall send a Power Request Failed (ACK Timeout) confirmation to the SA\_PC state machine (see 9.2.10.2).

If this state receives a Phy Disabled confirmation, then this state shall send a Power Request Failed (Phy Disabled) confirmation to the SA\_PC state machine (see 9.2.10.2).

#### 6.14.5.4.2 Transition SL\_P\_C\_2:Request\_Power to SL\_P\_C\_1:Idle

This transition shall occur after sending:

- a) a Power Request Failed confirmation to the SA\_PC state machine; or
- b) a Power Use Granted confirmation to the SA\_PC state machine that resulted from the receipt NOTIFY Received (Enable Spinup) message.

If a Phy Disabled confirmation is the cause of this transition, then this transition shall include a Phy Disabled argument.

#### 6.14.5.4.3 Transition SL\_P\_C\_2:Request\_Power to SL\_P\_C\_3:Wait\_Grant

This transition shall occur:

- a) after receiving a PWR\_ACK Received message.

#### 6.14.5.4.4 Transition SL\_P\_C\_2:Request\_Power to SL\_P\_C\_4:Wait\_Done

This transition shall occur after sending:

- a) a Transmit Power Request (PWR\_ACK) message to the SL\_CC0:Idle state; and
- b) a Power Use Granted confirmation to the SA\_PC state machine (see 9.2.10.2).

#### 6.14.5.5 SL\_P\_C\_3:Wait\_Grant state

##### 6.14.5.5.1 State description

This state waits for the power source device to allow additional consumption of power.

On entry this state shall initialize and start the Power Grant timer.

If this state receives a PWR\_GRANT Received message and the last Idle State Condition message received contained an Active argument, then this state shall:

- a) send a Transmit Power Request (PWR\_ACK) message to the SL\_CC0:Idle state; and
- b) send a Power Use Granted confirmation to the SA\_PC state machine (see 9.2.10.2).

If this state receives a PWR\_GRANT Received message and the last Idle State Condition message received contained an Inactive argument, then this state shall:

- 1) wait until receiving an Idle State Condition (Active) message; and
- 2) send:
  - A) a Transmit Power Request (PWR\_ACK) message to the SL\_CC0:Idle state; and
  - B) a Power Use Granted confirmation to the SA\_PC state machine (see 9.2.10.2).

If the Power Grant timer expires, then this state shall send a Power Request Failed (Grant timeout) confirmation to the SA\_PC state machine (see 9.2.10.2).

**6.14.5.5.2 Transition SL\_P\_C\_3:Wait\_Grant to SL\_P\_C\_1:Idle**

This transition shall occur:

- a) after sending a Power Request Failed confirmation to the SA\_PC state machine (see 9.2.10.2).

**6.14.5.5.3 Transition SL\_P\_C\_3:Wait\_Grant to SL\_P\_C\_4:Wait\_Done**

This transition shall occur after sending:

- a) a Transmit Power Request (PWR\_ACK) message to the SL\_CC0:Idle state; and
- b) a Power Use Granted confirmation to the SA\_PC state machine (see 9.2.10.2).

**6.14.5.6 SL\_P\_C\_4:Wait\_Done state****6.14.5.6.1 State description**

This state waits for the management application layer to indicate that it no longer requires additional consumption of power.

In this state the phy is enabled:

- a) if no Phy Disabled confirmation has been received since this state is entered; or
- b) if a Phy Enable confirmation has been received after the last Phy Disabled confirmation was received.

If this state:

- a) receives a Power Use Complete request;
- b) the phy is enabled; and
- c) the last Idle State Condition received contained an Active argument,

then this state shall send a Transmit Power Request (PWR\_DONE) message to the SL\_CC0:Idle state.

If this state receives a Power Use Complete request and either phy is not enabled or the last Idle State Condition received contained an Inactive argument, then this state shall:

- 1) wait until:
  - A) receiving an Idle State (Active) message; and
  - B) the phy is enabled;

and

- 2) send a Transmit Power Request (PWR\_DONE) message to the SL\_CC0:Idle state.

If this state sends a Transmit Power Request (PWR\_DONE) message to the SL\_CC0:Idle state, then this state shall initialize and start the ACK Timeout timer.

If the ACK Timeout timer expires, then this state shall, at least one time:

- 1) send a Transmit Power Request (PWR\_DONE) message to the SL\_CC0:Idle state after:
  - A) the phy is enabled; and
  - B) the last Idle State Condition message received contains an Active argument;

and

- 2) initialize and start the ACK Timeout timer.

The number of times this state waits for an acknowledgement from the power source device is vendor specific.

**6.14.5.6.2 Transition SL\_P\_C\_4:Wait\_Done to SL\_P\_C\_1:Idle**

This transition shall occur after:

- a) the vendor specific number of failed attempts to receive a PWR\_ACK Received message; or
- b) receiving a PWR\_ACK Received message.

## 6.15 SAS domain changes (Broadcast (Change) usage)

An expander device shall originate Broadcast (Change) from at least one phy in each of its expander ports other than the expander port that is the cause for originating Broadcast (Change).

Expander devices shall originate Broadcast (Change) for the following expander phy-related reasons:

- a) after an expander phy's SP state machine transitions from the SP15:SAS\_PHY\_Ready state, SP22:SATA\_PHY\_Ready state, SP31:SAS\_PS\_Low\_Phy\_Power state, SP32:SAS\_PS\_ALIGN0 state, or SP33:SAS\_PS\_ALIGN1 state to the SP0:OOB\_COMINIT state (see 5.14);

NOTE 29 - This occurs when the expander phy is reset or disabled with the SMP PHY CONTROL function DISABLE, LINK RESET, HARD RESET, or TRANSMIT SATA PORT SELECTION SIGNAL phy operations (see 9.4.3.28) as well as when dword synchronization is lost for any other reason.

- b) after an expander phy's SP state machine reaches the SP26:SATA\_SpinupHold state and sends a SATA Spinup Hold confirmation as defined in 5.14.8 and 5.21;
- c) after an expander phy's SP state machine sends a SATA Port Selector change confirmation to the link layer (see 5.14.3);
- d) after an expander phy completes the link reset sequence (see 6.11);
- e) after a virtual phy has been enabled or completed processing a reset requested by the SMP PHY CONTROL function LINK RESET or HARD RESET phy operations (see 9.4.3.28); and
- f) after an STP SATA bridge receives an initial Register - Device to host FIS (see 8.3.1).

In zoning expander devices with zoning enabled, forwarding Broadcasts is subject to restrictions defined in 4.8.5.

In zoning expander devices with zoning enabled, a Broadcast (Change) for an expander phy-related reason shall be originated from the source zone group of the expander phy causing the Broadcast (Change) or from zone group 1.

Expander devices shall originate Broadcast (Change) for the following expander device-related reasons:

- a) after a self-configuring expander device has changed its SELF CONFIGURING bit from one to zero in the SMP REPORT GENERAL response (see 9.4.3.4) as described in 4.6.4. In zoning expander devices with zoning enabled, the source zone group shall be 01h; and
- b) after a locked expander device is unlocked (i.e., a zoning expander device has changed its ZONE CONFIGURING bit from one to zero in the SMP REPORT GENERAL response) (see 4.8.6.5, 4.8.6.6, and 9.4.3.23), with the source zone group as specified in 4.8.6.5, 4.8.6.6, and 9.4.3.23.

Expander devices shall forward Broadcast (Change) after an expander phy receives Broadcast (Change).

For a virtual phy, if there is any time after a reset is originated during which connection requests to the attached SAS address result in connection responses of OPEN\_REJECT (NO DESTINATION), then the expander device shall originate the Broadcast (Change) twice, once at the start of the reset (i.e., when the SAS address becomes unavailable) and once at its completion (i.e., when the SAS address becomes available). If there is no such time window, then the expander device shall originate the Broadcast (Change) once.

SAS initiator ports may originate Broadcast (Change) to force other SAS initiator ports and expander ports to re-run the discover process. SAS target ports should not originate Broadcast (Change).

See 9.4.3.4 for details on counting Broadcast (Change) origination in an expander device.



## 6.16 Connections

### 6.16.1 Connections overview

A connection is opened between a SAS initiator port and a SAS target port before communication begins. A connection is established between one SAS initiator phy in the SAS initiator port and one SAS target phy in the SAS target port.

SSP initiator ports open SSP connections to transmit SCSI commands, task management functions, and transfer write data. SSP target ports open SSP connections to transfer read data, request write data, and transmit service responses.

SMP initiator ports open SMP connections to transmit SMP requests and receive SMP responses.

STP initiator ports and STP target ports open STP connections to transmit SATA frames. An STP target port in an expander device opens STP connections on behalf of SATA devices.

The OPEN address frame is used to request that a connection be opened (see 6.16.2.1). AIP primitive sequences, OPEN\_ACCEPT, and OPEN\_REJECT are the responses to an OPEN address frame (see 6.16.2.2). A BREAK primitive sequence is used to abort connection requests (see 6.16.7) and to break a connection (see 6.16.11). A CLOSE primitive sequence is used for orderly closing of a connection (see 6.16.9).

Connections use a single pathway from the SAS initiator phy to the SAS target phy. While a connection is open, only one pathway shall be used for that connection.

For STP connections, connections may be between the STP initiator port and an STP target port of an STP SATA bridge in an expander device. The SATA device behind the STP SATA bridge is not aware of SAS connection management.

A wide port may have separate connections on each of its logical phys.

### 6.16.2 Opening a connection

#### 6.16.2.1 Connection request

The OPEN address frame (see 6.10.3) is used to open a connection from a source port to a destination port using one source phy (i.e., one logical phy in the source port) and one destination phy (i.e., one logical phy in the destination port).

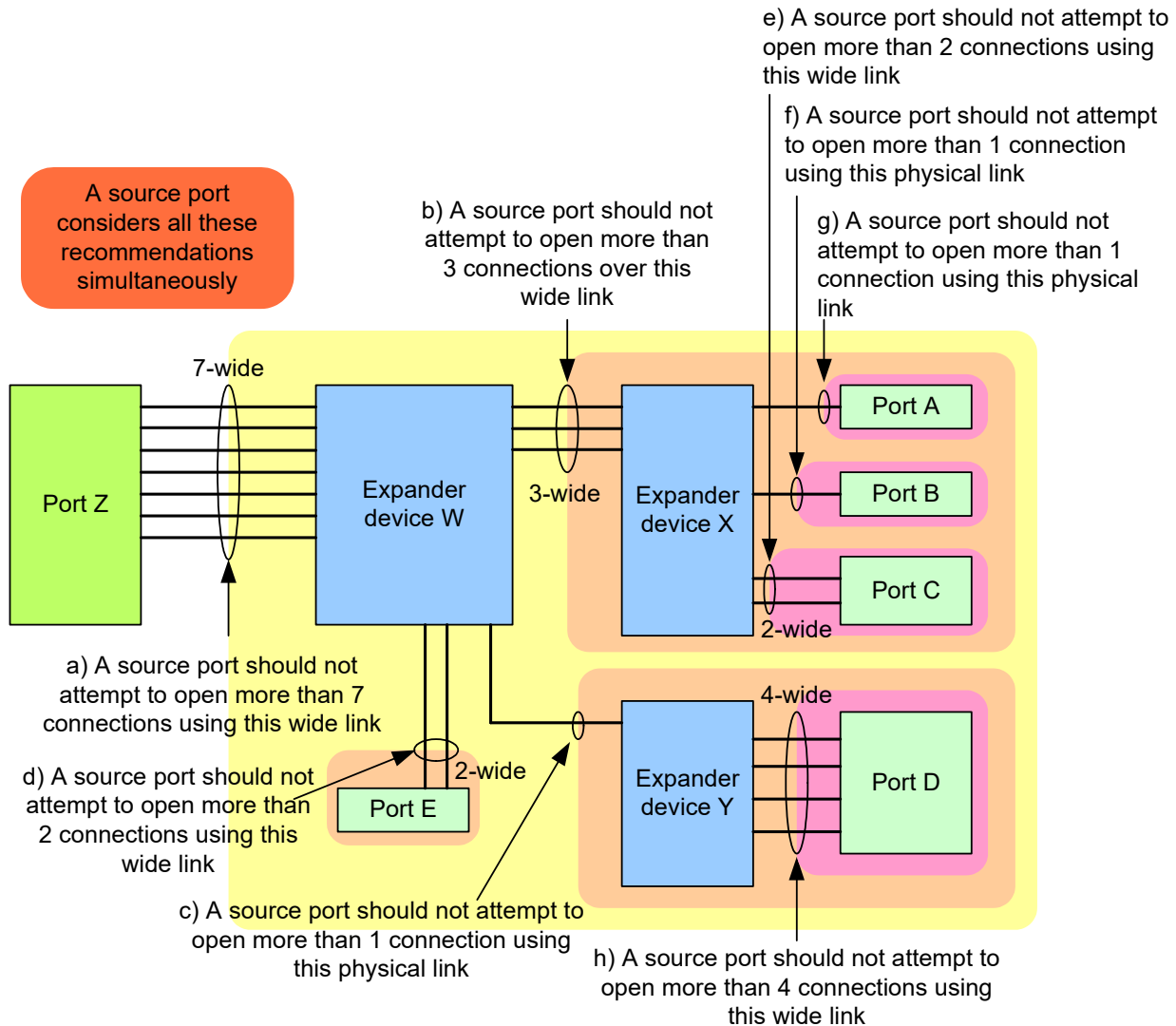
To make a connection request, the source port shall transmit an OPEN address frame through an available logical phy (i.e., the source phy). The source phy shall transmit idle dwords after the OPEN address frame until it receives a response or aborts the connection request with a BREAK primitive sequence.

After transmitting an OPEN address frame, the source phy shall initialize and start a 1 ms Open Timeout timer. Whenever an AIP is received, the source phy shall reinitialize and restart the Open Timeout timer. Source phys are not required to enforce a limit on the number of AIPs received before aborting the connection request. When any connection response is received, the source phy shall reinitialize the Open Timeout timer. If the Open Timeout timer expires before a connection response is received, then the source phy shall transmit a BREAK primitive sequence to abort the connection request (see 6.16.7).

The OPEN address frame flows through expander devices onto intermediate logical links. If an expander device on the pathway is unable to forward the connection request, then that expander device returns OPEN\_REJECT (see 6.16.5). If the OPEN address frame reaches the destination phy, then the destination phy returns either OPEN\_ACCEPT or OPEN\_REJECT unless the OPEN address frame passed an OPEN address frame from the destination phy with higher arbitration priority (see 6.16.4). Rate matching shall be used on any logical links in the pathway with negotiated logical link rates that are faster than the requested connection rate (see 6.17).

A wide port should not attempt to establish more connections to a destination port than the number of phys in the destination port or the number of phys in the narrowest logical link on the pathway to the destination port. A wide port should not attempt to establish more connections than the number of phys in the narrowest common logical link on the pathways to the destination ports of those connections. Additional requirements for STP connection requests are defined in 6.21.7. Additional requirements for SMP connection requests are defined in 6.22.4.

Figure 151 shows an example of the simultaneous connection recommendations for wide ports. Multiplexing is disabled in this example.



**Figure 151 –Example simultaneous connection recommendations for wide ports**

In figure 151, some of the recommendations are combined as follows:

- a) recommendations a), b), and e) together specify that port Z should not attempt to open more than two connections to port C;
- b) recommendations a), b), e), f), and g) together specify that if port Z has two connections open to ports A, B, or X, then it should not attempt to open more than one connection to port C. If port Z has six connections open to ports A, B, D, E, W, X, and Y, then port Z should not attempt to open more than one connection to port C; and

- c) recommendations a), c), and h) together specify that port Z should not attempt to open more than one connection to port D. If port Z has a connection open to port Y, then port Z should not attempt to open another connection to port D until the first connection is closed.

### 6.16.2.2 Results of a connection request

After a logical phy transmits an OPEN address frame, it shall expect one or more of the results listed in table 180.

**Table 180 – Connection results of a connection request**

Result	Description
Receive AIP	Arbitration in progress. While an expander device is trying to open a connection to the selected destination port (e.g., while it is internally arbitrating for access to an expander port), the expander device returns an AIP to the source phy. The source phy shall reinitialize and restart its Open Timeout timer each time the source phy receives an AIP.
Receive OPEN_ACCEPT	Connection request accepted. OPEN_ACCEPT is transmitted by the destination phy.
Receive OPEN_REJECT	Connection request rejected. OPEN_REJECT is transmitted by the destination phy or by an expander device in the partial pathway. The different versions are described in 6.2.6.10. See 4.4.3 for I_T nexus loss handling. See 6.10.3 for handling of OPEN_REJECT (CONNECTION RATE NOT SUPPORTED) for connection rates greater than 1.5 Gbit/s.
Receive OPEN address frame	If AIP has been received, then this indicates an overriding connection request.  If AIP has not yet been received, then this indicates two connection requests crossing on the logical link. Arbitration fairness determines which one wins (see 6.16.4).
Receive BREAK	The destination phy or an expander device in the partial pathway may reply with a BREAK primitive sequence indicating the connection is not being established. See 4.4.3 for I_T nexus loss handling.
Open Timeout timer expires	The source phy shall abort the connection request by transmitting a BREAK primitive sequence (see 6.16.7). See 4.4.3 for I_T nexus loss handling.

### 6.16.3 SMP frame priority

SMP frame priority is used by an expander device to prioritize OPEN address frames with the SAS PROTOCOL field set to SMP.

If an expander phy or a SAS phy supports SMP frame priority, then that phy shall set the SMP PRIORITY CAPABLE bit to one in the IDENTIFY address frame (see 6.10.2).

SMP frame priority is enabled on expander phy or a SAS phy that:

- sets the SMP PRIORITY CAPABLE bit to one in the IDENTIFY address frame; and
- receives an IDENTIFY address frame with the SMP PRIORITY CAPABLE bit set to one.

### 6.16.4 Arbitration fairness

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

Each SAS port and expander port shall include an Arbitration Wait Time timer that counts the time from the moment when the port makes a connection request until the request is accepted or rejected. The Arbitration Wait Time timer is in the port layer state machine (see 7.2.2). The Arbitration Wait Time timer shall count in microseconds from 0  $\mu$ s to 32 767  $\mu$ s and in milliseconds from 32 768  $\mu$ s to 32 767 ms + 32 768  $\mu$ s. The Arbitration Wait Time timer shall stop incrementing when its value reaches 32 767 ms + 32 768  $\mu$ s.

A SAS port (i.e., a SAS initiator port or a SAS target port) shall start the Arbitration Wait Time timer when it transmits the first OPEN address frame (see 6.10.3) for the connection request. When the SAS port retransmits the OPEN address frame (e.g., after losing arbitration and handling an inbound OPEN address frame), it shall set the ARBITRATION WAIT TIME field to the current value of the Arbitration Wait Time timer.

A SAS port should set the Arbitration Wait Time timer to zero when it transmits the first OPEN address frame for the connection request. A SAS initiator port or SAS target port may be unfair by setting the ARBITRATION WAIT TIME field in the OPEN address frame (see 6.10.3) to a higher value than its Arbitration Wait Time timer indicates. However, an unfair SAS port shall not set the ARBITRATION WAIT TIME field to a value greater than or equal to 8000h.

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

A SAS port shall stop the Arbitration Wait Time timer and set the Arbitration Wait Time timer to zero when the SAS port has no more frames to send.

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

- a) OPEN\_ACCEPT;
- b) OPEN\_REJECT (PROTOCOL NOT SUPPORTED);
- c) OPEN\_REJECT (ZONE VIOLATION);
- d) OPEN\_REJECT (RESERVED ABANDON 1);
- e) OPEN\_REJECT (RESERVED ABANDON 2);
- f) OPEN\_REJECT (RESERVED ABANDON 3);
- g) OPEN\_REJECT (STP RESOURCES BUSY); or
- h) OPEN\_REJECT (WRONG DESTINATION).

When an OPEN\_REJECT (RETRY), OPEN\_REJECT (RESERVED CONTINUE 0), or OPEN\_REJECT (RESERVED CONTINUE 1) is received:

- a) if the CONTINUE AWT bit is set to one in the Protocol Specific Port mode page (see 9.2.7.4), then a connection response of OPEN\_REJECT (RETRY), OPEN\_REJECT (RESERVED CONTINUE 0), or OPEN\_REJECT (RESERVED CONTINUE 1) shall not stop the Arbitration Wait Time timer and shall not set the Arbitration Wait Time timer to zero; or
- b) If the CONTINUE AWT bit is set to zero, then a SAS port shall stop the Arbitration Wait Time timer and set the Arbitration Wait Time timer to zero.

A SAS port should not stop the Arbitration Wait Time timer and set the Arbitration Wait Time timer to zero when the SAS port receives an incoming OPEN address frame that has priority over the outgoing OPEN address frame according to table 181, regardless of whether the SAS port replies with an OPEN\_ACCEPT or an OPEN\_REJECT.

When arbitrating for access to an outgoing expander port, the expander device shall select the connection request based on the rules described in 6.16.5.

If two connection requests pass on a logical link, then the logical phy shall determine the winner by comparing OPEN address frame field contents using the arbitration priority described in table 181.

**Table 181 – Arbitration priority for OPEN address frames passing on a logical link**

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

See 6.10.3 for details on the OPEN address frame, the ARBITRATION WAIT TIME field and the SOURCE SAS ADDRESS field.

### 6.16.5 Arbitration inside an expander device

#### 6.16.5.1 Expander logical phy arbitration requirements

An expander logical phy shall set its Request Path request High Priority argument to one when the expander logical phy requests a path after:

- a) the expander logical phy has forwarded an OPEN address frame to the logical link;
- b) the expander logical phy receives an OPEN address frame with higher arbitration priority (see 6.16.4); and
- c) the destination SAS address and connection rate of the received OPEN address frame are not equal to the source SAS address and connection rate of the transmitted OPEN address frame (see 6.19.4 and 6.19.8),

otherwise the expander logical phy shall set the High Priority argument to zero.

See the XL state machine (see 6.19) for detailed expander logical phy requirements.

#### 6.16.5.2 ECM arbitration requirements

##### 6.16.5.2.1 ECM arbitration requirements overview

The ECM shall arbitrate and assign or deny path resources for Request Path requests (see 4.5.6.3) from each expander logical phy.

Arbitration includes adherence to the SAS arbitration fairness algorithm and path recovery. Path recovery is used to avoid potential deadlock scenarios within the SAS topology by deterministically choosing which partial pathways to tear down to allow at least one connection to complete.

The ECM shall maintain an Arbitration Wait Time state machine variable for each Request Path request that the ECM is processing. The ECM shall initialize the Arbitration Wait Time state machine variable to the Arbitration Wait Time argument upon receiving the Request Path request and shall repeatedly increment the Arbitration Wait Time state machine variable until the ECM completes responding to the Request Path request.

The Source SAS Address argument and Connection Rate argument are set to values received in the received OPEN address frame.

The following are used by the ECM to compare Request Path requests:

- a) Source SAS Address argument;
- b) Connection Rate argument;
- c) High Priority argument; and
- d) Arbitration Wait Time state machine variable.

The High Priority argument is used to increase the priority of a Request Path request after a Backoff Retry response is sent by an expander logical phy (see 6.19.4).

When the ECM in an expander device receives a connection request:

- 1) if the destination SAS address is that of the expander device itself, then the ECM shall arbitrate for access to its SMP target port;
- 2) if:
  - A) the destination SAS address matches the SAS address attached to one or more of the expander logical phys; and
  - B) the ECM is:
    - a) not receiving Pause Phy responses from an expander logical phy associated with the connection request; or
    - b) receiving Pause Phy responses from an expander logical phy associated with the connection request and the phy identifier from the connection request is the same as the Phy Identifier argument of the Pause Phy responses,

then the ECM shall arbitrate for access to those expander logical phys;

- 3) if:
  - A) the destination SAS address matches an enabled SAS address in the expander route table for one or more expander logical phys that is using the table routing method; and
  - B) the ECM is:
    - a) not receiving Pause Phy responses from an expander logical phy associated with the connection request; or
    - b) receiving Pause Phy responses from an expander logical phy associated with the connection request and the phy identifier from the connection request is the same as the Phy Identifier argument of the Pause Phy responses,

then the ECM shall arbitrate for access to those expander logical phys;

and

- 4) if:
  - A) at least one expander logical phy is using the subtractive routing method;
  - B) the request did not come from one of those expander logical phys; and
  - C) the ECM is:
    - a) not receiving Pause Phy responses from an expander logical phy associated with the connection request; or
    - b) receiving Pause Phy responses from an expander logical phy associated with the connection request and the phy identifier from the connection request is the same as the Phy Identifier argument of the Pause Phy responses,

then the ECM shall arbitrate for access to one of those expander logical phys.

The ECM shall respond to each Request Path request by returning the following confirmations to the requesting expander logical phy while processing the Request Path request:

- a) Arbitrating (Normal) (see 6.16.5.2.2);
- b) Arbitrating (Waiting On Partial) (see 6.16.5.2.2);
- c) Arbitrating (Blocked On Partial) (see 6.16.5.2.2); and
- d) Arbitrating (Waiting On Connection) (see 6.16.5.2.2).

The ECM shall complete responding to each Request Path request by returning one of the following confirmations to the requesting expander logical phy:

- a) Arb Won (see 6.16.5.2.3);
- b) Arb Lost (see 6.16.5.2.4); or
- c) Arb Reject (see 6.16.5.2.5).

If the ECM receives an Idle request from a phy that is involved in a connection before it has received a Forward Close request from that phy and sent a Forward Close indication to that phy, then the ECM shall send a Forward Break indication to the destination phy.

**6.16.5.2.2 Arbitrating confirmations**

The ECM shall send an Arbitrating (Normal) confirmation after it has received a Request Path request.

The ECM shall send an Arbitrating (Waiting On Partial) confirmation if it is waiting on a partial pathway (see 4.1.11). The ECM is waiting on a partial pathway if:

- a) there is a destination port capable of routing to the requested destination SAS address;
- b) at least one expander logical phy within the destination port supports the requested connection rate;
- c) each of the expander logical phys within the destination port is returning a Phy Status (Partial Pathway) response or Phy Status (Blocked Partial Pathway) response; and
- d) at least one of the expander logical phys within the destination port is returning a Phy Status (Partial Pathway) response.

The ECM shall send an Arbitrating (Blocked On Partial) confirmation if it is waiting on a blocked partial pathway (see 4.1.11). The ECM is waiting on a blocked partial pathway if:

- a) there is a destination port capable of routing to the requested destination SAS address;
- b) at least one expander logical phy within the destination port supports the requested connection rate; and
- c) each of the expander logical phys within the destination port is returning a Phy Status (Blocked Partial Pathway) response.

The ECM shall send an Arbitrating (Waiting On Connection) confirmation if it is waiting on a connection to complete (see 4.1.12). The ECM is waiting on a connection to complete if:

- a) the connection request is blocked by an active connection;
- b) the ECM is receiving Pause Phy responses from an expander logical phy associated with the connection request and the phy identifier from the connection request is not the same as the Phy Identifier argument of the Pause Phy responses; or
- c) there are insufficient routing resources within the expander to complete the connection request.

A connection request shall be considered blocked by an active connection when:

- a) there is a destination port capable of routing to the requested destination SAS address;
- b) at least one expander logical phy within the destination port supports the requested connection rate;
- c) each of the expander logical phys within the destination port is returning:
  - A) a Phy Status (Partial Pathway) response;
  - B) a Phy Status (Blocked Partial Pathway) response;
  - C) a Phy Status (Breaking Connection) response; or
  - D) a Phy Status (Connection) response;
 and
- d) at least one of the expander logical phys within the destination port is returning a Phy Status (Connection) response.

**6.16.5.2.3 Arb Won confirmation**

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

- a) the Request Path request maps to a destination expander logical phy that:
  - A) supports the connection rate; and
  - B) is not reporting:
    - a) a Phy Status (Partial Pathway) response;
    - b) a Phy Status (Blocked Partial Pathway) response;
    - c) a Phy Status (Breaking Connection) response; or
    - d) a Phy Status (Connection) response,
 unless that expander logical phy is arbitrating for the requesting expander logical phy;
- b) the ECM is:

- A) not receiving Pause Phy responses from an expander logical phy associated with the connection request; or
- B) receiving Pause Phy responses from an expander logical phy associated with the connection request and the phy identifier from the connection request is the same as the Phy Identifier argument of the Pause Phy responses;
- c) there are sufficient routing resources to complete the connection request;
- d) no higher priority Request Path requests are present with the requesting expander logical phy as the destination; and
- e) the Request Path request is the highest priority Request Path request (see table 182) mapping to the destination expander logical phy (i.e., only send one Arb Won confirmation for Request Path requests to the same destination phy).

If two or more Request Path requests contend, then the ECM shall select the request with the highest arbitration priority. The arbitration priority for a Request Path request (see table 182) consists of the following:

- a) High Priority argument;
- b) SMP Open Priority argument;
- c) Arbitration Wait Time state machine variable;
- d) Source SAS Address argument; and
- e) Connection Rate argument.

**Table 182 – Arbitration priority for a Request Path request in the ECM**

Bit 85 (85 is MSB)	Bit 84	Bits 83 to 68	Bits 67 to 4	Bits 3 to 0 (0 is LSB)
High Priority argument	SMP Open Priority argument	Arbitration Wait Time state machine variable	Source SAS Address argument	Connection Rate argument

#### 6.16.5.2.4 Arb Lost confirmation

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

- a) the Request Path request maps to a destination expander logical phy that:
  - A) supports the connection rate; and
  - B) is not reporting a Phy Status (Partial Pathway) response, a Phy Status (Blocked Partial Pathway) response, a Phy Status (Breaking Connection) response, or a Phy Status (Connection) response unless that expander logical phy is arbitrating for the requesting expander logical phy;
- b) there are sufficient routing resources to complete the connection request; and
- c) one of the following conditions are met:
  - A) the destination expander logical phy is making a Request Path request with the requesting expander logical phy as its destination (i.e., when two expander logical phys both receive an OPEN address frame destined for each other, the ECM provides the Arb Lost confirmation to the expander logical phy that received the lowest priority OPEN address frame); or
  - B) the ECM is sending an Arb Won confirmation to another expander logical phy that is using the requesting expander logical phy as the destination.

#### 6.16.5.2.5 Arb Reject confirmation

The ECM shall generate one of the following Arb Reject confirmations when any of the following conditions are met and all the Arb Won conditions (see 6.16.5.2.3) are not met:

- 1) an Arb Reject (Bad Destination) confirmation if the source expander logical phy and destination expander logical phys are in the same expander port and are using the direct routing method;
- 2) an Arb Reject (Retry) confirmation if the expander device is unable to process the connection request because it has reduced functionality (see 4.5.8);



- 3) if the source expander logical phy and destination expander logical phys are in the same expander port and are using the table routing method or the subtractive routing method, then:
  - A) an Arb Reject (No Destination) confirmation if the expander device is not configuring (see 4.6.4); or
  - B) an Arb Reject (Retry) confirmation if the expander device is configuring;
- 4) if there are no destination expander logical phys (i.e., there is no direct routing or table routing match and there is no subtractive phy), then:
  - A) an Arb Reject (No Destination) confirmation if the expander device is not configuring; or
  - B) an Arb Reject (Retry) confirmation if the expander device is configuring;
- 5) if access to the destination expander logical phys is prohibited by zoning (see 4.8.3), then:
  - A) an Arb Reject (Zone Violation) confirmation if the zoning expander device is unlocked; or
  - B) an Arb Reject (Retry) confirmation if the zoning expander device is locked;
- 6) an Arb Reject (Connection Rate Not Supported) confirmation if none of the destination expander logical phys supports the connection rate; and
- 7) an Arb Reject (Pathway Blocked) confirmation if all the destination expander logical phys that support the connection rate contain blocked partial pathways (i.e., are all returning Phy Status (Blocked Partial Pathway) responses) and pathway recovery rules require this Request Path request be rejected to release path resources (see 6.16.5.5).

### 6.16.5.3 Arbitration status

Arbitration status shall be conveyed between expander devices and by expander devices to SAS endpoints using AIP primitive sequences (see 6.2.6.1). This status is used to monitor the progress of connection attempts and to facilitate pathway recovery as part of deadlock recovery.

The arbitration status of an expander logical phy is set to the last type of AIP received.

Before an expander device transmits an AIP primitive sequence, the expander device may have transmitted an OPEN address frame on the same physical link. Arbitration fairness dictates which OPEN address frame wins (see 6.16.4).

After transmitting an AIP primitive sequence, an expander device shall transmit at least one other dword (e.g., an idle dword) before transmitting another AIP primitive sequence.

Expander devices shall transmit at least one AIP primitive sequence every 128 dwords while originating AIP (NORMAL) primitive sequences, AIP (WAITING ON PARTIAL) primitive sequences, or AIP (WAITING ON CONNECTION) primitive sequence.

NOTE 30 - Expander devices compliant with SAS-1.1 were not required to transmit three consecutive AIP primitives, as AIP was defined as a single primitive sequence (see 6.2.4.2) rather than as an extended primitive sequence (see 6.2.4.5).

If SAS dword mode is enabled, then expander devices shall transmit an AIP primitive sequence (e.g., an AIP (NORMAL)) within 128 dwords of receiving an OPEN address frame.

If SAS packet mode is enabled, then expander devices shall transmit an AIP primitive sequence (e.g., an AIP (NORMAL)) within 300 ns of receiving an OPEN address frame.

### 6.16.5.4 Partial Pathway Timeout timer

Each expander logical 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 logical phy shall initialize the Partial Pathway Timeout timer to the time reported in the PARTIAL PATHWAY TIMEOUT VALUE field in the SMP DISCOVER response (see 9.4.3.10) and run the Partial Pathway Timeout timer whenever the ECM provides an Arbitrating (Blocked On Partial) confirmation to the expander logical phy that all expander logical phys within the requested destination port are blocked waiting on partial pathways.

NOTE 31 - The partial pathway timeout value allows flexibility in specifying how long an expander device waits before attempting pathway recovery. The recommended default value (see 9.4.3.10) was chosen to cover a wide range of topologies. Selecting small partial pathway timeout value values within a large topology

compromises performance as a result of the time a device waits after receiving OPEN\_REJECT (PATHWAY BLOCKED) before retrying the connection request. Similarly, selecting large partial pathway timeout value values within a small topology compromises performance due to waiting longer than necessary to detect pathway blockage.

When the Partial Pathway Timeout timer is not running, an expander logical phy shall initialize and start the Partial Pathway Timeout timer when all expander logical phys within the requested destination port contain a blocked partial pathway (i.e., are returning Phy Status (Blocked Partial Pathway)).

NOTE 32 - The Partial Pathway Timeout timer is not initialized and started if one or more of the expander logical phys within a requested destination port are being used for a connection.

When one of the conditions in this subclause is not met, the expander logical phy shall stop the Partial Pathway Timeout timer. If the Partial Pathway Timeout timer expires, then pathway recovery shall occur (see 6.16.5.5).

#### 6.16.5.5 Pathway recovery

Pathway recovery provides a means to abort connection requests in order to prevent deadlock using pathway recovery priority comparisons. Pathway recovery priority comparisons compare the PATHWAY BLOCKED COUNT fields and SOURCE SAS ADDRESS fields of the OPEN address frames of the blocked connection requests as described in table 183.

**Table 183 – Pathway recovery priority**

Bits 71 to 64 (71 is MSB)	Bits 63 to 0 (0 is LSB)
PATHWAY BLOCKED COUNT field value	SOURCE SAS ADDRESS field value

If the ECM receives a Partial Pathway Timeout Timer Expired request from an arbitrating expander logical phy expires (i.e., reaches a value of zero), the ECM shall determine whether to continue the connection request or to abort the connection request.

The ECM shall reply to a connection request with Arb Reject (Pathway Blocked) when:

- a) a Partial Pathway Timeout Timer Expired request has been received; and
- b) the pathway recovery priority of the arbitrating expander logical phy (i.e., the expander logical phy requesting the connection) is less than or equal to the pathway recovery priority of any of the expander logical phys within the destination port that are sending Phy Status (Blocked Partial Pathway) responses to the ECM.

The pathway blocked count and source SAS address values used to form the pathway recovery priority of a destination phy are those of the Request Path request if the expander logical phy sent a Request Path request to the ECM or those of the Forward Open indication if the expander logical phy received a Forward Open indication from the ECR.

#### 6.16.6 BREAK handling

A logical phy aborts a connection request (see 6.16.7) and breaks a connection (see 6.16.11) by transmitting a BREAK primitive sequence.

Logical phys shall enable the BREAK\_REPLY method of responding to received BREAK primitive sequences when:

- a) the BREAK\_REPLY CAPABLE bit transmitted by the logical phy in the outgoing IDENTIFY address frame is set to one; and
- b) the BREAK\_REPLY CAPABLE bit received by the logical phy in the incoming IDENTIFY address frame is set to one.

Logical phys shall disable the BREAK\_REPLY method of responding to received BREAK primitive sequences if the BREAK\_REPLY CAPABLE bit received by the logical phy in the incoming IDENTIFY address frame is set to zero.

Logical phys contained within SAS devices or expander devices that are compliant with this standard shall set the BREAK\_REPLY CAPABLE bit to one in their outgoing IDENTIFY address frame.

If the BREAK\_REPLY method of responding to received BREAK primitive sequences is enabled, then the logical phy shall transmit a BREAK\_REPLY primitive sequence in response to a received BREAK primitive sequence.

If the BREAK\_REPLY method of responding to received BREAK primitive sequences is disabled, then the logical phy shall transmit a BREAK primitive sequence in response to a received BREAK primitive sequence.

NOTE 33 - Phys compliant with SAS-1.1 do not set the BREAK\_REPLY CAPABLE bit to one in their outgoing IDENTIFY address frames.

### 6.16.7 Aborting a connection request

BREAK may be used to abort a connection request. The source phy shall transmit a BREAK primitive sequence after the Open Timeout timer expires or if the source phy chooses to abort its request for any other reason before a connection is established.

After transmitting a BREAK primitive sequence, the source phy shall initialize a Break Timeout timer to 1 ms and start the Break Timeout timer.

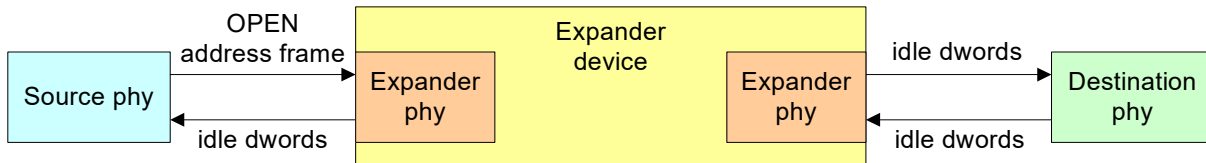
After a source phy transmits a BREAK primitive sequence to abort a connection request, the source phy shall expect one of the results listed in table 184.

**Table 184 – Results of aborting a connection request**

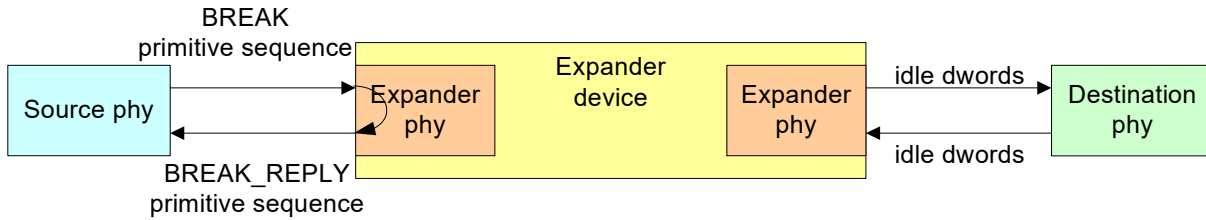
<b>BREAK_REPLY method of responding to received BREAK primitive sequences</b>	<b>Result</b>	<b>Description</b>
Disabled	Receive a BREAK primitive sequence	This confirms that the connection request has been aborted.
	Receive a BREAK_REPLY primitive sequence	Ignore the BREAK_REPLY primitive sequence.
Enabled	Receive a BREAK primitive sequence	The originating phy shall transmit a BREAK_REPLY primitive sequence and wait to receive a BREAK_REPLY primitive sequence or for the BREAK Timeout timer to expire.
	Receive a BREAK_REPLY primitive sequence	This confirms that the connection request has been aborted.
Enabled or disabled	Break Timeout timer expires	The originating phy shall assume the connection request has been aborted.

When a logical phy transmitting a BREAK primitive sequence is attached to an expander device, the BREAK or BREAK\_REPLY response to the logical phy is generated by the expander logical phy to which the logical phy is attached, not the other SAS logical phy in the connection. If the expander device has transmitted a connection request to the destination, then the expander device shall also transmit a BREAK primitive sequence to the destination. If the expander device has not transmitted a connection request to the destination, then the expander device shall not transmit a BREAK primitive sequence to the destination. After transmitting a BREAK primitive sequence or BREAK\_REPLY primitive sequence back to the source phy, the expander device shall ensure that a connection response does not occur (i.e., the expander device shall no longer forward dwords from the destination). Figure 152 shows an example of BREAK usage. Multiplexing is disabled on all phys in the example.

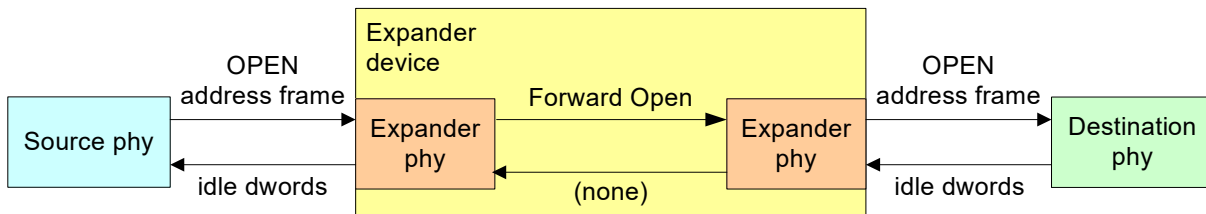
Case 1: OPEN address frame has not propagated through the expander device:



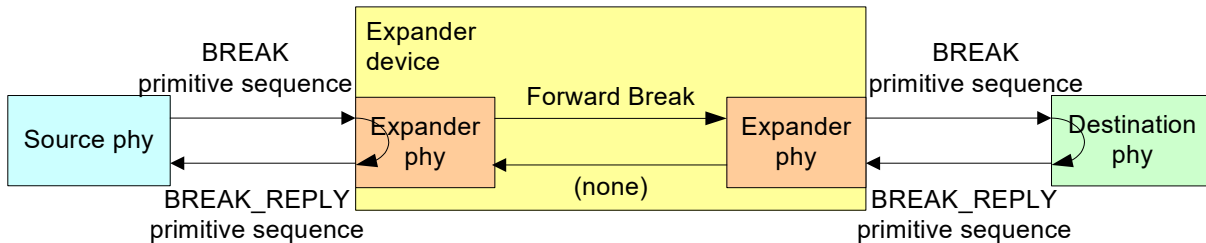
Case 1 result: Expander device transmits a BREAK\_REPLY primitive sequence to the source phy.



Case 2: OPEN address frame has propagated through the expander device:



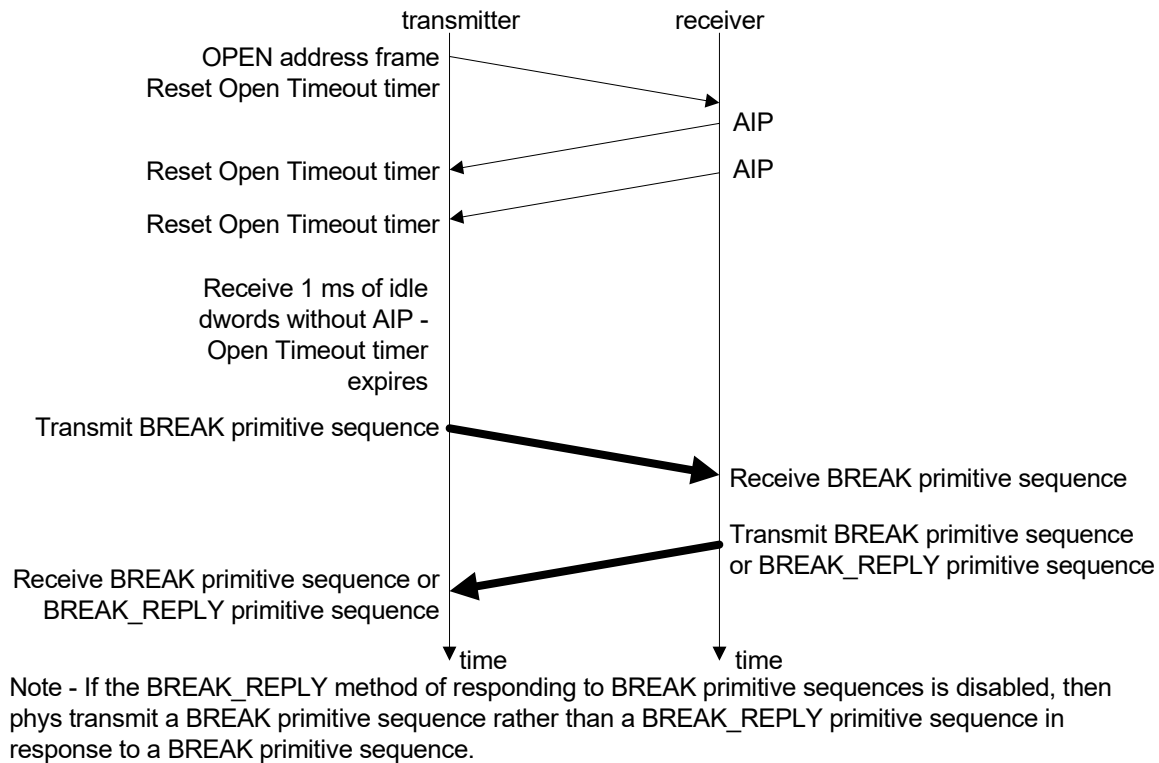
Case 2 result: Expander device transmits a BREAK\_REPLY primitive sequence to the source phy and a BREAK primitive sequence to the destination phy, then waits for a BREAK\_REPLY primitive sequence from the destination phy.



Note - If the BREAK\_REPLY method of responding to BREAK primitive sequences is disabled, then phys transmit a BREAK primitive sequence rather than a BREAK\_REPLY primitive sequence in response to a BREAK primitive sequence.

**Figure 152 –Aborting a connection request with a BREAK primitive sequence**

Figure 153 shows the sequence for a connection request where the Open Timeout timer expires.



**Figure 153 –Connection request timeout example**

#### 6.16.8 Expander device request for an SSP connection close

An expander device's expander function initiates the closing of an SSP connection by sending a Begin SSP Connection Close confirmation to the XL state machine (see 6.19.9) for each expander phy being used by the SSP connection. Upon receipt of a Begin SSP Connection Close confirmation each expander phy associated with the SSP connection being closed replaces the transmission of any:

- RRDY (NORMAL) with an RRDY (CLOSE) (see 6.19.2); and
- EXTEND\_CONNECTION (NORMAL) with an EXTEND\_CONNECTION (CLOSE) (see 6.19.2).

#### 6.16.9 Closing a connection

CLOSE is used to close a connection of any protocol. See 6.20.8 for details on closing SSP connections, 6.21.8 for details on closing STP connections, and 6.22.5 for details on closing SMP connections.

After transmitting a CLOSE primitive sequence and CLOSE primitive parameter (see 6.2.6.5.2), if any, the originating phy shall initialize a Close Timeout timer to 1 ms and start the Close Timeout timer.

After the logical phy transmits a CLOSE primitive sequence and CLOSE primitive parameter, if any, to close a connection, logical phy shall expect one of the results listed in table 185.

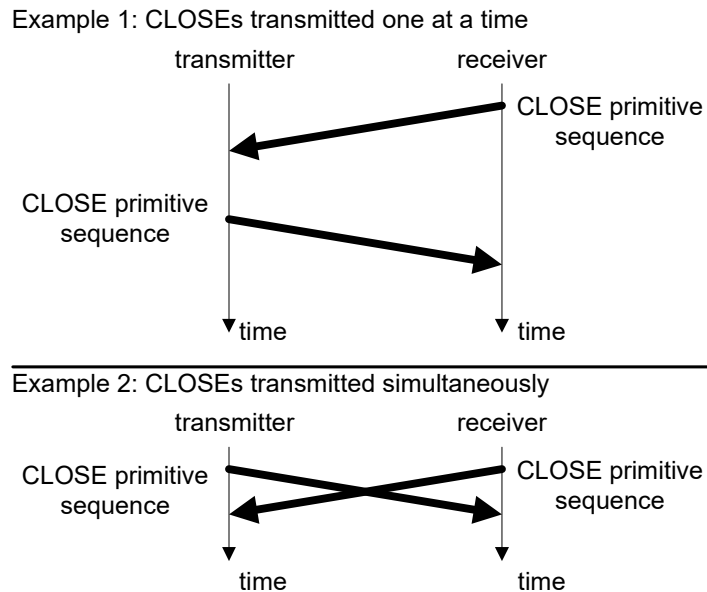
**Table 185 – Results of closing a connection**

Result	Description
Receive a CLOSE primitive sequence and CLOSE primitive parameter (see 6.2.6.5.2), if any	This confirms that the connection has been closed and provides a CLOSE primitive parameter, if any, describing the highest fairness priority of the next open request of an attached expander device.
Close Timeout timer expires	The originating phy shall attempt to break the connection (see 6.16.11).

No additional dwords for the connection shall follow the CLOSE primitive sequence except a CLOSE primitive parameter, if any. Expander devices shall close the full-duplex connection upon forwarding a CLOSE primitive sequence and a CLOSE primitive parameter, if any, in each direction.

When a logical phy has both transmitted and received a CLOSE primitive sequence, the logical phy shall consider the connection closed.

Figure 154 shows example sequences for closing a connection with no CLOSE primitive parameter.



**Figure 154 –Closing a connection example**

#### 6.16.10 Expander device closing a connection

If extended fairness priority is supported (i.e., EXTENDED FAIRNESS bit (see 9.4.3.4) is set to one), then the ECM shall respond to a Request Fairness Priority request with a Fairness Priority confirmation.

If an ECM receives a Request Fairness Priority request and there is:

- a) a blocked connection request associated with the expander logical phy that sent the Request Fairness request, then the ECM shall send a Fairness Priority confirmation to the requesting expander logical phy with the following arguments specifying the highest priority OPEN address frame requesting access to the logical phy specified by the Phy Identifier argument:
  - A) High Priority;

- B) SMP Open Priority;
- C) Arbitration Wait Time;
- D) Connection Rate; and
- E) Open Destination SAS Address;

or

- b) no blocked connection request associated with the expander logical phy that sent the Request Fairness request, then the ECM shall send a Fairness Priority confirmation to the requesting expander logical phy with no arguments.

#### 6.16.11 Breaking a connection

In addition to aborting a connection request, a BREAK primitive sequence may also be used to break a connection in cases where CLOSE is not available. After transmitting a BREAK primitive sequence, the originating phy shall ignore all incoming dwords except for BREAKs, BREAK\_REPLYs, and deletable primitives.

After transmitting a BREAK primitive sequence, the originating phy shall initialize a Break Timeout timer to 1 ms and start the Break Timeout timer.

After a logical phy transmits a BREAK primitive sequence to break a connection, the logical phy shall expect one of the results listed in table 186.

**Table 186 – Results of breaking a connection**

<b>BREAK_REPLY method of responding to received BREAK primitive sequences</b>	<b>Result</b>	<b>Description</b>
Disabled	Receive a BREAK primitive sequence	This confirms that the connection has been broken.
	Receive BREAK_REPLY primitive sequence	Ignore the BREAK_REPLY primitive sequence.
Enabled	Receive a BREAK primitive sequence	The originating phy shall transmit a BREAK_REPLY primitive sequence and wait to receive a BREAK_REPLY primitive sequence or for the BREAK Timeout timer to expire.
	Receive a BREAK_REPLY primitive sequence	This confirms that the connection has been broken.
Enabled or disabled	Break Timeout timer expires	The originating phy shall assume the connection has been broken. The originating phy may perform a link reset sequence.

In addition to a BREAK, a connection is considered broken if a link reset sequence starts (i.e., the SP state machine transitions from SP15:SAS\_PHY\_Ready or SP22:SATA\_PHY\_Ready to SP0:OOB\_COMINIT (see 5.14)).

See 6.20.7 for additional rules on breaking an SSP connection.

## 6.17 Rate matching

### 6.17.1 Rate matching overview

Each successful connection request contains the connection rate (see 4.1.12) of the pathway.

If the logical phy's logical link rate is faster than the connection rate, then that logical phy shall insert:

- a) deletable primitives between dwords; or
- b) SPL packets containing a scrambled idle segment between SPL packets.

### 6.17.2 Rate matching while in the SAS dword mode

Each logical phy in the pathway while in the SAS dword mode shall insert deletable primitives between dwords if the logical phy's logical link rate is faster than the connection rate as described in table 187.

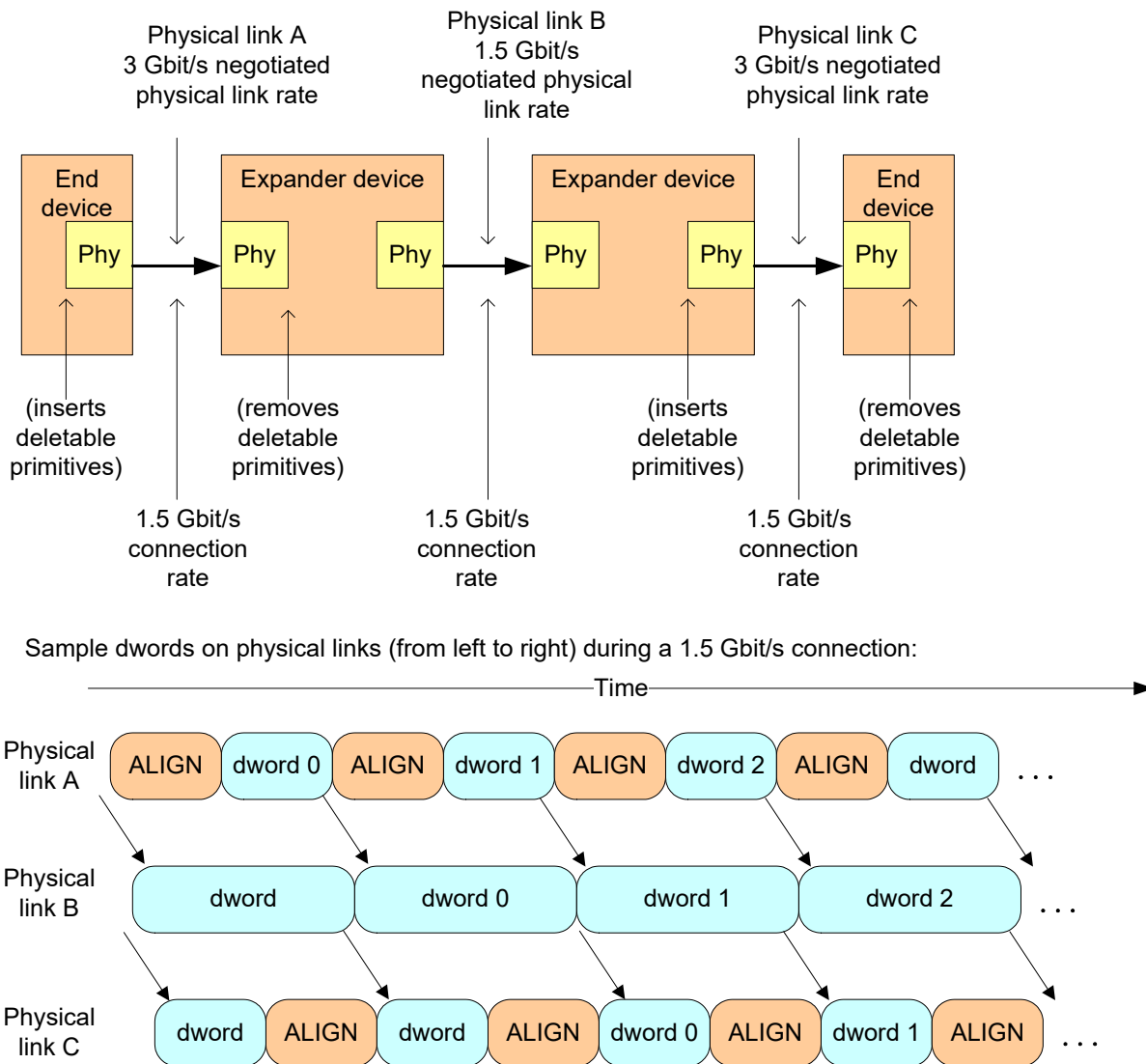
**Table 187 – Rate matching deletable primitive insertion requirements while in the SAS dword mode**

Logical link rate	Connection rate	Requirement
1.5 Gbit/s	1.5 Gbit/s	None
3 Gbit/s	1.5 Gbit/s	One deletable primitive within every two dwords that are not physical link rate tolerance management deletable primitives (i.e., every overlapping window of two dwords) (e.g., a repeating pattern of a deletable primitive followed by a dword or a repeating pattern of a dword followed by a deletable primitive)
	3 Gbit/s	None
6 Gbit/s	1.5 Gbit/s	Three deletable primitives within every four dwords that are not physical link rate tolerance management deletable primitives (i.e., 3 in every overlapping window of four dwords)
	3 Gbit/s	One deletable primitive within every two dwords that are not physical link rate tolerance management deletable primitives (i.e., every overlapping window of two dwords) (e.g., a repeating pattern of a deletable primitive followed by a dword or a repeating pattern of a dword followed by a deletable primitive)
	6 Gbit/s	None
12 Gbit/s	1.5 Gbit/s	Seven deletable primitives within every eight dwords that are not physical link rate tolerance management deletable primitives (i.e., seven in every overlapping window of eight dwords)
	3 Gbit/s	Three deletable primitives within every four dwords that are not physical link rate tolerance management deletable primitives (i.e., three in every overlapping window of four dwords)
	6 Gbit/s	One deletable primitive within every two dwords that are not physical link rate tolerance management deletable primitives (i.e., every overlapping window of two dwords) (e.g., a repeating pattern of a deletable primitive followed by a dword or a repeating pattern of a dword followed by a deletable primitive)
	12 Gbit/s	None



Deletable primitives inserted for rate matching are in addition to deletable primitives inserted for physical link rate tolerance management (see 6.5). See Annex H for a summary of their combined requirements.

Figure 155 shows an example of rate matching between a 3 Gbit/s originating phy and a 3 Gbit/s receiving phy, with an intermediate 1.5 Gbit/s physical link in between them. Multiplexing is disabled in this example.



**Figure 155 –Rate matching example while in the SAS dword mode**

A logical phy originating dwords shall start rate matching at the selected connection rate starting with the first dword that is not a deletable primitive inserted for physical link rate tolerance management following:

- transmitting the EOAF for an OPEN address frame; or
- transmitting an OPEN\_ACCEPT.

An expander logical phy forwarding dwords shall not insert deletable primitives for rate matching based on counting dwords transmitted and shall insert deletable primitives whenever it underflows.

The source phy transmits idle dwords including deletable primitives at the selected connection rate while waiting for the connection response. This enables each expander device to start forwarding dwords from the source phy to the destination phy after forwarding an OPEN\_ACCEPT.

A logical phy shall stop inserting deletable primitives for rate matching after:

- a) transmitting the first dword in a CLOSE primitive sequence;
- b) transmitting the first dword in a BREAK primitive sequence;
- c) transmitting the first dword in a BREAK\_REPLY primitive sequence;
- d) receiving an OPEN\_REJECT for a connection request; or
- e) losing arbitration to a received OPEN address frame.

### 6.17.3 Rate matching while in the SAS packet mode

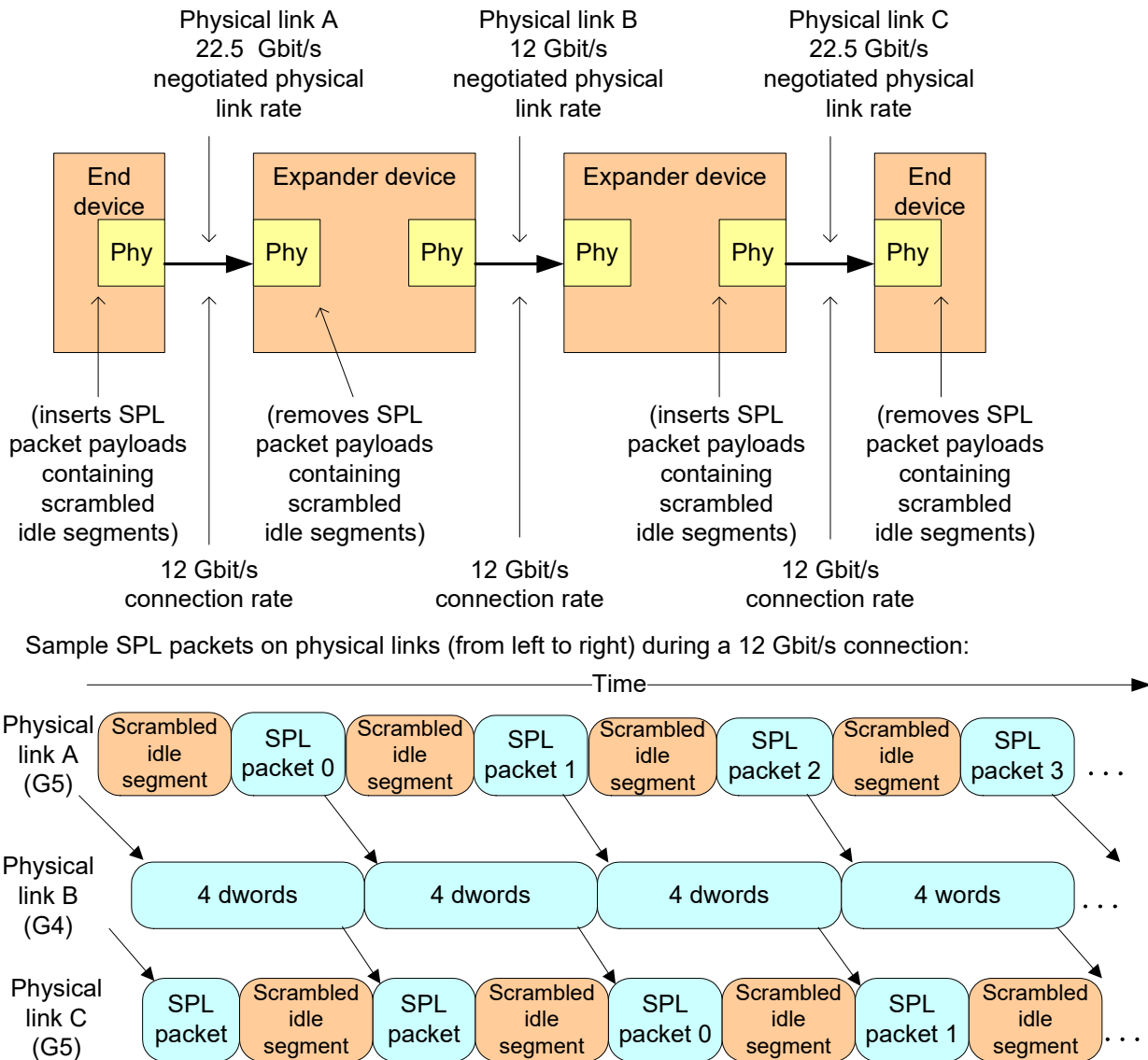
Each logical phy in the pathway while in the SAS packet mode shall insert SPL packets containing a scrambled idle segment between SPL packets if the logical phy's logical link rate is faster than the connection rate as described in table 188.

**Table 188 – Rate matching deletable primitive insertion requirements while in the SAS packet mode**

Logical link rate	Connection rate	Requirement
22.5 Gbit/s	1.5 Gbit/s	15 SPL packet payloads each containing a scrambled idle segment within every 16 SPL packets (i.e., 15 in every overlapping window of 16 SPL packet payloads)
	3 Gbit/s	Seven SPL packet payloads each containing a scrambled idle segment within every eight SPL packets (i.e., seven in every overlapping window of eight SPL packet payloads)
	6 Gbit/s	Three SPL packet payloads each containing a scrambled idle segment within every four SPL packets (i.e., three in every overlapping window of four SPL packet payloads)
	12 Gbit/s	One SPL packet payload containing a scrambled idle segment within every two SPL packets (i.e., every overlapping window of two SPL packet payloads)
	22.5 Gbit/s	None

SPL packet payloads containing a scrambled idle segment inserted for rate matching are in addition to deletable extended binary primitives contained in primitive segments for physical link rate tolerance management (see 6.5). See Annex H for a summary of their combined requirements.

Figure 156 shows an example of rate matching between a 22.5 Gbit/s originating phy and a 22.5 Gbit/s receiving phy, with an intermediate 12 Gbit/s physical link in between them.



**Figure 156 –Rate matching example while in the SAS packet mode**

A logical phy originating SPL packets shall start rate matching at the selected connection rate starting with the first SPL packet that does not contain a deletable extended binary primitive inserted for physical link rate tolerance management following an SPL packet containing:

- an EOAF for an OPEN address frame; or
- an OPEN\_ACCEPT.

An expander logical phy forwarding the contents of a received SPL packet:

- shall not insert scrambled idle segments for rate matching based on counting SPL packets transmitted; and
- shall insert scrambled idle segments whenever it underflows.

The source phy transmits SPL packets that include scrambled idle segments or deletable extended binary primitives at the selected connection rate while waiting for the connection response. This enables each expander device to start forwarding SPL packet payloads containing a scrambled idle segment from the source phy to the destination phy after forwarding an OPEN\_ACCEPT.

A logical phy shall stop inserting SPL packet payloads containing scrambled idle segments for rate matching after:

- a) transmitting the first SPL packet payload containing a CLOSE primitive sequence;
- b) transmitting the first SPL packet payload a BREAK primitive sequence;
- c) transmitting the first SPL packet payload a BREAK\_REPLY primitive sequence;
- d) receiving an OPEN\_REJECT for a connection request; or
- e) losing arbitration to a received OPEN address frame.

## 6.18 SL (link layer for SAS logical phys) state machines

### 6.18.1 SL state machines overview

The SL (link layer for SAS logical phys) state machines control connections, handles connection requests (i.e., OPEN address frames), CLOSEs, and BREAKs. The SL state machines are as follows:

- a) SL\_RA (receive OPEN address frame) state machine (see 6.18.3); and
- b) SL\_CC (connection control) state machine (see 6.18.4).

All the SL state machines shall begin after receiving an Enable Disable SAS Link (Enable) message from the SL\_IR state machines.

If a state machine consists of multiple states, then the initial state is as indicated in the state machine description.

Figure 157, figure 158, and figure 159 show the SL state machines.

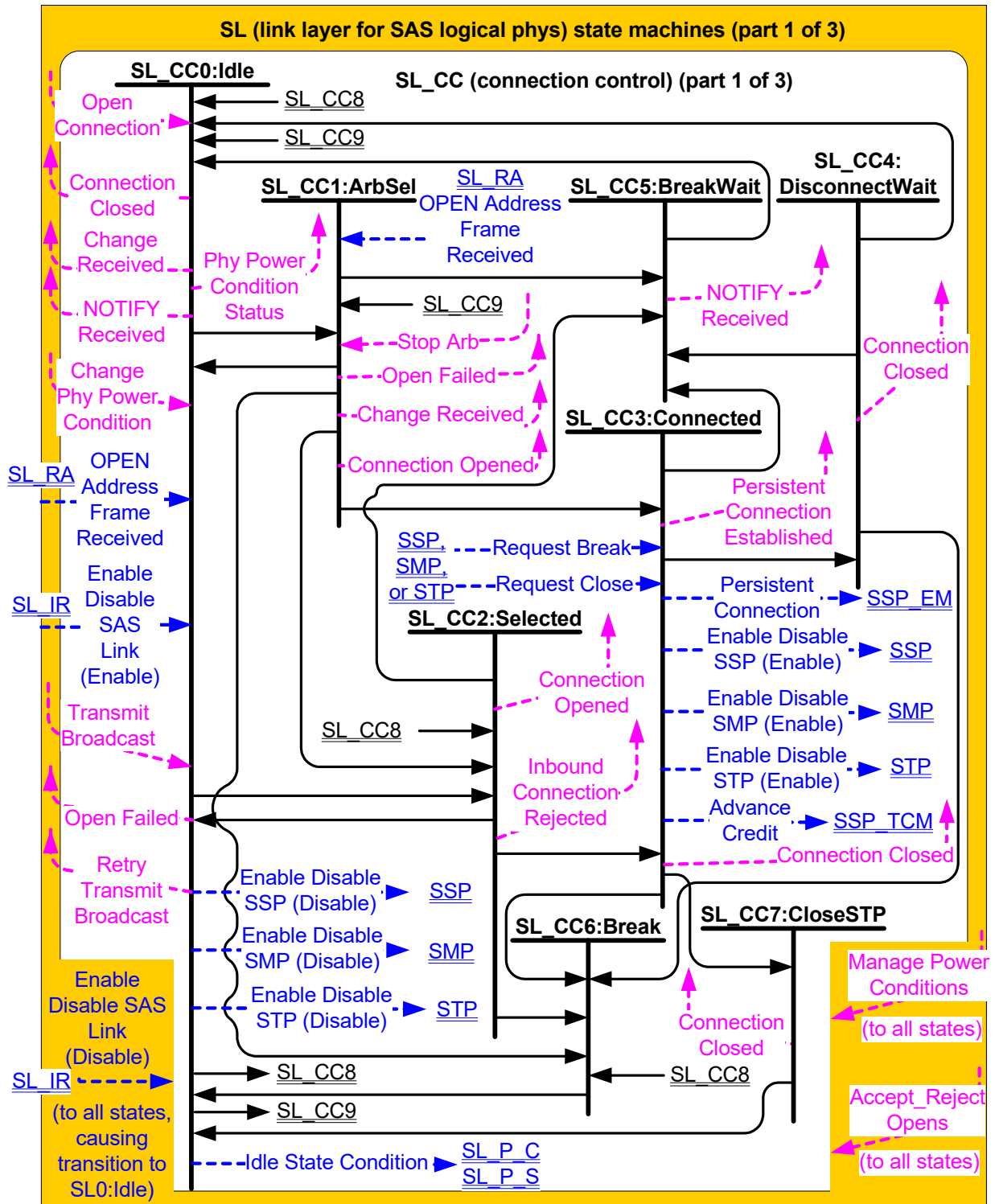


Figure 157 –SL (link layer for SAS logical phys) state machines (1 of 3)

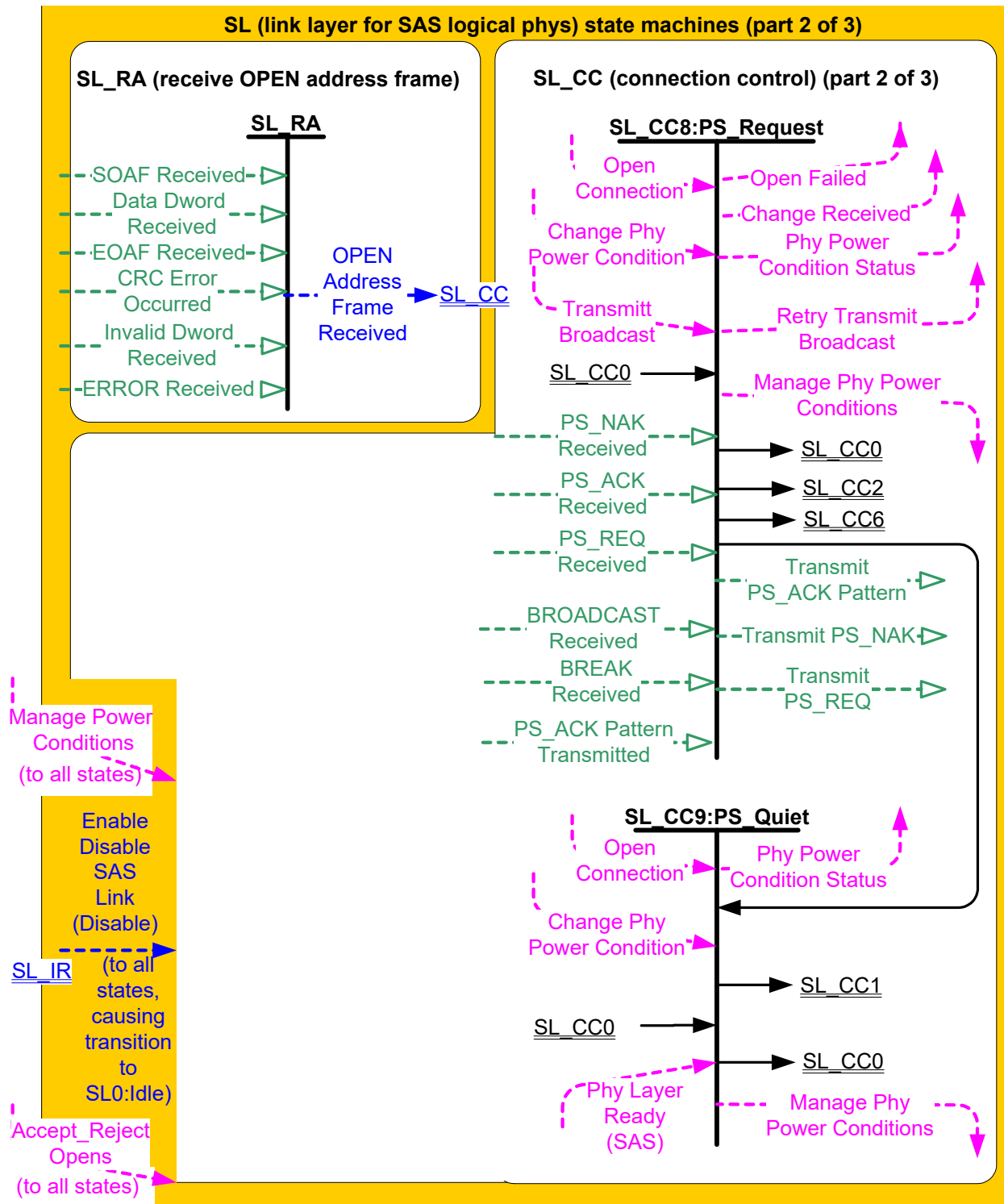


Figure 158 –SL (link layer for SAS logical phys) state machines (2 of 3)

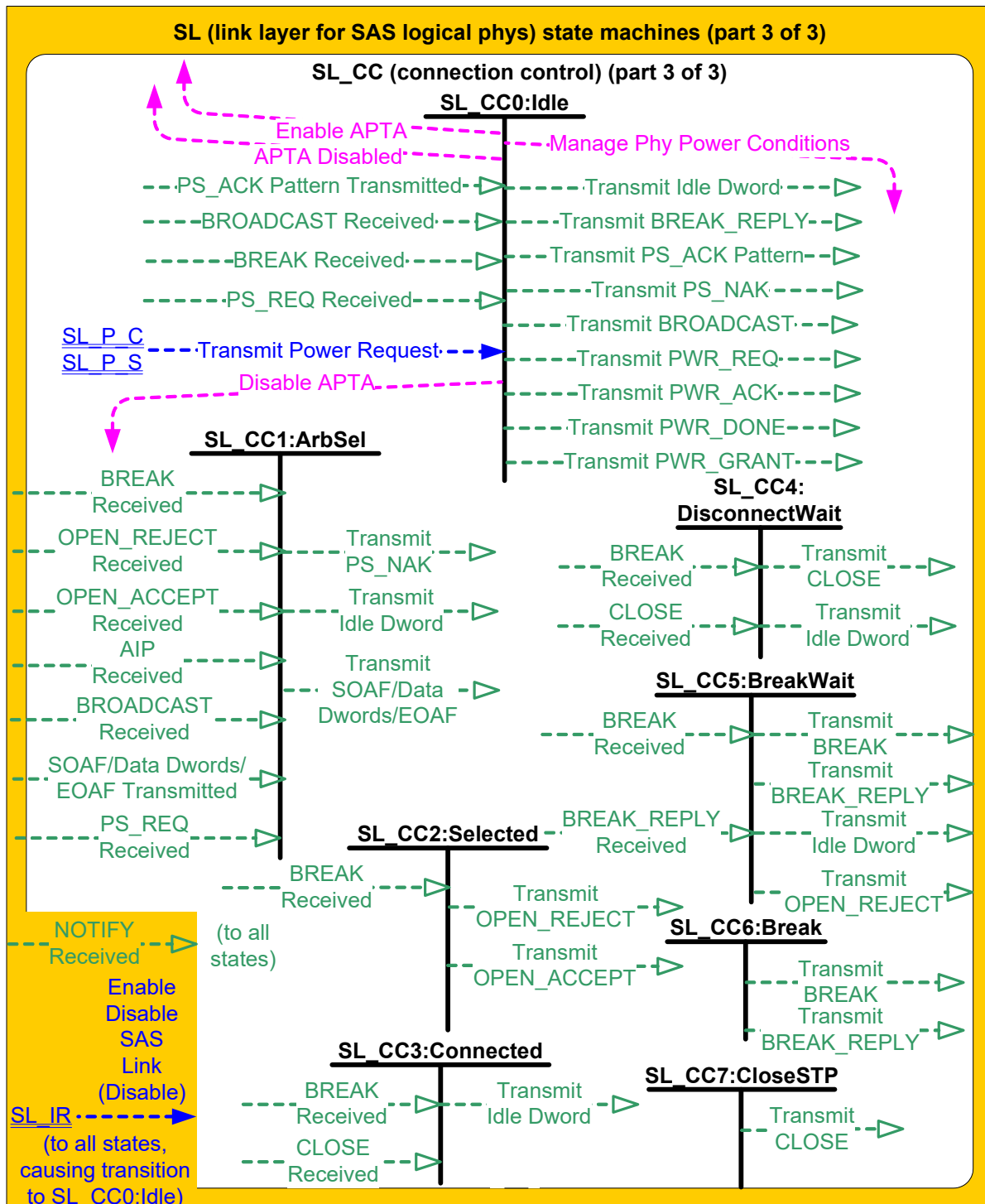


Figure 159 –SL (link layer for SAS logical phys) state machines (3 of 3)

### 6.18.2 SL transmitter and receiver

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

- Transmit Idle Dword;
- 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 BREAK\_REPLY;
- g) Transmit PS\_REQ with an argument indicating the specific type (e.g., Transmit PS\_REQ (Partial) or Transmit PS\_REQ (Slumber));
- h) Transmit PS\_ACK Pattern (see 6.13);
- i) Transmit PS\_NAK;
- j) Transmit BROADCAST;
- k) Transmit PWR\_REQ;
- l) Transmit PWR\_ACK;
- m) Transmit PWR\_DONE;
- n) Transmit PWR\_GRANT; and
- o) Transmit CLOSE with an argument indicating the specific type (e.g., Transmit CLOSE (Normal)).

When the SL transmitter is requested to transmit a dword from any state within any of the SL state machines, the SL transmitter shall transmit that dword. If there are multiple requests to transmit, then the following priority should be followed when selecting the dword to transmit:

- 1) BREAK\_REPLY;
- 2) BREAK;
- 3) CLOSE;
- 4) OPEN\_ACCEPT or OPEN\_REJECT;
- 5) SOAF, data dword, or EOAF;
- 6) PWR\_REQ, PWR\_ACK, PWR\_DONE, or PWR\_GRANT;
- 7) PS\_REQ, PS\_ACK Pattern, or PS\_NAK; and
- 8) idle dword.

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 to the SL state machines based on dwords that have been transmitted:

- a) SOAF/Data Dwords/EOAF Transmitted; and
- b) PS\_ACK Pattern Transmitted.

The SL receiver sends the following messages to the SL state machines indicating primitive sequences and dwords received from the SP\_DWS receiver (see 5.15.2) and the SP\_PS receiver (see 5.16.2):

- a) SOAF Received;
- b) Data Dword Received;
- c) EOAF Received;
- d) BROADCAST Received with an argument indicating the specific type (e.g., BROADCAST Received (Change));
- e) BREAK Received;
- f) BREAK\_REPLY 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;
- j) CLOSE Received with an argument indicating the specific type (e.g., CLOSE Received (Normal));
- k) NOTIFY Received (Power Loss Expected);
- l) PS\_REQ Received with an argument indicated the specific type (e.g., PS\_REQ Received (Partial) or PS\_REQ Received (Slumber));
- m) PS\_ACK Received;
- n) PS\_NAK Received;
- o) ERROR Received; and
- p) Invalid Dword Received.



The SL receiver shall ignore all other dwords.

The SL transmitter relationship to other transmitters is defined in 4.3.2. The SL receiver relationship to other receivers is defined in 4.3.3.

### 6.18.3 SL\_RA (receive OPEN address frame) state machine

The SL\_RA state machine's function is to receive address frames and determine if each received address frame is a valid OPEN address frame. This state machine consists of one state.

This state machine receives SOAFs, dwords of an OPEN address frames, and EOAFs.

This state machine shall ignore all messages except SOAF Received, Data Dword Received and EOAF Received.

If this state machine receives a subsequent SOAF Received message after receiving an SOAF Received message but before receiving an EOAF Received message, then this state machine shall discard the address frame in progress.

If this state machine receives more than eight Data Dword Received messages (i.e., 32 bytes) after an SOAF Received message and before an EOAF Received message, then this state machine shall discard the address frame.

If this state machine receives an Invalid Dword Received message or an ERROR Received message after an SOAF Received message and before an EOAF Received message, then this state machine shall either:

- a) ignore the invalid dword or ERROR; or
- b) discard the address frame.

After receiving an EOAF Received message, this state machine shall check if the address frame is a valid OPEN address frame.

This state machine shall accept an address frame if:

- a) the ADDRESS FRAME TYPE field is set to 1h (i.e., OPEN);
- b) the number of data dwords between the SOAF and EOAF is eight; and
- c) no CRC Error Occurred message was received for this address frame,

otherwise this state machine shall discard the address frame. If the frame is not discarded then this state machine shall send an OPEN Address Frame Received message to the SL\_CC0:Idle state and the SL\_CC1:ArbSel state with an argument that contains all the data dwords received in the OPEN address frame.

### 6.18.4 SL\_CC (connection control) state machine

#### 6.18.4.1 SL\_CC state machine overview

The SL\_CC state machine consists of the following states:

- a) SL\_CC0:Idle (see 6.18.4.2) (initial state);
- b) SL\_CC1:ArbSel (see 6.18.4.3);
- c) SL\_CC2:Selected (see 6.18.4.4);
- d) SL\_CC3:Connected (see 6.18.4.5);
- e) SL\_CC4:DisconnectWait (see 6.18.4.6);
- f) SL\_CC5:BreakWait (see 6.18.4.7);
- g) SL\_CC6:Break (see 6.18.4.8);
- h) SL\_CC7:CloseSTP (see 6.18.4.9);
- i) SL\_CC8:PS\_Request (see 6.18.4.10); and
- j) SL\_CC9:PS\_Quiet (see 6.18.4.11).

This state machine receives the following requests from the management application layer:

- a) Transmit Broadcast.

This state machine shall start in the SL\_CC0:Idle state. The state machine shall transition to the SL\_CC0:Idle state from any other state after receiving an Enable Disable SAS Link (Disable) message from the SL\_IR state machines (see 6.12).

This machine receives the following messages from the SSP link layer state machine (see 6.20.9), the STP link layer state machine, and SMP link layer state machine (see 6.22.6):

- a) Request Break; and
- b) Request Close.

This state machine sends the following messages to the SL\_P\_S link layer state machine (see 6.14.4) and SL\_P\_C link layer state machine (see 6.14.5):

- a) Idle State Condition (Active); and
- b) Idle State Condition (Inactive).

This state machine sends the following messages to the SSP link layer state machine, the STP link layer state machine, and SMP link layer state machine:

- a) Enable Disable SSP (Enable);
- b) Enable Disable SSP (Disable);
- c) Enable Disable STP (Enable);
- d) Enable Disable STP (Disable);
- e) Enable Disable SMP (Enable); and
- f) Enable Disable SMP (Disable).

This state machine receives the following messages from the SL\_IR state machines (see 6.12):

- a) Enable Disable SAS Link (Enable); and
- b) Enable Disable SAS Link (Disable).

Any message received by a state that is not referred to in the description of that state or in this subclause shall be ignored.

If this state machine receives an Accept\_Reject OPENS (Accept SSP) request, then this state machine shall set the Reject SSP Opens state machine variable (see table 190) to NO. If this state machine receives an Accept\_Reject OPENS (Reject SSP) request, then this state machine shall set the Reject SSP Opens state machine variable to YES.

If this state machine receives an Accept\_Reject OPENS (Accept SMP) request, then this state machine shall set the Reject SMP Opens state machine variable (see table 190) to NO. If this state machine receives an Accept\_Reject OPENS (Reject SMP) request, then this state machine shall set the Reject SMP Opens state machine variable to YES.

If this state machine receives an Accept\_Reject OPENS (Accept STP) request, then this state machine shall set the Reject STP Opens state machine variable (see table 190) to NO. If this state machine receives an Accept\_Reject OPENS (Reject STP) request, then this state machine shall set the Reject STP Opens state machine variable to YES.

If this state machine receives a Manage Power Conditions (Accept Partial) request from the management application layer, then this state machine shall set the Accept Partial state machine variable (see table 190) to YES. If this state machine receives a Manage Power Conditions (Accept Slumber) request from the management application layer, then this state machine shall set the Accept Slumber state machine variable (see table 190) to YES.

If this state machine receives a Manage Power Conditions (Reject Partial) request from the management application layer, then this state machine shall set the Accept Partial state machine variable to NO. If this state machine receives a Manage Power Conditions (Reject Slumber) request from the management application layer, then this state machine shall set the Accept Slumber state machine variable to NO.

The default value of the Accept Partial state machine variable shall be set to NO and default value of the Accept Slumber state machine variable shall be set to NO.

Any detection of an internal error shall cause the SL\_CC state machine to transition to the SL\_CC5:BreakWait state.

This state machine shall maintain the timers listed in table 189.

**Table 189 – SL\_CC state machine timers**

Timer	Initial value
Open Timeout timer	1 ms
Close Timeout timer	1 ms
Break Timeout timer	1 ms
Power Condition Request Timeout timer	1 ms
Idle timer	1 ms

This state machine shall maintain the state machine variables listed in table 190.

**Table 190 – SL\_CC state machine variables**

State machine variable	Description
Reject SSP Opens	Used to determine if the SCSI application layer is permitting SSP connection requests to be accepted on this logical phy.
Reject SMP Opens	Used to determine if the management application layer is permitting SMP connection requests to be accepted on this logical phy.
Reject STP Opens	Used to determine if the ATA application layer is permitting STP connection requests to be accepted on this logical phy.
Accept Partial	Used to determine if the management application layer is permitting this phy to enter a partial phy power condition.
Accept Slumber	Used to determine if the management application layer is permitting this phy to enter a slumber phy power condition.

#### 6.18.4.2 SL\_CC0:Idle state

##### 6.18.4.2.1 State description

This state is the initial state and is the state that is used while there is no connection pending or established.

Upon entry into this state, this state shall:

- send an Enable Disable SSP (Disable) message to the SSP link layer state machines;
- send an Enable Disable SMP (Disable) message to the SMP link layer state machines;
- send an Enable Disable STP (Disable) message to the STP link layer state machines;
- send an Idle State Condition (Active) message to the SL\_P\_S link layer state machine (see 6.14.4) and SL\_P\_C link layer state machine (see 6.14.5);
- initialize and start the Idle timer;
- send a Connection Closed (Transition to Idle) confirmation to the port layer; and
- send an Enable APTA confirmation to the management application layer.

This state shall request that idle dwords be transmitted by repeatedly sending Transmit Idle Dword messages to the SL transmitter (see 6.6).

If a BROADCAST Received (Change) message, BROADCAST Received (Reserved Change 0) message, or BROADCAST Received (Reserved Change 1) message is received, then this state shall send a Change Received confirmation to the management application layer.

If a Transmit Broadcast request is received with any argument and this state has not sent a Transmit PS\_ACK Pattern message, then this state shall send a Transmit BROADCAST message with the same argument to the SL transmitter. If a Transmit Broadcast request is received and this state has sent a Transmit PS\_ACK Pattern message to the SL transmitter, then this state shall send a Retry Transmit Broadcast confirmation to the management application layer.

If the Transmit Broadcast request is received coincident with receiving either an Open Connection request or an OPEN Address Frame Received message, then this state shall send a Transmit BROADCAST message with the same argument to the SL transmitter before transitioning to another state.

If a BREAK Received message is received and the BREAK\_REPLY method of responding to received BREAK primitive sequences is enabled (see 6.16.6), then this state shall send a Transmit BREAK\_REPLY message to the SL transmitter.

After this state receives an Enable Disable SAS Link (Enable) message, this state shall:

- a) set the Reject SSP Opens state machine variable to a vendor specific default value (i.e., YES or NO);
- b) set the Reject SMP Opens state machine variable to a vendor specific default value (i.e., YES or NO); and
- c) set the Reject STP Opens state machine variable to a vendor specific default value (i.e., YES or NO).

If this state receives a NOTIFY Received (Power Loss Expected) message and the SAS port that contains this state machine supports NOTIFY (Power Loss Expected) (e.g., the SAS port is an SSP target port), then this state shall send a NOTIFY Received (Power Loss Expected) confirmation to the port layer.

If a PS\_REQ Received (Partial) message is received and the Accept Partial state machine variable is set to YES, then this state shall send:

- a) a Transmit PS\_ACK Pattern message to the SL transmitter;
- b) a Manage Phy Power Conditions (Stop DWS) request to the phy layer; and
- c) a Manage Phy Power Conditions (Stop PS) request to the phy layer.

If a PS\_REQ Received (Slumber) message is received and the Accept Slumber state machine variable is set to YES, then this state shall send:

- a) a Transmit PS\_ACK Pattern message to the SL transmitter;
- b) a Manage Phy Power Conditions (Stop DWS) request to the phy layer; and
- c) a Manage Phy Power Conditions (Stop PS) request to the phy layer.

If a PS\_REQ Received (Partial) message is received and the Accept Partial state machine variable is set to NO, then this state shall send a Transmit PS\_NAK message to the SL transmitter. If a PS\_REQ Received (Slumber) message is received and the Accept Slumber state machine variable is set to NO, then this state shall send a Transmit PS\_NAK message to the SL transmitter.

If this state receives an Open Connection request and this state has sent a Transmit PS\_ACK Pattern message to the SL transmitter, then this state shall send an Open Failed (Low Phy Power Condition) confirmation to the port layer.

If this state receives a Change Phy Power Condition request and this state has sent a Transmit PS\_ACK Pattern message to the SL transmitter, then this state shall send a Phy Power Condition Status (Retry Change Phy Power Condition Request) confirmation to the management application layer.

This state shall ignore any Change Phy Power Condition (Exit Power Condition) requests.

If this state receives a:

- a) Transmit Power Request (PWR\_REQ) message, then this state shall send a Transmit PWR\_REQ message to the SL transmitter;
- b) Transmit Power Request (PWR\_ACK) message, then this state shall send a Transmit PWR\_ACK message to the SL transmitter;

- c) Transmit Power Request (PWR\_DONE) message, then this state shall send a Transmit PWR\_DONE message to the SL transmitter; or
- d) Transmit Power Request (PWR\_GRANT) message, then this state shall send a Transmit PWR\_GRANT message to the SL transmitter.

If the Idle timer expires, then this state shall:

- a) send an Idle State Condition (Active) message to the SL\_P\_S link layer state machine (see 6.14.4) and SL\_P\_C link layer state machine (see 6.14.5); and
- b) initialize and start the Idle timer.

#### 6.18.4.2.2 Transition SL\_CC0:Idle to SL\_CC1:ArbSel

If this state has not sent a Transmit PS\_ACK Pattern message to the SL transmitter, then this transition shall occur after receiving:

- a) an Enable Disable SAS Link (Enable) message; and
- b) an Open Connection request.

The Open Connection request includes these arguments:

- a) Initiator Port Bit;
- b) Protocol;
- c) Connection Rate;
- d) Initiator Connection Tag;
- e) Destination SAS Address;
- f) Source SAS Address;
- g) Pathway Blocked Count; and
- h) Arbitration Wait Time.

If persistent connections are supported (see 4.1.13.2), then the Open Connection request includes the following argument:

- a) Send Extend Bit.

If credit advance is implemented (see 4.1.14), then the Open Connection request includes the following argument:

- a) Credit Advance Bit.

Before this transition, this state shall send:

- a) an Idle State Condition (Inactive) message to the SL\_P\_S link layer state machine (see 6.14.4);
- b) an Idle State Condition (Inactive) message to the SL\_P\_C link layer state machine (see 6.14.5);
- c) an APTA Disabled (Active Connection) confirmation to the management application layer; and
- d) a Disable APTA request to the phy layer.

#### 6.18.4.2.3 Transition SL\_CC0:Idle to SL\_CC2:Selected

This transition shall occur:

- a) after receiving both an Enable Disable SAS Link (Enable) message and an OPEN Address Frame Received message.

Before this transition, this state shall send:

- a) an Idle State Condition (Inactive) message to the SL\_P\_S link layer state machine (see 6.14.4);
- b) an Idle State Condition (Inactive) message to the SL\_P\_C link layer state machine (see 6.14.5);
- c) an APTA Disabled (Active Connection) confirmation to the management application layer; and
- d) a Disable APTA request to the phy layer.

If persistent connections are supported (see 4.1.13.2) and:

- a) the SEND EXTEND bit is set to one in the received OPEN address frame and the INITIATOR PORT bit is set to one in the received OPEN address frame, then this transition shall include an Extend Connection (Transmit) argument;
- b) the SEND EXTEND bit is set to zero in the received OPEN address frame, then this transition shall include an Extend Connection (Off) argument; or
- c) the INITIATOR PORT bit is set to zero in the received OPEN address frame, then this transition shall include an Extend Connection (Off) argument.

If credit advance is implemented (see 4.1.14) and:

- a) the CREDIT ADVANCE bit is set to one in the received OPEN address frame; and
- b) the SAS PROTOCOL field is set to 001b (i.e., SSP) in the received OPEN address frame,

then this transition shall include an Advance Credit (Received) argument.

#### **6.18.4.2.4 Transition SL\_CC0:Idle to SL\_CC8:PS\_Request**

If this state has not sent a Transmit PS\_ACK Pattern message to the SL transmitter, then this transition shall occur after receiving:

- a) an Enable Disable SAS Link (Enable) message; and
- b) a Change Phy Power Condition (Enter Slumber) or Change Phy Power Condition (Enter Partial) request from the management application layer.

This transition shall include the Phy Power Condition (Enter Partial) argument or Phy Power Condition (Enter Slumber) argument that corresponds to the Change Phy Power Condition request.

Before this transition, this state shall send an Idle State Condition (Inactive) message to the SL\_P\_S link layer state machine (see 6.14.4) and SL\_P\_C link layer state machine (see 6.14.5).

#### **6.18.4.2.5 Transition SL\_CC0:Idle to SL\_CC9:PS\_Quiet**

This transition shall occur:

- a) after receiving a PS\_ACK Pattern Transmitted message from the SL transmitter.

This transition shall include the Phy Power Condition (Enter Partial) argument or Phy Power Condition (Enter Slumber) argument that corresponds to the PS\_REQ Received message.

Before this transition, this state shall send an Idle State Condition (Inactive) message to the SL\_P\_S link layer state machine (see 6.14.4) and SL\_P\_C link layer state machine (see 6.14.5).

### **6.18.4.3 SL\_CC1:ArbSel state**

#### **6.18.4.3.1 State description**

This state is used to make a connection request.

Upon entry into this state, this state shall:

- 1) request an OPEN address frame be transmitted by sending a Transmit SOAF/Data Dwords/EOAF message to the SL transmitter with the dwords containing the OPEN address frame with its fields set to the arguments received with the Open Connection request;
- 2) initialize and start the Open Timeout timer; and
- 3) request idle dwords be transmitted by repeatedly sending Transmit Idle Dword messages to the SL transmitter.

See 6.16 for details on rate matching when opening a connection.

If this state receives an SOAF/Data Dwords/EOAF Transmitted message followed by an OPEN\_ACCEPT Received message and the SAS PROTOCOL field in the transmitted OPEN address frame was set to:

- a) STP, then this state shall send a Connection Opened (STP, Source Opened) confirmation to the port layer;
- b) SSP, then this state shall send a Connection Opened (SSP, Source Opened) confirmation to the port layer; or
- c) SMP, then this state shall send a Connection Opened (SMP, Source Opened) confirmation to the port layer.

After this state sends the Connection Opened confirmation the SMP connection, SSP connection, or SMP connection has been opened between the source phy and the destination phy.

This state shall ignore OPEN\_REJECT Received and OPEN\_ACCEPT Received messages from the time a Transmit SOAF/Data Dwords/EOAF message is sent to the SL transmitter until an SOAF/Data Dwords/EOAF Transmitted message is received from the SL transmitter.

If a BROADCAST Received (Change) message, BROADCAST Received (Reserved Change 0) message, or BROADCAST Received (Reserved Change 1) message is received, then this state shall send a Change Received confirmation to the management application layer.

If a PS\_REQ Received message is received, then this state shall send a Transmit PS\_NAK to the SL transmitter.

If an AIP Received message is received after requesting the OPEN address frame be transmitted, then this state shall reinitialize and restart the Open Timeout timer. The state machine shall not enforce a limit on the number of AIPs received.

If a Stop Arb request is received, then this state shall send an Open Failed (Arb Stopped) confirmation to the port layer.

If there is no response to the OPEN address frame before the Open Timeout timer expires, then this state shall send an Open Failed (Open Timeout Occurred) confirmation to the port layer.

If a BREAK Received message is received, then this state shall send an Open Failed (Break Received) confirmation to the port layer.

If this state receives an OPEN\_REJECT Received message listed in table 191 after transmitting the OPEN address frame, then this state shall send the corresponding Open Failed confirmation listed in table 191 to the port layer.

**Table 191 – OPEN\_REJECT Received message to Open Failed confirmation mapping**

<b>OPEN_REJECT Received message</b>	<b>Open Failed confirmation</b>
OPEN_REJECT Received (Bad Destination)	Open Failed (Bad Destination)
OPEN_REJECT Received (Connection Rate Not Supported)	Open Failed (Connection Rate Not Supported)
OPEN_REJECT Received (Protocol Not Supported)	Open Failed (Protocol Not Supported)
OPEN_REJECT Received (Reserved Abandon 1)	Open Failed (Reserved Abandon 1)
OPEN_REJECT Received (Reserved Abandon 2)	Open Failed (Reserved Abandon 2)
OPEN_REJECT Received (Reserved Abandon 3)	Open Failed (Reserved Abandon 3)
OPEN_REJECT Received (STP Resources Busy)	Open Failed (STP Resources Busy)
OPEN_REJECT Received (Wrong Destination)	Open Failed (Wrong Destination)

**Table 191 – OPEN\_REJECT Received message to Open Failed confirmation mapping**

<b>OPEN_REJECT Received message</b>	<b>Open Failed confirmation</b>
OPEN_REJECT Received (Zone Violation)	Open Failed (Zone Violation)
OPEN_REJECT Received (No Destination)	Open Failed (No Destination)
OPEN_REJECT Received (Pathway Blocked)	Open Failed (Pathway Blocked)
OPEN_REJECT Received (Reserved Continue 0)	Open Failed (Reserved Continue 0)
OPEN_REJECT Received (Reserved Continue 1)	Open Failed (Reserved Continue 1)
OPEN_REJECT Received (Reserved Initialize 0)	Open Failed (Reserved Initialize 0)
OPEN_REJECT Received (Reserved Initialize 1)	Open Failed (Reserved Initialize 1)
OPEN_REJECT Received (Reserved Stop 0)	Open Failed (Reserved Stop 0)
OPEN_REJECT Received (Reserved Stop 1)	Open Failed (Reserved Stop 1)
OPEN_REJECT Received (Retry)	Open Failed (Retry)

**6.18.4.3.2 Transition SL\_CC1:ArbSel to SL\_CC0:Idle**

This transition shall occur:

- a) after sending an Open Failed confirmation.

**6.18.4.3.3 Transition SL\_CC1:ArbSel to SL\_CC2:Selected**

This transition shall occur after receiving an SOAF/Data Dwords/EOAF Transmitted message if:

- a) one or more AIP Received messages have been received before an OPEN Address Frame Received message is received (i.e., the incoming OPEN address frame overrides the outgoing OPEN address frame); or
- b) no AIP Received messages have been received before an OPEN Address Frame Received message is received, and:
  - A) SMP frame priority is enabled (see 6.16.3):
    - a) the SAS PROTOCOL field in the transmitted OPEN address frame was set to other than SMP; and
    - b) the SAS PROTOCOL field in the received OPEN address frame is set to SMP;
  - or
  - B) the arbitration fairness rules (see 6.16.4) indicate the received OPEN address frame overrides the outgoing OPEN address frame if:
    - a) SMP frame priority is disabled (see 6.16.3);
    - b) SMP frame priority is enabled and:
      - A) the SAS PROTOCOL field in the transmitted OPEN address frame was set to other than SMP; and
      - B) the SAS PROTOCOL field in the received OPEN address frame is set to other than SMP;
  - or
  - c) SMP frame priority is enabled and:
    - A) the SAS PROTOCOL field in the transmitted OPEN address frame was set to SMP; and
    - B) the SAS PROTOCOL field in the received OPEN address frame is set to SMP.



The arbitration fairness comparison shall compare:

- a) the value of the arbitration wait time argument in the Open Connection request for the outgoing OPEN address frame; and
- b) the value of the ARBITRATION WAIT TIME field received in the incoming OPEN address frame.

If persistent connections are supported (see 4.1.13.2) and:

- a) the SEND EXTEND bit is set to one in the received OPEN address frame and the INITIATOR PORT bit is set to one in the received OPEN address frame, then this transition shall include an Extend Connection (Transmit) argument;
- b) the SEND EXTEND bit is set to zero in the received OPEN address frame, then this transition shall include an Extend Connection (Off) argument; or
- c) the INITIATOR PORT bit is set to zero in the received OPEN address frame, then this transition shall include an Extend Connection (Off) argument.

If credit advance is implemented (see 4.1.14) and:

- a) the CREDIT ADVANCE bit is set to one in the received OPEN address frame; and
- b) the SAS PROTOCOL field is set to 001b (i.e., SSP) in the received OPEN address frame,

then this transition shall include an Advance Credit (Received) argument.

#### **6.18.4.3.4 Transition SL\_CC1:ArbSel to SL\_CC3:Connected**

This transition shall occur:

- a) after sending a Connection Opened confirmation.

If the SAS PROTOCOL field in the transmitted OPEN address frame was set to:

- a) STP, then this transition shall include an Open STP Connection argument;
- b) SSP, then this transition shall include an Open SSP Connection argument; or
- c) SMP, then this transition shall include an Open SMP Connection argument.

If persistent connections are supported (see 4.1.13.2) and:

- a) the SEND EXTEND bit is set to one in the transmitted OPEN address frame and the INITIATOR PORT bit was set to one in the transmitted OPEN address frame, then this transition shall include an Extend Connection (Wait) argument;
- b) the SEND EXTEND bit is set to zero in the transmitted OPEN address frame, then this transition shall include an Extend Connection (Off) argument; or
- c) the INITIATOR PORT bit was set to zero in the transmitted OPEN address frame, then this transition shall include an Extend Connection (Off) argument.

If credit advance is implemented (see 4.1.14) and:

- a) the CREDIT ADVANCE bit is set to one in the transmitted OPEN address frame; and
- b) the SAS PROTOCOL field is set to 001b (i.e., SSP) in the transmitted OPEN address frame,

then this transition shall include an Advance Credit (Transmitted) argument.

#### **6.18.4.3.5 Transition SL\_CC1:ArbSel to SL\_CC5:BreakWait**

This transition shall occur after receiving an SOAF/Data Dwords/EOAF Transmitted message if a BREAK Received message has not been received and after:

- a) sending an Open Failed (Arb Stopped) confirmation to the port layer;
- b) sending an Open Failed (Open Timeout Occurred) confirmation to the port layer; or
- c) a NOTIFY Received (Power Loss Expected) message is received.

If a NOTIFY Received (Power Loss Expected) message was received and the SAS port that contains this state machine supports NOTIFY (Power Loss Expected) (e.g., the SAS port is an SSP target port), then this transition shall include a Power Loss Expected argument.

#### 6.18.4.3.6 Transition SL\_CC1:ArbSel to SL\_CC6:Break

This transition shall occur after:

- a) receiving an SOAF/Data Dwords/EOAF Transmitted message;
- b) receiving a BREAK Received message; and
- c) sending an Open Failed (Break Received) confirmation to the port layer.

If a NOTIFY Received (Power Loss Expected) message was received and the SAS port that contains this state machine supports NOTIFY (Power Loss Expected) (e.g., the SAS port is an SSP target port), then this transition shall include a Power Loss Expected argument.

#### 6.18.4.4 SL\_CC2:Selected state

##### 6.18.4.4.1 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, then this state shall send a Transmit OPEN\_REJECT (Wrong Destination) message to the SL transmitter (see 6.18.4.4.2);
- 2) if the OPEN address frame INITIATOR PORT bit, SAS 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), then this state shall send a Transmit OPEN\_REJECT (Protocol Not Supported) message to the SL transmitter (see 6.18.4.4.2);
- 3) if the OPEN address frame CONNECTION RATE field is set to a connection rate that is not supported, then this state shall send a Transmit OPEN\_REJECT (Connection Rate Not Supported) message to the SL transmitter (see 6.18.4.4.2);
- 4) if the OPEN address frame SAS PROTOCOL field is set to STP, the STP target port supports affiliations, and the source SAS address is not that of an STP initiator port with an affiliation established (see 6.21.6), then this state shall send a Transmit OPEN\_REJECT (STP Resources Busy) message to the SL transmitter (see 6.18.4.4.2);
- 5) if the OPEN address frame SAS PROTOCOL field is set to SSP and the Reject SSP Opens state machine variable is set to YES, then this state shall send a Transmit OPEN\_REJECT (Retry) message to the SL transmitter (see 6.18.4.4.2);
- 6) if the OPEN address frame SAS PROTOCOL field is set to SMP and the Reject SMP Opens state machine variable is set to YES, then this state shall send a Transmit OPEN\_REJECT (Retry) message to the SL transmitter (see 6.18.4.4.2);
- 7) if the OPEN address frame SAS PROTOCOL field is set to STP and the Reject STP Opens state machine variable is set to YES, then this state shall send a Transmit OPEN\_REJECT (Retry) message to the SL transmitter (see 6.18.4.4.2);
- 8) if the OPEN address frame SAS PROTOCOL field is set to SSP and the Reject SSP Opens state machine variable is set to NO, 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 (see 6.18.4.4.3);
- 9) if the OPEN address frame SAS PROTOCOL field is set to SMP and the Reject SMP Opens state machine variable is set to NO, 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 (see 6.18.4.4.3); or
- 10) if the OPEN address frame SAS PROTOCOL field is set to STP and the Reject STP Opens state machine variable is set to NO, 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 (see 6.18.4.4.3).

If this state sends a Transmit OPEN\_REJECT message to the SL transmitter, then it shall also send an Inbound Connection Rejected confirmation to the port layer.

NOTE 34 - Possible livelock scenarios occur if the BREAK\_REPLY method of responding to BREAK primitive sequences is disabled and a SAS logical phy transmits a BREAK primitive sequence to abort a connection request (e.g., if its Open Timeout timer expires). SAS logical phys responding to OPEN Address frames faster than 1 ms reduce susceptibility to this problem.

#### **6.18.4.4.2 Transition SL\_CC2:Selected to SL\_CC0:Idle**

This transition shall occur after sending a Transmit OPEN\_REJECT message to the SL transmitter.

#### **6.18.4.4.3 Transition SL\_CC2:Selected to SL\_CC3:Connected**

This transition shall occur after sending a Connection Opened confirmation to the port layer.

This transition shall include:

- a) an Open SSP Connection, Open STP Connection, or Open SMP Connection argument based on the requested protocol;
- b) the received OPEN address frame;
- c) the Extend Connection argument, if persistent connections are supported (see 4.1.13.2); and
- d) the Advance Credit argument, if the transition into this state included an Advance Credit argument.

#### **6.18.4.4.4 Transition SL\_CC2:Selected to SL\_CC5:BreakWait**

If the SAS port that contains this state machine supports NOTIFY (Power Loss Expected) (e.g., the SAS port is an SSP target port), then this transition shall occur:

- a) after receiving a NOTIFY Received (Power Loss Expected) message and shall include a Power Loss Expected argument.

#### **6.18.4.4.5 Transition SL\_CC2:Selected to SL\_CC6:Break**

This transition shall occur:

- a) after a BREAK Received message is received.

### **6.18.4.5 SL\_CC3:Connected state**

#### **6.18.4.5.1 State description**

This state enables the SSP, STP, or SMP link layer state machine to transmit dwords during a connection. See 6.17 for details on rate matching during the connection.

If this state is entered from SL\_CC1:ArbSel state or the SL\_CC2:Selected state with an argument of Open SMP Connection, then this state shall send an Enable Disable SMP (Enable) message to the SMP link layer state machines (see 6.22.6).

If this state is entered from SL\_CC1:ArbSel state or the SL\_CC2:Selected state with an argument of Open SSP Connection, then:

- 1) this state shall send an Enable Disable SSP (Enable) message to the SSP link layer state machines (see 6.20.9);
- 2) if this state is entered with an Advance Credit (Received) argument, then this state shall send an Advance Credit (Received) message to the SSP link layer state machines (see 6.20.9); and
- 3) if this state is entered with an Advance Credit (Transmitted) argument, then this state shall send an Advance Credit (Transmitted) message to the SSP link layer state machines (see 6.20.9).

If this state is entered from SL\_CC1:ArbSel state or the SL\_CC2:Selected state with an argument of Open STP Connection, then this state shall send an Enable Disable STP (Enable) message to the STP link layer state machines (see 6.21.10).

If this state is entered from SL\_CC1:ArbSel state or the SL\_CC2:Selected state with an argument of Open SSP Connection and persistent connections are supported (see 4.1.13.2), then this state if entered with:

- a) an Extend Connection (Transmit) argument, shall:
    - A) send a Persistent Connection (Transmit) message to the SSP\_EM link layer state machine (see 6.20.9.12); and
    - B) send a Persistent Connection Established (Disabled) confirmation to the port layer;
  - b) an Extend Connection (Wait) argument, shall:
    - A) send a Persistent Connection (Wait) message to the SSP\_EM link layer state machine (see 6.20.9.12); and
    - B) send a Persistent Connection Established (Disabled) confirmation to the port layer;
- or
- c) an Extend Connection (Off) argument, send a Persistent Connection (Off) message to the SSP\_EM link layer state machine (see 6.20.9.12).

This state shall request that idle dwords be transmitted by repeatedly sending Transmit Idle Dword messages to the SL transmitter until the SSP, SMP, or STP link layer state machine starts transmitting.

A CLOSE Received message may be received at any time while in this state, but shall be ignored during SSP and SMP connections. If a CLOSE Received (Clear Affiliation) message is received during an STP connection, then this state shall clear any affiliation (see 6.21.6).

If a Request Break message is received and a BREAK Received message has not been received, then this state shall send a Connection Closed (Break Requested) confirmation to the port layer.

If a BREAK Received message is received, then this state shall send a Connection Closed (Break Received) confirmation to the port layer.

#### **6.18.4.5.2 Transition SL\_CC3:Connected to SL\_CC4:DisconnectWait**

This transition shall occur:

- a) if a Request Close message is received.

#### **6.18.4.5.3 Transition SL\_CC3:Connected to SL\_CC5:BreakWait**

This transition shall occur after:

- a) sending a Connection Closed (Break Requested) confirmation to the port layer; or
- b) receiving a NOTIFY Received (Power Loss Expected) message, if the SAS port that contains this state machine supports NOTIFY (Power Loss Expected) (e.g., the SAS port is an SSP target port).

If a NOTIFY Received (Power Loss Expected) message was received and the SAS port that contains this state machine supports NOTIFY (Power Loss Expected) (e.g., the SAS port is an SSP target port), then this state shall include:

- a) a Power Loss Expected argument.

#### **6.18.4.5.4 Transition SL\_CC3:Connected to SL\_CC6:Break**

This transition shall occur:

- a) after sending a Connection Closed (Break Received) confirmation to the port layer.

**6.18.4.5.5 Transition SL\_CC3:Connected to SL\_CC7:CloseSTP**

This transition shall occur:

- a) if a CLOSE Received message is received during an STP connection.

**6.18.4.6 SL\_CC4:DisconnectWait state****6.18.4.6.1 State description**

This state closes the connection and releases all resources associated with the connection.

Upon entry into this state, this state shall:

- 1) send a Transmit CLOSE (Normal) message or Transmit CLOSE (Clear Affiliation) message to the SL transmitter (see 6.21.8); and
- 2) initialize and start the Close Timeout timer.

A CLOSE Received message may be received at any time while in this state. If a CLOSE Received (Clear Affiliation) message is received during an STP connection, then this state shall clear any affiliation (see 6.21.6).

NOTE 35 - Possible livelock scenarios occur if the BREAK\_REPLY method of responding to received BREAK primitive sequences is disabled and a SAS logical phy transmits a BREAK primitive sequence to break a connection (e.g., if its Close Timeout timer expires). SAS logical phys responding to CLOSE faster than 1 ms reduce susceptibility to this problem.

If a CLOSE Received message is received, then this state shall send a Connection Closed (Normal) confirmation to the port layer.

If a BREAK Received message is received, then this state shall send a Connection Closed (Break Received) confirmation to the port layer.

If a BREAK Received message has not been received and no CLOSE Received message is received in response to a Transmit CLOSE message before the Close Timeout timer expires, then this state shall send a Connection Closed (Close Timeout) confirmation to the port layer.

**6.18.4.6.2 Transition SL\_CC4:DisconnectWait to SL\_CC0:Idle**

This transition shall occur:

- a) after sending a Connection Closed (Normal) confirmation to the port layer.

**6.18.4.6.3 Transition SL\_CC4:DisconnectWait to SL\_CC5:BreakWait**

This transition shall occur after:

- a) receiving a NOTIFY Received (Power Loss Expected) message; or
- b) sending a Connection Closed (Close Timeout) confirmation to the port layer.

If a NOTIFY Received (Power Loss Expected) message was received and the SAS port that contains this state machine supports NOTIFY (Power Loss Expected) (e.g., the SAS port is an SSP target port), then this state shall include:

- a) a Power Loss Expected argument.

**6.18.4.6.4 Transition SL\_CC4:DisconnectWait to SL\_CC6:Break**

This transition shall occur:

- a) after sending a Connection Closed (Break Received) confirmation to the port layer.

**6.18.4.7 SL\_CC5:BreakWait state****6.18.4.7.1 State description**

This state closes the connection if one is established and releases all resources associated with the connection.

Upon entry into this state, this state shall:

- 1) send a Transmit BREAK message to the SL transmitter; and
- 2) initialize and start the Break Timeout timer.

If this state:

- a) is entered with a Power Loss Expected argument; or
- b) receives a NOTIFY Received (Power Loss Expected) message and the SAS port that contains this state machine supports NOTIFY (Power Loss Expected) (e.g., the SAS port is an SSP target port),

then this state shall send a NOTIFY Received (Power Loss Expected) confirmation to the port layer.

If a BREAK Received message is received and the BREAK\_REPLY method of responding to received BREAK primitive sequences is enabled (see 6.16.6), then this state shall send a Transmit BREAK\_REPLY message to the SL transmitter.

NOTE 36 - Some SAS logical phys compliant with SAS-1.1 send a Transmit OPEN\_REJECT (Retry) message to the SL transmitter in response to each OPEN Address Frame Received message received while in this state.

**6.18.4.7.2 Transition SL\_CC5:BreakWait to SL\_CC0:Idle**

This transition shall occur after:

- a) receiving a BREAK\_REPLY Received message if the BREAK\_REPLY method of responding to received BREAK primitive sequences is enabled (see 6.16.6);
- b) receiving a BREAK Received message if the BREAK\_REPLY method of responding to received BREAK primitive sequences is disabled (see 6.16.6); or
- c) the Break Timeout timer expires.

**6.18.4.8 SL\_CC6:Break state****6.18.4.8.1 State description**

This state closes any connection and releases all resources associated with this connection.

Upon entry into this state:

- a) if the BREAK\_REPLY method of responding to received BREAK primitive sequences is enabled (see 6.16.6), then this state shall send a Transmit BREAK\_REPLY message to the SL transmitter (see 6.18.4.8.2); and
- b) if the BREAK\_REPLY method of responding to received BREAK primitive sequences is disabled (see 6.16.6), then this state shall send a Transmit BREAK message to the SL transmitter.

If this state:

- a) is entered with a Power Loss Expected argument; or
- b) receives a NOTIFY Received (Power Loss Expected) message and the SAS port that contains this state machine supports NOTIFY (Power Loss Expected) (e.g., the SAS port is an SSP target port),

then this state shall send a NOTIFY Received (Power Loss Expected) confirmation to the port layer.

**6.18.4.8.2 Transition SL\_CC6:Break to SL\_CC0:Idle**

This transition shall occur:

- a) after sending a Transmit BREAK message or a Transmit BREAK\_REPLY message to the SL transmitter.

**6.18.4.9 SL\_CC7:CloseSTP state****6.18.4.9.1 State description**

This state closes an STP connection and releases all resources associated with the connection.

Upon entry into this state, this state shall:

- 1) send a Transmit CLOSE (Normal) message or Transmit CLOSE (Clear Affiliation) message to the SL transmitter (see 6.21.8); and
- 2) send a Connection Closed (Normal) confirmation to the port layer (see 6.18.4.9.2).

NOTE 37 - Possible livelock scenarios occur if the BREAK\_REPLY method of responding to received BREAK primitive sequences is disabled and a SAS logical phy transmits a BREAK primitive sequence to break a connection (e.g., if its Close Timeout timer expires). SAS logical phys responding to a CLOSE primitive sequence faster than 1 ms reduce susceptibility to this problem.

**6.18.4.9.2 Transition SL\_CC7:CloseSTP to SL\_CC0:Idle**

This transition shall occur:

- a) after sending a Connection Closed (Normal) confirmation to the port layer.

**6.18.4.10 SL\_CC8:PS\_Request state****6.18.4.10.1 State description**

This state requests the attached phy change to a low phy power condition.

Upon entry into this state, this state shall:

- a) if entered with the Phy Power Condition (Enter Partial) argument, then send a Transmit PS\_REQ (Partial) message to the SL transmitter;
- b) if entered with the Phy Power Condition (Enter Slumber) argument, then send a Transmit PS\_REQ (Slumber) message to the SL transmitter; and
- c) initialize and start the Power Condition Request Timeout timer.

If this state:

- a) is entered with the Phy Power Condition (Enter Partial) argument; and
- b) receives a PS\_REQ Received (Slumber) message,

then this state shall set the Phy Power Condition argument to Enter Partial.

If this state:

- a) is entered with the Phy Power Condition (Enter Slumber) argument; and
- b) receives a PS\_REQ Received (Partial) message,

then this state shall:

- a) set the Phy Power Condition argument to Enter Partial; and
- b) if the Accept Partial state machine variable is set to:
  - A) YES, then send a Transmit PS\_ACK Pattern message to the SL transmitter; or
  - B) NO, then send a Transmit PS\_NAK message to the SL transmitter.

If this state:

- a) receives a PS\_REQ Received message;
- b) the PS\_REQ Received message argument is requesting the same power condition as the Phy Power Condition argument; and
- c) the attached SAS address is greater than the port identifier (i.e., SAS address),

then this state shall send:

- a) a Transmit PS\_ACK Pattern message to the SL transmitter;
- b) a Manage Phy Power Conditions (Stop DWS) request to the phy layer; and
- c) a Manage Phy Power Conditions (Stop PS) request to the phy layer.

If a Change Phy Power Condition request is received, then this state shall send a Phy Power Condition Status (Retry Change Phy Power Condition Request) confirmation to the management application layer.

If an Open Connection request is received, then this state shall send an Open Failed (Low Phy Power Condition) confirmation to the port layer.

If a BROADCAST Received (Change) message, BROADCAST Received (Reserved Change 0) message, or BROADCAST Received (Reserved Change 1) message is received, then this state shall send a Change Received confirmation to the management application layer.

If a Transmit Broadcast request is received, then this state shall send a Retry Transmit Broadcast confirmation to the management application layer.

#### **6.18.4.10.2 Transition SL\_CC8:PS\_Request to SL\_CC9:PS\_Quiet**

This transition shall occur after:

- a) receiving a PS\_ACK Pattern Transmitted message from the SL transmitter; or
- b) receiving a PS\_ACK Received message.

This transition shall include the:

- a) Phy Power Condition argument.

#### **6.18.4.10.3 Transition SL\_CC8:PS\_Request to SL\_CC0:Idle**

This transition shall occur after:

- a) sending a Transmit PS\_NAK message to the SL transmitter;
- b) receiving a PS\_NAK Received message; or
- c) the Power Condition Request Timeout timer expires.

#### **6.18.4.10.4 Transition SL\_CC8:PS\_Request to SL\_CC2:Selected**

This transition shall occur:

- a) after receiving an OPEN Address Frame Received message.

#### **6.18.4.10.5 Transition SL\_CC8:PS\_Request to SL\_CC6:Break**

This transition shall occur:

- a) after receiving a BREAK Received message.

#### **6.18.4.11 SL\_CC9:PS\_Quiet state**

##### **6.18.4.11.1 State description**

This state requests the phy be placed into a low phy power condition.



If this state is entered with a Phy Power Condition (Enter Partial) argument, then this state shall send:

- a) a Manage Phy Power Conditions (Enter Partial) request to the phy layer; and
- b) a Phy Power Condition Status (In Partial) confirmation to the management application layer.

If this state is entered with Phy Power Condition (Enter Slumber) argument, then this state shall send:

- a) a Manage Phy Power Conditions (Enter Slumber) request to the phy layer; and
- b) a Phy Power Condition Status (In Slumber) confirmation to the management application layer.

If this state receives a:

- a) Change Phy Power Condition (Enter Slumber) request and the current phy power condition is partial;  
or
- b) Change Phy Power Condition (Enter Partial) request and the current phy power condition is slumber,

then this state shall send a Phy Power Condition Status (Request Exit Power Condition) confirmation to the management application layer.

If a Transmit Broadcast request is received, then this state shall send a Retry Transmit Broadcast confirmation to the management application layer.

If this state receives an Open Connection request or a Change Phy Power Condition (Exit Power Condition) request, then this state shall send a Manage Phy Power Conditions (Exit) request to the phy layer.

#### **6.18.4.11.2 Transition SL\_CC9:PS\_Quiet to SL\_CC0:Idle**

This transition shall occur:

- a) after a Phy Layer Ready (SAS) confirmation is received; and
- b) an Open Connection request has not been received.

#### **6.18.4.11.3 Transition SL\_CC9:PS\_Quiet to SL\_CC1:ArbSel**

This transition shall occur after receiving both a Phy Layer Ready (SAS) confirmation and an Open Connection request. The Open Connection request includes these arguments:

- a) Initiator Port Bit;
- b) Protocol;
- c) Connection Rate;
- d) Initiator Connection Tag;
- e) Destination SAS Address;
- f) Source SAS Address;
- g) Pathway Blocked Count; and
- h) Arbitration Wait Time.

If persistent connections are supported (see 4.1.13.2), then the Open Connection request includes the following argument:

- a) Send Extend Bit.

If credit advance is implemented (see 4.1.14), then the Open Connection request includes the following argument:

- a) Credit Advance Bit.

### **6.19 XL (link layer for expander logical phys) state machine**

#### **6.19.1 XL state machine overview**

The XL state machine controls the flow of dwords on the logical link and establishes and maintains connections with another XL state machine as facilitated by the expander function (e.g., the ECM and ECR).

This state machine consists of the following states:

- a) XL0:Idle (see 6.19.3) (initial state);
- b) XL1:Request\_Path (see 6.19.4);
- c) XL2:Request\_Open (see 6.19.5);
- d) XL3:Open\_Confirm\_Wait (see 6.19.6);
- e) XL5:Forward\_Open (see 6.19.7);
- f) XL6:Open\_Response\_Wait (see 6.19.8);
- g) XL7:Connected (see 6.19.9);
- h) XL8:Close\_Wait (see 6.19.10);
- i) XL9:Break (see 6.19.11);
- j) XL10:Break\_Wait (see 6.19.12);
- k) XL11:PS\_Request (see 6.19.13); and
- l) XL12:PS\_Quiet (see 6.19.14).

NOTE 38 - Actions contained in the removed XL4:Open\_Reject state have been placed into the XL1:Request\_Path state.

This state machine shall start in the XL0:Idle state. The XL state machine shall transition to the XL0:Idle state from any other state after receiving an Enable Disable SAS Link (Disable) message from the SL\_IR state machines (see 6.12).

This state machine receives the following messages from the SL\_IR state machines:

- a) Enable Disable SAS Link (Enable); and
- b) Enable Disable SAS Link (Disable).

This state machine sends the following messages to the SL\_P\_S link layer state machine (see 6.14.4):

- a) Idle State Condition (Active); and
- b) Idle State Condition (Inactive).

Any message received by a state that is not referred to in the description of that state shall be ignored.

If this state machine receives a Manage Power Conditions (Accept Partial) request from the ECM, then this state machine shall set the Accept Partial state machine variable (see table 193) to YES. If this state machine receives a Manage Power Conditions (Accept Slumber) request from the ECM, then this state machine shall set the Accept Slumber state machine variable (see table 193) to YES.

If this state machine receives a Manage Power Conditions (Reject Partial) request from the ECM, then this state machine shall set the Accept Partial state machine variable to NO. If this state machine receives a Manage Power Conditions (Reject Slumber) request from the ECM, then this state machine shall set the Accept Slumber state machine variable to NO.

The default value of the Accept Partial state machine variable and Accept Slumber state machine variable shall be NO.

This state machine shall maintain the timers listed in table 192.

**Table 192 – XL state machine timers**

Timer	Initial value
Partial Pathway Timeout timer	Partial pathway timeout value (see 6.16.5.4)
Break Timeout timer	1 ms
Power Condition Request Timeout timer	1 ms

**Table 192 – XL state machine timers**

Timer	Initial value
Idle timer	1 ms
SSP Maximum Connection Time Limit timer	For SSP connections, the value in the SSP CONNECT TIME LIMIT field (see 9.4.3.4). For SMP connections and STP connections this timer is not used.
Wait For Frame Timeout timer	1 ms
Delay Expander Forward Open Indication timer	The value in the INITIAL TIME TO DELAY EXPANDER FORWARD OPEN INDICATION field (see 9.4.3.4).

The XL state machine shall maintain the state machine variables listed in table 193.

**Table 193 – XL state machine variable**

State machine variable	Description
Accept Partial	Used to determine if the ECM is permitting this phy to enter a partial phy power condition.
Accept Slumber	Used to determine if the ECM is permitting this phy to enter a slumber phy power condition.
Open Source SAS Address	The SAS address of the SAS port that originated the OPEN address frame.
Open Destination SAS Address	The SAS address of the SAS port that is the destination of the OPEN address frame.

Figure 160 shows several states in the XL state machine.

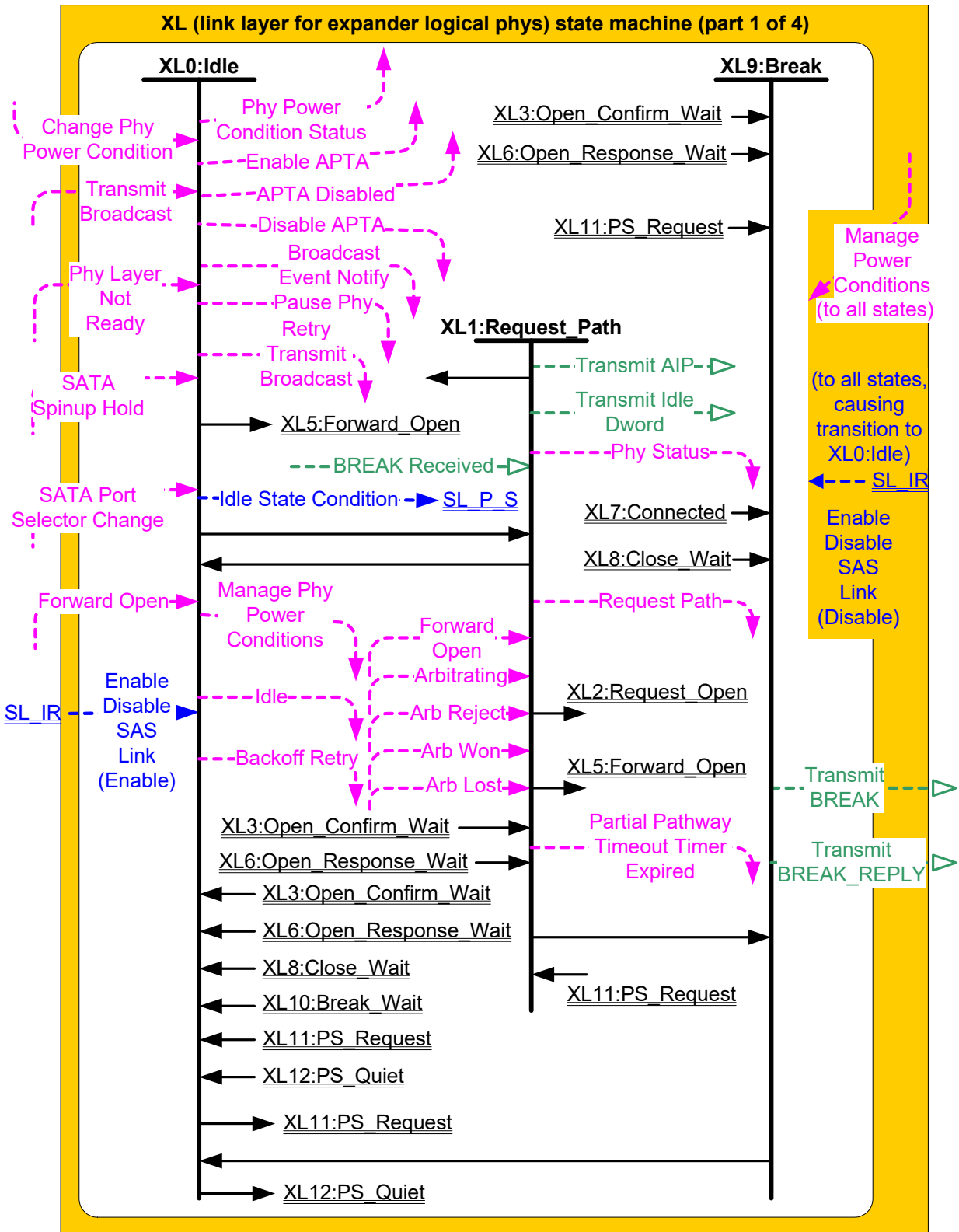


Figure 160 –XL (link layer for expander logical phys) state machine (1 of 4)

Figure 161 shows additional states in the XL state machine.

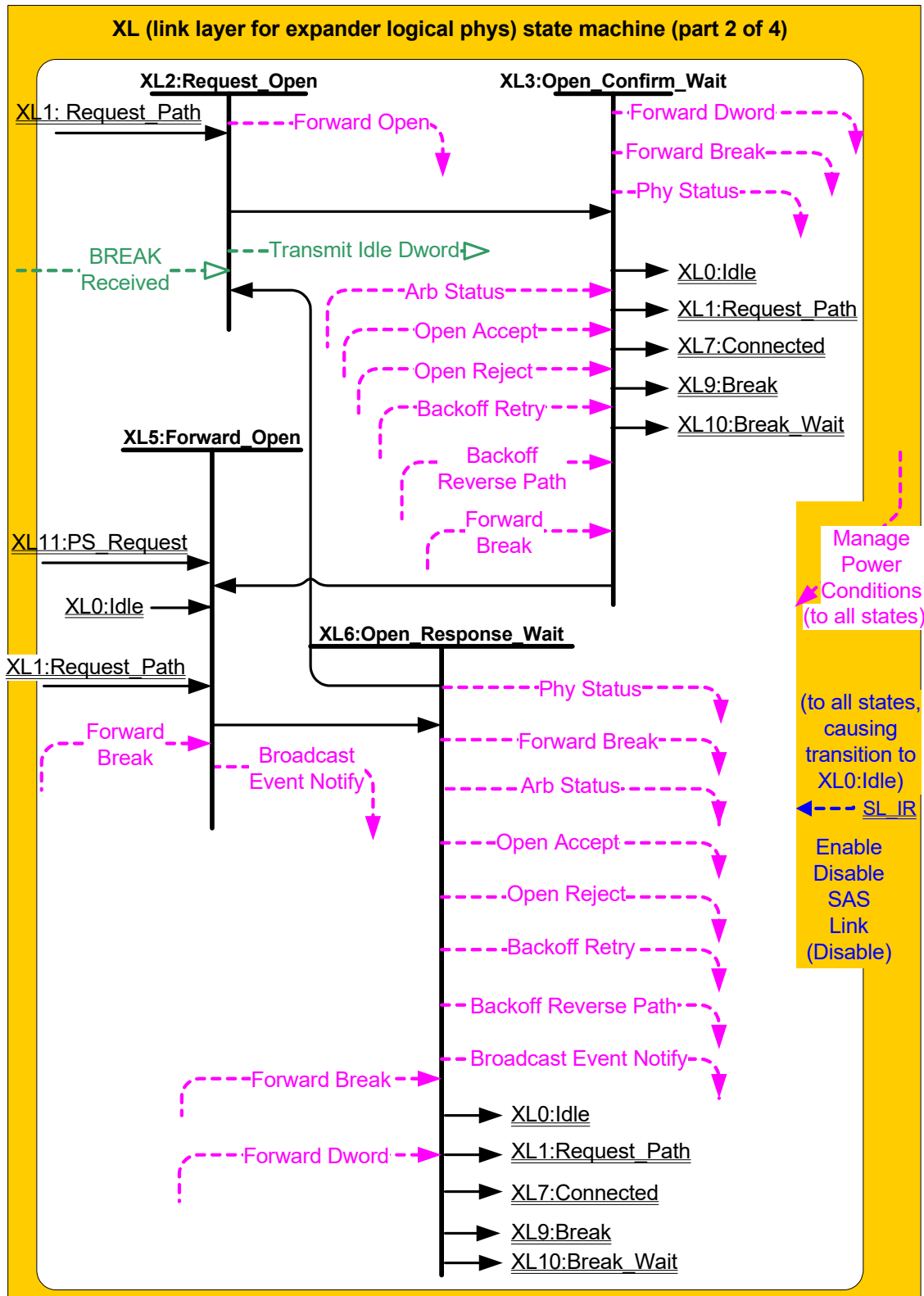


Figure 161 –XL (link layer for expander logical phys) state machine (2 of 4)

Figure 162 shows additional states in the XL state machine.

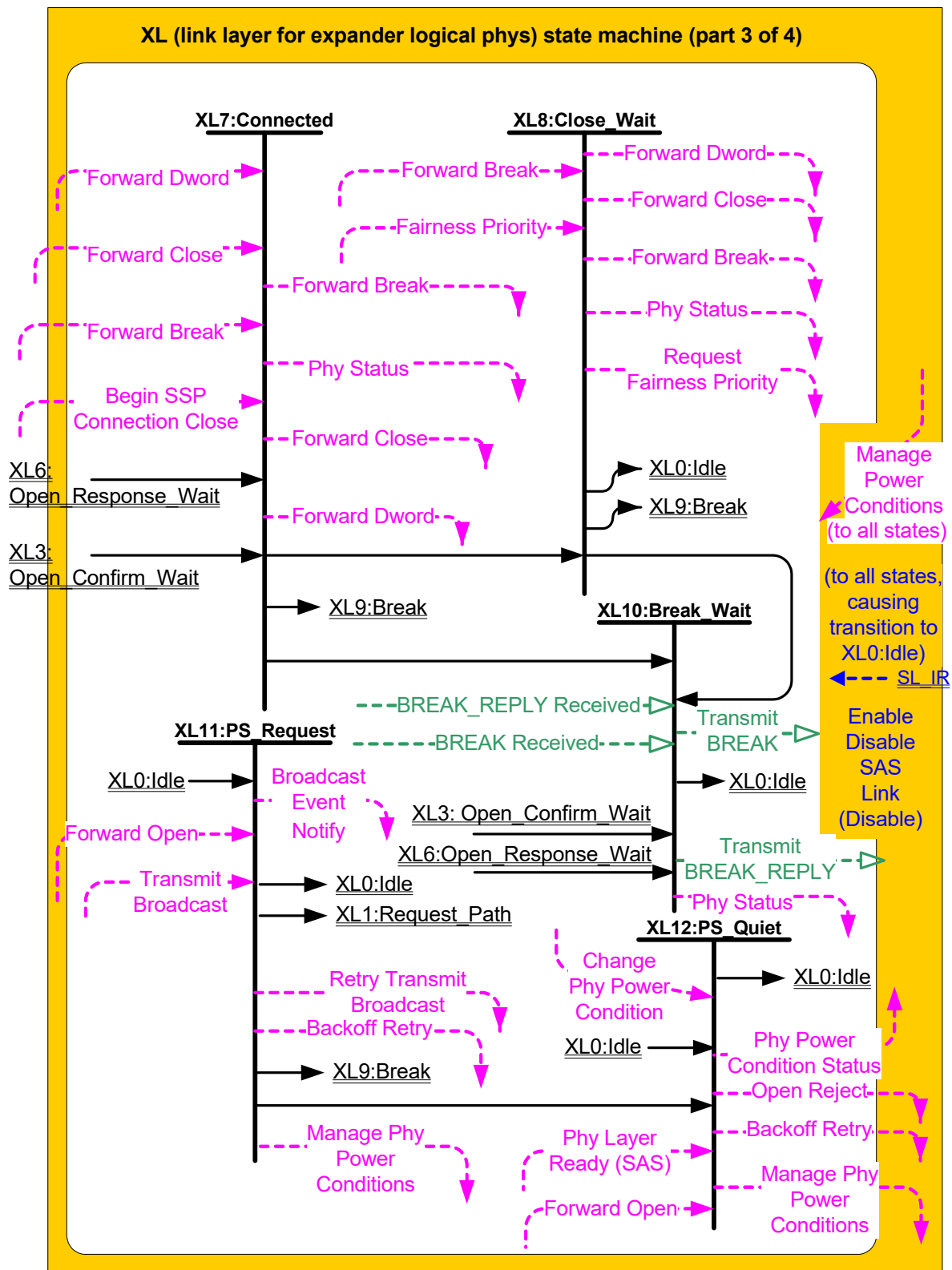


Figure 162 –XL (link layer for expander logical phys) state machine (3 of 4)

Figure 163 shows additional states in the XL state machine.

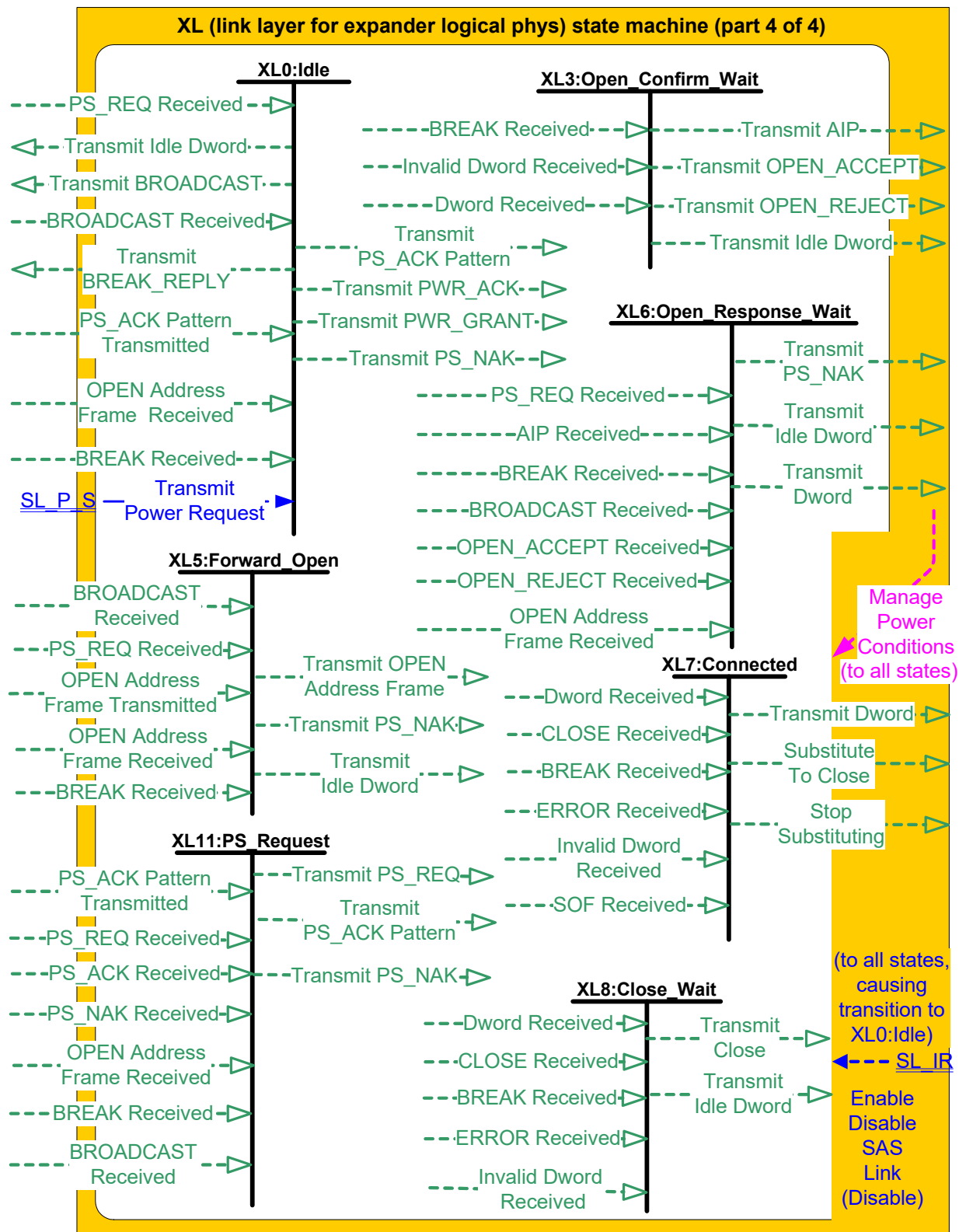


Figure 163 –XL (link layer for expander logical phys) state machine (4 of 4)

### 6.19.2 XL transmitter and receiver

The XL transmitter receives the following messages from the XL state machine 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 BREAK\_REPLY;
- e) Transmit BROADCAST with an argument indicating the specific type (e.g., Transmit BROADCAST (Change));
- f) Transmit CLOSE with an argument indicating the specific type (e.g., Transmit CLOSE (Normal)) and with the following arguments, if any:
  - A) Open Connection Rate;
  - B) Open Destination SAS Address;
  - C) Open Arbitration Wait Time;
  - D) SMP Open Priority;
  - E) High Priority; and
  - F) Hop Count;
- g) Transmit OPEN\_ACCEPT;
- h) Transmit OPEN\_REJECT, with an argument indicating the specific type (e.g., Transmit OPEN\_REJECT (No Destination));
- i) Transmit PS\_REQ with an argument indicating the specific type (e.g., Transmit PS\_REQ (Partial) or Transmit PS\_REQ (Slumber));
- j) Transmit PS\_ACK Pattern (see 6.13);
- k) Transmit PS\_NAK;
- l) Transmit PWR\_ACK;
- m) Transmit PWR\_GRANT;
- n) Substitute to Close;
- o) Stop Substituting;
- p) Transmit OPEN Address Frame; and
- q) Transmit Dword.

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

- a) OPEN Address Frame Transmitted; and
- b) PS\_ACK Pattern Transmitted.

The XL transmitter shall ensure physical link rate tolerance management requirements are met (see 6.5) while originating dwords or originating SPL packets.

The XL transmitter shall ensure physical link rate tolerance management requirements are met while forwarding dwords or forwarding SPL packets (i.e., during a connection) by inserting or deleting as many deletable primitives or deletable extended binary primitives as required to match the transmit and receive connection rates (see 6.5.4).

The XL transmitter shall ensure physical link rate tolerance management requirements are met (see 6.5) during and after switching from forwarding dwords or forwarding SPL packets to originating dwords or forwarding SPL packets, including, for example:

- a) when transmitting a BREAK primitive sequence;
- b) when transmitting a BREAK\_REPLY primitive sequence;
- c) when transmitting a CLOSE primitive sequence;
- d) when transmitting an idle dword after closing a connection (i.e., after receiving a BREAK primitive sequence, BREAK\_REPLY primitive sequence, or CLOSE primitive sequence);
- e) while transmitting a SATA frame to a SAS logical link during an STP connection, when transmitting the first SATA\_HOLD in response to detection of SATA\_HOLD; and
- f) while receiving dwords of a SATA frame from a SAS logical link during an STP connection, when transmitting SATA\_HOLD.



The XL transmitter may insert a deletable primitive or a deletable extended binary primitive before transmitting a BREAK, BREAK\_REPLY, CLOSE, or SATA\_HOLD to meet physical link rate tolerance management requirements.

The XL transmitter shall insert a deletable primitive or a deletable extended binary primitive before switching from originating dwords or originating SPL packets to forwarding dwords or forwarding SPL packets, including, for example:

- a) when transmitting OPEN\_ACCEPT;
- b) when transmitting the last idle dword before a connection is established (i.e., after receiving OPEN\_ACCEPT);
- c) while transmitting a SATA frame to a SAS logical link during an STP connection, when transmitting the last dword from the STP flow control buffer in response to release of SATA\_HOLD;
- d) while transmitting a SATA frame to a SAS logical link during an STP connection, when transmitting the last SATA\_HOLD in response to release of SATA\_HOLD (e.g., if the STP flow control buffer is empty); and
- e) while receiving dwords of a SATA frame from a SAS logical link during an STP connection, when transmitting the last SATA\_HOLD.

NOTE 39 - This ensures that physical link rate tolerance management requirements are met, even if the forwarded dword stream does not include a deletable primitive until the last possible dword.

The XL transmitter shall ensure rate matching requirements are met during a connection (see 6.17).

After receiving a Substitute To Close message the XL transmitter shall transmit:

- a) an RRDY (CLOSE) in place of each received RRDY (NORMAL); and
- b) an EXTEND\_CONNECTION (CLOSE) in place of each received EXTEND\_CONNECTION (NORMAL).

If the XL transmitter receives a Stop Substituting message after receiving a Substitute To Close message, then the XL transmitter shall cancel the replacement:

- a) of RRDY (NORMAL) with RRDY (CLOSE); and
- b) of EXTEND\_CONNECTION (NORMAL) with EXTEND\_CONNECTION (CLOSE).

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, primitive parameters, frames, and dwords received from the SP\_DWS receiver (see 5.15.2) and the SP\_PS receiver (see 5.16.2):

- a) AIP Received with an argument indicating the specific type (e.g., AIP Received (Normal));
- b) BREAK Received;
- c) BREAK\_REPLY Received;
- d) BROADCAST Received;
- e) CLOSE Received with an argument indicating the specific type (e.g., CLOSE Received (Normal)) and with the following arguments, if any, received in Dword Received (Primitive Parameter) confirmations associated with the received CLOSE:
  - A) Open Connection Rate;
  - B) Open Destination SAS Address;
  - C) Open Arbitration Wait Time;
  - D) SMP Open Priority;
  - E) High Priority;
  - F) Hop Count; and
  - G) Source Phy Identifier (i.e., the phy identifier of the expander logical phy that received the CLOSE);
- f) OPEN\_ACCEPT Received;
- g) OPEN\_REJECT Received;
- h) OPEN Address Frame Received;

- i) PS\_REQ Received with an argument indicated the specific type (e.g., PS\_REQ Received (Partial)) or PS\_REQ Received (Slumber));
- j) PS\_ACK Received;
- k) PS\_NAK Received;
- l) SOF Received;
- m) Dword Received with an argument indicating the data dword or primitive received. Deletable primitives are not included; and
- n) Invalid Dword Received.

The XL receiver shall ignore all other dwords.

While receiving an address frame, if the XL receiver receives an invalid dword or ERROR, then the XL receiver shall either:

- a) ignore the invalid dword or ERROR; or
- b) discard the address frame.

The XL transmitter relationship to other transmitters is defined in 4.3.2. The XL receiver relationship to other receivers is defined in 4.3.3.

### 6.19.3 XL0:Idle state

#### 6.19.3.1 State description

This state is the initial state and is the state that is used when there is no connection pending or established.

Upon entry into this state, this state shall:

- a) send an Idle State Condition (Active) message to the SL\_P\_S link layer state machine;
- b) initialize the Delay Expander Forward Open Indication timer to the Delay Expander Forward Open Indication Timer argument and start the Delay Expander Forward Open Indication timer;
- c) send an Enable APTA confirmation to the ECM; and
- d) initialize and start the Idle timer.

This state shall repeatedly send Idle requests to the ECM.

This state shall repeatedly send a Pause Phy response with the Phy Identifier argument set to the Source Phy Identifier argument to the ECM until the Delay Expander Forward Open Indication timer expires.

If a Phy Layer Not Ready confirmation is received, then this state shall send a Broadcast Event Notify (Phy Not Ready) request to the BPP.

If a SATA Spinup Hold confirmation is received, then this state shall send a Broadcast Event Notify (SATA Spinup Hold) request to the BPP.

If an Enable Disable SAS Link (Enable) message is received, then this state shall send a Broadcast Event Notify (Identification Sequence Complete) request to the BPP.

If a SATA Port Selector Change confirmation is received, then this state shall send a Broadcast Event Notify (SATA Port Selector Change) request to the BPP.

If a BROADCAST Received message is received, then this state shall send a Broadcast Event Notify request to the BPP with the argument indicating the specific BROADCAST primitive sequence received (e.g., Change Received).

If a Transmit Broadcast indication is received and this state has not sent a Transmit PS\_ACK Pattern message, then this state shall send a Transmit BROADCAST message to the XL transmitter with an argument specifying the specific type from the Transmit Broadcast indication, otherwise this state shall request that idle dwords be transmitted by repeatedly sending Transmit Idle Dword messages to the XL transmitter. If a Transmit Broadcast indication is received and this state has sent a Transmit PS\_ACK Pattern message to the XL transmitter, then this state shall send a Retry Transmit Broadcast request to the BPP.

If a BREAK Received message is received and the BREAK\_REPLY method of responding to received BREAK primitive sequences is enabled (see 6.16.6), then this state shall send a Transmit BREAK\_REPLY message to the XL transmitter.

If a PS\_REQ Received (Partial) message is received and the Accept Partial state machine variable is set to YES, then this state shall send:

- a) a Transmit PS\_ACK Pattern message to the XL transmitter;
- b) a Manage Phy Power Conditions (Stop DWS) request to the phy layer; and
- c) a Manage Phy Power Conditions (Stop PS) request to the phy layer.

If a PS\_REQ Received (Slumber) message is received and the Accept Slumber state machine variable is set to YES, then this state shall send:

- a) a Transmit PS\_ACK Pattern message to the XL transmitter;
- b) a Manage Phy Power Conditions (Stop DWS) request to the phy layer; and
- c) a Manage Phy Power Conditions (Stop PS) request to the phy layer.

If a PS\_REQ Received (Partial) message is received and the Accept Partial state machine variable is set to NO, then this state shall send a Transmit PS\_NAK message to the XL transmitter. If a PS\_REQ Received (Slumber) message is received and the Accept Slumber state machine variable is set to NO, then this state shall send a Transmit PS\_NAK message to the XL transmitter.

If a Forward Open indication is received from the ECM and this state has sent a Transmit PS\_ACK Pattern message to the XL transmitter, then this state shall send a Backoff Retry response to the ECR.

If a Change Phy Power Condition request is received and this state has sent a Transmit PS\_ACK Pattern message to the XL transmitter, then this state shall send a Phy Power Condition Status (Retry Change Phy Power Condition Request) confirmation to the ECM.

This state shall ignore any Change Phy Power Condition (Exit Power Condition) requests.

If this state receives a:

- a) Transmit Power Request (PWR\_ACK) message, then this state shall send a Transmit PWR\_ACK message to the SL transmitter; or
- b) Transmit Power Request (PWR\_GRANT) message, then this state shall send a Transmit PWR\_GRANT message to the SL transmitter.

If the Idle timer expires, then this state shall:

- a) send an Idle State Condition (Active) message to the SL\_P\_S link layer state machine; and
- b) initialize and start the Idle timer.

### 6.19.3.2 Transition XL0:Idle to XL1:Request\_Path

This transition shall occur if:

- a) an Enable Disable SAS Link (Enable) message has been received;
- b) a Forward Open indication is not being received; and
- c) an OPEN Address Frame Received message is received.

This state shall include an OPEN Address Frame Received argument that contains the arguments received in the OPEN Address Frame Received message.

Before this transition, this state shall:

- a) send an Idle State Condition (Inactive) message to the SL\_P\_S link layer state machine (see 6.14.4);
- b) send an APTA Disabled (Active Connection) confirmation to the ECM;
- c) send a Disable APTA request to the phy layer;
- d) if a Transmit Broadcast indication is received coincident with receiving the OPEN Address Frame Received message, then send a Transmit BROADCAST message with the same argument to the XL transmitter;
- e) if a BREAK Received message is received coincident with receiving the OPEN Address Frame Received message, then:

- A) send a Transmit BREAK\_REPLY message to the XL transmitter if the BREAK\_REPLY method of responding to received BREAK primitive sequences is enabled (see 6.16.6); and
- B) send a Transmit BREAK message to the XL transmitter if the BREAK\_REPLY method of responding to received BREAK primitive sequences is disabled (see 6.16.6);
- f) stop the Delay Expander Forward Open Indication timer; and
- g) stop the Idle timer.

#### 6.19.3.3 Transition XL0:Idle to XL5:Forward\_Open

If this state has not sent a Transmit PS\_ACK Pattern message to the XL transmitter, then this transition shall occur after receiving:

- a) an Enable Disable SAS Link (Enable) message; and
- b) a Forward Open indication.

This transition shall include an ECR Forward Open argument that contains the arguments received in the Forward Open indication.

If an OPEN Address Frame Received message is received, then this transition shall include:

- a) an OPEN Address Frame Received argument that contains the arguments received in the OPEN Address Frame Received message.

Before this transition, this state shall:

- a) send an Idle State Condition (Inactive) message to the SL\_P\_S link layer state machine (see 6.14.4);
- b) send an APTA Disabled (Active Connection) confirmation to the ECM;
- c) send a Disable APTA request to the phy layer;
- d) if a Transmit Broadcast indication is received coincident with receiving either a Forward Open indication or an OPEN Address Frame Received message, then send a Transmit BROADCAST message with the same argument to the XL transmitter;
- e) if a BREAK Received message is received coincident with receiving either a Forward Open indication or an OPEN Address Frame Received message, then:
  - A) send a Transmit BREAK\_REPLY message to the XL transmitter if the BREAK\_REPLY method of responding to received BREAK primitive sequences is enabled (see 6.16.6); and
  - B) send a Transmit BREAK message to the XL transmitter if the BREAK\_REPLY method of responding to received BREAK primitive sequences is disabled (see 6.16.6);
- f) stop the Delay Expander Forward Open Indication timer; and
- g) stop the Idle timer.

#### 6.19.3.4 Transition XL0:Idle to XL11:PS\_Request

If this state has not sent a Transmit PS\_ACK Pattern message to the XL transmitter, then this transition shall occur after receiving:

- a) an Enable Disable SAS Link (Enable) message; and
- b) a Change Phy Power Condition request.

This transition shall include the following arguments that corresponds to the Change Phy Power Condition request:

- a) Phy Power Condition (Enter Partial); or
- b) Phy Power Condition (Enter Slumber).

Before this transition, this state shall:

- a) send an Idle State Condition (Inactive) message to the SL\_P\_S link layer state machine (see 6.14.4);
- b) stop the Delay Expander Forward Open Indication timer; and
- c) stop the Idle timer.

**6.19.3.5 Transition XL0:Idle to XL12:PS\_Quiet**

This transition shall occur:

- a) after receiving a PS\_ACK Pattern Transmitted message from the XL transmitter.

This transition shall include the following arguments that corresponds to the PS\_REQ Received message:

- a) Phy Power Condition (Enter Partial); or
- b) Phy Power Condition (Enter Slumber).

Before this transition, this state shall:

- a) send an Idle State Condition (Inactive) message to the SL\_P\_S link layer state machine (see 6.14.4);
- b) stop the Delay Expander Forward Open Indication timer; and
- c) stop the Idle timer.

**6.19.4 XL1:Request\_Path state****6.19.4.1 State description**

If this state is entered from the XL6:Open\_Response\_Wait state, then this state shall set the High Priority argument to one. If this state is entered from any other state, then this state shall set the High Priority argument to zero.

If the OPEN Address Frame Received (Protocol) argument is set to SMP, then this state shall set the SMP Open Priority argument to one. If the OPEN Address Frame Received (Protocol) argument is set to SSP or STP, then this state shall set the SMP Open Priority argument to zero.

Upon entry into this state, this state shall:

- a) send a Request Path request to the ECM with:
  - A) the following OPEN Address Frame Received arguments:
    - a) Initiator Port Bit;
    - b) Protocol;
    - c) Connection Rate;
    - d) Initiator Connection Tag;
    - e) Destination SAS Address;
    - f) Source SAS Address;
    - g) Pathway Blocked Count;
    - h) Arbitration Wait Time;
  - B) an SMP Open Priority argument; and
  - C) a High Priority argument;
- b) set the Open Source SAS Address state machine variable to the Source SAS Address argument; and
- c) set the Open Destination SAS Address state machine variable to the Destination SAS Address argument.

This state is used to arbitrate for connection resources and to specify the destination of the connection.

If an Arbitrating (Normal) confirmation is received, then this state shall repeatedly send Transmit AIP (Normal) messages and Transmit Idle Dword messages to the XL transmitter in accordance with AIP transmission rules (see 6.16.5.3).

If an Arbitrating (Waiting On Partial) confirmation or an Arbitrating (Blocked On Partial) confirmation is received, then this state shall repeatedly send Transmit AIP (Waiting On Partial) messages and Transmit Idle Dword messages to the XL transmitter in accordance with AIP transmission rules (see 6.16.5.3).

If an Arbitrating (Waiting On Partial) confirmation is received, then this state shall repeatedly send a Phy Status (Partial Pathway) response to the ECM.

If an Arbitrating (Blocked On Partial) confirmation is received, then this state shall repeatedly send a Phy Status (Blocked Partial Pathway) response to the ECM.

If an Arbitrating (Waiting On Connection) confirmation is received, then this state shall repeatedly send Transmit AIP (Waiting On Connection) messages and Transmit Idle Dword messages to the XL transmitter in accordance with AIP transmission rules (see 6.16.5.3).

If an Arbitrating (Waiting On Connection) confirmation is received, then this state shall repeatedly send a Phy Status (Connection) response to the ECM.

This state shall send one of the following messages to the XL transmitter:

- a) a Transmit OPEN\_REJECT (No Destination) message when an Arb Reject (No Destination) confirmation is received;
- b) a Transmit OPEN\_REJECT (Bad Destination) message when an Arb Reject (Bad Destination) confirmation is received;
- c) a Transmit OPEN\_REJECT (Connection Rate Not Supported) message when an Arb Reject (Connection Rate Not Supported) confirmation is received;
- d) a Transmit OPEN\_REJECT (Zone Violation) message when an Arb Reject (Zone Violation) confirmation is received;
- e) a Transmit OPEN\_REJECT (Pathway Blocked) message when an Arb Reject (Pathway Blocked) confirmation is received; or
- f) a Transmit OPEN\_REJECT (Retry) message when an Arb Reject (Retry) confirmation is received.

This state maintains the Partial Pathway Timeout timer.

If the Partial Pathway Timeout timer is not already running, then the Partial Pathway Timeout timer shall be initialized and started when an Arbitrating (Blocked On Partial) confirmation is received.

If the Partial Pathway Timeout timer is already running, then the Partial Pathway Timeout timer shall continue to run if an Arbitrating (Blocked On Partial) confirmation is received.

The Partial Pathway Timeout timer shall be stopped when one of the following confirmations is received:

- a) Arbitrating (Waiting On Partial); or
- b) Arbitrating (Waiting On Connection).

If the Partial Pathway Timeout timer expires, then this state shall send a Partial Pathway Timeout Timer Expired request to the ECM.

#### **6.19.4.2 Transition XL1:Request\_Path to XL0:Idle**

This transition shall occur if:

- a) a BREAK Received message has not been received; and
- b) an Arb Lost confirmation is received or after sending a Transmit OPEN\_REJECT message to the XL transmitter.

#### **6.19.4.3 Transition XL1:Request\_Path to XL2:Request\_Open**

This transition shall occur if:

- a) a BREAK Received message has not been received; and
- b) an Arb Won confirmation is received.

This transition shall include an OPEN Address Frame Received argument containing the arguments in the received OPEN Address Frame Received argument.

#### **6.19.4.4 Transition XL1:Request\_Path to XL5:Forward\_Open**

This transition shall occur if a Forward Open indication is received and none of the following confirmations have been received:

- a) Arbitrating (Normal);
- b) Arbitrating (Waiting On Partial);
- c) Arbitrating (Blocked On Partial);

- d) Arbitrating (Waiting On Connection);
- e) Arb Won;
- f) Arb Lost;
- g) Arb Reject (No Destination);
- h) Arb Reject (Bad Destination);
- i) Arb Reject (Connection Rate Not Supported);
- j) Arb Reject (Zone Violation);
- k) Arb Reject (Pathway Blocked); or
- l) Arb Reject (Retry).

This transition shall include:

- a) an OPEN Address Frame Received argument containing the arguments in the received OPEN Address Frame Received argument;
- b) an ECR Forward Open argument containing the arguments received in the Forward Open indication; and
- c) a BREAK Received argument if a BREAK Received message was received.

#### **6.19.4.5 Transition XL1:Request\_Path to XL9:Break**

This transition shall occur:

- a) after receiving a BREAK Received message if a Forward Open indication has not been received.

#### **6.19.5 XL2:Request\_Open state**

##### **6.19.5.1 State description**

This state is used to forward an OPEN address frame through the ECR to a destination phy.

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

Upon entry into this state, this state shall:

- a) send a Forward Open request to the ECR, with the following received OPEN Address Frame Received arguments:
  - A) Initiator Port Bit;
  - B) Protocol;
  - C) Features;
  - D) Connection Rate;
  - E) Initiator Connection Tag;
  - F) Destination SAS Address;
  - G) Source SAS Address;
  - H) Source Zone Group;
  - I) Pathway Blocked Count;
  - J) Arbitration Wait Time;
  - K) Compatible Features; and
  - L) More Compatible Features;
- b) set the Open Source SAS Address state machine variable to the Source SAS Address argument; and
- c) set the Open Destination SAS Address state machine variable to the Destination SAS Address argument.

##### **6.19.5.2 Transition XL2:Request\_Open to XL3:Open\_Confirm\_Wait**

This transition shall occur:

- a) after sending a Forward Open request to the ECR.

This transition shall include an OPEN Address Frame Received argument containing the arguments in the received OPEN Address Frame Received argument.

If a BREAK Received message is received, then this transition shall include:

- a) a BREAK Received argument.

### 6.19.6 XL3:Open\_Confirm\_Wait state

#### 6.19.6.1 State description

This state waits for confirmation for an OPEN address frame sent on a destination phy.

This state shall send the following messages to the XL transmitter:

- a) a Transmit AIP (Normal) message when an Arb Status (Normal) confirmation is received;
- b) a Transmit AIP (Waiting On Partial) message when an Arb Status (Waiting On Partial) confirmation is received;
- c) a Transmit AIP (Waiting On Connection) message when an Arb Status (Waiting On Connection) confirmation is received;
- d) a Transmit AIP (Waiting On Device) message when an Arb Status (Waiting On Device) confirmation is received;
- e) a Transmit OPEN\_ACCEPT message when an Open Accept confirmation is received;
- f) a Transmit OPEN\_REJECT message when an Open Reject confirmation is received with the argument from the Open Reject confirmation, after releasing path resources; or
- g) request idle dwords be transmitted by repeatedly sending Transmit Idle Dword messages when none of the previous conditions are present.

If a Backoff Retry confirmation is received, then this state shall release path resources.

If a BREAK Received message is received or a BREAK Received argument is included in the transition into this state, then this state shall send a Forward Break request to the ECR.

This state shall repeatedly send a Phy Status (Partial Pathway) response to the ECM until an Arb Status (Waiting On Partial) confirmation is received. After an Arb Status (Waiting on Partial) confirmation is received, this state shall repeatedly send a Phy Status (Blocked Partial Pathway) response to the ECM.

If a Dword Received message is received containing a valid dword except a BREAK, then this state shall send a Forward Dword request to the ECR containing that dword.

If:

- a) an Invalid Dword Received message is received; and
- b) the expander logical phy is forwarding to an expander logical phy attached to a SAS logical link,

then the expander logical phy shall:

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

If:

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

then the expander logical phy shall:

- a) send a SATA\_ERROR with the Forward Dword request instead of the invalid dword or ERROR primitive; or
- b) delete the ERROR or invalid dword.

#### 6.19.6.2 Transition XL3:Open\_Confirm\_Wait to XL0:Idle

This transition shall occur after sending a Transmit OPEN\_REJECT message to the XL transmitter if:

- a) a BREAK Received message has not been received; and
- b) a BREAK Received argument was not included in the transition into this state.



**6.19.6.3 Transition XL3:Open\_Confirm\_Wait to XL1:Request\_Path**

This transition shall occur after receiving a Backoff Retry confirmation, after releasing path resources if:

- a) a BREAK Received message has not been received; and
- b) a BREAK Received argument was not included in the transition into this state.

This transition shall include an OPEN Address Frame Received argument containing the arguments in the received OPEN Address Frame Received argument.

**6.19.6.4 Transition XL3:Open\_Confirm\_Wait to XL5:Forward\_Open**

This transition shall occur after receiving a Backoff Reverse Path confirmation if:

- a) a BREAK Received message has not been received; and
- b) a BREAK Received argument was not included in the transition into this state.

The transition shall include:

- a) OPEN Address Frame Received arguments contained in the received Backoff Reverse Path arguments (i.e., the OPEN address frame).

**6.19.6.5 Transition XL3:Open\_Confirm\_Wait to XL7:Connected**

This transition shall occur after sending a Transmit OPEN\_ACCEPT message to the XL transmitter if:

- a) a BREAK Received message has not been received; and
- b) a BREAK Received argument was not included in the transition into this state.

**6.19.6.6 Transition XL3:Open\_Confirm\_Wait to XL9:Break**

This transition shall occur:

- a) after sending a Forward Break request to the ECR.

**6.19.6.7 Transition XL3:Open\_Confirm\_Wait to XL10:Break\_Wait**

This transition shall occur after receiving a Forward Break indication if:

- a) a BREAK Received message has not been received; and
- b) a BREAK Received argument was not included in the transition into this state.

**6.19.7 XL5:Forward\_Open state****6.19.7.1 State description**

This state is used to transmit an OPEN address frame passed with the transition into this state.

If a BROADCAST Received message is received, then this state shall send a Broadcast Event Notify request to the BPP with the argument indicating the specific BROADCAST primitive sequence received (e.g., Change Received).

Upon entry into this state, this state shall:

- a) send a Transmit OPEN Address Frame message to the XL transmitter with the following arguments set to the values specified in the ECR Forward Open argument included in the transition into this state:
  - A) Initiator Port Bit;
  - B) Protocol;
  - C) Features;
  - D) Connection Rate;
  - E) Initiator Connection Tag;

- F) Destination SAS Address;
- G) Source SAS Address;
- H) Source Zone Group;
- I) Pathway Blocked Count;
- J) Arbitration Wait Time;
- K) Compatible Features; and
- L) More Compatible Features;
- b) set the Open Source SAS Address state machine variable to the Source SAS Address argument; and
- c) set the Open Destination SAS Address state machine variable to the Destination SAS Address argument.

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

If a PS\_REQ Received message is received, then this state shall send a Transmit PS\_NAK to the XL transmitter.

#### **6.19.7.2 Transition XL5:Forward\_Open to XL6:Open\_Response\_Wait**

This transition shall occur:

- a) after receiving an OPEN Address Frame Transmitted message.

This transition shall include an ECR Forward Open argument containing the arguments received in the ECR Forward Open argument that was included in the transition into this state.

If an OPEN Address Frame Received message is received or an OPEN Address Frame Received argument is included in the transition into this state, then this transition shall include:

- a) an OPEN Address Frame Received argument.

If a BREAK Received message is received or a BREAK Received argument is included in the transition into this state, then this transition shall include:

- a) a BREAK Received argument.

If a Forward Break Indication is received, then this transition shall include:

- a) a Forward Break argument.

#### **6.19.8 XL6:Open\_Response\_Wait state**

##### **6.19.8.1 State description**

This state waits for a response to a transmitted OPEN address frame and determines the appropriate action to take based on the response.

This state shall either:

- a) request idle dwords be transmitted by repeatedly sending Transmit Idle Dword messages to the XL transmitter, honoring ALIGN insertion rules for rate matching and physical link rate tolerance management; or
- b) send Transmit Dword messages to the XL transmitter to transmit all dwords received with Forward Dword indications.

If a BROADCAST Received message is received before an AIP Received message is received, then this state shall send a Broadcast Event Notify request to the BPP with the argument indicating the specific BROADCAST primitive sequence received (e.g., Broadcast Event Notify (Change Received)).

This state shall send the following responses to the ECR, which are received by the source phy as confirmations:

- a) an Open Accept response when an OPEN\_ACCEPT Received message is received;

- b) an Open Reject response when an OPEN\_REJECT Received message is received, after releasing any path resources;
- c) a Backoff Retry response, after releasing path resources, when:
  - A) an AIP Received message has not been received;
  - B) an OPEN Address Frame Received message is received or an OPEN Address Frame Received argument is included in the transition into this state;
  - C) SMP frame priority is enabled (see 6.16.3);
  - D) the SAS Protocol argument in:
    - a) the ECR Forward Open argument is set to other than SMP; and
    - b) the OPEN Address Frame Received message set to SMP or an OPEN Address Frame Received SAS Protocol argument is set to SMP;

and

- E) the destination SAS address and connection rate contained in the OPEN Address Frame Received message or the OPEN Address Frame Received argument are not equal to the source SAS address and connection rate of the ECR Forward Open argument;
- d) a Backoff Retry response, after releasing path resources, when:
  - A) an AIP Received message has not been received;
  - B) an OPEN Address Frame Received message is received or an OPEN Address Frame Received argument is included in the transition into this state containing a higher priority OPEN address frame according to the arbitration fairness comparison (see 6.16.4) if:
    - a) SMP frame priority is disabled (see 6.16.3);
    - b) SMP frame priority is enabled and the SAS Protocol argument in:
      - A) the ECR Forward Open argument is set to other than SMP; and
      - B) the OPEN Address Frame Received message is set to other than SMP or an OPEN Address Frame Received argument is set to other than SMP;

or

- c) SMP frame priority is enabled and the SAS Protocol argument in:
  - A) the ECR Forward Open argument is set to SMP; and
  - B) the OPEN Address Frame Received message is set to SMP or an OPEN Address Frame Received argument is set to SMP;

and

- C) the destination SAS address and connection rate contained in the OPEN Address Frame Received message or the OPEN Address Frame Received argument are not equal to the source SAS address and connection rate of the ECR Forward Open argument;
- e) a Backoff Retry response, after releasing path resources, when:
  - A) an AIP Received message has been received;
  - B) an OPEN Address Frame Received message is received or an OPEN Address Frame Received argument is included in the transition into this state; and
  - C) the destination SAS address and connection rate contained in the OPEN Address Frame Received message or the OPEN Address Frame Received argument are not equal to the source SAS address and connection rate of the ECR Forward Open argument;
- f) a Backoff Reverse Path response when:
  - A) an AIP Received message has not been received;
  - B) an OPEN Address Frame Received message is received or an OPEN Address Frame Received argument is included in the transition into this state;
  - C) SMP frame priority is enabled (see 6.16.3);
  - D) the SAS Protocol argument in:
    - a) the ECR Forward Open argument is set to other than SMP;
    - b) the OPEN Address Frame Received message set to SMP or an OPEN Address Frame Received SAS Protocol argument is set to SMP;

and

- E) the destination SAS address and connection rate contained in the OPEN Address Frame Received message or the OPEN Address Frame Received argument are equal to the source SAS address and connection rate of the ECR Forward Open argument;
- g) a Backoff Reverse Path response when:
  - A) an AIP Received message has not been received;
  - B) an OPEN Address Frame Received message is received or an OPEN Address Frame Received argument is included in the transition into this state containing a higher priority OPEN address frame according to the arbitration fairness comparison (see 6.16.4); if:
    - a) SMP frame priority is disabled (see 6.16.3);
    - b) SMP frame priority is enabled and the SAS Protocol argument in:
      - A) the ECR Forward Open argument is set to other than SMP; and
      - B) the OPEN Address Frame Received message is set to other than SMP or an OPEN Address Frame Received argument is set to other than SMP;
  - or
  - c) SMP frame priority is enabled and the SAS Protocol argument in:
    - A) the ECR Forward Open argument is set to SMP; and
    - B) the OPEN Address Frame Received message is set to SMP or an OPEN Address Frame Received argument is set to SMP;
- and
- C) the destination SAS address and connection rate contained in the OPEN Address Frame Received message or the OPEN Address Frame Received argument are equal to the source SAS address and connection rate of the ECR Forward Open argument;
- and
- h) a Backoff Reverse Path response when:
  - A) an AIP Received message has been received;
  - B) an OPEN Address Frame Received message is received or an OPEN Address Frame Received argument is included in the transition into this state; and
  - C) the destination SAS address and connection rate contained in the OPEN Address Frame Received message or the OPEN Address Frame Received argument are equal to the source SAS address and connection rate of the ECR Forward Open argument.

A Backoff Reverse Path response shall include the contents of:

- a) the OPEN Address Frame Received message; or
- b) the OPEN Address Frame Received argument included in the transition into this state.

This state shall send the following responses to the ECR, which are received by the source phy as confirmations:

- a) an Arb Status (Waiting On Device) response upon entry into this state;
- b) an Arb Status (Normal) response when an AIP Received (Normal) message is received;
- c) an Arb Status (Waiting On Partial) response when an AIP Received (Waiting On Partial) message is received;
- d) an Arb Status (Waiting On Connection) response when an AIP Received (Waiting On Connection) message is received; and
- e) an Arb Status (Waiting On Device) response when an AIP Received (Waiting On Device) message is received.

If a BREAK Received message is received or a BREAK Received argument is included in the transition into this state, then this state shall send a Forward Break request to the ECR (see 6.19.8.6).

This state shall repeatedly send a Phy Status (Partial Pathway) response to the ECM until an AIP Received (Waiting On Partial) message is received. After an AIP Received (Waiting On Partial) message is received, this state shall repeatedly send a Phy Status (Blocked Partial Pathway) response to the ECM.

If a PS\_REQ Received message is received, then this state shall send a Transmit PS\_NAK to the XL transmitter.

**6.19.8.2 Transition XL6:Open\_Response\_Wait to XL0:Idle**

This transition shall occur:

- a) after sending an Open Reject response to the ECR.

**6.19.8.3 Transition XL6:Open\_Response\_Wait to XL1:Request\_Path**

This transition shall occur:

- a) after sending a Backoff Retry response to the ECR.

This transition shall include an OPEN Address Frame Received argument containing the arguments in:

- a) the received OPEN Address Frame Received message; or
- b) the OPEN Address Frame Received argument included in the transition into this state.

**6.19.8.4 Transition XL6:Open\_Response\_Wait to XL2:Request\_Open**

This transition shall occur:

- a) after sending a Backoff Reverse Path response to the ECR.

This transition shall include an OPEN Address Frame Received argument containing the arguments in:

- a) the received OPEN Address Frame Received message; or
- b) the OPEN Address Frame Received argument included in the transition into this state.

**6.19.8.5 Transition XL6:Open\_Response\_Wait to XL7:Connected**

This transition shall occur:

- a) after sending an Open Accept response to the ECR.

**6.19.8.6 Transition XL6:Open\_Response\_Wait to XL9:Break**

This transition shall occur:

- a) after sending a Forward Break response to the ECR.

**6.19.8.7 Transition XL6:Open\_Response\_Wait to XL10:Break\_Wait**

This transition shall occur if:

- a) a Forward Break argument was included in the transition into this state;
- b) a BREAK Received message has not been received; and
- c) a BREAK Received argument was not included in the transition into this state.

This transition shall occur after receiving a Forward Break indication if:

- a) a BREAK Received message has not been received; and
- b) a BREAK Received argument was not included in the transition into this state.

**6.19.9 XL7:Connected state****6.19.9.1 State description**

This state provides a full-duplex path between two phys within an expander device.

If the connection is an SSP connection, then upon entry into this state, this state shall initialize and start the Wait For Frame Timeout timer.

If the connection is an SSP connection, then this state shall initialize and start the SSP Maximum Connection Time Limit timer when:

- a) the Wait For Frame Timeout timer expires; or
- b) an SOF Received message is received.

This state shall send Transmit Dword messages to the XL transmitter to transmit all dwords received with Forward Dword indications. During an STP connection, the expander device may expand or contract a repeated or continued primitive sequence (see 6.2.5).

If this state has not sent a Forward Close request to the ECR and a Dword Received message is received containing a valid dword, except a BREAK or CLOSE, then this state shall send Forward Dword request containing a valid dword to the ECR. During an STP connection, the expander device may expand or contract a repeated or continued primitive sequence (see 6.2.5).

If:

- a) an Invalid Dword Received message is received; and
- b) the expander phy is forwarding to an expander logical phy attached to a SAS logical link,

then the expander logical phy shall:

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

If:

- a) an ERROR Received message 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 logical phy attached to a SATA phy,

then the expander logical phy shall:

- a) send a SATA\_ERROR with the Forward Dword request instead of the invalid dword or ERROR primitive; or
- b) delete the ERROR or invalid dword.

If:

- a) this state receives a Begin SSP Connection Close confirmation and:
  - A) the Wait For Frame Timeout timer has expired; or
  - B) an SOF Received message was received;
 or
- b) the SSP Maximum Connection Time Limit timer expires,

then this state shall send a Substitute To Close message to the XL transmitter.

If this state receives a Begin SSP Connection Close confirmation and:

- a) the Wait For Frame Timeout timer has not expired; and
- b) an SOF Received message has not been received,

then this state shall:

- 1) wait until:
  - A) the Wait For Frame Timeout timer expires; or
  - B) an SOF Received message is received;
 and
- 2) send a Substitute To Close message to the XL transmitter.

If a CLOSE Received message is received, then this state shall:

- 1) stop the SSP Maximum Connection Time Limit timer;
- 2) send a Stop Substituting message to the XL transmitter; and

- 3) send a Forward Close request to the ECR with the arguments from the CLOSE Received message (e.g., Type, Primitive Parameter, if any, and Source Phy Identifier, if any).

If a BREAK Received message is received, then this state shall:

- 1) stop the SSP Maximum Connection Time Limit timer;
- 2) send a Stop Substituting message to the XL transmitter; and
- 3) send a Forward Break request to the ECR (see 6.19.9.3).

This state shall repeatedly send a Phy Status (Connection) response to the ECM.

#### **6.19.9.2 Transition XL7:Connected to XL8:Close\_Wait**

This transition shall occur:

- a) after receiving a Forward Close indication if a BREAK Received message has not been received.

If this state has sent a Forward Close request to the ECR, then this transition shall include a Close Forwarded argument.

This transition shall include the following arguments from the Forward Close indication:

- a) Type (e.g., Normal or Clear Affiliation);
- b) Forwarded Close Primitive Parameter, if any; and
- c) Source Phy Identifier, if any.

#### **6.19.9.3 Transition XL7:Connected to XL9:Break**

This transition shall occur:

- a) after sending a Forward Break request to the ECR.

#### **6.19.9.4 Transition XL7:Connected to XL10:Break\_Wait**

This transition shall occur after receiving a Forward Break indication if a BREAK Received message has not been received.

This transition shall occur after:

- a) stopping the SSP Maximum Connection Time Limit timer; and
- b) sending a Stop Substituting message to the XL transmitter.

### **6.19.10 XL8:Close\_Wait state**

#### **6.19.10.1 State description**

This state closes a connection and releases path resources.

Upon entry into this state, this state shall:

- 1) if extended fairness priority is supported (i.e., EXTENDED FAIRNESS bit (see 9.4.3.4) is set to one), then:
  - 1) send a Request Fairness Priority request;
  - 2) wait for a Fairness Priority confirmation;
  - 3) select the CLOSE primitive parameter to send to the XL transmitter; and
  - 4) send a Transmit CLOSE message to the XL transmitter with the selected CLOSE primitive parameters;
- 2) if extended fairness priority is not supported, then:
  - 1) send a Transmit CLOSE message to the XL transmitter with the Type argument from the received Forward Close indication Type argument; and
  - 2) request that idle dwords be transmitted by repeatedly sending Transmit Idle Dword messages to the XL transmitter.

NOTE 40 - Possible livelock scenarios occur if the BREAK\_REPLY method of responding to received BREAK primitive sequences is disabled and a phy transmits a BREAK primitive sequence to break a connection (e.g., if its Close Timeout timer expires). Phys responding to a CLOSE primitive sequence faster than 1 ms reduce susceptibility to this problem.

If extended fairness priority is supported, then this state shall select the highest extended fairness priority between the Fairness Priority confirmation from the ECM and the Forwarded Close Primitive Parameter argument. The extended fairness priority (see table 194) consists of the following:

- a) High Priority argument;
- b) SMP Open Priority argument;
- c) Arbitration Wait Time argument; and
- d) Connection Rate argument.

**Table 194 – Extended fairness priority**

<b>Bit 21 (21 is MSB)</b>	<b>Bit 20</b>	<b>Bits 19 to 4</b>	<b>Bits 3 to 0 (0 is LSB)</b>
High Priority argument	SMP Open Priority argument	Arbitration Wait Time argument	Connection Rate argument



If SAS packet mode is enabled and extended fairness priority is supported, then this state shall set the CLOSE primitive parameters and the Delay Expander Forward Open Indication Timer argument as described in table 195.

**Table 195 – Setting CLOSE primitive parameters and Delay Expander Forward Open Indication timer**

Forwarded Close has Primitive Parameter arguments	Fairness Priority confirmation has Primitive Parameter arguments	Open Destination SAS Address from Forwarded Close Primitive Parameter equals			State action
		Open Destination SAS Address in <sup>a</sup>	Open Source SAS Address in <sup>b</sup>	Open Destination SAS Address in <sup>c</sup>	
yes	yes	yes	x	x	Select the highest extended fairness priority between Forwarded Close Primitive Parameter arguments and Fairness Priority confirmation Primitive Parameter arguments. If the selected highest fairness priority is associated with: a) this expander device, then see <sup>d</sup> ; or b) another expander device, then see <sup>e</sup> .
		x	yes		
		x	x	yes	
		no	no	no	See <sup>d</sup>
no	yes	x	x	x	See <sup>e</sup>
yes	no	x	yes	x	
		x	x	yes	
no	no	x	no	no	Not use any CLOSE primitive parameter arguments on the Transmit CLOSE message and set the Delay Expander Forward Open Indication Timer argument to zero.
		x	x	x	

**Key:**  
x = don't care

<sup>a</sup> The Open Destination SAS Address argument from the Fairness Priority confirmation.  
<sup>b</sup> The Open Source SAS Address state machine variable.  
<sup>c</sup> The Open Destination SAS Address state machine variable.  
<sup>d</sup> Set the CLOSE primitive parameters to the contents of the arguments of the Fairness Priority confirmation from the ECM, set the HOP COUNT field to one, and set the Delay Expander Forward Open Indication Timer argument to zero.  
<sup>e</sup> Set the CLOSE primitive parameter to the contents of the Forwarded Close Primitive Parameter argument, set the HOP COUNT field to the current hop count plus one, and set the Delay Expander Forward Open Indication Timer argument to the contents of the INITIAL TIME TO DELAY EXPANDER FORWARD OPEN INDICATION field (see 9.4.3.4) multiplied by the Forwarded Close Primitive Parameter Hop Count argument.

If SAS dword mode is enabled or extended fairness priority is not supported, then this state shall:

- a) not use any CLOSE primitive parameter arguments on the Transmit CLOSE message; and
- b) set the Delay Expander Forward Open Indication Timer argument to zero.

If a Dword Received message is received containing a valid dword except a BREAK or CLOSE, then this state shall send a Forward Dword request to the ECR containing that dword. During an STP connection, the expander device may expand or contract a repeated or continued primitive sequence (see 6.2.5).

If:

- a) an Invalid Dword Received message is received; and
- b) the expander logical phy is forwarding to an expander logical phy attached to a SAS logical link,

then the expander logical phy shall:

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

If:

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

then the expander logical phy shall:

- a) send a SATA\_ERROR with the Forward Dword request instead of the invalid dword or ERROR primitive; or
- b) delete the ERROR or invalid dword.

If a CLOSE Received message is received, then this state shall release path resources and send a Forward Close request to the ECR with the argument from the CLOSE Received message (see 6.19.10.2).

If this state was entered with an argument of Close Forwarded, then this state shall release path resources after all the CLOSE primitive parameters, if any, have been received.

If a BREAK Received message is received, then this state shall send a Forward Break request to the ECR (see 6.19.10.3).

This state shall repeatedly send a Phy Status (Connection) response to the ECM.

#### **6.19.10.2 Transition XL8:Close\_Wait to XL0:Idle**

This transition shall occur after:

- a) sending a Forward Close request to the ECR; or
- b) sending a Transmit CLOSE message to the XL transmitter if this state was entered with an argument of Close Forwarded.

This transition shall include:

- a) the Delay Expander Forward Open Indication Timer argument; and
- b) the Source Phy Identifier argument.

#### **6.19.10.3 Transition XL8:Close\_Wait to XL9:Break**

This transition shall occur:

- a) after sending a Forward Break request to the ECR.

#### **6.19.10.4 Transition XL8:Close\_Wait to XL10:Break\_Wait**

This transition shall occur:

- a) after receiving a Forward Break indication if a BREAK Received message has not been received.

### **6.19.11 XL9:Break state**

#### **6.19.11.1 State description**

This state closes the connection, if there is one, and releases all path resources associated with the connection.

This state shall:

- a) send a Transmit BREAK\_REPLY message to the XL transmitter if the BREAK\_REPLY method of responding to received BREAK primitive sequences is enabled (see 6.16.6); and
- b) send a Transmit BREAK message to the XL transmitter if the BREAK\_REPLY method of responding to received BREAK primitive sequences is disabled (see 6.16.6).

#### **6.19.11.2 Transition XL9:Break to XL0:Idle**

This transition shall occur:

- a) after sending a Transmit BREAK message or a Transmit BREAK\_REPLY message to the XL transmitter.

#### **6.19.12 XL10:Break\_Wait state**

##### **6.19.12.1 State description**

This state closes the connection, if there is one, and releases path resources associated with the connection.

Upon entry into this state, this state shall:

- 1) send a Transmit BREAK message to the XL transmitter;
- 2) initialize and start the Break Timeout timer; and
- 3) repeatedly send a Phy Status (Breaking Connection) response to the ECM.

If a BREAK Received message is received and the BREAK\_REPLY method of responding to received BREAK primitive sequences is enabled (see 6.16.6), then this state shall send a Transmit BREAK\_REPLY message to the XL transmitter.

##### **6.19.12.2 Transition XL10:Break\_Wait to XL0:Idle**

This transition shall occur after:

- a) a BREAK\_REPLY Received message is received if the BREAK\_REPLY method of responding to received BREAK primitive sequences is enabled (see 6.16.6);
- b) a BREAK Received message is received if the BREAK\_REPLY method of responding to received BREAK primitive sequences is disabled (see 6.16.6); or
- c) the Break Timeout timer expires.

#### **6.19.13 XL11:PS\_Request state**

##### **6.19.13.1 State description**

This state requests the attached phy change to a low phy power condition.

Upon entry into this state, this state shall:

- a) if entered with the Phy Power Condition (Enter Partial) argument, then send a Transmit PS\_REQ (Partial) message to the XL transmitter;
- b) if entered with the Phy Power Condition (Enter Slumber) argument, then send a Transmit PS\_REQ (Slumber) message to the XL transmitter; and
- c) initialize and start the Power Condition Request Timeout timer.

If this state:

- a) is entered with the Phy Power Condition (Enter Partial) argument; and
- b) receives a PS\_REQ Received (Slumber) message,

then this state shall set the Phy Power Condition argument to Partial.

If this state:

- a) is entered with the Phy Power Condition (Enter Slumber) argument; and
- b) receives a PS\_REQ Received (Partial) message,

then this state shall:

- a) set the Phy Power Condition argument to Partial; and
- b) if the Accept Partial state machine variable is set to:
  - A) YES, then send a Transmit PS\_ACK Pattern message to the XL transmitter; or
  - B) NO, then send a Transmit PS\_NAK message to the XL transmitter.

If this state:

- a) receives a PS\_REQ Received message;
- b) the PS\_REQ Received message argument is requesting the same power condition as the Phy Power Condition argument; and
- c) the attached SAS address is greater than the port identifier (i.e., SAS Address),

then this state shall send:

- a) a Transmit PS\_ACK Pattern message to the XL transmitter;
- b) a Manage Phy Power Conditions (Stop DWS) request to the phy layer; and
- c) a Manage Phy Power Conditions (Stop PS) request to the phy layer.

If this state receives a Forward Open indication, then this state shall send a Backoff Retry response to the ECR.

If a BROADCAST Received message is received, then this state shall send a Broadcast Event Notify request to the BPP with the argument indicating the specific BROADCAST primitive sequence received (e.g., Change Received).

If a Transmit Broadcast indication is received, then this state shall send a Retry Transmit Broadcast request to the BPP.

#### **6.19.13.2 Transition XL11:PS\_Request to XL12:PS\_Quiet**

This transition shall occur after receiving:

- a) a PS\_ACK Pattern Transmitted message from the XL transmitter; or
- b) a PS\_ACK Received message.

This transition shall include:

- a) the Phy Power Condition argument.

#### **6.19.13.3 Transition XL11:PS\_Request to XL0:Idle**

This transition shall occur after:

- a) sending a Transmit PS\_NAK message to the XL transmitter;
- b) a PS\_NAK Received message is received; or
- c) the Power Condition Request Timeout timer expires.

#### **6.19.13.4 Transition XL11:PS\_Request to XL1:Request\_Path**

This transition shall occur:

- a) after receiving an OPEN Address Frame Received message.

This transition shall include:

- a) an OPEN Address Frame Received argument containing the arguments in the received OPEN Address Frame Received message.

**6.19.13.5 Transition XL11:PS\_Request to XL9:Break**

This transition shall occur:

- a) after receiving a BREAK Received message.

**6.19.14 XL12:PS\_Quiet state****6.19.14.1 State description**

This state requests the phy be placed into a low phy power condition.

If this state is entered with a Phy Power Condition (Enter Partial) argument, then this state shall send:

- a) a Manage Phy Power Conditions (Enter Partial) request to the phy layer; and
- b) a Phy Power Condition Status (In Partial) confirmation to the ECM.

If this state is entered with a Phy Power Condition (Enter Slumber) argument, then this state shall send:

- a) a Manage Phy Power Conditions (Enter Slumber) request to the phy layer; and
- b) a Phy Power Condition Status (In Slumber) confirmation to the ECM.

If this state receives a:

- a) Change Phy Power Condition (Enter Slumber) request and the current phy power condition is partial;  
or
- b) Change Phy Power Condition (Enter Partial) request and the current phy power condition is slumber,

then this state shall send a Phy Power Condition Status (Request Exit Power Condition) confirmation to the ECM.

If a Transmit Broadcast indication is received, then this state shall send a Retry Transmit Broadcast request to the BPP.

If this state receives a Change Phy Power Condition (Exit Power Condition) request, then this state shall send a Manage Phy Power Conditions (Exit) request to the phy layer.

If this state receives a Forward Open indication from the ECM, then this state shall send a Manage Phy Power Conditions (Exit) request to the phy layer and if the phy is in:

- a) a slumber phy power condition, send an Open Reject (Retry) response to the ECR; or
- b) a partial phy power condition, send a Backoff Retry response to the ECR.

**6.19.14.2 Transition XL12:PS\_Quiet to XL0:Idle**

This transition shall occur:

- a) after a Phy Layer Ready (SAS) confirmation is received.

**6.20 SSP link layer****6.20.1 Opening an SSP connection**

An SSP phy that accepts a connection request (i.e., an OPEN address frame) shall:

- a) transmit at least one RRDY in that connection within 1 ms of transmitting an OPEN\_ACCEPT; and
- b) if persistent connections are supported (see 4.1.13.2), the SEND\_EXTEND bit is set to one in the received OPEN address frame, and the INITIATOR\_PORT bit is set to one in the received OPEN address frame, then transmit an EXTEND\_CONNECTION (NORMAL).

If the SSP phy is not able to grant SSP frame credit, then it shall respond with OPEN\_REJECT (RETRY) and not accept the connection request.

To prevent livelocks (e.g., where ports are waiting on each other to accept a connection request):

- a) a SAS phy shall not reject an incoming connection request to an SSP initiator port with OPEN\_REJECT (RETRY) as a result of the SAS port containing that SAS phy is waiting for an outgoing connection request to be accepted (e.g., if the SAS phy is used by an SSP initiator port and an SSP target port, they share a buffer, that buffer is being used by the SSP target port, and the SSP target port is waiting to transmit a frame to another SSP initiator port before it is able to free that buffer);
- b) a SAS phy may reject an incoming connection request to an SSP initiator port with OPEN\_REJECT (RETRY) for any reason that is not dependent on the SAS port containing that SAS phy having an outgoing connection request accepted (e.g., a temporary buffer full condition); and
- c) a SAS phy may reject an incoming connection request to an SSP target port with OPEN\_REJECT (RETRY) for any reason, including because the SAS port containing that SAS phy is waiting for an outgoing connection request to be accepted (e.g., to transmit a frame and empty a buffer).

### 6.20.2 Full duplex

SSP is a full duplex protocol. An SSP phy may receive an SSP frame or primitive in a connection while the SSP phy is transmitting an SSP frame or primitive in the same connection. A wide SSP port may send and/or receive SSP frames or primitives concurrently on different connections (i.e., on different phys).

If:

- a) a connection is open;
- b) a persistent connection has:
  - A) not been requested (i.e., the SEND\_EXTEND bit was set to zero in the OPEN address frame); or
  - B) been:
    - 1) requested (i.e., the SEND\_EXTEND bit was set to one in the OPEN address frame);
    - 2) established (see 4.1.13); and
    - 3) disabled (see 6.20.9.12.4);
- and
- c) an SSP phy has no more SSP frames to transmit on that connection,

then the SSP phy transmits a DONE to start closing the connection (see 7.2.2.3.9). The other direction may still be active, so the DONE may be followed by one or more of the following primitives:

- a) CREDIT\_BLOCKED;
- b) RRDY;
- c) ACK; or
- d) NAK.

### 6.20.3 SSP frame transmission and reception

#### 6.20.3.1 SSP frame transmission and reception overview

During an SSP connection the receiving SSP phy shall acknowledge SSP frames within 1 ms, if not discarded as described in 6.20.9.7, with either:

- a) ACK (i.e., positive acknowledgement) if the SSP frame was received into a frame buffer without errors; or
- b) NAK (CRC ERROR) (i.e., negative acknowledgement) if the SSP frame was received with a CRC error (i.e., a bad CRC), an invalid dword, or an ERROR.

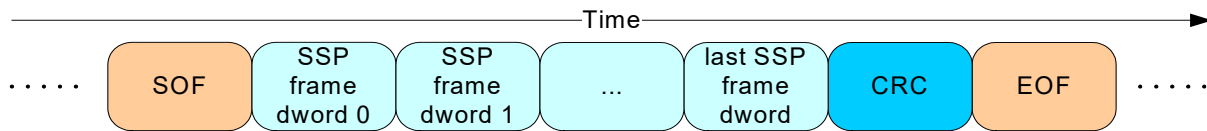
NOTE 41 - It is not required that frame recipients generate NAK (CRC ERROR) from invalid dwords and ERRORS (see 6.20.9.2).

If an SSP frame results in a link layer error (e.g., is NAKed or creates an ACK/NAK timeout), then:

- a) the transport layer (see 8.2.4) retries sending the SSP frame; or
- b) the SCSI application layer aborts the SCSI command associated with that SSP frame.

### 6.20.3.2 SSP frame transmission and reception while in the SAS dword mode

During an SSP connection, while in the SAS dword mode, SSP frames are preceded by SOF and followed by EOF as shown in figure 164.

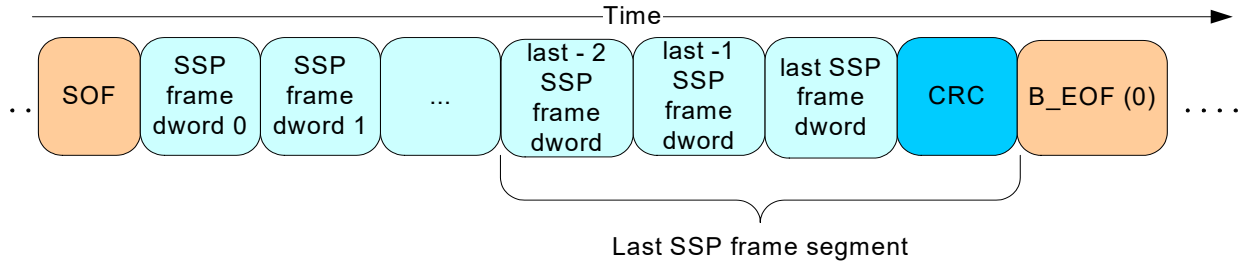


**Figure 164 –SSP frame transmission**

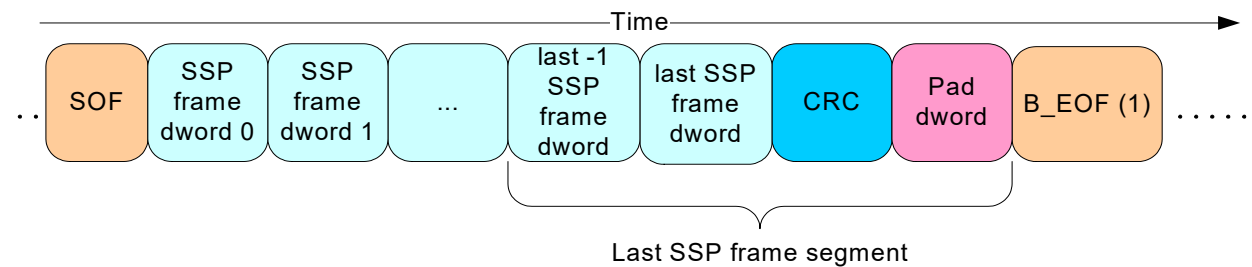
The last data dword after the SOF prior to the EOF always contains a CRC (see 6.7). The SSP link layer state machine checks that the frame is a valid length and that the CRC is valid (see 6.20.9.7). Other primitives (e.g., CREDIT\_BLOCKED, RRDY, ACK, and NAK) may be interspersed between the SOF, data dwords, and EOF.

### 6.20.3.3 SSP frame transmission and reception while in SAS packet mode

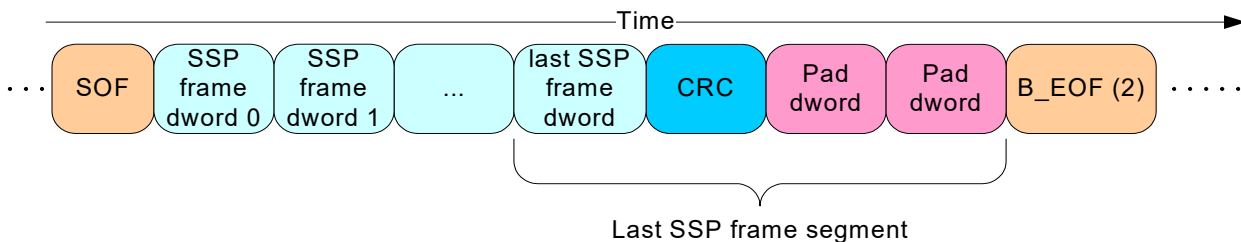
During an SSP connection while in SAS packet mode, SSP frames are preceded by SOF and followed by B\_EOF (0), B\_EOF (1), B\_EOF (2), or B\_EOF (3) as shown in figure 165, figure 166, figure 167, and figure 168.



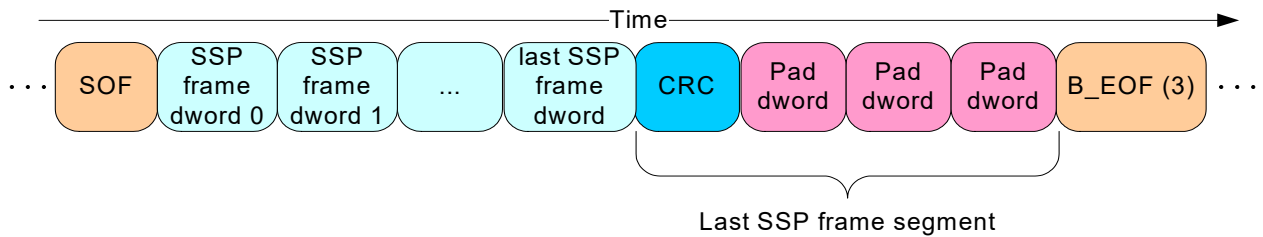
**Figure 165 –SSP frame transmission with no pad dword**



**Figure 166 –SSP frame transmission with one pad dword**



**Figure 167 –SSP frame transmission with two pad dwords**



**Figure 168 –SSP frame transmission with three pad dwords**



The SSP frame transmission and reception while in SAS packet mode requirements for placement of the CRC, pad dwords, B\_EOF (0), B\_EOF (1), B\_EOF (2), and B\_EOF (3) are as described in 6.20.3.4.

#### 6.20.3.4 SSP frame and SMP frame transmission and reception while in SAS packet mode

The last data dword after the SOF, before the pad dwords, if any, and before the B\_EOF (0), the B\_EOF (1), the B\_EOF (2), or the B\_EOF (3) shall contain a CRC (see 6.7). The position of the CRC in the last SPL frame segment shall be determined by the first B\_EOF (0), the B\_EOF (1), the B\_EOF (2), or the B\_EOF (3) following the last SPL frame segment as follows:

- B\_EOF (0) specifies the CRC is the fourth dword of the last SPL frame segment (i.e., there are no pad dwords) (see figure 165);
- B\_EOF (1) specifies the CRC is the third dword of the last SPL frame segment (i.e., there is one pad dword) (see figure 166);
- B\_EOF (2) specifies the CRC is the second dword of the last SPL frame segment (i.e., there are two pad dwords) (see figure 167); and
- B\_EOF (3) specifies the CRC is the first dword of the last SPL frame segment (i.e., there are three pad dwords) (see figure 168).

If a B\_EOF is received on a SAS physical link that is in the SAS packet mode and that B\_EOF is to be forwarded to a SAS physical link that is in:

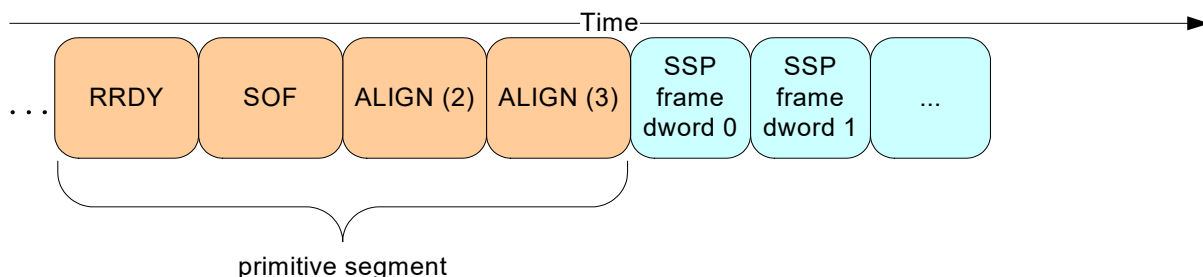
- the SAS packet mode, then the expander device shall not make any changes to the SSP frame's B\_EOF; or
- the SAS dword mode, then the expander device shall replace that B\_EOF with an EOF positioned in the next dword after the CRC (see 6.7) before forwarding.

If an EOF is received on a SAS physical link that is in the SAS dword mode and that EOF is to be forwarded to a SAS physical link that is in:

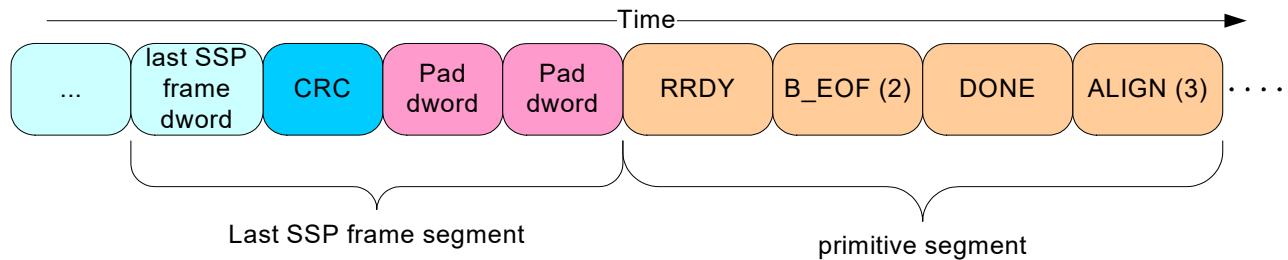
- the SAS dword mode, then the expander device shall not make any changes to the SSP frame's EOF; or
- the SAS packet mode, then the expander device shall replace that EOF with a B\_EOF (0), B\_EOF (1), B\_EOF (2), or B\_EOF (3) after the pad dwords, if any, as described in this subclause.

The SSP link layer state machine checks that the frame is a valid length and that the CRC is valid (see 6.20.9.7). Other primitives (e.g., CREDIT\_BLOCKED, RRDY, ACK, and NAK), binary primitives, or extended binary primitives may be interspersed between:

- the SOF (see figure 169);
- the SSP frame segments or SMP frame segments; and
- the B\_EOF (0), B\_EOF (1), B\_EOF (2), or B\_EOF (3) (see figure 170).



**Figure 169 –Unaligned SOF at start of SSP frame example**



**Figure 170 –Unaligned B\_EOF (2) at end of SSP frame example**

#### 6.20.4 SSP flow control

An SSP phy uses the CREDIT ADVANCE bit in the OPEN address frame and RRDY to grant SSP frame credit for permission for the other SSP phy in the connection to transmit frames.

If credit advance is implemented (see 4.1.14) and an OPEN address frame with the CREDIT ADVANCE bit set to one is received, then:

- 1) at the beginning of each connection a transmit SSP frame credit of zero is established;
- 2) the transmit SSP frame credit is incremented by one (see 6.20.9.4);
- 3) the first RRDY received does not increment transmit SSP frame credit (see 6.20.9.4); and
- 4) each subsequent RRDY increments transmit SSP frame credit by one frame.

If credit advance is not implemented (see 4.1.14) or an OPEN address frame with the CREDIT ADVANCE bit set to zero is received, then:

- a) at the beginning of each connection a transmit SSP frame credit of zero is established; and
- b) each RRDY increments transmit SSP frame credit by one frame.

Frame transmission decrements transmit SSP frame credit by one frame.

SSP phys shall not grant more than 255 SSP frame credits (i.e., outstanding received SSP frame credits shall not exceed 255).

To prevent deadlocks where an SSP initiator port and SSP target port are both waiting on each other to provide SSP frame credit, an SSP initiator port shall not refuse to provide SSP frame credit by withholding RRDY because the SSP initiator port is waiting to transmit a frame. An SSP initiator port may refuse to provide SSP frame credit for other reasons (e.g., temporary buffer full conditions).

An SSP target port may refuse to provide SSP frame credit for any reason, including because the SSP target port is waiting to transmit a frame.

If SSP frame credit is zero and a persistent connection:

- a) has not been established;
- b) was established and has been disabled (see 6.20.9.12.4); or
- c) has been established and the Persistent Connection Timeout timer (see table 198) has expired,

then SSP phys that are going to be unable to provide SSP frame credit for 1 ms may send CREDIT\_BLOCKED. The other phy may use this to avoid waiting 1 ms to transmit DONE (CREDIT TIMEOUT) (see 6.20.9).

If SSP frame credit is nonzero and a persistent connection:

- a) has not been established;
- b) was established and has been disabled (see 6.20.9.12.4); or
- c) has been established and the Persistent Connection Timeout timer (see table 198) has expired,

then SSP phys that are going to be unable to provide additional SSP frame credit for 1 ms, even if they receive frames per the existing SSP frame credit, may transmit CREDIT\_BLOCKED.

After sending CREDIT\_BLOCKED, an SSP phy shall not transmit any additional RRDYs in the connection.

### 6.20.5 Extending an SSP connection with persistent connections

If a persistent connection has been established (see 4.1.13), then to maintain the persistent connection an SSP phy shall transmit an EXTEND\_CONNECTION within the interval specified by the Transmit Extend Connection timer (see table 198) while the SSP phy has no SSP frame to transmit.

An EXTEND\_CONNECTION is not transmitted within an SSP frame.

The management application layer initiates the closing of a persistent connection by sending a Close Persistent Connection request for the SSP\_EM state machine (see 6.20.9.12).

### 6.20.6 Interlocked frames

Table 196 shows which SSP frames shall be interlocked and which are non-interlocked.

**Table 196 – SSP frame interlock requirements**

SSP frame type	Interlock requirement
COMMAND	Interlocked
TASK	Interlocked
XFER_RDY	Interlocked
DATA	Non-interlocked
RESPONSE	Interlocked
Note - See 8.2 for SSP frame type definitions.	

Before transmitting an interlocked frame, an SSP phy shall wait for all SSP frames to be acknowledged with ACK or NAK, even if transmit SSP frame credit is available. After transmitting an interlocked frame, an SSP phy shall not transmit another SSP frame until that interlocked frame has been acknowledged with ACK or NAK, even if transmit SSP frame credit is available.

Before transmitting a non-interlocked frame, an SSP phy shall wait for:

- a) all non-interlocked frames with different initiator port transfer tags; and
- b) all interlocked frames,

to be acknowledged with ACK or NAK, even if transmit SSP frame credit is available.

After transmitting a non-interlocked frame, an SSP phy may transmit another non-interlocked frame with the same initiator port transfer tag if transmit SSP frame credit is available. The phy shall not transmit:

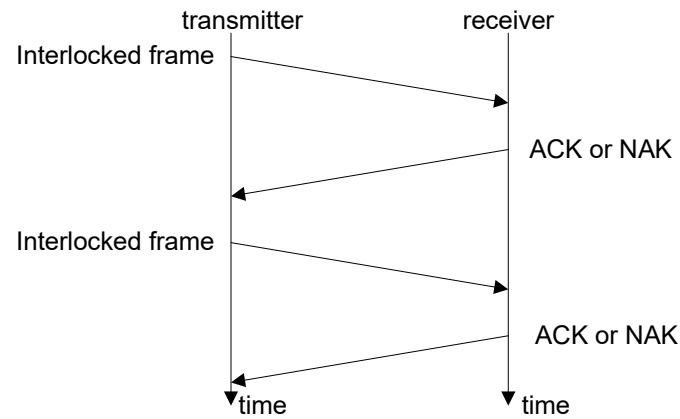
- a) a non-interlocked frame with a different initiator port transfer tag; or
- b) an interlocked frame,

until all SSP frames have been acknowledged with ACK or NAK, even if transmit SSP frame credit is available.

Transmitting an interlocked frame does not prevent the transmitting phy from receiving frames at the same time (e.g., it is possible for an SSP initiator phy to transmit a COMMAND frame while receiving XFER\_RDY, DATA, or RESPONSE frames).

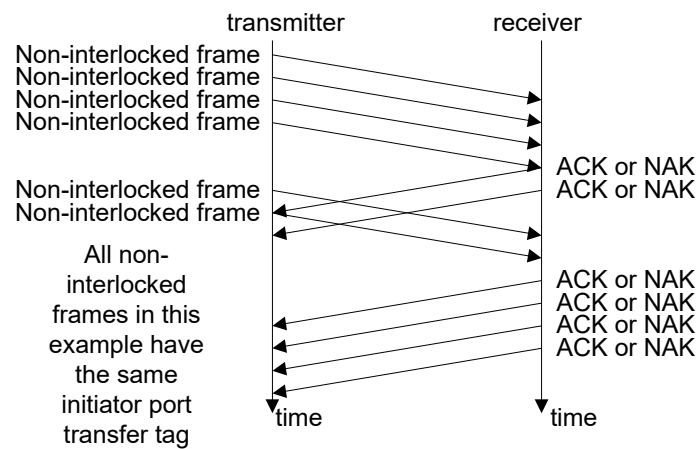
An SSP phy may transmit primitives responding to traffic that it is receiving (e.g., an ACK or NAK to acknowledge an SSP frame, an RRDY to grant more transmit SSP frame credit, or a CREDIT\_BLOCKED to specify that no more RRDYs are going to be transmitted in the connection) while waiting for an interlocked frame that it transmitted to be acknowledged. These primitives may also be interspersed within an SSP frame.

Figure 171 shows an example of interlocked frame transmission.



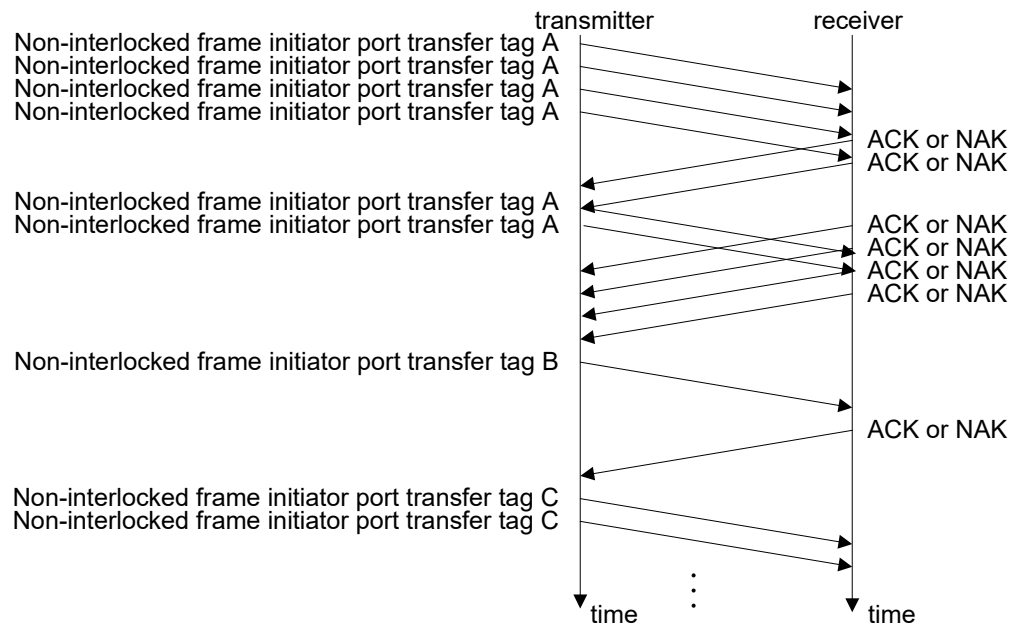
**Figure 171 –Interlocked frames**

Figure 172 shows an example of non-interlocked frame transmission with the same initiator port transfer tags.



**Figure 172 –Non-interlocked frames with the same initiator port transfer tags**

Figure 173 shows an example of non-interlocked frame transmission with different initiator port transfer tags.



**Figure 173 –Non-interlocked frames with different initiator port transfer tags**

#### 6.20.7 Breaking an SSP connection

In addition to the actions described in 6.16.11, the following shall be the responses by an SSP phy to a broken connection:

- received frames having no CRC error may be considered valid regardless of whether an ACK has been transmitted in response to the frame prior to the broken connection;
- transmitted frames for which an ACK has been received prior to a broken connection shall be considered transmitted without error; and
- transmitted frames for which an ACK or NAK has not been received prior to a broken connection shall be considered to have been transmitted with an error.

#### 6.20.8 Closing an SSP connection

DONE shall be exchanged prior to closing an SSP connection (see 7.2.2.3.9). The type of DONE indicates additional information about why the SSP connection is being closed as follows:

- DONE (NORMAL) specifies that the transmitter has no more SSP frames to transmit (i.e., normal completion);
- DONE (CLOSE) specifies that an end device has received an RRDY (CLOSE) (i.e., forced normal completion);
- DONE (CREDIT TIMEOUT) specifies that the transmitter still has SSP frames to transmit but did not receive an RRDY granting transmit SSP frame credit within 1 ms or the transmitter has received a CREDIT\_BLOCKED and has consumed all RRDYs received; and
- DONE (ACK/NAK TIMEOUT) specifies that 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 primitive sequence in 1 ms unless the recipient replies with DONE and the connection is closed.

A DONE exchange may be requested as follows:

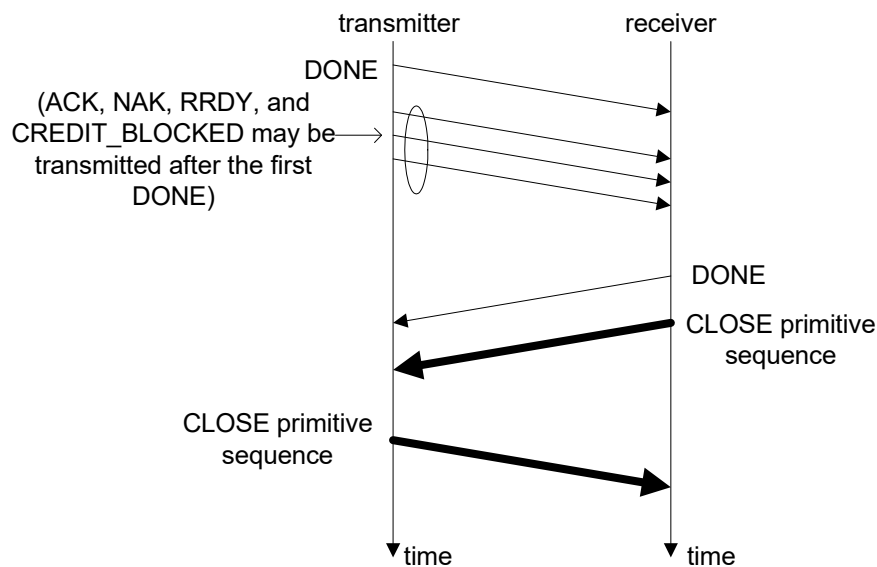
- a) RRDY (CLOSE) specifies that an expander device or an end device has requested that the SSP connection or the SSP persistent connection be closed (i.e., requested forced normal completion); and
- b) EXTEND\_CONNECTION (CLOSE) specifies that an expander device has requested that the SSP persistent connection be closed (i.e., requested forced normal completion).

If the transmitter has no more SSP frames to transmit and receives a CREDIT\_BLOCKED, then it may transmit either DONE (NORMAL) or DONE (CREDIT TIMEOUT).

After transmitting DONE, the transmitting phy initializes and starts a 1 ms DONE Timeout timer (see 6.20.9.5).

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 as needed 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 a CLOSE (NORMAL) primitive sequence (see 6.16.9).

Figure 174 shows the sequence for a closing an SSP connection.



**Figure 174 –Closing an SSP connection example**

A Force Completion request sent to the SSP link layer (see 6.20.9.10) by a management application layer in an end device requests an RRDY (CLOSE) be substituted (i.e., forced normal completion of a connection). After a Force Completion request is received by the SSP link layer, the SSP link layer substitutes RRDY (CLOSE) when granting more transmit SSP frame credit instead of transmitting RRDY (NORMAL) until the connection is closed.

By sending a Begin SSP Connection Close confirmation to the XL link layer (see 6.19.9) a management application layer in an expander device may request:

- a) an RRDY (CLOSE) be substituted for an RRDY (NORMAL); and
- b) an EXTEND\_CONNECTION (CLOSE) be substituted for an EXTEND\_CONNECTION (NORMAL).

In response to receiving an RRDY (CLOSE) or an EXTEND\_CONNECTION (CLOSE) the SSP link layer requests the transmission of a DONE (CLOSE). The transmission of a DONE (CLOSE) occurs if:

- a) there are no SSP frames waiting to be transmitted by the SSP link layer (see 6.20.9.6.3);
- b) there are no SSP frames being transmitted by the SSP link layer (see 6.20.9.6.4); and
- c) the ACK/NAK count is balanced (see 6.20.9.3).

## 6.20.9 SSP (link layer for SSP phys) state machines

### 6.20.9.1 SSP state machines overview

The SSP link layer contains several state machines that run in parallel to control the flow of dwords on the physical link during an SSP connection. The SSP state machines are as follows:

- a) SSP\_TIM (transmit interlocked frame monitor) state machine (see 6.20.9.3);
- b) SSP\_TCM (transmit frame credit monitor) state machine (see 6.20.9.4);
- c) SSP\_D (DONE control) state machine (see 6.20.9.5);
- d) SSP\_TF (transmit frame control) state machine (see 6.20.9.6);
- e) SSP\_RF (receive frame control) state machine (see 6.20.9.7);
- f) SSP\_RCM (receive frame credit monitor) state machine (see 6.20.9.8);
- g) SSP\_RIM (receive interlocked frame monitor) state machine (see 6.20.9.9);
- h) SSP\_TC (transmit credit control) state machine (see 6.20.9.10);
- i) SSP\_TAN (transmit ACK/NAK control) state machine (see 6.20.9.11); and
- j) SSP\_EM (establish and manage persistent connection) state machine (see 6.20.9.12).

All the SSP state machines, except SSP\_EM, shall start after receiving an Enable Disable SSP (Enable) message from the SL state machines (see 6.18).

If persistent connections are supported (see 4.1.13.2), then the SSP\_EM state machine shall start after receiving an Enable Disable SSP (Enable) message from the SL state machines.

All the SSP state machines shall terminate after:

- a) receiving an Enable Disable SSP (Disable) message from the SL state machines;
- b) receiving a Request Close message from the SSP\_D state machine indicating that the connection has been closed;
- c) receiving a Request Break message from the SSP\_D state machine indicating that a BREAK primitive sequence has been transmitted; or
- d) receiving a NOTIFY Received (Power Loss Expected) message from the SP\_DWS receiver or the SP\_PS receiver (see 5.16.2) if the SAS port that contains this state machine supports NOTIFY (Power Loss Expected) (e.g., the SAS port is an SSP target port).

If a state machine consists of multiple states, then the initial state is as indicated in the state machine description in this subclause.

The SSP state machines shall maintain the timers listed in table 197.

**Table 197 – SSP state machines timers**

Timer	Initial value	State machine	Reference
ACK/NAK Timeout timer	1 ms	SSP_TIM	6.20.9.3
DONE Timeout timer	1 ms	SSP_D	6.20.9.5
Credit Timeout timer	1 ms	SSP_TF	6.20.9.6

If persistent connections are supported (see 4.1.13.2), then the SSP state machines shall maintain the timers listed in table 198.

**Table 198 – SSP state machines timers for persistent connections**

Timer	Initial value	State machine	Reference
Persistent Connection Timeout timer	1 ms	SSP_EM	6.20.9.12
Transmit Extend Connection timer	100 $\mu$ s	SSP_EM	6.20.9.12

Figure 175 shows the SSP state machines and states related to frame transmission.

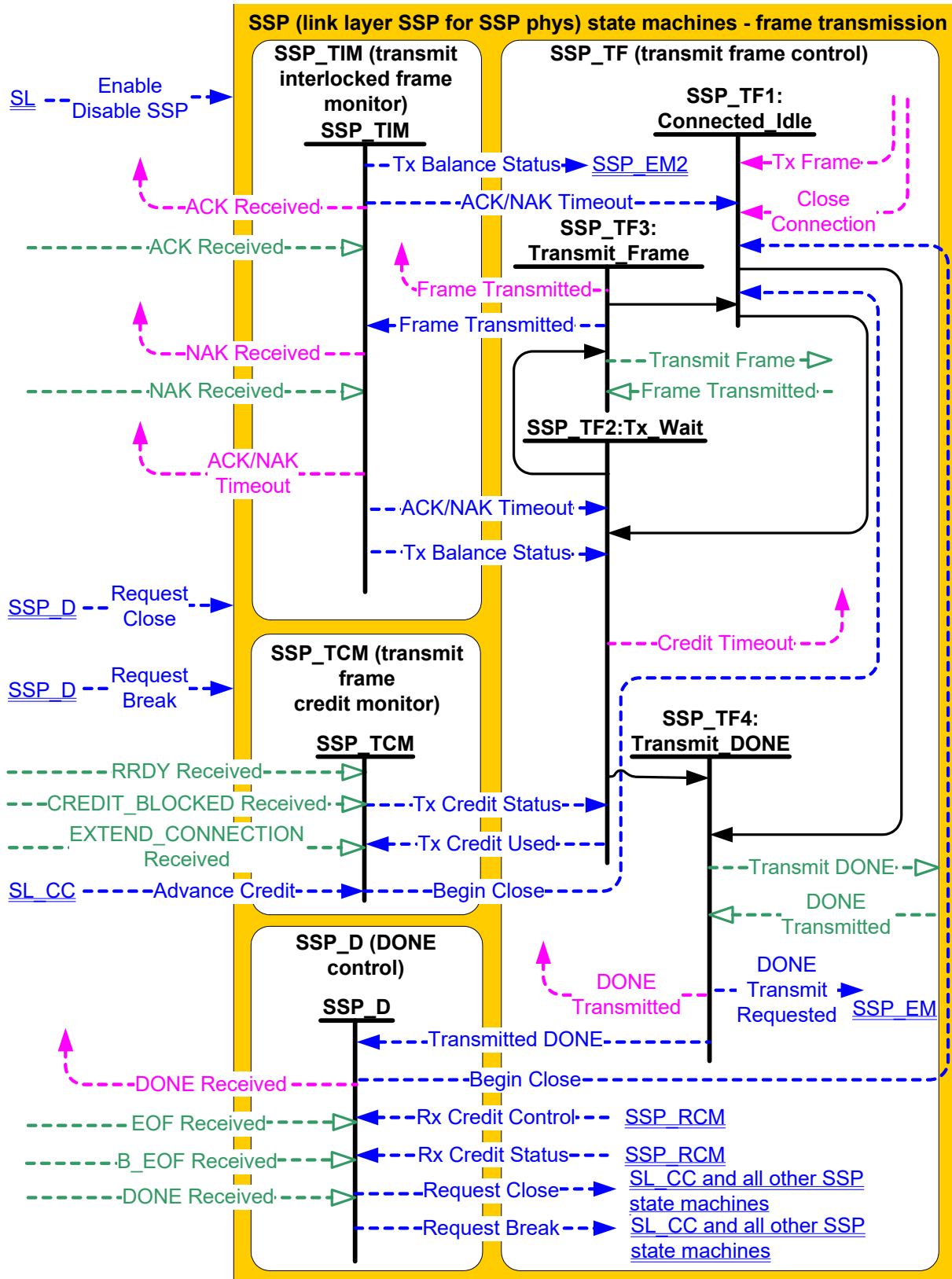


Figure 175 –SSP (link layer for SSP phys) state machines (1 of 3 - frame transmission)



Figure 176 shows the SSP state machines and states related to frame reception.

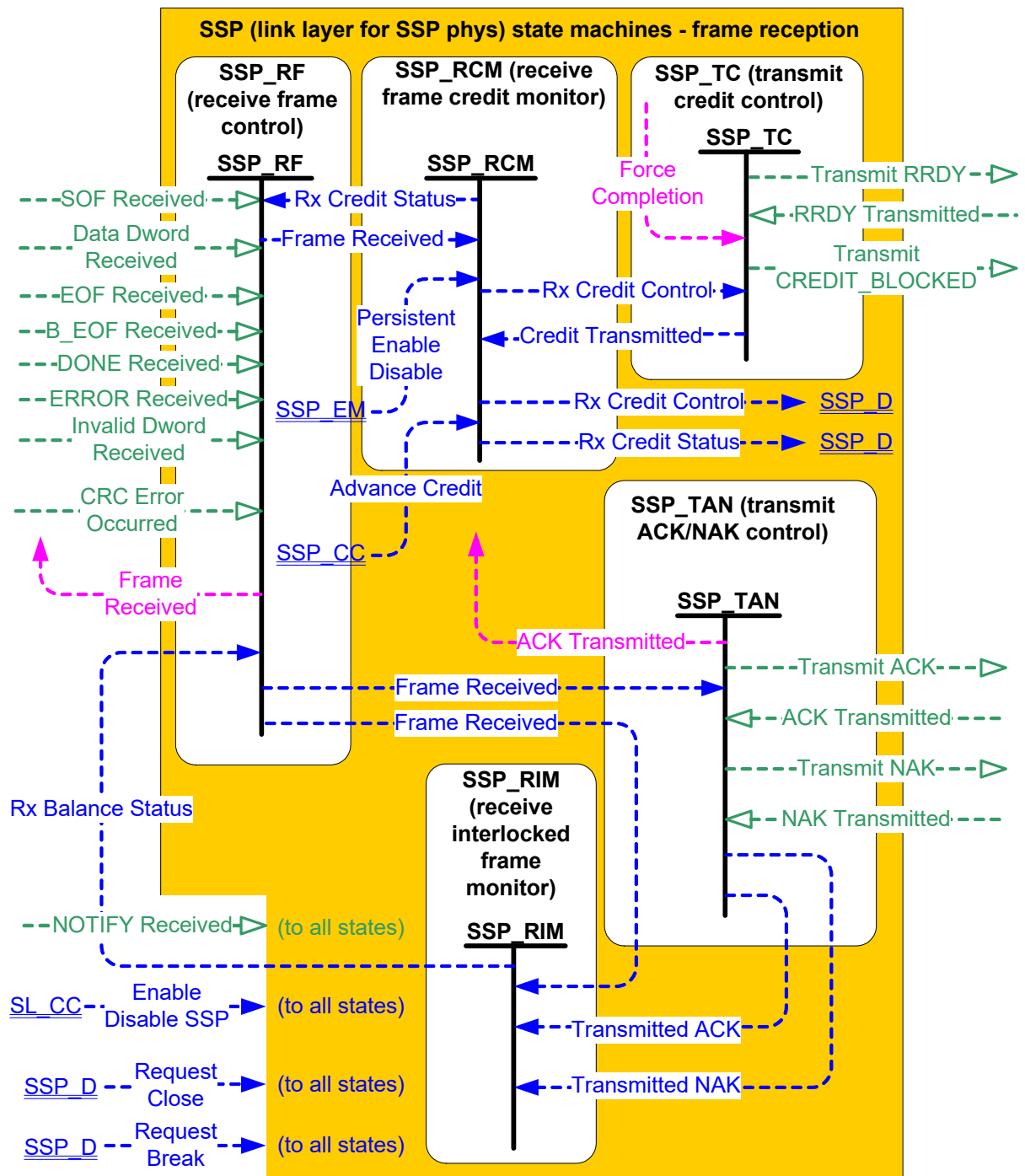
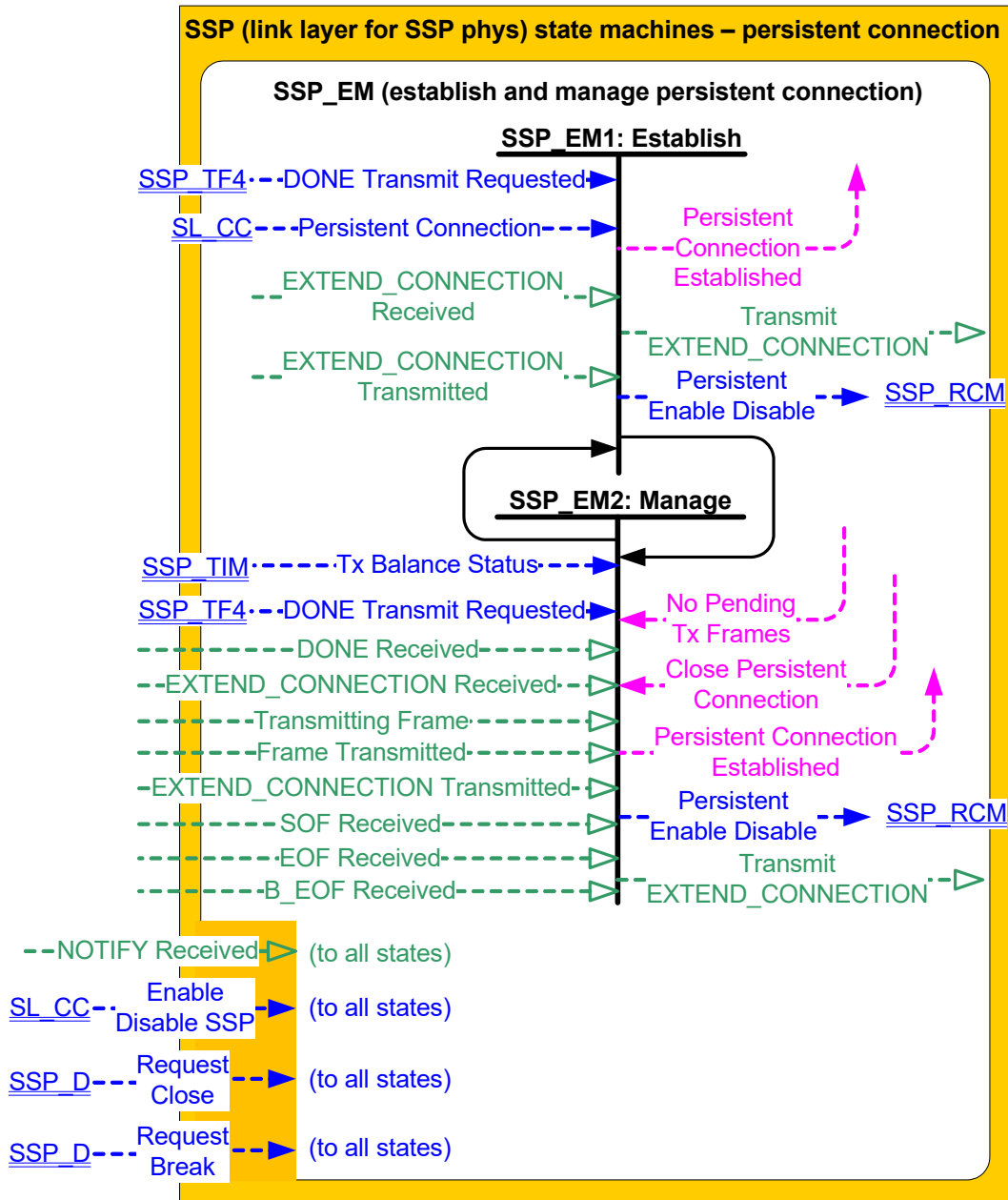


Figure 176 –SSP (link layer for SSP phys) state machines (2 of 3 - frame reception)

Figure 177 shows the SSP state machine and states related to persistent connections.



**Figure 177 –SSP (link layer for SSP phys) state machines (3 of 3 - persistent connection)**

#### 6.20.9.2 SSP transmitter and receiver

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

- Transmit RRDY with an argument indicating the specific type (e.g., Transmit RRDY (Normal));
- Transmit CREDIT\_BLOCKED;
- Transmit ACK;
- Transmit NAK with an argument indicating the specific type (e.g., Transmit NAK (CRC Error));
- Transmit Frame with an argument containing the frame contents;
- Transmit DONE with an argument indicating the specific type (e.g., Transmit DONE (Normal)); and

- g) Transmit `EXTEND_CONNECTION` with an argument indicating the specific type (e.g., Transmit `EXTEND_CONNECTION (Normal)`).

If SAS dword mode is enabled, then, in response to the Transmit Frame message, the SSP transmitter transmits:

- 1) SOF;
- 2) the frame contents;
- 3) CRC; and
- 4) EOF.

If SAS packet mode is enabled, then, in response to the Transmit Frame message, the SSP transmitter transmits:

- 1) SOF;
- 2) the frame contents;
- 3) CRC;
- 4) pad dwords, if any, as described in 6.20.3.3; and
- 5) `B_EOF (0)`, `B_EOF (1)`, `B_EOF (2)`, or `B_EOF (3)` as described in 6.20.3.3.

If persistent connections are supported (see 4.1.13.2), then in response to the Transmit Frame message, the SSP transmitter shall send a Transmitting Frame message to the SSP state machines before transmitting an SOF.

The SSP transmitter shall not transmit an `EXTEND_CONNECTION` while processing a Transmit Frame message (i.e., the `EXTEND_CONNECTION` shall only be transmitted outside an SSP frame).

When the SSP transmitter is requested to transmit a dword from any state within any of the SSP state machines, the SSP transmitter shall transmit that dword. If there are multiple requests to transmit, then the following priority should be followed when selecting the dwords to transmit:

- 1) `RRDY`, `CREDIT_BLOCKED`, `ACK`, `NAK`, or `DONE`;
- 2) SOF, the frame contents, CRC, and:
  - A) EOF;
  - B) `B_EOF (0)`;
  - C) `B_EOF (1)`;
  - D) `B_EOF (2)`; or
  - E) `B_EOF (3)`;
- 3) `EXTEND_CONNECTION`; and
- 4) idle dword.

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) `EXTEND_CONNECTION` Transmitted;
- e) `ACK` Transmitted;
- f) `NAK` Transmitted;
- g) Transmitting Frame; and
- h) Frame Transmitted.

While 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 from the `SP_DWS` receiver (see 5.15.2) or `SP_PS` receiver (see 5.16.2):

- a) `ACK` Received;
- b) `NAK` Received;
- c) `RRDY` Received with an argument indicating the specific type (e.g., `RRDY Received (Normal)`);
- d) `CREDIT_BLOCKED` Received;

- e) DONE Received with an argument indicating the specific type (e.g., DONE Received (Normal));
- f) SOF Received;
- g) Data Dword Received;
- h) EOF Received;
- i) B\_EOF Received;
- j) NOTIFY Received (Power Loss Expected);
- k) ERROR Received;
- l) EXTEND\_CONNECTION Received with an argument indicating the specific type (e.g., EXTEND\_CONNECTION Received (Normal)); and
- m) Invalid Dword Received.

The SSP receiver shall ignore:

- a) pad dwords, if any, received before a B\_EOF as described in 6.20.3.3; and
- b) all dwords not described in this subclause.

The SSP transmitter relationship to other transmitters is defined in 4.3.2. The SSP receiver relationship to other receivers is defined in 4.3.3.

### 6.20.9.3 SSP\_TIM (transmit interlocked frame monitor) state machine

The SSP\_TIM state machine's function is to ensure that ACKs or NAKs are received for each transmitted frame before the ACK/NAK timeout. This state machine consists of one state.

This state machine monitors the number of frames transmitted with a Number Of Frames Transmitted counter and monitors the number of ACKs and NAKs received with a Number Of ACKs/NAKs Received counter. This state machine ensures that an ACK or NAK is received for each frame transmitted and reports an ACK/NAK timeout if they are not.

When the Number Of Frames Transmitted counter equals the Number Of ACKs/NAKs Received counter, the ACK/NAK count is balanced and this state machine shall repeatedly send the Tx Balance Status (Balanced) message to the SSP\_TF2:Tx\_Wait state and the SSP\_EM state machine. When the Number Of Frames Transmitted counter does not equal the Number Of ACKs/NAKs Received counter, the ACK/NAK count is not balanced and this state machine shall send the Tx Balance Status (Not Balanced) message to the SSP\_TF2:Tx\_Wait state and the SSP\_EM state machine.

Each time a Frame Transmitted message is received, this state machine shall increment the Number Of Frames Transmitted counter.

If the ACK/NAK count is not balanced, then each time an ACK Received message is received, this state machine shall:

- a) increment the Number Of ACKs/NAKs Received counter; and
- b) send an ACK Received confirmation to the port layer.

If the ACK/NAK count is not balanced, then each time a NAK Received message is received, this state machine shall:

- a) increment the Number Of ACKs/NAKs Received counter; and
- b) send a NAK Received confirmation to the port layer.

If the ACK/NAK count is balanced, then the ACK Received message and NAK Received message shall be ignored and the ACK/NAK Timeout timer shall be stopped.

Each time the ACK/NAK count is not balanced, the ACK/NAK Timeout timer shall be initialized and started. The ACK/NAK Timeout timer shall be reinitialized each time the Number Of ACKs/NAKs Received counter is incremented. If the ACK/NAK Timeout timer expires, then this state machine shall send the ACK/NAK Timeout confirmation to the port layer and to the following states:

- a) SSP\_TF1:Connected\_Idle; and
- b) SSP\_TF2:Tx\_Wait state.

When this state machine receives an Enable Disable SSP (Enable) message, Request Close message, or Request Break message, the Number Of Frames Transmitted counter shall be set to zero and the Number Of ACKs/NAKs Received counter shall be set to zero.

#### 6.20.9.4 SSP\_TCM (transmit frame credit monitor) state machine

The SSP\_TCM state machine's function is to ensure that transmit SSP frame credit is available before a frame is transmitted. This state machine consists of one state.

This state machine shall keep track of the number of transmit SSP frame credits available.

If an Advance Credit (Received) message is received, then this state machine shall:

- 1) increment transmit SSP frame credit by one frame;
- 2) ignore the first RRDY Received message received; and
- 3) add one transmit SSP frame credit for each subsequent RRDY Received message received.

If an Advance Credit (Received) message is not received, then this state machine shall add one transmit SSP frame credit for each RRDY Received message received.

This state machine shall subtract one transmit SSP frame credit for each Tx Credit Used message received.

Upon starting this state machine, the number of transmit SSP frame credits available value shall be set to zero.

The CREDIT\_BLOCKED Received message indicates that transmit SSP frame credit is blocked. After receiving a CREDIT\_BLOCKED Received message, this state machine may ignore additional RRDY Received messages until it receives a Request Close message or a Request Break message.

This state machine shall handle an EXTEND\_CONNECTION Received (Close) message as if a CREDIT\_BLOCKED Received message was received.

The RRDY Received (Close) message indicates:

- a) that transmit SSP frame credit is available; and
- b) this state machine shall handle that RRDY Received (Close) message as if a CREDIT\_BLOCKED Received message was received.

After receiving an RRDY Received (Close) message, this state machine may ignore additional RRDY Received messages until it receives a Request Close message or a Request Break message.

If this state receives an RRDY Received (Close) message or an EXTEND\_CONNECTION Received (Close) message, then this state shall repeatedly send a Begin Close message to the SSP\_TF1:Connected\_Idle state.

While transmit SSP frame credit is available, this state machine repeatedly shall send the Tx Credit Status (Available) message to the SSP\_TF2:Tx\_Wait state.

While transmit SSP frame credit is not available and transmit SSP frame credit is not blocked, this state machine shall repeatedly send the Tx Credit Status (Not Available) message to the SSP\_TF2:Tx\_Wait state.

While transmit SSP frame credit is not available and transmit SSP frame credit is blocked, this state machine shall repeatedly send the Tx Credit Status (Blocked) message to the SSP\_TF2:Tx\_Wait state.

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

#### 6.20.9.5 SSP\_D (DONE control) state machine

The SSP\_D state machine's function is to ensure a DONE has been received and transmitted before the SL\_CC state machine disables the SSP state machines. This state machine consists of one state.

This state machine ensures that a DONE is received and transmitted before the connection is closed. The DONE may be transmitted and received in any order.

If a DONE (Close) Received message is received, then this state shall repeatedly send a Begin Close message to the SSP\_TF1:Connected\_Idle state.

If a DONE Received message has been received before a Transmitted DONE message is received, then this state machine shall send a Request Close message to the SL\_CC state machine (see 6.18) and all the SSP state machines after receiving the Transmitted DONE message.

If a DONE Received message, a Transmitted DONE (Normal) message, or a Transmitted DONE (Credit Timeout) message has not been received and an Rx Credit Status (Extended) message or an Rx Credit Control (Blocked) message has been received, then this state shall initialize and start the DONE Timeout timer after receiving a Transmitted DONE (Normal) message or a Transmitted DONE (Credit Timeout) message.

If a DONE Received message has not been received and a Transmitted DONE (Normal) message or a Transmitted DONE (Credit Timeout) message has been received, then this state machine shall initialize and start the DONE Timeout timer each time:

- a) an Rx Credit Status (Extended) message is received; or
- b) an Rx Credit Control (Blocked) message is received.

If a Transmitted DONE (Normal) message or a Transmitted DONE (Credit Timeout) message has been received, then the DONE Timeout timer shall be reinitialized each time an EOF Received message or B\_EOF Received message is received.

If a Transmitted DONE (Normal) message or a Transmitted DONE (Credit Timeout) message has been received, then the DONE Timeout timer shall be stopped after:

- a) an Rx Credit Status (Exhausted) message is received; and
- b) an Rx Credit Control (Blocked) message has not been received.

NOTE 42 - Stopping the timer ensures that, if SSP frame credit remains exhausted long enough that the Credit Timeout timer of the other phy in the connection expires, then the other phy is able to transmit a DONE (CREDIT TIMEOUT).

If a DONE Received message has not been received and a Transmitted DONE (ACK/NAK Timeout) message has been received, then:

- a) this state machine shall initialize and start the DONE Timeout timer; and
- b) this state shall not reinitialize the DONE Timeout timer if an EOF Received message or B\_EOF Received message is received.

If a DONE Received message is received before the DONE Timeout timer expires, then this state machine shall send a Request Close message to the SL\_CC state machine and all the SSP state machines.

If a DONE Received message is not received before the DONE Timeout timer expires, then this state machine shall send a Request Break message to the SL\_CC state machine and all the SSP state machines.

If a DONE Received message is received, then this state machine shall send a DONE Received confirmation to the port layer. A DONE Received (ACK/NAK Timeout) confirmation informs the port layer that the SSP transmitter is going to close the connection within 1 ms. Other DONE Received confirmations (e.g., DONE Received (Normal), DONE Received (Close), and DONE Received (Credit Timeout)) may be used by the SCSI application layer to decide when to reuse initiator port transfer tags (see 9.2.2).

It is possible for the DONE Timeout timer in one phy (e.g., phy A) may expire concurrently with the ACK/NAK Timeout timer in the other phy (e.g., phy B) in a connection.

EXAMPLE - If phy A receives DONE (NORMAL) indicating phy B has no more frames to transmit and phy A then transmits a series of non-interlocked frames where one or more of the SOFs is corrupted, then phy A waits to receive all the ACKs and/or NAKs after transmitting the series of non-interlocked frames. However, since phy B did not receive the full number of SOFs, it does not transmit as many ACKs and/or NAKs as phy A is expecting. The ACK/NAK Timeout timer in phy A expires and phy A transmits DONE (ACK/NAK TIMEOUT). Meanwhile, despite having transmitted DONE, phy B stops receiving frames while phy A is waiting for the final ACKs and/or NAKs. Since phy B does not receive DONE or any more frames, its DONE Timeout timer expires and phy B transmits a BREAK primitive sequence. Since the timers may expire at different times (e.g., due to timer resolution differences), the DONE (ACK/NAK TIMEOUT) may be transmitted before,

concurrently with, or after the BREAK primitive sequence. Nevertheless, the phys handle the link layer error (i.e., the ACK/NAK timeout or the DONE timeout) the same way (see 8.2.4.5 and 8.2.4.6).

### **6.20.9.6 SSP\_TF (transmit frame control) state machine**

#### **6.20.9.6.1 SSP\_TF state machine overview**

The SSP\_TF state machine's function is to control when the SSP transmitter transmits SOF, frame dwords, EOF, B\_EOF, and DONE. This state machine consists of the following states:

- a) SSP\_TF1:Connected\_Idle (see 6.20.9.6.2) (initial state);
- b) SSP\_TF2:Tx\_Wait (see 6.20.9.6.3);
- c) SSP\_TF3:Transmit\_Frame (see 6.20.9.6.4); and
- d) SSP\_TF4:Transmit\_DONE (see 6.20.9.6.5).

This state machine shall start in the SSP\_TF1:Connected\_Idle state.

#### **6.20.9.6.2 SSP\_TF1:Connected\_Idle state**

##### **6.20.9.6.2.1 State description**

This state waits for a request to transmit a frame or to close the connection.

##### **6.20.9.6.2.2 Transition SSP\_TF1:Connected\_Idle to SSP\_TF2:Tx\_Wait**

This transition shall occur after:

- a) a Tx Frame request is received;
- b) a Close Connection request is received; or
- c) a Begin Close message is received.

If:

- a) a Tx Frame (Balance Required) request was received; and
- b) no Begin Close message has been received,

then this transition shall include a Transmit Frame Balance Required argument.

If:

- a) a Tx Frame (Balance Not Required) request was received; and
- b) no Begin Close message has been received,

then this transition shall include a Transmit Frame Balance Not Required argument.

If a Close Connection request was received, then this transition shall include a Close Connection argument.

If a Begin Close message was received, then this transition shall include a Full Duplex Close argument.

If a Begin Close message and a Close Connection request were received, then this transition:

- a) shall include a Full Duplex Close argument; and
- b) shall not include a Close Connection argument.

If a Begin Close message and a Tx Frame request were received, then this transition:

- a) shall include a Full Duplex Close argument; and
- b) shall not include a Transmit Frame Balance Required argument or a Transmit Frame Balance Not Required argument.

##### **6.20.9.6.2.3 Transition SSP\_TF1:Connected\_Idle to SSP\_TF4:Transmit\_DONE**

This transition shall occur:

- a) if an ACK/NAK Timeout message is received.

This transition shall include an ACK/NAK Timeout argument.

#### **6.20.9.6.3 SSP\_TF2:Tx\_Wait state**

##### **6.20.9.6.3.1 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 if the last Tx Credit Status message:

- a) received had an argument of Not Available, then this state shall initialize and start the Credit Timeout timer; or
- b) had an argument other than Not Available, then this state shall stop the Credit Timeout timer.

##### **6.20.9.6.3.2 Transition SSP\_TF2:Tx\_Wait to SSP\_TF3:Transmit\_Frame**

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 the last:

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

This transition shall occur:

- a) after sending a Tx Credit Used message to the SSP\_TCM state machine.

##### **6.20.9.6.3.3 Transition SSP\_TF2:Tx\_Wait to SSP\_TF4:Transmit\_DONE**

This transition shall occur and include:

- a) 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 Full Duplex Close argument if:

- a) this state was entered from the SSP\_TF1:Connected\_Idle state with an argument of Full Duplex Close; and
- b) the last Tx Balance Status message received had an argument of Balanced.

This transition shall occur after sending a Credit Timeout confirmation 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.



**6.20.9.6.4 SSP\_TF3:Transmit\_Frame state****6.20.9.6.4.1 State description**

This state shall request a frame transmission by sending a Transmit Frame message to the SSP transmitter with an argument containing the frame contents. Each time a Transmit Frame message is sent to the SSP transmitter, one SSP frame (i.e., SOF, frame contents, CRC, and EOF or B\_EOF) is transmitted.

In this state receiving a Frame Transmitted message indicates that the frame has been transmitted.

**6.20.9.6.4.2 Transition SSP\_TF3:Transmit\_Frame to SSP\_TF1:Connected\_Idle**

This transition shall occur after:

- a) receiving a Frame Transmitted message;
- b) sending a Frame Transmitted message to the SSP\_TIM state machine; and
- c) sending a Frame Transmitted confirmation to the port layer.

**6.20.9.6.5 SSP\_TF4:Transmit\_DONE state**

This state shall send one of the following messages to an SSP transmitter:

- a) a Transmit DONE (Normal) message if this state was entered from the SSP\_TF2:Tx\_Wait state with an argument of Close Connection;
- b) a Transmit DONE (Close) message if this state was entered from the SSP\_TF2:Tx\_Wait state with an argument of Full Duplex Close;
- c) a Transmit DONE (ACK/NAK Timeout) message if this state was entered from the SSP\_TF2:Tx\_Wait state or the SSP\_TF1:Connected\_Idle state with an argument of ACK/NAK Timeout; or
- d) a Transmit DONE (Credit Timeout) message if this state was entered from the SSP\_TF2:Tx\_Wait state with an argument of Credit Timeout.

NOTE 43 - Possible livelock scenarios occur if the BREAK\_REPLY method of responding to received BREAK primitive sequences is disabled and a SAS logical phy transmits a BREAK primitive sequence to break a connection (e.g., if its Done Timeout timer expires). SAS logical phys responding to DONE faster than 1 ms reduce susceptibility to this problem.

This state shall send a DONE Transmit Requested message to the SSP\_EM1:Establish state and the SSP\_EM2:Manage state.

After a DONE Transmitted message is received, this state shall send the DONE Transmitted confirmation to the port layer and send one of the following messages to the SSP\_D state machine:

- a) a Transmitted DONE (Normal) message if this state was entered from the SSP\_TF2:Tx\_Wait state with an argument of Close Connection or Full Duplex Close;
- b) a Transmitted DONE (ACK/NAK Timeout) message if this state was entered from the SSP\_TF2:Tx\_Wait state or the SSP\_TF1:Connected\_Idle state with an argument of ACK/NAK Timeout; or
- c) a Transmitted DONE (Credit Timeout) message if this state was entered from the SSP\_TF2:Tx\_Wait state with an argument of Credit Timeout.

**6.20.9.7 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 without error. 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.

If this state receives a subsequent SOF Received message after receiving an SOF Received message but before receiving an EOF Received message or B\_EOF Received message (e.g., SOF, data dwords, SOF, data dwords, and EOF instead of SOF, data dwords, EOF, SOF, data dwords, and EOF), then this state shall discard the frame in progress.

This state shall discard the frame if this state receives:

- a) more than 263 Data Dword Received messages (i.e., 1 052 bytes) after an SOF Received message and before:
  - A) an EOF Received message; or
  - B) a B\_EOF Received message;
- b) fewer than 7 Data Dword Received messages (i.e., 28 bytes) after an SOF Received message and before:
  - A) an EOF Received message; or
  - B) a B\_EOF Received message;
- c) an Rx Credit Status (Credit Exhausted) message; or
- d) a DONE Received message.

If this state receives an Invalid Dword Received message or an ERROR Received message after receiving an SOF Received message and before receiving an EOF Received message or B\_EOF Received message, then this state machine shall either:

- a) ignore the invalid dword or ERROR; or
- b) discard the frame, send a Frame Received message to the SSP\_RCM state machine, send a Frame Received message to the SSP\_RIM state machine, and send a Frame Received (Unsuccessful) message to the SSP\_TAN state machine.

If the frame is not discarded and a CRC Error Occurred message was received for the frame, then this state machine shall send:

- a) a Frame Received message to the SSP\_RCM state machine;
- b) a Frame Received message to the SSP\_RIM state machine; and
- c) a Frame Received (Unsuccessful) message to the SSP\_TAN state machine.

If the frame is not discarded and no CRC Error Occurred message was received for the frame, then this state machine shall send:

- a) a Frame Received message to the SSP\_RCM state machine;
- b) a Frame Received message to the SSP\_RIM state machine; and
- c) a Frame Received (Successful) message to the SSP\_TAN state machine and if the last Rx Balance Status message received had an argument of:
  - A) Balanced, then send a Frame Received (ACK/NAK Balanced) confirmation to the port layer; or
  - B) Not Balanced, then send a Frame Received (ACK/NAK Not Balanced) confirmation to the port layer.

#### 6.20.9.8 SSP\_RCM (receive frame credit monitor) state machine

The SSP\_RCM state machine's function is to ensure that there was transmit SSP frame 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.

Upon starting this state machine, the following values shall be set to zero:

- a) the number of RRDYs transmitted; and
- b) the number of frames received.

If an Advance Credit (Transmitted) message is received, then this state machine shall send an Rx Credit Control (Available) message to the SSP\_TC state machine.

If:

- a) resources are released or become available; and

- b) this state machine has not sent the Rx Credit Control (Blocked) message to the SSP\_TC state machine and the SSP\_D state machine,

then this state machine shall send the Rx Credit Control (Available) message to the SSP\_TC state machine. This state machine shall 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.

If this state machine has:

- a) not received a Persistent Enable Disable (Enabled) message; or
- b) received a Persistent Enable Disable (Enabled) message followed by a Persistent Enable Disable (Disabled) message,

then this state machine may send the Rx Credit Control (Blocked) message 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), even if frames are received per the existing receive frame credit. 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.

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 five frames, then the maximum number of Rx Credit Control requests with the Available argument outstanding is five).

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.

If the number of Credit Transmitted messages received exceeds the number of Frame Received messages received, then this state machine shall send an Rx Credit Status (Extended) message to the SSP\_RF state machine and the SSP\_D state machine.

If the number of Credit Transmitted messages received equals the number of Frame Received messages received, then this state machine shall send an Rx Credit Status (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, then the frame receive resources shall be initialized to the no SSP frame credit value for the current connection.

#### **6.20.9.9 SSP\_RIM (receive interlocked frame monitor) state machine**

The SSP\_RIM state machine's function is to inform the SSP\_RF state machine when the number of ACKs and NAKs transmitted equals the number of the EOFs received and B\_EOFs received. This state machine consists of one state.

This state machine monitors the number of frames received with a Number Of Frames Received counter and monitors the number of ACKs and NAKs transmitted with a Number Of ACKs/NAKs Transmitted counter.

Each time a Frame Received message is received, this state machine shall increment the Number Of Frames Received counter.

Each time a Transmitted ACK message or a Transmitted NAK message is received, this state machine shall increment the Number Of ACKs/NAKs Transmitted counter.

While the Number Of Frames Received counter equals the Number Of ACKs/NAKs Transmitted counter, this state machine shall repeatedly send an Rx Balance Status (Balanced) message to the SSP\_RF state machine.

While the Number Of Frames Received counter does not equal the Number Of ACKs/NAKs Transmitted counter, this state machine shall repeatedly send an Rx Balance Status (Not Balanced) message to the SSP\_RF state machine.

When this state machine receives an Enable Disable SSP (Enable) message, Request Close message, or Request Break message, the Number Of Frames Received counter shall be set to zero and the Number Of ACKs/NAKs Transmitted counter shall be set to zero.

#### **6.20.9.10 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.

If this state machine receives an Rx Credit Control (Available) message, then this state machine shall send to the SSP transmitter the indicated amount of resources available to handle received frames (e.g., if the Available argument indicates five RRDYs are to be transmitted, then this state machine sends five Transmit RRDY (Normal) messages to the SSP transmitter) and if this state has:

- a) not received a Force Completion request, then this state machine shall send that number of Transmit RRDY (Normal) messages; or
- b) received a Force Completion request, then this state machine shall send that number of Transmit RRDY (Close) messages.

If this state machine receives an RRDY Transmitted message, then this state machine shall send a Credit Transmitted message to the SSP\_RCM state machine.

If this state machine receives an Rx Credit Control (Blocked) message, then this state machine shall send a Transmit CREDIT\_BLOCKED message to the SSP transmitter.

#### **6.20.9.11 SSP\_TAN (transmit ACK/NAK control) state machine**

The SSP\_TAN state machine's function is to control the sending of requests to transmit an ACK or NAK to the SSP transmitter. This state machine consists of one state.

If this state machine receives a Frame Received (Successful) message, then this state machine shall send a Transmit ACK message to the SSP transmitter.

If this state machine receives a Frame Received (Unsuccessful) message, then this state machine shall send a Transmit NAK (CRC Error) message to the SSP transmitter.

If multiple Frame Received (Unsuccessful) messages and Frame Received (Successful) messages are received, then the order in which the Transmit ACK messages and Transmit NAK messages are sent to the SSP transmitter shall be the same order as the Frame Received (Unsuccessful) messages and Frame Received (Successful) messages were received.

If this state machine receives an ACK Transmitted message, then this state machine shall send:

- a) a Transmitted ACK message to the SSP\_RIM state machine; and
- b) an ACK Transmitted confirmation to the port layer.

If this state machine receives a NAK Transmitted argument, then this state machine shall send a Transmitted NAK message to the SSP\_RIM state machine.

#### **6.20.9.12 SSP\_EM (establish and manage persistent connection) state machine**

##### **6.20.9.12.1 SSP\_EM state machine overview**

The SSP\_EM state machine's function is to establish a persistent connection and manage the sending and receiving of EXTEND\_CONNECTIONs. This state machine consists of the following states:

- a) SSP\_EM1:Establish (see 6.20.9.12.2) (initial state); and
- b) SSP\_EM2:Manage (see 6.20.9.12.4).

This state machine shall start in the SSP\_EM1:Establish state.

**6.20.9.12.2 SSP\_EM1:Establish state**

This state establishes a persistent connection (see 6.20.1).

If this state receives a Persistent Connection (Transmit) message, then:

- a) this state shall send a Transmit EXTEND\_CONNECTION (Normal) message to the SSP transmitter;
- b) this state shall initialize and start the Transmit Extend Connection timer; and
- c) if this state receives an EXTEND\_CONNECTION Received message and this state has not received a DONE Transmit Requested message, then this state shall send:
  - A) a Persistent Connection Established (Enabled) confirmation to the port layer; and
  - B) a Persistent Enable Disable (Enable) message to the SSP\_RCM state machine.

If this state receives a Persistent Connection (Wait) message, has not received a DONE Transmit Requested message, and receives an EXTEND\_CONNECTION Received message, then this state shall send:

- a) a Transmit EXTEND\_CONNECTION (Normal) message to the SSP transmitter;
- b) a Persistent Connection Established (Enabled) confirmation to the port layer; and
- c) a Persistent Enable Disable (Enable) message to the SSP\_RCM state machine.

If this state receives a Persistent Connection (Off) message, then this state shall:

- a) ignore EXTEND\_CONNECTION Received messages; and
- b) send a Persistent Connection Established (Disabled) confirmation to the port layer.

If the Transmit Extend Connection timer expires and this state has not received a DONE Transmit Requested message, then this state shall:

- 1) wait for a Tx Balance Status (Balanced) message;
- 2) send a Transmit EXTEND\_CONNECTION (Normal) message to the SSP transmitter; and
- 3) after receiving an EXTEND\_CONNECTION Transmitted message, initialize and start the Transmit Extend Connection timer.

**6.20.9.12.3 Transition SSP\_EM1:Establish to SSP\_EM2:Manage**

This transition shall occur after:

- a) sending a Persistent Connection Established (Enabled) confirmation to the port layer; and
- b) receiving an EXTEND\_CONNECTION Transmitted message.

**6.20.9.12.4 SSP\_EM2:Manage**

This state:

- a) requests the SSP transmitter transmit EXTEND\_CONNECTION; and
- b) monitors the receipt of EXTEND\_CONNECTION.

Upon entry into this state, this state shall initialize and start:

- a) the Transmit Extend Connection timer; and
- b) the Persistent Connection Timeout timer.

If this state receives:

- a) a Frame Transmitted message, then this state shall initialize and start the Transmit Extend Connection timer; or
- b) a Transmitting Frame message, then this state shall stop the Transmit Extend Connection timer.

If the Transmit Extend Connection timer expires, then this state shall:

- 1) wait for a Tx Balance Status (Balanced) message;
- 2) send a Transmit EXTEND\_CONNECTION (Normal) message to the SSP transmitter; and
- 3) after receiving an EXTEND\_CONNECTION Transmitted message, initialize and start the Transmit Extend Connection timer.

If this state receives an EXTEND\_CONNECTION Received message, then this state shall initialize and start the Persistent Connection Timeout timer.

If this state receives:

- a) an SOF Received message, then this state shall stop the Persistent Connection Timeout timer; or
- b) an EOF Received message or B\_EOF Received message and the Persistent Connection Timeout timer is currently stopped, then this state shall initialize and start the Persistent Connection Timeout timer.

If:

- a) the Persistent Connection Timeout timer expires;
- b) this state receives a DONE Received message;
- c) this state receives a Close Persistent Connection request; or
- d) this state receives a DONE Transmit Requested message,

then this state shall send:

- a) a Persistent Connection Established (Disabled) confirmation to the port layer; and
- b) a Persistent Enable Disable (Disable) message to the SSP\_RCM state machine.

If this state receives a No Pending Tx Frames request, then this state shall:

- 1) wait for a Tx Balance Status (Balanced) message; and
- 2) send a Transmit EXTEND\_CONNECTION (Normal) message to the SSP transmitter.

#### 6.20.9.12.5 Transition SSP\_EM2:Manage to SSP\_EM1:Establish

This transition shall occur:

- a) after a Persistent Connection Established (Disabled) confirmation is sent to the port layer.

## 6.21 STP link layer

### 6.21.1 STP frame transmission and reception overview

STP frame transmission is defined by SATA. Other primitives may be interspersed during the connection as defined by SATA.

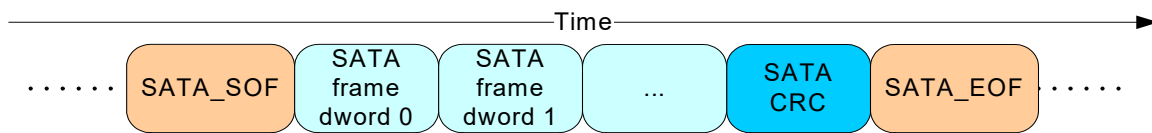
STP encapsulates SATA with connection management. Table 199 summarizes STP link layer differences from the SATA link layer (see SATA) that affect behavior during an STP connection.

**Table 199 – STP link layer differences from SATA link layer during an STP connection**

Feature	Description	Reference
STP flow control	Flow control through an STP connection is point-to-point, not end-to-end. Expander devices accept dwords in an STP flow control buffer after transmitting SATA_HOLD to avoid losing data en-route before the transmitting phy acknowledges the SATA_HOLD with SATA_HOLD_A.	6.21.4
Continued primitive sequence	Sustain the continued primitive sequence if a SATA_CONT appears after the continued primitive sequence has begun.	6.21.5

### 6.21.2 STP frame transmission and reception while in the SAS dword mode

During an STP connection, frames are preceded by SATA\_SOF and followed by SATA\_EOF as shown in figure 178.



**Figure 178 –STP frame transmission**

The last data dword after the SOF prior to the EOF always contains a CRC (see 6.7).

### 6.21.3 STP frame transmission and reception while in the SAS packet mode

During an STP connection while in the SAS packet mode, STP frames are preceded by SATA\_SOFTWARE and followed by B\_EOF (0), B\_EOF (1), B\_EOF (2), or B\_EOF (3) as shown in figure 179, figure 180, figure 181, and figure 182.

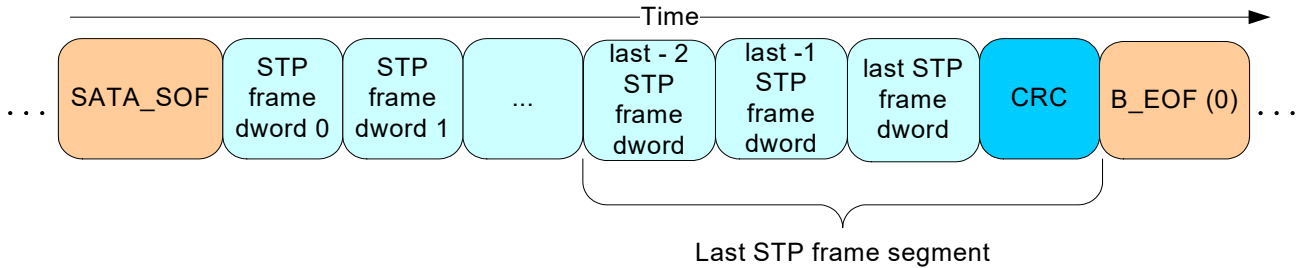


Figure 179 –STP frame transmission with no pad dword

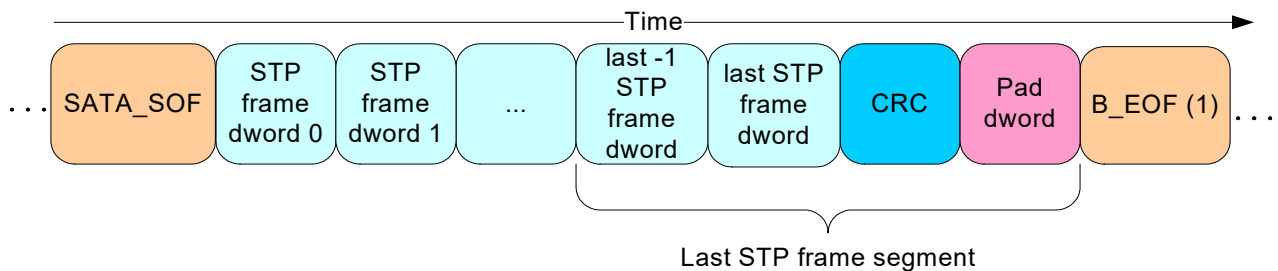


Figure 180 –STP frame transmission with one pad dword

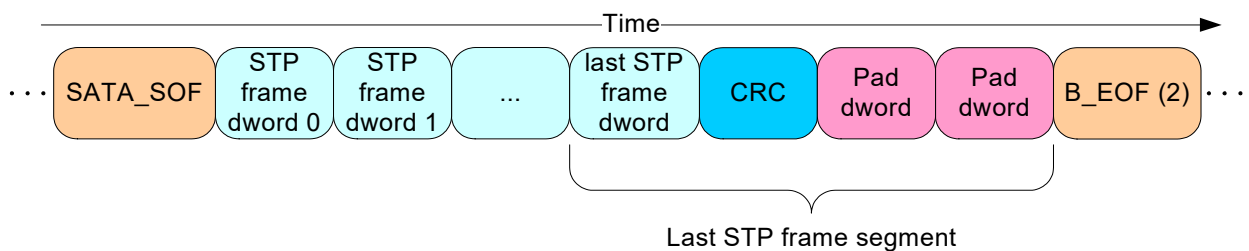


Figure 181 –STP frame transmission with two pad dwords

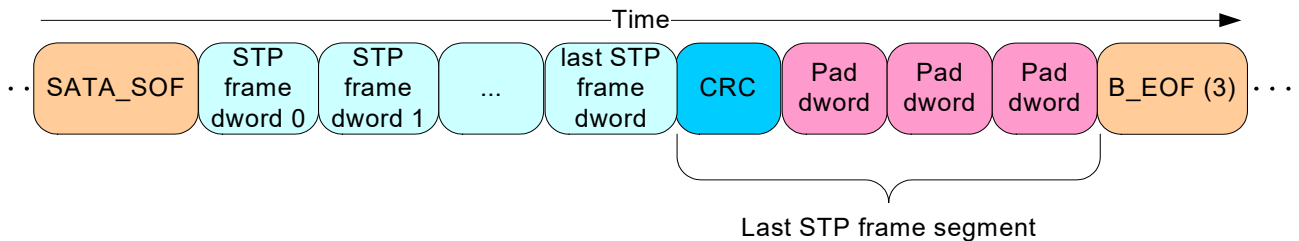


Figure 182 –STP frame transmission with three pad dwords



The last data dword after the SATA\_SOF, before the pad dwords, if any, and before the B\_EOF (0), the B\_EOF (1), the B\_EOF (2), or the B\_EOF (3) shall contain a CRC (see 6.7). An expander device shall determine the position of the CRC in the last STP frame segment by the first B\_EOF (0), the B\_EOF (1), the B\_EOF (2), or the B\_EOF (3) following the last SPL frame segment as follows:

- a) B\_EOF (0) specifies the CRC is the fourth dword of the last STP frame segment (i.e., there are no pad dwords) (see figure 179);
- b) B\_EOF (1) specifies the CRC is the third dword of the last STP frame segment (i.e., there is one pad dword) (see figure 180);
- c) B\_EOF (2) specifies the CRC is the second dword of the last STP frame segment (i.e., there are two pad dwords) (see figure 181); and
- d) B\_EOF (3) specifies the CRC is the first dword of the last STP frame segment (i.e., there are three pad dwords) (see figure 182).

If a B\_EOF is received on a SAS physical link that is in the SAS packet mode and that B\_EOF is to be forwarded to:

- a) a SAS physical link that is in the SAS packet mode, then the expander device shall not make any changes to the STP frame's B\_EOF;
- b) a SAS physical link that is in the SAS dword mode, then the expander device shall replace that B\_EOF with a SATA\_EOF positioned in the next dword after the CRC (see 6.7) before forwarding; or
- c) a SATA physical link, then the STP target device shall replace that B\_EOF with a SATA\_EOF positioned in the next dword after the CRC (see 6.7) before forwarding.

If a SATA\_EOF is received on a SAS physical link that is in the SAS dword mode and that SATA\_EOF is to be forwarded to a SAS physical link that is in:

- a) the SAS dword mode, then the expander device shall not make any changes to the STP frame's SATA\_EOF; or
- b) the SAS packet mode, then the expander device shall replace that SATA\_EOF with a B\_EOF after the pad dwords, if any, as described in this subclause.

## 6.21.4 STP flow control

### 6.21.4.1 STP flow control overview

Each STP phy (i.e., STP initiator phy and STP target phy) and expander logical phy through which the STP connection is routed shall implement the SATA flow control protocol on each logical link in the pathway. The flow control primitives are not forwarded through expander devices like other dwords.

### 6.21.4.2 SATA frame buffering

When an STP phy or expander phy during an STP connection is receiving a SATA frame and its STP flow control buffer begins to fill up, that STP phy or expander phy shall transmit SATA\_HOLD. After transmitting SATA\_HOLD, the STP phy or expander phy shall accept at least the number of data dwords, SATA\_EOFs, or B\_EOFs for the SATA frame defined in 6.21.4.3 into its STP flow control buffer and receives SATA\_HOLD\_A within that number of data dwords, SATA\_EOFs, or B\_EOFs. While receiving SATA\_HOLD\_A, the STP phy or expander phy does not place any data dwords into the STP flow control buffer. The STP phy or expander phy shall stop transmitting SATA\_HOLD when the STP flow control buffer empties enough to hold at least that number of data dwords, SATA\_EOFs, or B\_EOFs.

When an STP phy or expander phy during an STP connection is transmitting a SATA frame and receives SATA\_HOLD, that STP phy or expander phy shall transmit no more than:

- a) 20 data dwords or SATA\_EOFs if SAS dword mode is enabled; or
- b) 24 SPL frame segments or primitive segments containing B\_EOF if SAS packet mode is enabled,

for the SATA frame and respond with SATA\_HOLD\_A.

When a SATA host phy in an STP SATA bridge is receiving a SATA frame from a SATA physical link, that SATA host phy shall transmit a SATA\_HOLD when it is only capable of receiving 21 more data dwords for Gen1 and Gen2 or 25 more data dwords for Gen3 (see SATA). The SATA host phy shall stop transmitting SATA\_HOLD (e.g., return to transmitting SATA\_R\_IP) when it is capable of receiving at least 21 more data dwords for Gen1 and Gen2 or 24 more data dwords for Gen3.

NOTE 44 - SATA requires that frame transmission cease and SATA\_HOLD be transmitted within 20 data dwords of receiving SATA\_HOLD. Since the SATA physical link has non-zero propagation time for Gen1 or Gen2, one dword of margin is included.

When a SATA host phy in an STP SATA bridge is transmitting a SATA frame to a SATA physical link, that SATA host phy shall transmit no more than 19 dwords for Gen1 and Gen2 or 20 dwords for Gen3 (see SATA) (e.g., including data dwords, deletable primitives, SATA\_EOF, and SATA\_HOLDs followed by data dwords) after receiving SATA\_HOLD before responding with SATA\_HOLDA.

NOTE 45 - SATA assumes that once a SATA\_HOLD is transmitted, frame transmission ceases and SATA\_HOLDA arrives within 20 dwords for Gen1 or Gen2. Since the SATA physical link has non-zero propagation time for Gen1 or Gen2, one dword of margin is included.

While transmitting SATA\_HOLD or SATA\_HOLDA, the expander device is considered to be originating (see 6.5.2) rather than forwarding (see 6.5.4) for purposes of deletable primitive insertion.

#### 6.21.4.3 STP flow control buffer size

All logical phys in a port shall support the same STP flow control buffer size. The STP flow control buffer size is determined for each connector category (see SAS-4) to which the phy is attached as follows:

- a) for the unmanaged passive connector category (see SAS-4) if SAS dword mode is enabled, then the STP flow control buffer size shall be at least:
  - A) 24 data dwords, SATA\_EOFs at the 1.5 Gbit/s logical link rate;
  - B) 28 data dwords, SATA\_EOFs at the 3 Gbit/s logical link rate; or
  - C) 36 data dwords, SATA\_EOFs at the 6 Gbit/s logical link rate;

NOTE 46 - For connectors in the unmanaged connector category the STP flow control buffer requirements are based on  $(20 + (4 \times 2^n))$  if SAS dword mode is enabled, where  $n$  is zero for 1.5 Gbit/s, one for 3 Gbit/s, and two for 6 Gbit/s. The 20 portion of this equation is based on the frame transmitter requirements (see SATA). The  $(4 \times 2^n)$  portion of this equation is based on:

- a) one-way propagation time on a 10 m cable = 53 ns (see SAS-4);
  - b) round-trip propagation time on a 10 m cable = 106 ns (e.g., time to send SATA\_HOLD and receive SATA\_HOLD A);
  - c) time to transmit a 1.5 Gbit/s dword =  $(0.6 \text{ ns/bit unit interval (see SAS-4)}) \times (40 \text{ bits/dword}) = 26.6 \text{ ns}$ ;
  - d) number of 1.5 Gbit/s dwords on the wire during round-trip propagation time =  $(106 \text{ ns} / 26.6 \text{ ns}) = 3.98$  data dwords; and
  - e) rounding to four data dwords.
- b) for the unmanaged active connector category (see SAS-4) if SAS dword mode is enabled, then the STP flow control buffer size shall be at least:
  - A) 30 data dwords or SATA\_EOFs at the 1.5 Gbit/s logical link rate;
  - B) 40 data dwords or SATA\_EOFs at the 3 Gbit/s logical link rate; or
  - C) 60 data dwords or SATA\_EOFs at the 6 Gbit/s logical link rate;

NOTE 47 - For connectors in the unmanaged active connector category the STP flow control buffer requirements are based on  $(20 + (9.375 \times 2^n))$  if SAS dword mode is enabled, where  $n$  is zero for 1.5 Gbit/s, one for 3 Gbit/s, and two for 6 Gbit/s. The 20 portion of this equation is based on the frame transmitter requirements (see SATA). The  $(9.375 \times 2^n)$  portion of this equation is based on:

- a) one-way propagation time on a 25 m cable = 133 ns (see SAS-4);
  - b) round-trip propagation time on a 25 m cable = 266 ns (e.g., time to send SATA\_HOLD and receive SATA\_HOLD A);
  - c) time to transmit a 1.5 Gbit/s dword =  $(0.6 \text{ ns/bit unit interval (see SAS-4)}) \times (40 \text{ bits/dword}) = 26.6 \text{ ns}$ ; and

- d) number of 1.5 Gbit/s dwords on the wire during round-trip propagation time =  $(266 \text{ ns} / 26.6 \text{ ns}) = 10$  data dwords.
- c) for the managed connector category (see SAS-4) if SAS dword mode is enabled, then the total cable assembly propagation delay is reported through the management protocol (see SAS-4). If the propagation delay is less than or equal to 53 ns, then the STP flow control buffer size is 24 data dwords, 28 data dwords, or 36 data dwords depending on the logical link rate (see a)). If the propagation delay exceeds 53 ns, then the minimum STP flow control buffer size in dwords shall be calculated using the following equation:

$$\text{Minimum STP flow control buffer size} = (20 + ((2 \times Pd \times R) / 40))$$

where:

- $Pd$  is the propagation delay of cable assembly (e.g., in nanoseconds) (see SAS-4);  
and  
 $R$  is the nominal logical link rate (e.g., in gigabit per second).

If the minimum buffer size is not an integer, then the minimum buffer size value shall be rounded to the next highest integer value;

NOTE 48 - For connectors in the managed active connector category where the cable reports total propagation delay of 250 ns and connection rate at 6 Gbit/s and SAS dword mode is enabled, then the STP flow control buffer requirements are based on Minimum buffer size =  $[20 + ((2 \times Pd \times R) / 40)]$ . The 20 portion of this equation is based on the frame transmitter requirements (see SATA). The  $[(2 \times Pd \times R) / 40]$  portion of this equation is the number of dwords on the wire during a round trip,  $2 \times Pd$  is the round trip propagation time. The  $40 / R$  portion of this equation is the time to transmit a dword. For example, for a 100 m optical cable:

- one-way propagation time = 500 ns;
- round-trip propagation time = 1 000 ns (e.g., time to send SATA\_HOLD and receive SATA\_HOLD);
- time to transmit a 6 Gbit/s dword =  $(0.16 \text{ ns/bit unit interval}) \times (40 \text{ bits/dword}) = 6.6 \text{ ns}$ ;
- number of 6 Gbit/s dwords on the wire during round-trip propagation time =  $(1\,000 \text{ ns} / 6.6 \text{ ns}) = 150$  data dwords; and
- 20 is the buffer size required by SATA, resulting in a minimum STP flow control buffer size = 170 data dwords

or

- d) for the managed connector category (see SAS-4) if packet mode is enabled, then the total cable assembly propagation delay is reported through the management protocol (see SAS-4). The minimum STP flow control buffer size in SPL packets shall be calculated using the following equation:

$$\text{Minimum STP flow control buffer size} = (24 + ((2 \times Pd \times R) / B))$$

where:

- $Pd$  is the propagation delay of cable assembly (e.g., in nanoseconds) (see SAS-4);  
 $R$  is the nominal logical link rate (e.g., in gigabit per second); and  
 $B$  is 160 if the logical link rate is less than 22.5 Gbit/s and 150 if the logical link rate is equal to 22.5 Gbit/s.

If the minimum buffer size is not an integer, then the minimum buffer size value shall be rounded to the next highest integer value;

NOTE 49 - For connectors in the managed active connector category where the cable reports total propagation delay of 500 ns, a connection rate at 22.5 Gbit/s, and if packet mode is enabled, then the STP flow control buffer requirements are based on Minimum buffer size =  $[24 + ((2 \times Pd \times R) / B)]$ . The 24 portion of this equation is based on the frame transmitter requirements described in this subclause. The  $[(2 \times Pd \times R) / B]$  portion of this equation is the number of SPL frame segments on the wire during a round trip and  $2 \times Pd$  is the round trip propagation time. The  $B / R$  portion of this equation is the time to transmit an SPL packet

containing an SPL frame segment or the time to transmit an equivalent amount of data if the logical link rate is less than 22.5 Gbit/s. For example, for a 100 m optical cable:

- a) one-way propagation time = 500 ns;
- b) round-trip propagation time = 1 000 ns (e.g., time to send SATA\_HOLD and receive SATA\_HOLD\_A);
- c) transmission period of SPL frame segments in a 22.5 Gbit/s logical link rate =  $(0.04 \text{ ns/bit}) \times (150 \text{ bits / SPL frame segment}) = 6.6 \text{ ns}$ ;
- d) number of 22.5 Gbit/s SPL frame segments on the wire during round-trip propagation time =  $(1\,000 \text{ ns} / 6.6 \text{ ns}) = 150 \text{ SPL frame segments}$ ; and
- e) 24 is the buffer size required by the frame transmitter requirements described in this subclause, resulting in a minimum STP flow control buffer size = 174 SPL frame segments.

#### 6.21.4.4 STP flow control example

Figure 183 shows STP flow control between:

- a) an STP initiator phy receiving a frame;
- b) an expander device (the first expander device);
- c) an expander device with an STP SATA bridge (the second expander device); and
- d) a SATA device phy transmitting a frame.

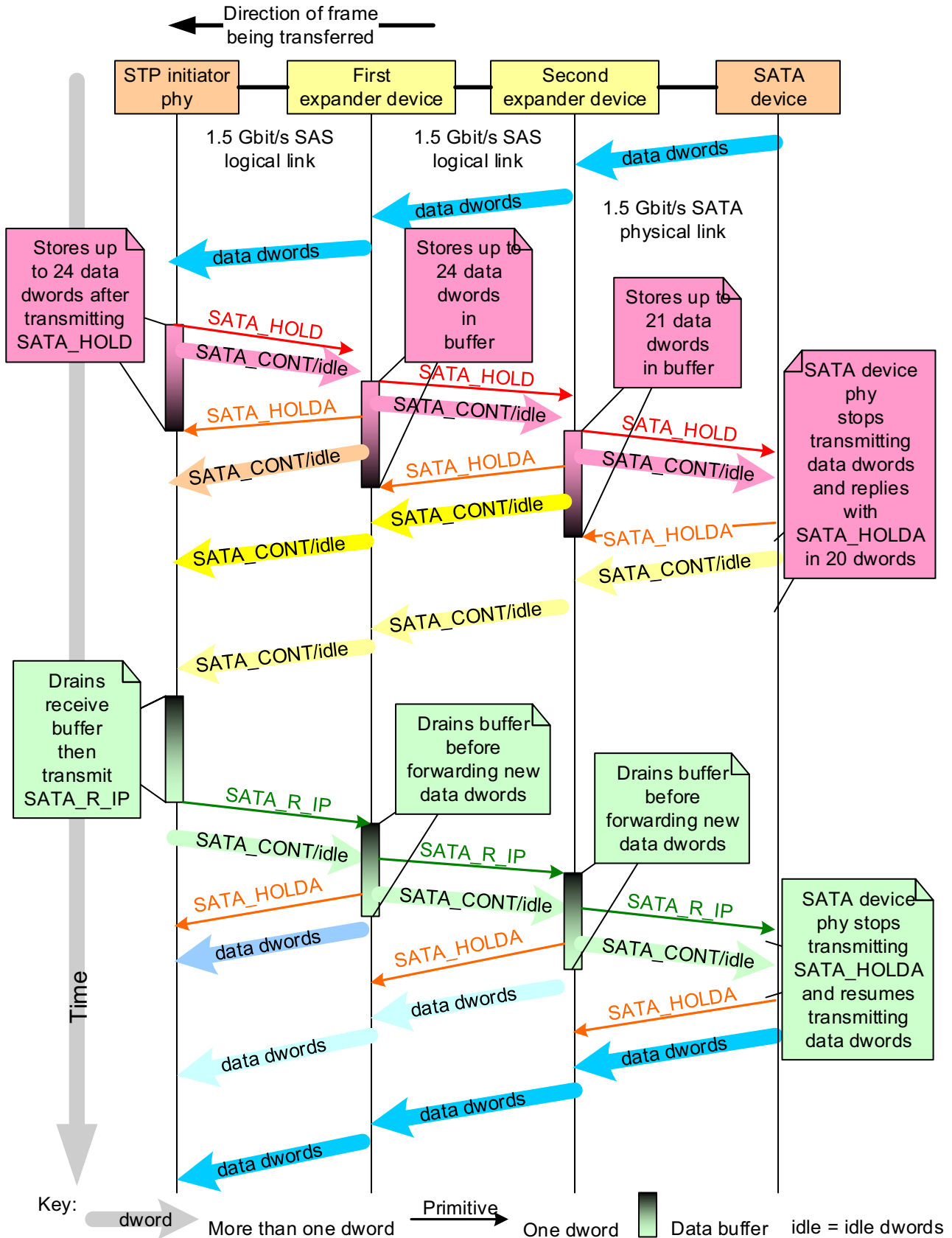


Figure 183 –STP flow control

After the STP initiator phy transmits SATA\_HOLD, it receives a SATA\_HOLDA reply from the first expander device within 24 dwords. The first expander device transmits SATA\_HOLD to the second expander device and receives SATA\_HOLDA within 24 dwords, buffering data dwords in the STP flow control buffer that the first expander device is no longer able to forward to the STP initiator phy. The second expander device transmits SATA\_HOLD to the SATA device phy and receives SATA\_HOLDA within 21 dwords, buffering data dwords in the STP flow control buffer that it is no longer able to forward to the first expander device. When the SATA device phy stops transmitting data dwords, its previous data dwords are stored in the STP flow control buffers in both expander devices and the STP initiator phy.

After the STP initiator phy drains its STP flow control buffer and transmits SATA\_R\_IP, it receives data dwords from the first expander device's STP flow control buffer, followed by data dwords from the second expander device's STP flow control buffer, followed by data dwords from the SATA device phy.

#### 6.21.4.5 STP insufficient buffer support

If an STP phy is attached to a connector in the managed connector category (see SAS-4) and that STP phy does not support the minimum buffer size for the logical link rate, then all phys in the STP port shall perform a link reset sequence indicating the STP protocol is not supported with:

- a) the STP INITIATOR PORT bit set to zero in the IDENTIFY address frame (see 6.10.2); and
- b) the STP TARGET PORT bit set to zero in the IDENTIFY address frame (see 6.10.2).

An STP initiator port shall not originate an STP connection request to a destination STP target port on a pathway that contains an expander device reporting insufficient buffer for STP support.

If an expander phy is attached to a connector in the managed connector category (see SAS-4) and that expander phy does not support the minimum buffer size for the logical link rate, then all phys in the expander port shall perform a link reset sequence indicating the STP protocol is not supported with the STP BUFFER TOO SMALL bit set to one in:

- a) the SMP DISCOVER response (see 9.4.3.10); and
- b) the SMP DISCOVER LIST response (see 9.4.3.15).

If an expander phy does not support the minimum buffer size for the logical link rate, then that expander phy shall respond to an STP connection request with OPEN\_REJECT (PROTOCOL NOT SUPPORTED).

#### 6.21.5 Continued primitive sequence

If the SAS dword mode is enabled, then primitives that form continued primitive sequences (e.g., SATA\_HOLD) shall be:

- 1) transmitted two times;
- 2) be followed by SATA\_CONT, if needed; and
- 3) be followed by vendor specific scrambled data dwords, if needed.

If the SAS packet mode is enabled, then primitives that form continued primitive sequences (e.g., SATA\_HOLD) shall be:

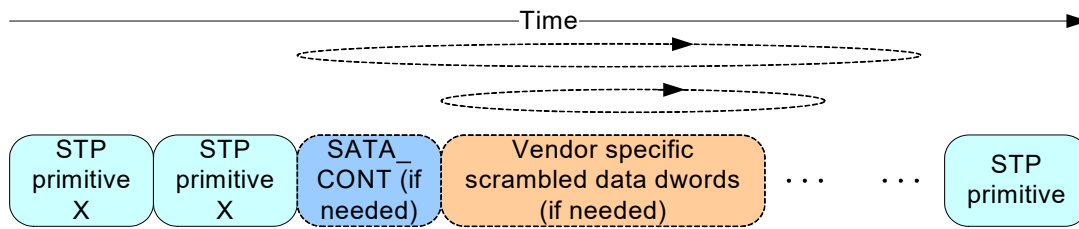
- 1) transmitted two times;
- 2) followed by SATA\_CONT, if needed; and
- 3) followed by idle dword segments, if needed.

Deletable primitives may be transmitted inside continued primitive sequences as described in 6.2.4.1.

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 184 shows an example of transmitting a continued primitive sequence while in the SAS dword mode.

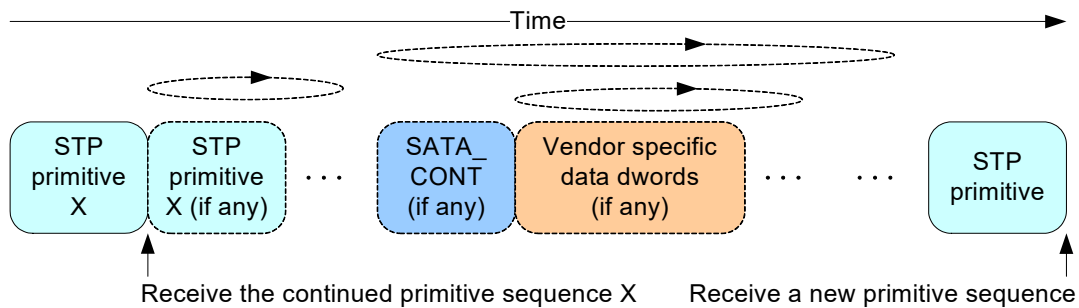


**Figure 184 –Transmitting a continued primitive sequence while in the SAS dword mode**

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\_CONTS. 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 in the state of receiving the primitive).

Expanders forwarding dwords may or may not detect an incoming sequence of the same primitive and convert the incoming sequence into a continued primitive sequence.

Figure 185 shows an example of receiving a continued primitive sequence while in the SAS dword mode.



**Figure 185 –Receiving a continued primitive sequence while in the SAS dword mode**

An expander device forwarding a continued primitive sequence may transmit more dwords in the continued primitive sequence than it receives (i.e., expand) or transmit fewer dwords in the continued primitive sequence than it receives (i.e., contract). While transmitting a continued primitive sequence, the expander device is considered to be originating (see 6.5.2) rather than forwarding (see 6.5.4) for purposes of deletable primitive insertion.

### 6.21.6 Affiliations

The STP target port shall provide coherent access to a set of registers called an affiliation context for each STP initiator port from which the STP target port accepts connections. An affiliation is a state entered by an STP target port in which the STP target port refuses to accept connection requests from STP initiator ports other than those that have established an affiliation.

An STP target port shall implement one of the affiliation policies defined in table 200.

**Table 200 – Affiliation policies**

Affiliation policy	Description
No affiliations	An unlimited number of STP initiator ports are allowed to access the STP target port concurrently. The STP target port is cognizant of the SAS address of the STP initiator port that sends each ATA command.
Multiple affiliations	The STP target port implements more than one affiliation, so a limited number of STP initiator ports are allowed to access the STP target port concurrently. The STP target port implements no more than one affiliation context per STP initiator port.
Single affiliation	The STP target port implements one affiliation, so one STP initiator port is allowed to access the STP target port at a time.

An STP SATA bridge that supports either no affiliations or multiple affiliations shall:

- ensure that the SATA NCQ tags in commands issued to the SATA device are unique across all affiliations;
- ensure that a non-queued command received in one affiliation context is not issued to the SATA device while another affiliation context has a queued command outstanding to the SATA device (e.g., the STP target port shall allow all queued commands in the SATA device to complete prior to issuing the non-queued command);
- ensure that a non-queued command received in one affiliation context is not issued to the SATA device while another affiliation context has a non-queued command outstanding to the SATA device (e.g., the STP target port shall allow the non-queued command in the SATA device to complete prior to issuing the new non-queued command); and
- ensure that a queued command received in one affiliation context is not issued to the SATA device while another affiliation context has a non-queued command outstanding to the SATA device (e.g., the STP target port shall allow any non-queued command in the SATA device to complete prior to issuing the queued commands).

An STP SATA bridge that supports multiple affiliations may modify the queue depth reported in the ATA IDENTIFY DEVICE data (see ACS-4) to each STP initiator port to ensure that all the STP initiator ports with affiliations do not send more commands than the SATA device supports.

An STP target port that supports affiliations shall establish an affiliation whenever it accepts a connection request from an STP initiator port that does not already have an affiliation. While all affiliation contexts are in use, 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:

- power on;
- the management device server receives an SMP PHY CONTROL request specifying the phy with the affiliation and specifying a phy operation of HARD RESET (see 9.4.3.28) from any SMP initiator port;
- the management device server receives an SMP PHY CONTROL request specifying the phy with the affiliation and specifying a phy operation of TRANSMIT SATA PORT SELECTION SIGNAL (see 9.4.3.28) from any SMP initiator port;
- the management device server receives an SMP PHY CONTROL request specifying the phy with the affiliation and specifying a phy operation of CLEAR AFFILIATION (see 9.4.3.28) from the same SAS initiator port that has the affiliation;
- an STP connection to a phy in the STP target port is closed with a CLOSE (CLEAR AFFILIATION) primitive sequence; or
- the STP target port encounters an I\_T nexus loss.



The STP initiator port shall maintain an affiliation starting with the connection in which a command is transmitted until all frames for the command have been delivered. An STP initiator port implementing command queuing (see ACS-4 and SATA) shall maintain an affiliation while any commands are outstanding. STP initiator ports should not keep affiliations while commands are not outstanding.

An STP target port that implements affiliations shall implement at least one affiliation context per STP target port. Multiple phys on the same STP target port shall use the same set of affiliation contexts. Support for affiliations is indicated in the SMP REPORT PHY SATA response (see 9.4.3.12).

An STP target port implementing multiple affiliations shall sort the affiliation contexts in a vendor specific order. In the SMP REPORT PHY SATA response, if the SMP initiator port has the same SAS address as an affiliated STP initiator port, then the management device server shall report the affiliation for that SAS address as relative identifier 0 and shall report all additional affiliations with incrementing relative identifiers following the sorted order. If the SMP initiator port does not have the same SAS address as an affiliated STP initiator port, then the management device server shall report the affiliation contexts in the vendor specific order.

For example, if the STP target port supports four affiliation contexts sorted in order A, B, C, and D, when returning the SMP REPORT PHY SATA response to an SMP initiator port, then the management device server reports the affiliation contexts as described in table 201.

**Table 201 – Affiliation context relative identifier example**

Affiliation context containing the SAS address of the SMP initiator port	Affiliation context relative identifier assignment			
	0	1	2	3
A	A	B	C	D
B	B	C	D	A
C	C	D	A	B
D	D	A	B	C
None	A	B	C	D

#### 6.21.7 Opening an STP connection

When the SATA host port in an STP SATA bridge receives a SATA\_X\_RDY from the attached SATA device, the STP target port in the STP SATA bridge shall establish an STP connection to the appropriate STP initiator port. If there is no affiliation, then the SATA host port may either:

- perform a link reset on the SATA physical link; or
- wait for an affiliation to be established.

If an STP SATA bridge receives a connection request for a SATA device that has delivered the initial Register – Device to Host FIS in error, then it shall return an OPEN\_REJECT (NO DESTINATION).

If there is a problem receiving the expected initial Register - Device to Host FIS, then the STP SATA bridge should use SATA\_R\_ERR to retry until the FIS is received without error. In the DISCOVER response, the ATTACHED SATA DEVICE bit is set to one and the ATTACHED SAS ADDRESS field is valid, but the ATTACHED SAS DEVICE TYPE field is set to 000b (i.e., no device attached) during this time.

If an STP SATA bridge that retrieves IDENTIFY DEVICE data (see ACS-4) receives a connection request for a SATA device before it has retrieved the IDENTIFY DEVICE data, then it shall return an OPEN\_REJECT (NO DESTINATION). If the STP SATA bridge has a problem retrieving the IDENTIFY DEVICE data (e.g., word 255 (i.e., the Integrity Word) is not correct), then it shall set the ATTACHED DEVICE NAME field to zero, set the ATTACHED SAS DEVICE TYPE field to 001b (i.e., end device), and start accepting connections.

A wide STP initiator port shall not request more than one connection at a time to a specific STP target port.

While a wide STP initiator port is waiting for a response to a connection request to an STP target port, a SAS phy in the STP initiator port shall not reject an incoming connection request from that STP target port with OPEN\_REJECT (RETRY) because the SAS port containing that SAS phy is waiting for an outgoing connection request to be accepted. The SAS phy may reject an incoming connection request from that STP target port with OPEN\_REJECT (RETRY) for any reason that is not dependent on the SAS port containing that SAS phy having an outgoing connection request accepted (e.g., because of a temporary buffer full condition).

If a wide STP initiator port receives an incoming connection request from an STP target port while it has a connection established with that STP target port, then the wide STP initiator port shall reject the request with OPEN\_REJECT (RETRY).

A wide STP target port shall not request more than one connection at a time to a specific 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, the wide STP target port shall:

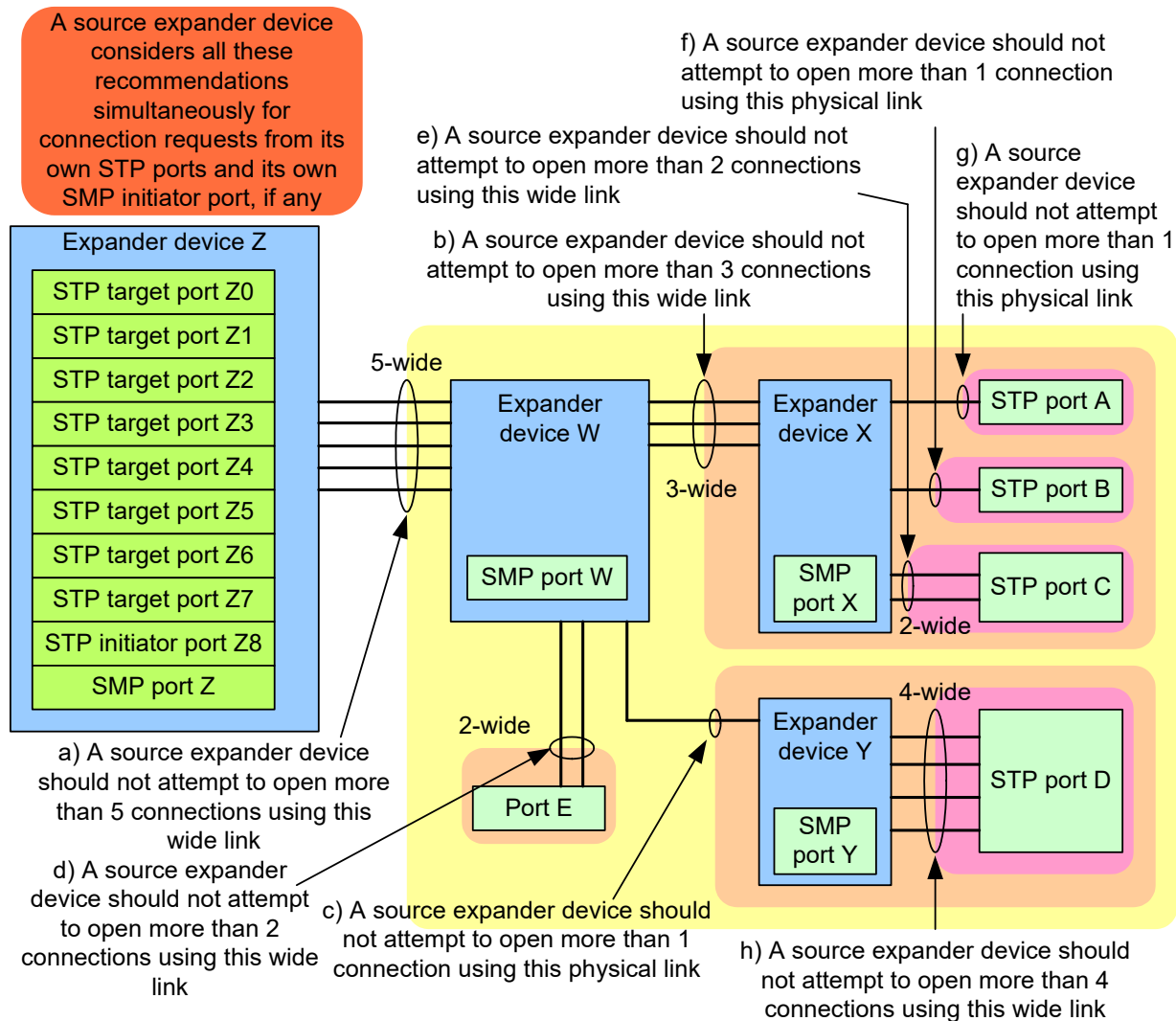
- a) reject incoming connection requests from that STP initiator port with OPEN\_REJECT (RETRY); and
- b) if affiliations are supported and the maximum number of affiliations has been established (i.e., all affiliation contexts are in use), then reject incoming connection requests from other STP initiator ports that do not have affiliations with OPEN\_REJECT (STP RESOURCES BUSY).

A SAS phy may reject an incoming connection request (i.e., an OPEN address frame) to an STP target port with OPEN\_REJECT (RETRY) for any reason, including because the SAS port containing that SAS phy is waiting for an outgoing connection request to be accepted (e.g., to transmit a frame and empty a buffer).

An expander device should not allow its STP ports (e.g., the STP target ports in STP SATA bridges and any STP initiator ports in the expander device) to attempt to establish more connections to a specific destination port than the destination port width or the width of the narrowest physical link on the pathway to the destination port. This does not apply to connection requests being forwarded by the expander device.

An expander device should not allow its STP ports (e.g., the STP target ports in STP SATA bridges and any STP initiator ports in the expander device) to attempt to establish more connections than the width of the narrowest common physical link on the pathways to the destination ports of those connections. This does not apply to connection requests being forwarded by the expander device.

Figure 186 shows an example of the simultaneous connection recommendations for an expander device containing STP ports. Multiplexing is disabled in this example.



**Figure 186 –Example simultaneous connection recommendations for an expander device**

In figure 186, some of the recommendations are combined as follows:

- a) recommendations a), b), and e) together specify that expander device Z should not attempt to open more than 2 connections to port C;
- b) recommendations a), b), e), f), and g) together specify that if expander device Z has two connections open to ports A, B, and X, then expander device Z should not attempt to open more than one connection to port C. If expander device Z has four connections open to ports A, B, D, E, W, X, and Y, then expander device Z should not attempt to open more than one connection to port C; and
- c) recommendations a), c), and h) together specify that expander device Z should not attempt to open more than one connection to port D. If expander device Z has a connection open to port Y, then expander device Z should not attempt to open another connection to port D until the first connection is closed.

The first dword that an STP phy sends inside an STP connection after OPEN\_ACCEPT that is not a deletable primitive shall be an STP primitive (e.g., SATA\_SYNC).

### 6.21.8 Closing an STP connection

Either STP port (i.e., either the STP initiator port or the STP target port) may originate closing an STP connection. An STP port shall not originate closing an STP connection after sending a SATA\_X\_RDY or SATA\_R\_RDY until after both sending and receiving SATA\_SYNC. An STP port shall transmit a CLOSE primitive sequence after receiving a CLOSE primitive sequence if it has not already transmitted a CLOSE primitive sequence.

If an STP port receives a CLOSE primitive sequence after transmitting a SATA\_X\_RDY but before receiving a SATA\_R\_RDY, then the STP port shall complete closing the connection (i.e., transmit a CLOSE primitive sequence) and retransmit the SATA\_X\_RDY in a new connection.

When an STP initiator port closes an STP connection, it shall transmit a CLOSE (NORMAL) primitive sequence or CLOSE (CLEAR AFFILIATION) primitive sequence. When an STP target port closes an STP connection, it shall transmit a CLOSE (NORMAL) primitive sequence.

An STP initiator port may issue a CLOSE (CLEAR AFFILIATION) primitive sequence in place of a CLOSE (NORMAL) primitive sequence to cause the STP target port to clear the affiliation (see 6.21.6) along with closing the connection. If an STP target port receives a CLOSE (CLEAR AFFILIATION) primitive sequence, then the STP target port shall clear the affiliation for the STP initiator port from which that CLOSE (CLEAR AFFILIATION) primitive sequence was received.

See 6.16.9 for additional details on closing connections.

An STP SATA bridge shall break an STP connection if its SATA host phy loses dword synchronization (see 6.16.11).

### 6.21.9 STP connection management examples

The STP SATA bridge adds the outgoing OPEN address frames and CLOSEs so the STP initiator port sees an STP target port. The STP SATA bridge removes incoming OPEN address frame and CLOSEs so the SATA device port sees only a SATA host port. While the connection is open, the STP SATA bridge passes through all dwords without modification. Both STP initiator port and STP target port use SATA, with SATA flow control (see 6.21.4), while the connection is open.

Figure 187 shows an STP initiator port opening a connection when SAS dword mode is enabled, transmitting a single SATA frame, and closing the connection.

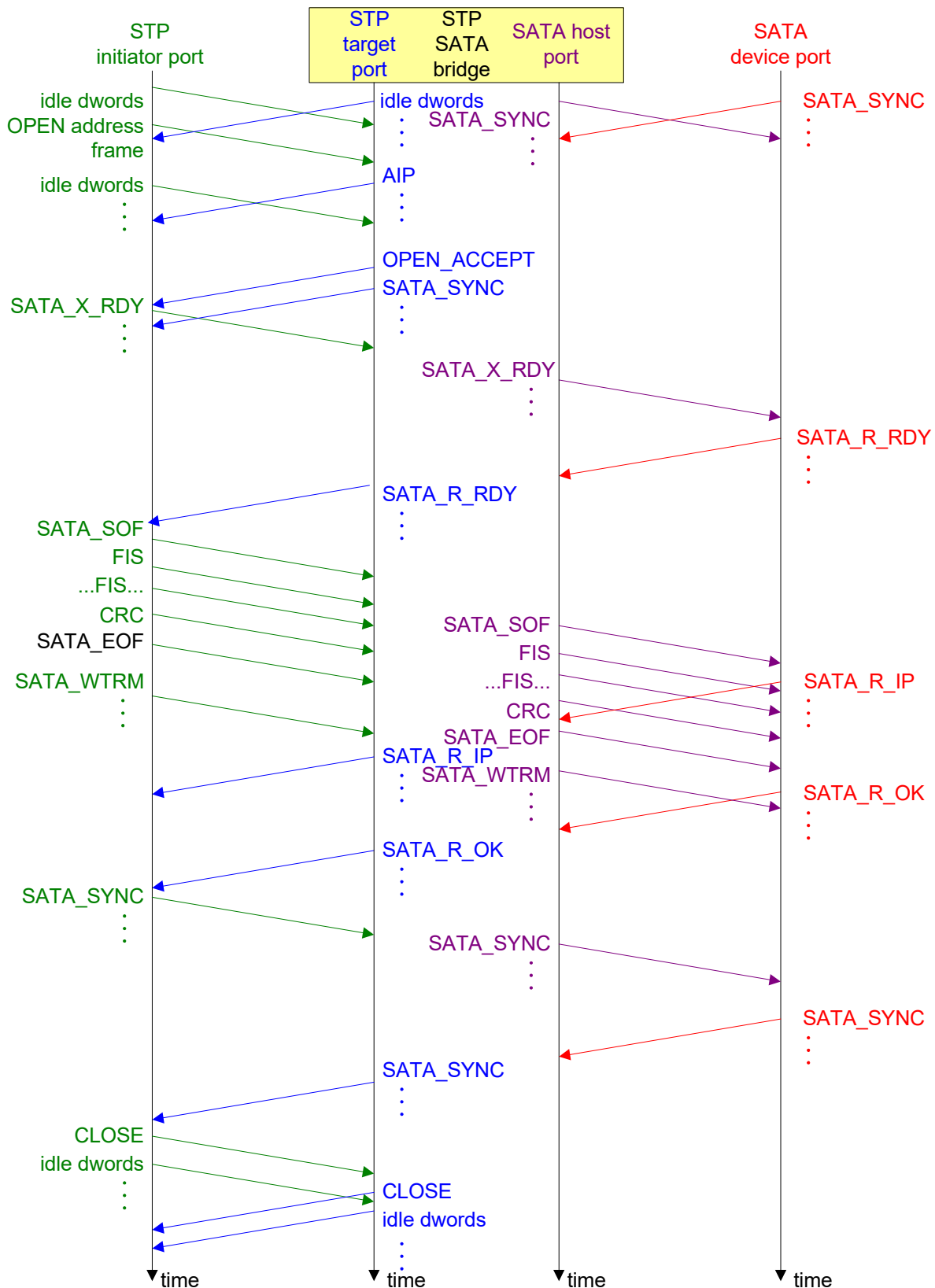


Figure 187 –STP initiator port opening an STP connection while SAS dword mode is enabled

Figure 188 shows a SATA device transmitting a SATA frame when SAS dword mode is enabled. In this example, the STP target port in the STP SATA bridge opens a connection to an STP initiator port to send just one frame, then closes the connection.

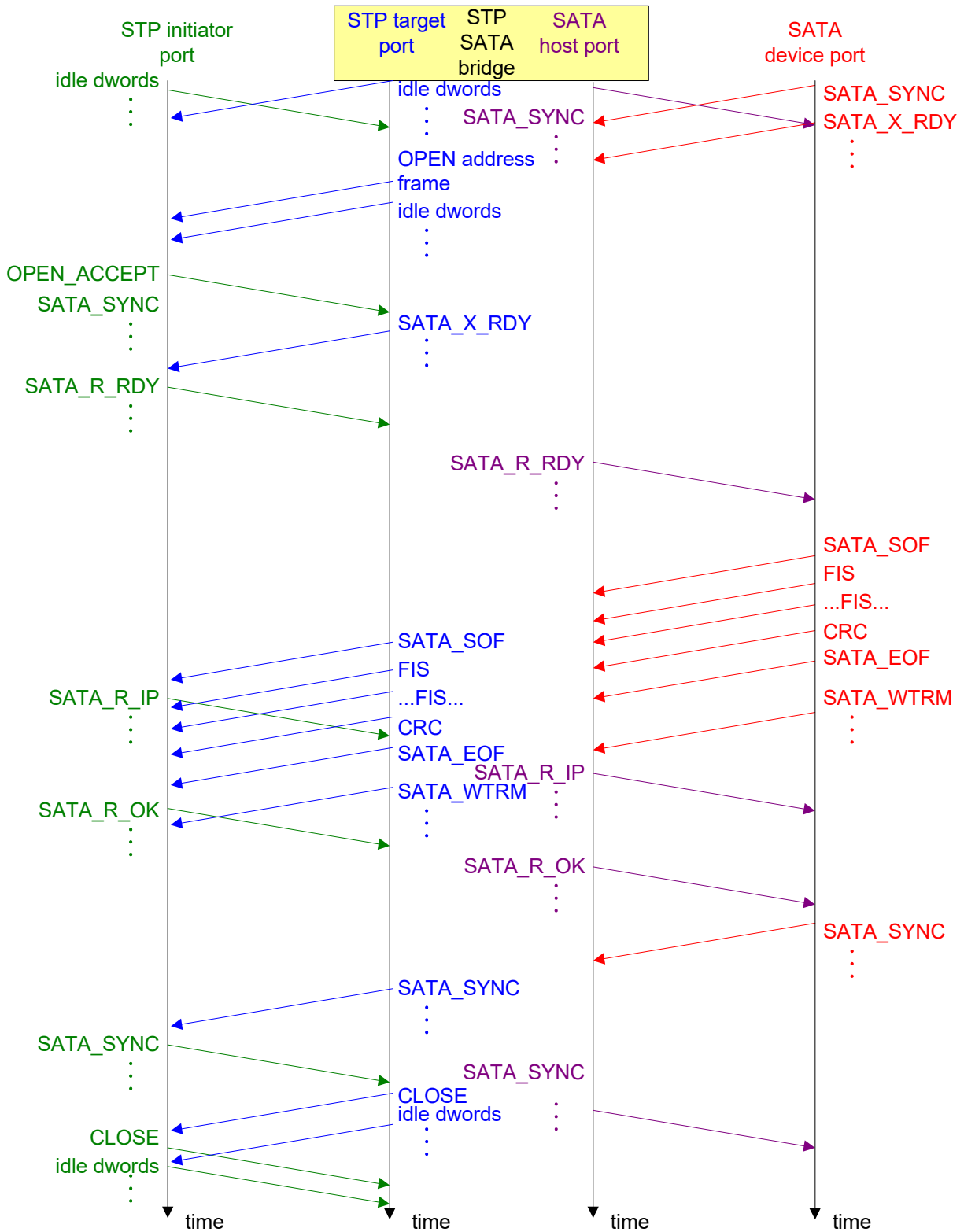


Figure 188 –STP target port opening an STP connection while SAS dword mode is enabled

### 6.21.10 STP (link layer for STP phys) state machines

The STP link layer uses the SATA link layer state machines (see SATA), 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 and the first SATA primitive);
- c) communicate with an STP transmitter and receiver; and
- d) support an affiliation policy (see 6.21.6).

These modifications are not described in this standard.

The STP transmitter relationship to other transmitters is defined in 4.3.2. The STP receiver relationship to other receivers is defined in 4.3.3.

### 6.21.11 SMP target port support

A SAS device that contains an STP target port shall also contain an SMP target port.

## 6.22 SMP link layer

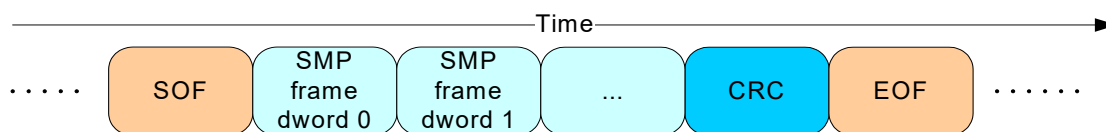
### 6.22.1 SMP frame transmission and reception

Inside an SMP connection, the SMP initiator phy transmits a single SMP\_REQUEST frame within 100  $\mu$ s and the SMP target phy responds with a single SMP\_RESPONSE frame (see 8.4) within 1 900  $\mu$ s.

#### 6.22.1.1 SMP frame transmission and reception while in SAS dword mode

While in SAS dword mode, frames are surrounded by SOF and EOF as shown in figure 189. See 6.22.6 for error handling details.

NOTE 50 - Unlike SSP, there is no acknowledgement of SMP frames with ACK and NAK and there is no frame credit exchange with RRDY.

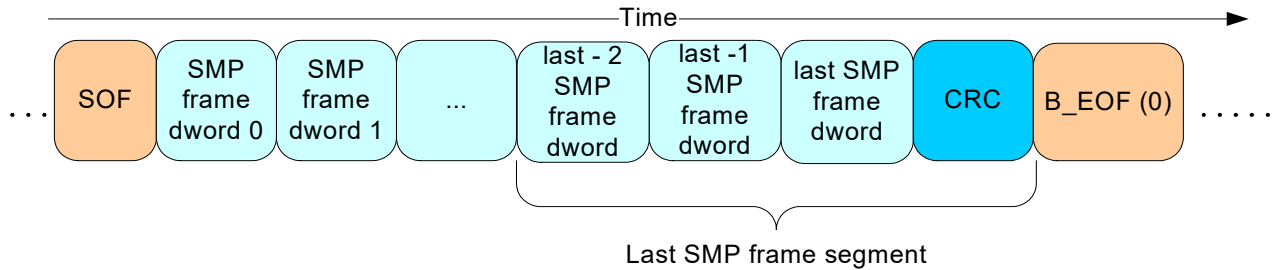


**Figure 189 –SMP frame transmission while in the SAS dword mode**

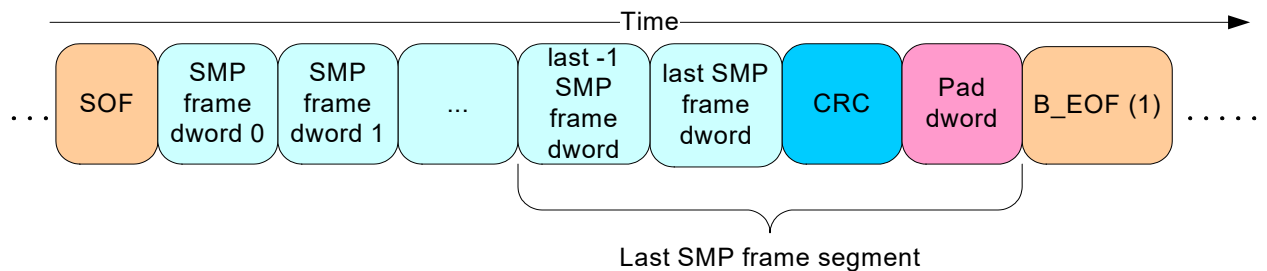
The last data dword after the SOF prior to the EOF always contains a CRC (see 6.7). The SMP link layer state machine checks that the frame is not too short and that the CRC is valid (see 6.22.6).

### 6.22.2 SMP frame transmission and reception while in the SAS packet mode

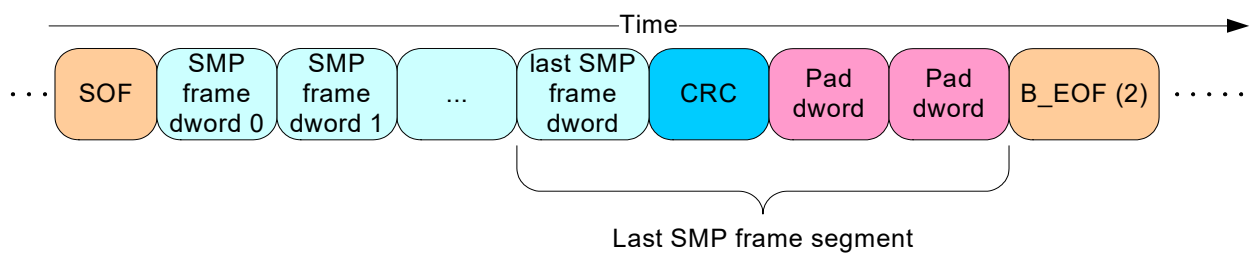
During an SMP connection while in SAS packet mode, SMP frames are preceded by SOF and followed by B\_EOF (0), B\_EOF (1), B\_EOF (2), or B\_EOF (3) as shown in figure 190, figure 191, figure 192, and figure 193.



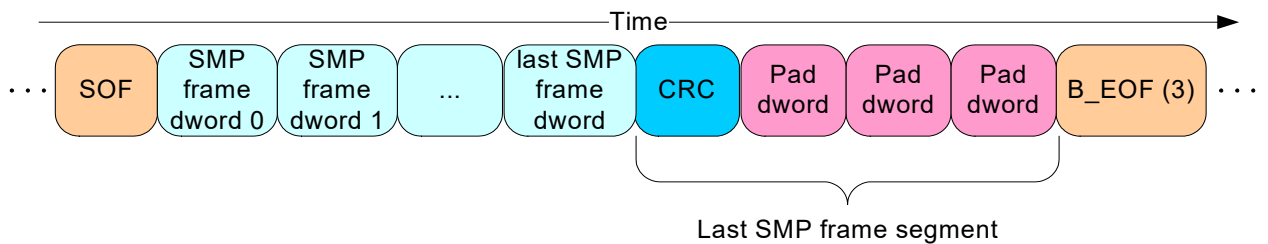
**Figure 190 –SMP frame transmission with no pad dword**



**Figure 191 –SMP frame transmission with one pad dword**



**Figure 192 –SMP frame transmission with two pad dword**



**Figure 193 –SMP frame transmission with three pad dword**



The requirements for placement of the CRC, pad dwords, B\_EOF (0), B\_EOF (1), B\_EOF (2), and B\_EOF (3) while processing SMP frame transmission and reception in SAS packet mode are as described in 6.20.3.4.

### 6.22.3 SMP flow control

By accepting an SMP connection, the SMP target phy indicates it is ready to receive one SMP\_REQUEST frame.

After the SMP initiator phy transmits one SMP\_REQUEST frame, it shall be ready to receive one SMP\_RESPONSE frame.

### 6.22.4 Opening an SMP connection

An SMP target port shall not attempt to establish an SMP connection.

A SAS phy may reject an incoming connection request (i.e., OPEN address frame) to an SMP target port with OPEN\_REJECT (RETRY) for any reason, including because the SAS port containing that SAS phy is waiting for an outgoing connection request to be accepted (e.g., to transmit a frame and empty a buffer).

### 6.22.5 Closing an SMP connection

After receiving the SMP\_RESPONSE frame, the SMP initiator phy shall transmit a CLOSE (NORMAL) primitive sequence to close the connection.

After transmitting the SMP\_RESPONSE frame, the SMP target phy shall reply with a CLOSE (NORMAL) primitive sequence.

See 6.16.9 for additional details on closing connections.

### 6.22.6 SMP (link layer for SMP phys) state machines

#### 6.22.6.1 SMP state machines overview

The SMP state machines control the flow of dwords on the physical link during an SMP connection. The SMP state machines are as follows:

- a) SMP\_IP (link layer for SMP initiator phys) state machine (see 6.22.6.3); and
- b) SMP\_TP (link layer for SMP target phys) state machine (see 6.22.6.4).

#### 6.22.6.2 SMP transmitter and receiver

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

- a) Transmit Idle Dword; and
- b) Transmit Frame with an argument containing the frame contents.

If SAS dword mode is enabled, then in response to the Transmit Frame message, the SMP transmitter transmits:

- 1) SOF;
- 2) the frame contents;
- 3) CRC; and
- 4) EOF.

If SAS packet mode is enabled, then in response to the Transmit Frame message, the SMP transmitter transmits:

- 1) SOF;
- 2) the frame contents;
- 3) CRC;
- 4) pad dwords, if any, as described in 6.20.3.3; and

- 5) B\_EOF (0), B\_EOF (1), B\_EOF (2), or B\_EOF (3) as described in 6.20.3.3.

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

- a) Frame Transmitted.

While 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 from the SP\_DWS receiver (see 5.15.2) or SP\_PS receiver (see 5.16.2):

- a) SOF Received;
- b) Data Dword Received;
- c) EOF Received;
- d) B\_EOF Received;
- e) ERROR Received; and
- f) Invalid Dword Received.

The SMP receiver shall ignore:

- a) pad dwords, if any, received before a B\_EOF as described in 6.20.3.3; and
- b) all dwords not described in this subclause.

The SMP transmitter relationship to other transmitters is defined in 4.3.2. The SMP receiver relationship to other receivers is defined in 4.3.3.

### **6.22.6.3 SMP\_IP (link layer for SMP initiator phys) state machine**

#### **6.22.6.3.1 SMP\_IP state machine overview**

The SMP\_IP state machine's function is to transmit an SMP request frame and then receive the corresponding response frame. This state machine consists of the following states:

- a) SMP\_IP1:Idle (see 6.22.6.3.2) (initial state);
- b) SMP\_IP2:Transmit\_Frame (see 6.22.6.3.3); and
- c) SMP\_IP3:Receive\_Frame (see 6.22.6.3.4).

This state machine shall start in the SMP\_IP1:Idle state on receipt of an Enable Disable SMP (Enable) message from the SL state machines (see 6.18).

The SMP\_IP state machine shall terminate after receiving an Enable Disable SMP (Disable) message from the SL state machines.

Figure 194 shows the SMP\_IP state machine.

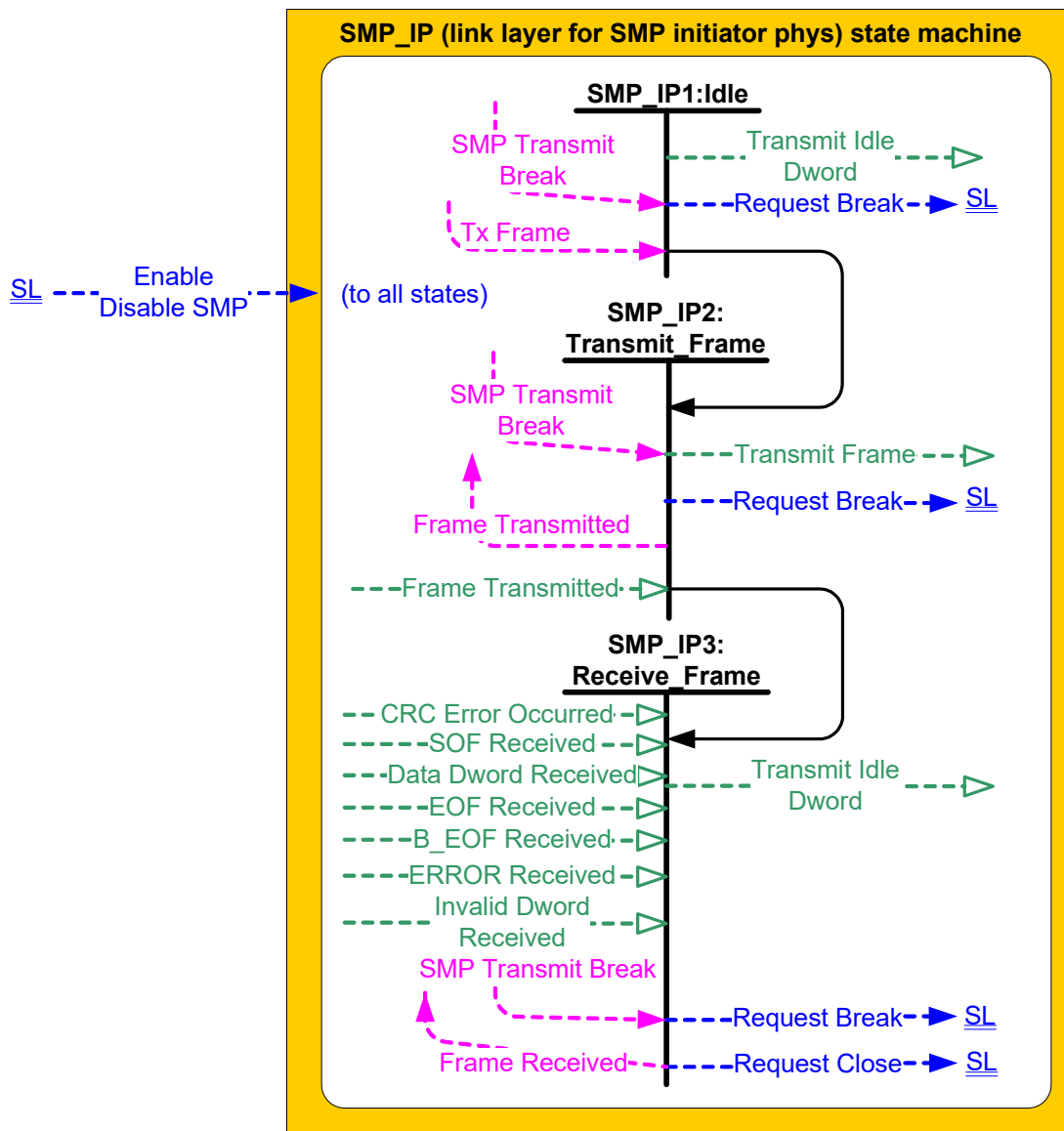


Figure 194 –SMP\_IP (link layer for SMP initiator phys) state machine

#### 6.22.6.3.2 SMP\_IP1:Idle state

##### 6.22.6.3.2.1 State description

This state is the initial state.

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

If an SMP Transmit Break request is received, then this state shall send a Request Break message to the SL state machines (see 6.18).

**6.22.6.3.2.2 Transition SMP\_IP1:Idle to SMP\_IP2:Transmit\_Frame**

This transition shall occur:

- a) after a Tx Frame request is received.

**6.22.6.3.3 SMP\_IP2:Transmit\_Frame state****6.22.6.3.3.1 State description**

This state shall send a Transmit Frame message to the SMP transmitter with an argument containing the frame contents.

If an SMP Transmit Break request is received, then this state shall send a Request Break message to the SL state machines (see 6.18) and terminate this state machine.

After the Frame Transmitted message is received, this state shall send a Frame Transmitted confirmation to the port layer.

**6.22.6.3.3.2 Transition SMP\_IP2:Transmit\_Frame to SMP\_IP3:Receive\_Frame**

This transition shall occur:

- a) after sending a Frame Transmitted confirmation to the port layer.

**6.22.6.3.4 SMP\_IP3:Receive\_Frame state**

This state checks the SMP response frame and determines if the SMP response frame was received without error (e.g., no CRC error).

If this state receives a subsequent SOF Received message after receiving an SOF Received message but before receiving an EOF Received message or B\_EOF Received message (e.g., SOF, data dwords, SOF, data dwords, and EOF instead of SOF, data dwords, EOF, SOF, data dwords, and EOF), then this state shall discard the frame in progress.

This state shall discard the frame, send a Frame Received (SMP Unsuccessful) confirmation to the port layer, send a Request Break message to the SL state machines, and terminate this state machine if:

- a) this state receives more than 257 Data Dword Received messages (i.e., 1 028 bytes) after an SOF Received message and before an EOF Received message or B\_EOF Received message; or

NOTE 51 - SMP target phys compliant with SAS 1.1 is allowed to send vendor specific SMP frames containing 258 data dwords (i.e., 1 032 bytes).

- b) this state receives fewer than two Data Dword Received messages (i.e., 8 bytes) after an SOF Received message and before an EOF Received message or B\_EOF Received message.

If this state receives an Invalid Dword Received message or an ERROR Received message after an SOF Received message and before an EOF Received message, then this state machine shall either:

- a) ignore the invalid dword or ERROR; or
- b) discard the frame, send a Frame Received (SMP Unsuccessful) confirmation to the port layer, send a Request Break message to the SL state machines, and terminate this state machine.

If a CRC Error Occurred message was received for the SMP response frame, then this state shall discard the SMP response frame, send a Frame Received (SMP Unsuccessful) confirmation to the port layer, send a Request Break message to the SL state machines, and terminate this state machine.

If no CRC Error Occurred message was received for the SMP response frame and the SMP response frame is valid, then this state shall send:

- a) a Frame Received (SMP Successful) confirmation to the port layer; and
- b) a Request Close message to the SL state machines (see 6.18).

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

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

#### 6.22.6.4 SMP\_TP (link layer for SMP target phys) state machine

##### 6.22.6.4.1 SMP\_TP state machine overview

The SMP\_TP state machine's function is to receive an SMP request frame and then transmit the corresponding SMP response frame. The SMP\_TP state machine consists of the following states:

- SMP\_TP1:Receive\_Frame (see 6.22.6.4.2) (initial state); and
- SMP\_TP2:Transmit\_Frame (see 6.22.6.4.3).

This state machine shall start in the SMP\_TP1:Receive\_Frame state after receiving an Enable Disable SMP (Enable) message from the SL state machines (see 6.18).

This state machine shall terminate after receiving an Enable Disable SMP (Disable) message from the SL state machines.

Figure 195 shows the SMP\_TP state machine.

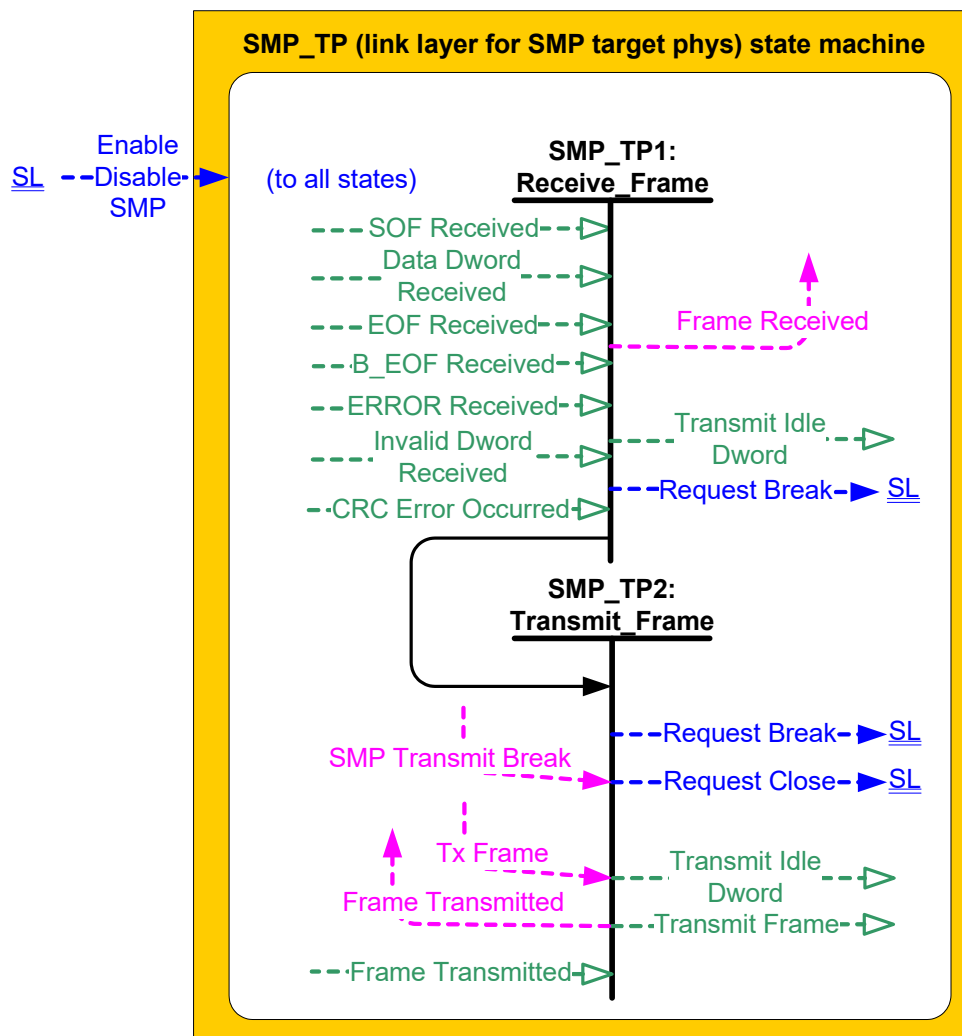


Figure 195 –SMP\_TP (link layer for SMP target phys) state machine

**6.22.6.4.2 SMP\_TP1:Receive\_Frame state****6.22.6.4.2.1 State description**

This state waits for an SMP frame and determines if the SMP frame was received without error (e.g., no CRC error).

If this state receives a subsequent SOF Received message after receiving an SOF Received message but before receiving an EOF Received message or a B\_EOF Received message (e.g., SOF, data dwords, SOF, data dwords, and EOF instead of SOF, data dwords, EOF, SOF, data dwords, and EOF), then this state shall discard the frame in progress.

This state shall discard the frame, send a Request Break message to the SL state machines (see 6.18) and shall terminate this state machine if:

- a) this state receives more than 257 Data Dword Received messages (i.e., 1 028 bytes) after an SOF Received message and before an EOF Received message or a B\_EOF Received message; or

NOTE 52 - SMP initiator phys compliant with SAS 1.1 is allowed to send vendor specific SMP frames containing 258 data dwords (i.e., 1 032 bytes).

- b) this state receives fewer than two Data Dword Received messages (i.e., 8 bytes) after an SOF Received message and before an EOF Received message or a B\_EOF Received message.

If this state receives an Invalid Dword Received message or an ERROR Received message after an SOF Received message and before an EOF Received message or a B\_EOF Received message, then this state machine shall either:

- a) ignore the invalid dword or ERROR; or
- b) discard the frame, send a Request Break message to the SL state machines (see 6.18) and shall terminate this state machine.

If a CRC Error Occurred message was received for the SMP request frame, then this state shall discard the SMP request frame, send a Request Break message to the SL state machines (see 6.18) and shall terminate this state machine, otherwise this state shall send a Frame Received (SMP Successful) confirmation to the port layer.

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

**6.22.6.4.2.2 Transition SMP\_TP1:Receive\_Frame to SMP\_TP2:Transmit\_Frame**

This transition shall occur after sending a Frame Received (SMP Successful) confirmation to the port layer.

**6.22.6.4.3 SMP\_TP2:Transmit\_Frame state**

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

If this state receives a Tx Frame request, then this state shall send a Transmit Frame message to the SMP transmitter with an argument containing the frame contents, then wait for a Frame Transmitted message. After receiving a Frame Transmitted message, this state shall send a Frame Transmitted confirmation to the port layer, send a Request Close message to the SL state machines (see 6.18) and terminate this state machine.

After sending Transmit Frame message to the SMP transmitter, this state shall request that idle dwords be transmitted by repeatedly sending Transmit Idle Dword messages to the SMP transmitter.

## 7 Port layer

### 7.1 Port layer overview

The port layer state machines interface with one or more SAS link layer state machines and one or more SSP, SMP, and STP transport layer state machines to establish port connections and disconnections. The port layer state machines also interpret or pass transmit data, receive data, commands, and confirmations between the link layer, transport layer, and the management application layer.

### 7.2 Port layer state machines

#### 7.2.1 Port layer state machines overview

The port layer consists of state machines that run in parallel and perform the following functions:

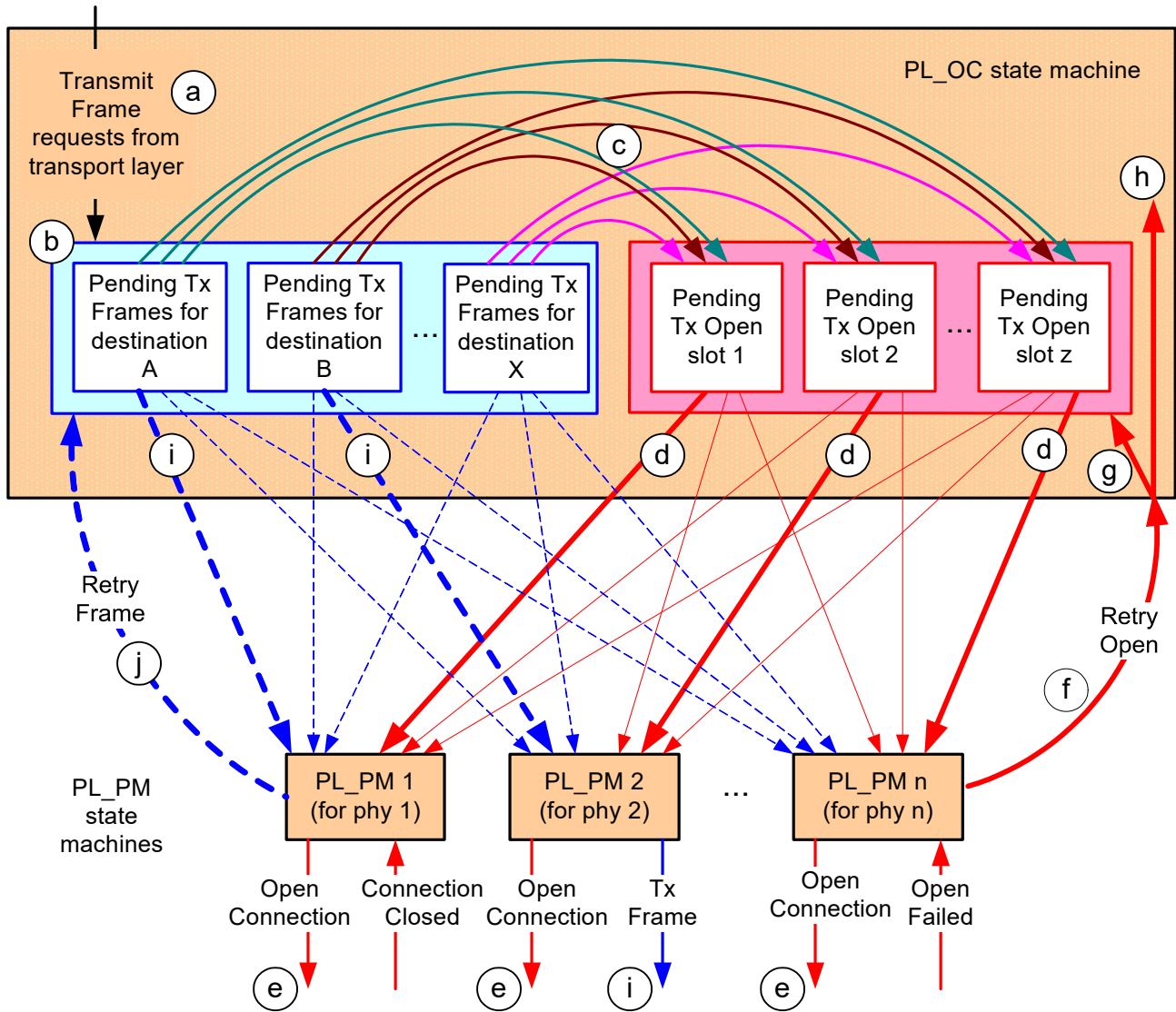
- a) receive requests from the SSP, SMP, and STP transport layer state machines for connection management (e.g., requests to open or close connections) and frame transmission;
- b) receive requests from the management application layer;
- c) send requests to the SAS link layer state machines for connection management and frame transmission;
- d) receive confirmation from the SAS link layer state machines;
- e) send confirmations to the SSP, SMP, and STP transport layer state machines; and
- f) send confirmations to the management application layer.

The port layer state machines are as follows:

- a) PL\_OC (port layer overall control) state machines (see 7.2.2); and
- b) PL\_PM (port layer phy manager) state machines (see 7.2.3).

There is one PL\_OC state machine per port (see 4.1.4). There is one PL\_PM state machine for each phy contained in the port. Phys are assigned to ports by the management application layer. More than one port in a SAS device may have the same SAS address if the ports are in different SAS domains (see 4.2.9).

Figure 196 shows examples of the port layer state machines and their interaction with the transport and link layers.



**Figure 196 – Port layer examples**

The following is a description of the example processes in figure 196. These example processes do not describe all of the possible condition or actions:

- Transmit Frame requests are received by the PL\_OC state machine;
- the PL\_OC state machine converts Transmit Frame requests into pending Tx Frame messages associated with the destination SAS address;
- the PL\_OC state machine generates a pending Tx Open message for a pending Tx Frame message when there is a pending Tx Open slot available (i.e., the number of pending Tx Open messages is less than or equal to the number of destination SAS addresses);
- the PL\_OC state machine sends a pending Tx Open message as a Tx Open message to a PL\_PM state machine when a PL\_PM machine is available, a slot is then available for a new pending Tx Open message;
- when a PL\_PM state machine receives a Tx Open message, the PL\_PM state machine attempts to establish a connection with the destination SAS address through the link layer;



- f) if a PL\_PM state machine is unable to establish a connection with the destination SAS address, then the PL\_PM state machine sends a Retry Open message to the PL\_OC state machine;
- g) if there is a pending Tx Open slot available, then the PL\_OC state machine converts a Retry Open message to a pending Tx Open message with the pathway blocked count and arbitration wait time context from the Retry Open message applied to the pending Tx Open message, and may start the Reject To Open Limit timer;
- h) if the PL\_OC state machine does not convert a Retry Open to a pending Tx Open message, then the PL\_OC discards the Retry Open message. The PL\_OC state machine may create a new Tx Open message for the same pending Tx Frame at a later time or send the appropriate Transmission Status confirmation to the transport layer. If the PL\_OC state machine discards a Retry Open message, then the pathway blocked count and arbitration wait time context from the Retry Open message are also discarded;
- i) after the Reject To Open Limit timer, if any, has expired and after a PL\_PM state machine establishes a connection with a destination SAS address, the PL\_OC state machine sends pending Tx Frame messages for the destination to the PL\_PM state machine as Tx Frame messages;
- j) if a PL\_PM state machine is unable to send a Tx Frame message to the link layer as a Tx Frame request (e.g., due to an SSP frame credit timeout), then the PL\_PM state machine sends a Retry Frame message to the PL\_OC state machine, and the PL\_OC state machine converts the Retry Frame message into a pending Tx Frame message; and
- k) if the PL\_PM state machine is able to send a Tx Frame message as a Tx Frame request to the link layer, then the PL\_PM state machine sends a Transmission Status confirmation to the transport layer.

The Transmission Status confirmation from either the PL\_OC state machine or a PL\_PM state machine shall include the following as arguments:

- a) initiator port transfer tag;
- b) Destination SAS Address; and
- c) Source SAS Address.

## 7.2.2 PL\_OC (port layer overall control) state machine

### 7.2.2.1 PL\_OC state machine overview

The PL\_OC state machine:

- a) receives requests from the SSP, SMP, and STP transport layers;
- b) sends messages to the PL\_PM state machine;
- c) receives messages from the PL\_PM state machine;
- d) receives requests from the management application layer;
- e) selects frames to transmit;
- f) selects phys on which to transmit frames;
- g) receives confirmations from the link layer;
- h) sends confirmations to the transport layer;
- i) sends confirmations to the management application layer;
- j) has Arbitration Wait Time timers;
- k) has I\_T Nexus Loss timers; and
- l) may have Reject To Open Limit timers.

This state machine consists of the following states:

- a) PL\_OC1:Idle (see 7.2.2.2) (initial state); and
- b) PL\_OC2:Overall\_Control (see 7.2.2.3).

This state machine shall start in the PL\_OC1:Idle state after power on.

This state machine shall maintain a pool of pending:

- a) Tx Frame messages for each destination SAS address; and
- b) Tx Open message slots. There shall only be at most a single pending Tx Open message slot for each destination SAS address. There may be fewer total pending Tx Open message slots than the total number of destination SAS addresses.

This state machine shall maintain the timers listed in table 202.

**Table 202 – PL\_OC state machine timers**

Timer	Maximum number of timers	Initial value
I_T Nexus Loss timer	One per destination SAS address	Depending on the protocol used by the port: a) for SSP target ports, the value in the I_T NEXUS LOSS TIME field in the Protocol Specific Port mode page (see 9.2.7.4); b) for SSP initiator ports, the value in the I_T NEXUS LOSS TIME field in the Protocol Specific Port mode page for the SSP target port with that destination SAS address (see 9.2.7.4); c) for STP target ports, the value in the STP SMP I_T NEXUS LOSS TIME field in the SMP CONFIGURE GENERAL function (see 9.4.3.18); d) for STP initiator ports, the value in the STP SMP I_T NEXUS LOSS TIME field in the SMP REPORT GENERAL function (see 9.4.3.4) for the STP target port with that destination SAS address; e) for SMP initiator ports managed by a management device server, the value in the STP SMP I_T NEXUS LOSS TIME field in the SMP CONFIGURE GENERAL function (see 9.4.3.18); or f) for SMP initiator ports not managed by a management device server, the value in the STP SMP I_T NEXUS LOSS TIME field in the SMP REPORT GENERAL function (see 9.4.3.4) for the SMP target port with that destination SAS address.
Arbitration Wait Time timer	One per pending Tx Open message	0000h, a vendor specific value less than 8000h (see 6.16.4) or the value received with a Retry Open message.
Reject To Open Limit timer	One per Retry Open message	If an OPEN_REJECT retry-class primitive is received with an OPEN_REJECT retry-class primitive parameter and the OPEN_RETRY_DELAY field in the OPEN_REJECT retry-class primitive parameter is not equal to zero, then the value in the OPEN_RETRY_DELAY field in the OPEN_REJECT retry-class primitive parameter (see 6.2.6.10.2). Otherwise depending on the protocol used by the port: a) for SSP target ports, the value in the REJECT TO OPEN LIMIT field in the Protocol Specific Port mode page (see 9.2.7.4); b) for SSP initiator ports, a vendor specific value; c) for STP target ports, the value in the STP REJECT TO OPEN LIMIT field in the SMP CONFIGURE GENERAL function (see 9.4.3.18); or d) for STP initiator ports, a vendor specific value.

Figure 197 shows the PL\_OC state machine.

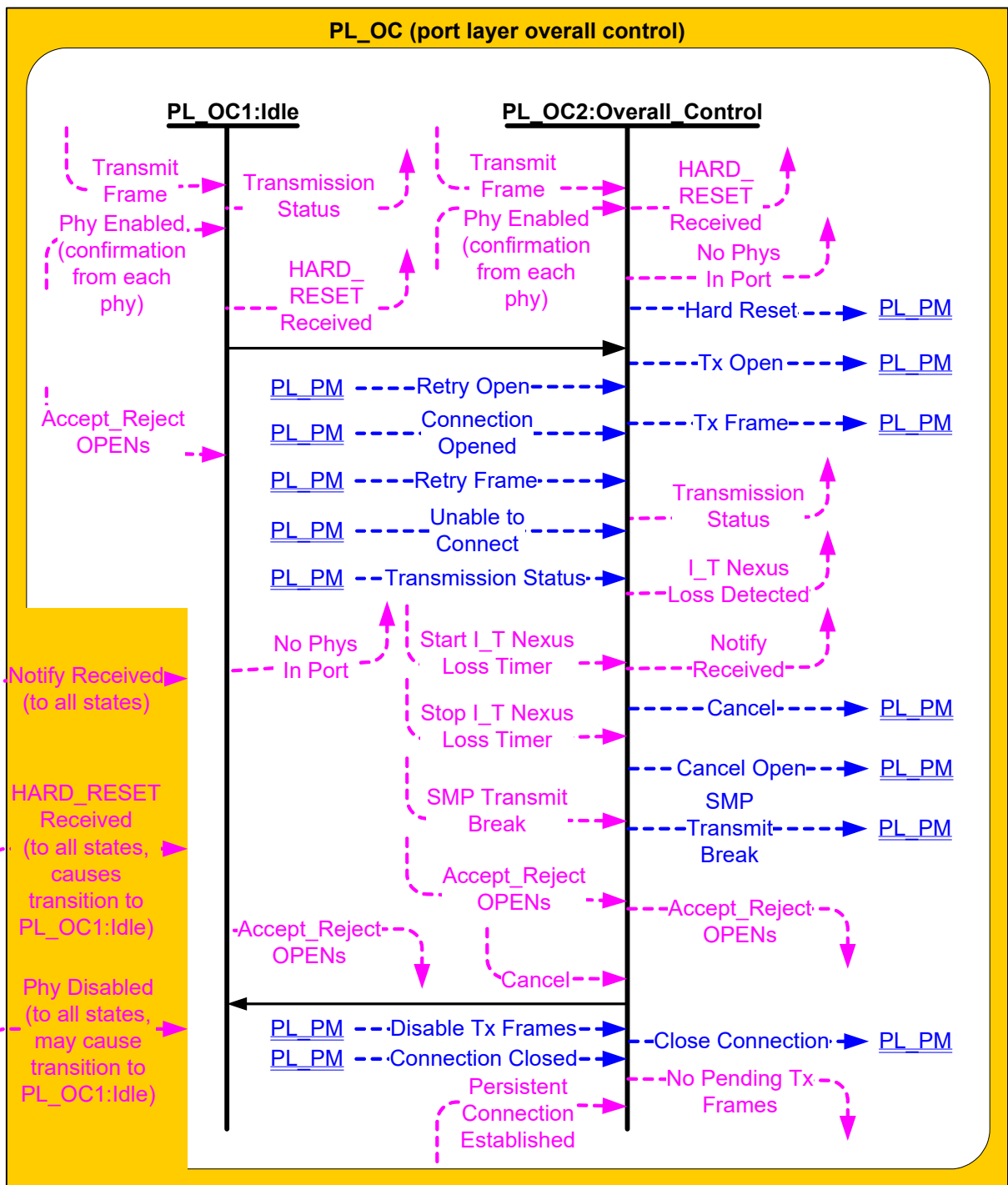


Figure 197 – PL\_OC (port layer overall control) state machine

#### 7.2.2.2 PL\_OC1:Idle state

##### 7.2.2.2.1 PL\_OC1:Idle state description

This state is the initial state of the PL\_OC state machine.

If this state receives a HARD\_RESET Received confirmation, then this state shall send a HARD\_RESET Received confirmation to the transport layer.

If this state receives a NOTIFY Received (Power Loss Expected) confirmation, then this state shall send a NOTIFY Received (Power Loss Expected) confirmation to the transport layer.

If this state receives an Accept\_Reject OPENs request, then this state shall send an Accept\_Reject OPENs request with the same arguments to all link layers in the port.

If this state receives a Transmit Frame request, then this state shall send a No Phys In Port confirmation to the transport layer.

If the port is an STP target port or an STP initiator port, then the port shall handle all pending commands as described in 4.4.3.

#### **7.2.2.2.2 Transition PL\_OC1:Idle to PL\_OC2:Overall\_Control**

This transition shall occur after a Phy Enabled confirmation is received for at least one phy assigned to the port.

#### **7.2.2.3 PL\_OC2:Overall\_Control state**

##### **7.2.2.3.1 PL\_OC2:Overall\_Control state overview**

This state may receive Transmit Frame requests from the transport layers (i.e., SSP and SMP) and Retry Frame messages from PL\_PM state machines. This state shall create a pending Tx Frame message for each received Transmit Frame request and Retry Frame message. There may be more than one pending Tx Frame message at a time for each SSP transport layer. There shall be only one pending Tx Frame message at a time for each SMP transport layer.

This state selects PL\_PM state machines through which connections are established. This state shall only attempt to establish connections through PL\_PM state machines whose phys are enabled. In a vendor specific manner, this state selects PL\_PM state machines on which connections are established to transmit frames. This state shall receive a response to a message from a PL\_PM state machine before sending another message to that PL\_PM state machine.

This state also:

- a) receives connection management requests from the transport layers;
- b) sends connection management messages to PL\_PM state machines;
- c) receives connection management messages from PL\_PM state machines;
- d) sends connection management confirmations to the transport layers;
- e) receives requests from the management application layer; and
- f) sends confirmations to the management application layer.

##### **7.2.2.3.2 PL\_OC2: Non-connection specific confirmations and requests**

###### **7.2.2.3.2.1 PL\_OC2: Transmit Frame request**

After receiving a Transmit Frame request for a destination SAS address for which there is no connection established and for which no I\_T Nexus Loss timer has been created, this state shall create an I\_T Nexus Loss timer for that SAS address if the protocol is:

- a) SSP, the port is an SSP target port, the Protocol Specific Port mode page is implemented, and the I\_T NEXUS LOSS TIME field in the Protocol Specific Port mode page (see 9.2.7.4) is not set to 0000h;
- b) STP, the port is an STP target port, and the STP SMP I\_T NEXUS LOSS TIME field in the SMP CONFIGURE GENERAL function is not set to 0000h; or
- c) SMP, the port is an SMP initiator port, and the STP SMP I\_T NEXUS LOSS TIME field in the SMP CONFIGURE GENERAL function is not set to 0000h.

After receiving a Transmit Frame request for a destination SAS address for which there is no connection established and for which no I\_T Nexus Loss timer has been created this state may create an I\_T Nexus Loss timer for that SAS address if the protocol is:

- a) SSP and the port is an SSP initiator port; or
- b) STP and the port is an STP initiator port.

If this state creates an I\_T Nexus Loss timer after receiving a Transmit Frame request for a destination SAS address, then this state shall:

- 1) initialize that I\_T Nexus Loss timer as specified in table 202 (see 7.2.2.1); and
- 2) not start that I\_T Nexus Loss timer.

If there are no pending Tx Frame messages for a destination SAS address and an I\_T Nexus Loss timer has been created for that destination SAS address, then this state shall delete the I\_T Nexus Loss timer for that destination SAS address.

#### **7.2.2.3.2.2 PL\_OC2: HARD\_RESET Received confirmation**

If this state receives a HARD\_RESET Received confirmation, then this state shall:

- a) discard all pending Tx Frame messages;
- b) discard all pending Tx Open messages;
- c) delete all timers (e.g., I\_T Nexus Loss timers, Arbitration Wait Time timers, and Reject To Open Limit timers);
- d) send a Hard Reset message to each PL\_PM state machine; and
- e) send a HARD\_RESET Received confirmation to the transport layer.

#### **7.2.2.3.2.3 PL\_OC2: NOTIFY Received (Power Loss Expected) confirmation**

If this state receives a NOTIFY Received (Power Loss Expected) confirmation, then this state shall:

- a) discard all pending Tx Frame messages;
- b) discard all pending Tx Open messages;
- c) delete all timers (e.g., I\_T Nexus Loss timers, Arbitration Wait Time timers, and Reject To Open Limit timers);
- d) send a Close Connection message to each of the PL\_PM state machines;
- e) send a Cancel Open message to each of the PL\_PM state machines; and
- f) send a NOTIFY Received (Power Loss Expected) confirmation to the transport layer.

#### **7.2.2.3.2.4 PL\_OC2: Phy Disabled confirmation**

If this state receives a Phy Disabled confirmation from all the link layers in the port, then for each destination SAS address accessible through this port for which no I\_T Nexus Loss timer has been created, this state should create an I\_T Nexus Loss timer for that SAS address if the protocol is:

- a) SSP, the port is an SSP target port, the Protocol-Specific Port mode page is implemented, and the I\_T NEXUS LOSS TIME field is not set to 0000h in the Protocol-Specific Port mode page (see 9.2.7.4);
- b) STP, the port is an STP target port, and the STP SMP I\_T NEXUS LOSS TIME field is not set to 0000h in the SMP CONFIGURE GENERAL function; or
- c) SMP, the port is an SMP initiator port, and the STP SMP I\_T NEXUS LOSS TIME field is not set to 0000h in the SMP CONFIGURE GENERAL function.

If this state receives a Phy Disabled confirmation from all the link layers in the port, then, for each destination SAS address accessible through this port, this state may create an I\_T Nexus Loss timer for that SAS address if the protocol is:

- a) SSP and the port is an SSP initiator port; or
- b) STP and the port is an STP initiator port.

If this state receives a Phy Disabled confirmation from all the link layers in the port, an I\_T Nexus Loss timer has been created for a destination SAS address, and that I\_T Nexus Loss timer is not running, then this state shall:

- 1) initialize that I\_T Nexus Loss timer; and
- 2) start that I\_T Nexus Loss timer.

If this state receives a Phy Disabled confirmation from all the link layers in the port and an I\_T Nexus Loss timer has not been created for the destination SAS address (e.g., the destination port is an SSP target port does not support the I\_T NEXUS LOSS TIME field in the Protocol Specific Port mode page), then this state shall, for the destination SAS address:

- a) discard all pending Tx Frame messages;
- b) discard all pending Tx Open messages;
- c) delete all timers (e.g., Arbitration Wait Time timers and Reject To Open Limit timers); and
- d) send a No Phys In Port confirmation to the transport layer.

After this state receives a Phy Disabled confirmation from all the link layers in the port, if a Phy Enabled confirmation is not received from any phy in the port before the I\_T Nexus Loss timer for a destination SAS address has expired, then this state shall, for the destination SAS address:

- a) discard all pending Tx Frame messages;
- b) discard all pending Tx Open messages;
- c) delete all timers (e.g., I\_T Nexus Loss timers, Arbitration Wait Time timers, and Reject To Open Limit timers); and
- d) send a No Phys In Port confirmation to the transport layer.

#### **7.2.3.2.5 PL\_OC2: Start I\_T Nexus Loss Timer request**

If this state receives a Start I\_T Nexus Loss Timer request from the management application layer for a destination SAS Address accessible through this port for which no I\_T Nexus Loss timer has been created, then this state shall create an I\_T Nexus Loss timer for the specified SAS address if the protocol is:

- a) SSP, the port is an SSP target port, the Protocol-Specific Port mode page is implemented, and the I\_T NEXUS LOSS TIME field is not set to 0000h in the Protocol-Specific Port mode page (see 9.2.7.4);
- b) STP, the port is an STP target port, and the STP SMP I\_T NEXUS LOSS TIME field is not set to 0000h in the SMP CONFIGURE GENERAL function; or
- c) SMP, the port is an SMP initiator port, and the STP SMP I\_T NEXUS LOSS TIME field is not set to 0000h in the SMP CONFIGURE GENERAL function.

If this state receives a Start I\_T Nexus Loss Timer request from the management application layer, then this state may create an I\_T Nexus Loss timer for the specified SAS address if the protocol is:

- a) SSP and the port is an SSP initiator port; or
- b) STP and the port is an STP initiator port.

If this state receives a Start I\_T Nexus Loss Timer request from the management application layer, an I\_T Nexus Loss timer has been created for the specified destination SAS address, and that I\_T Nexus Loss timer is not running, then this state shall:

- 1) initialize that I\_T Nexus Loss timer; and
- 2) start that I\_T Nexus Loss timer.

#### **7.2.3.3 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. This state shall discard a pending Tx Open message if there are no pending Tx Frame messages for that destination (e.g., after accepting an incoming connection in which the other phy provides transmit SSP frame credit).

See 6.16.2.1 for additional requirements and recommendations on how this state decides to create pending Tx Open messages.

If this state receives a Retry Open (Retry) message or a Retry Open (Low Phy Power Condition) message, then this state shall process the Retry Open message.

If this state receives a Retry Open (No Destination) message 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., the destination port is an SSP target port does not support the I\_T NEXUS LOSS TIME field in the Protocol Specific Port mode page), 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 or Retry Open (Break Received) 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) message, a Retry Open (Open Timeout Occurred) message, Retry Open (Break Received) message, or a 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:

- a) this state receives a Retry Open (No Destination) message, a Retry Open (Break Received) message, or a Retry Open (Open Timeout Occurred) message;
- b) an I\_T Nexus Loss timer has been created for the destination SAS address; and
- c) 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, the I\_T nexus loss time is not set to FFFFh, and the CONFIGURING bit is set to zero in the REPORT GENERAL response (see 9.4.3.4) for each expander device between this port and the destination port that is two or more levels away from this port, then this state shall start the I\_T Nexus Loss 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 (see 7.2.2.3.6).

If this state:

- a) receives a Retry Open (Pathway Blocked) message or Retry Open (Low Phy Power Condition) message;
- b) an I\_T Nexus Loss timer has been created for the destination SAS address; and
- c) there is no connection established with the destination SAS address,

then this state shall check the I\_T Nexus Loss timer and if the I\_T Nexus Loss timer:

- a) is running, then this state shall not stop the timer; or
- b) has expired, then this state shall process the Retry Open message as if it were an Unable To Connect message (see 7.2.2.3.6).

If this state receives a Retry Open (Retry) message 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; and
- b) initialize the I\_T Nexus Loss timer (see table 202).

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) there is a pending Tx Open message slot available for the destination SAS address;
- 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) there is a pending Tx Open message slot available for the destination SAS address.

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., there is not an available pending Tx Open message slot for the destination SAS address), 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 port bit 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 and state has a pending Tx Open slot available, then this state may create a pending Tx Open message from the Retry Open message.

If a connection is established by another port as indicated by a Retry Open (Opened By Destination) message, then transmit SSP frame 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 transmit SSP frame 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, then this state should send a Tx Open message for the pending Tx Open message that has been pending for the longest time first.

If low phy power conditions (see 4.10.1) are enabled, then this state should use the following precedence in selecting the PL\_PM state machine to which a pending Tx Open message is to be sent:

- 1) PL\_PM state machine with its phy in the active phy power condition (see 4.10.1.2);
- 2) PL\_PM state machine with its phy in the partial phy power condition (see 4.10.1.3); and
- 3) PL\_PM state machine with its phy in the slumber phy power condition (see 4.10.1.4).

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

- a) Retry Open (Opened By Destination);
- b) Retry Open (Opened By Other);
- c) Retry Open (Collided);
- d) Retry Open (Pathway Blocked);
- e) Retry Open (Low Phy Power Condition); or
- f) Retry Open (Retry) if the CONTINUE AWT bit is set to one in the Protocol specific Port mode page (see 9.2.7.4),

then this state shall:

- 1) create an Arbitration Wait Time timer for the pending Tx Open message;
- 2) set the Arbitration Wait Time timer for the pending Tx Open message to the arbitration wait time argument from the Retry Open message; and
- 3) 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 Port 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 persistent connections are supported (see 4.1.13.2), then the Tx Open message sent to a PL\_PM state machine shall contain the following additional argument to be used in an OPEN address frame:

- a) Send Extend Bit from the Transmit Frame request.

If credit advance is implemented (see 4.1.14), then the Tx Open message sent to a PL\_PM state machine shall contain the following additional argument to be used in an OPEN address frame:

- a) Credit Advance Bit from the Transmit Frame request.

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;
- d) a Retry Open (Break Received) message; or
- e) a Retry Open (Retry) message if the CONTINUE AWT bit is set to zero in the Protocol Specific Port mode page (see 9.2.7.4),

then this state shall set:

- a) the pathway blocked count argument in the Tx Open message to zero; and
- b) the arbitration wait time argument in the Tx Open message to zero or a vendor specific value less than 8000h (see 6.16.4).

If a pending Tx Open message was created as the result of this state receiving a Retry Open (Retry) message and the protocol for the connection is:

- a) SSP, the Protocol Specific Port mode page is implemented, and the REJECT TO OPEN LIMIT field in the Protocol Specific Port mode page (see 9.2.7.4) is not set to zero; or
- b) STP and the STP REJECT TO OPEN LIMIT field is not set to zero in the SMP REPORT GENERAL response (see 9.4.3.4),

then this state shall:

- 1) create a Reject To Open Limit timer associated with the pending Tx Open message that received the Retry Open (Retry) message;
- 2) initialize the Reject To Open Limit timer as specified in table 202 (see 7.2.2.1);
- 3) start the Reject To Open Limit timer; and
- 4) wait at least until the Reject To Open Limit timer expires before sending a Tx Open message.

If a pending Tx Open message was created as the result this state receiving a Retry Open (Pathway Blocked) message and the Retry Open pathway blocked count argument is:

- a) FFh, then this state shall set the Tx Open pathway blocked count argument to FFh; or
- b) less than FFh, then this state shall set the Tx Open pathway blocked count argument to the Retry Open pathway blocked count argument plus 01h.

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;
- c) a Retry Open (Collided) message;

- d) a Retry Open (Pathway Blocked) message;
- e) a Retry Open (Low Phy Power Condition) message; or
- f) a Retry Open (Retry) message if the CONTINUE AWT bit is set to one in the Protocol Specific Port mode page (see 9.2.7.4),

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 message 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 port bit and protocol arguments match those in a pending Tx Open message, then any Reject To Open Limit timer associated with that pending Tx Open message shall be discarded.

If this state receives a Connection Opened message and the initiator port bit 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.

#### 7.2.2.3.4 PL\_OC2:Overall\_Control state connection established

If this state receives a Connection Opened message or a Retry Open (Opened By Destination) message for a SAS address, and an I\_T Nexus Loss timer has been created for the SAS address, then this state shall:

- a) if the I\_T Nexus Loss timer for the SAS address has been running, then stop the timer; and
- b) initialize the I\_T Nexus Loss timer (see table 202).

#### 7.2.2.3.5 PL\_OC2:Overall\_Control state unable to establish a connection — Unable To Connect message

If this state receives an Unable To Connect message shown in table 203, then this state shall establish an I\_T nexus loss event (see 7.2.2.3.8).

Table 203 defines the confirmation to be sent to the transport layer for each Unable To Connect message received.

**Table 203 – Confirmations from Unable To Connect messages**

Message received	Confirmation sent to the transport layer
Unable To Connect (Bad Destination)	Transmission Status (I_T Nexus Loss)
Unable To Connect (Connection Rate Not Supported)	Transmission Status (I_T Nexus Loss)
Unable To Connect (Protocol Not Supported)	Transmission Status (I_T Nexus Loss)
Unable To Connect (Reserved Abandon 1)	Transmission Status (I_T Nexus Loss)
Unable To Connect (Reserved Abandon 2)	Transmission Status (I_T Nexus Loss)
Unable To Connect (Reserved Abandon 3)	Transmission Status (I_T Nexus Loss)
Unable To Connect (STP Resources Busy)	Transmission Status (I_T Nexus Loss)
Unable To Connect (Wrong Destination)	Transmission Status (I_T Nexus Loss)
Unable To Connect (Zone Violation)	Transmission Status (I_T Nexus Loss)

#### **7.2.2.3.6 PL\_OC2:Overall\_Control state unable to establish a connection — Unable To Connect message - Retry Open message processed as an Unable To Connect message**

If this state receives a Retry Open (No Destination) message, a Retry Open (Open Timeout Occurred) message, Retry Open (Low Phy Power Condition), or a Retry Open (Pathway Blocked) message and processes it as an Unable To Connect message, then this state shall establish an I\_T nexus loss event (see 7.2.2.3.8).

#### **7.2.2.3.7 PL\_OC2:Overall\_Control state unable to establish a connection — I\_T Nexus Loss timer expires**

If this state receives a Retry Open (No Destination) message, a Retry Open (Open Timeout Occurred) message, Retry Open (Break Received) message, Retry Open (Low Phy Power Condition) message, or a Retry Open (Pathway Blocked) message and the I\_T Nexus Loss timer for the SAS address has expired, then this state shall establish an I\_T nexus loss event (see 7.2.2.3.8).

#### **7.2.2.3.8 PL\_OC2:Overall\_Control state - I\_T nexus loss event**

If an I\_T nexus loss event occurs, then this state shall perform the following:

- a) delete the I\_T Nexus Loss timer for the SAS address;
- b) discard the Retry Open message;
- c) send a Transmission Status (I\_T Nexus Loss) confirmation for the pending Tx Frame message from which the Retry Open message resulted;
- d) discard the pending Tx Frame message from which the Retry Open message resulted;
- e) send an I\_T Nexus Loss Detected confirmation to the management application layer;
- f) if this state has any pending Tx Frame messages with the same destination SAS address and protocol as the Retry Open message, and this state has not sent a Tx Open message to a PL\_PM state machine for the messages, then this state shall send a Transmission Status (I\_T Nexus Loss) confirmation for each pending Tx Frame message and discard the pending Tx Frame messages and any corresponding pending Tx Open messages; and
- g) if this state has any pending Tx Frame messages with the same destination SAS address and protocol as the Retry Open message, and this state has sent a Tx Open message to a PL\_PM state machine for a message, then this state shall send a Cancel Open message to each PL\_PM state machine to which it has sent a Tx Open message. After receiving an Unable To Connect (Arb Stopped) message from a PL\_PM state machine in response to the Cancel Open message, this state shall send a Transmission Status (I\_T Nexus Loss) confirmation for each pending Tx Frame message and discard the pending Tx Frame messages and any corresponding pending Tx Open messages.

#### **7.2.2.3.9 PL\_OC2:Overall\_Control state connection management**

If this state receives an Accept\_Reject OPENs request, then this state shall send an Accept\_Reject OPENs request with the same arguments to all phys in the port.

If this state receives an SMP Transmit Break request, then this state shall send an SMP Transmit Break message to the PL\_PM state machine associated with the corresponding SMP transport state machine. If there is no PL\_PM state machine associated with the request, then the PL\_OC state machine shall ignore the request.

If this state receives one of the following:

- a) a Connection Closed (Close Timeout) message;
- b) a Connection Closed (Break Requested) message; or
- c) a Connection Closed (Break Received) message,

then this state shall not send a Tx Open or Tx Frame message to the PL\_PM state machine that sent the message until this state receives a Connection Closed (Transition to Idle) message from that PL\_PM state machine.

If this state receives a Connection Closed (Normal) message or a Connection Closed (Transition to Idle) message indicating that a connection with a destination SAS address is no longer open, and this state has pending Tx Open messages, then this state may send a Tx Open message to the PL\_PM state machine that sent the Connection Closed message.

If this port is a wide SSP port, then this state shall not reject an incoming connection request on one phy because this state has an outgoing connection request on another phy.

If:

- a) persistent connections are not supported (see 4.1.13.2);
- b) this port is an SSP port;
- c) there are no pending Tx Frame messages for a destination SAS address with which a PL\_PM state machine has established a connection; and
- d) the connection was established by a message from this state,

then this state should send a Close Connection message to the PL\_PM state machine.

If:

- a) persistent connections are not supported (see 4.1.13.2);
- b) this port is an SSP port;
- c) has no pending Tx Frame messages for a destination SAS address with which a PL\_PM state machine has established a connection; and
- d) the connection was established by the destination,

then this state may wait a vendor specific time and then shall send a Close Connection message to the PL\_PM state machine.

If:

- a) persistent connections are supported (see 4.1.13.2);
- b) this port is an SSP port;
- c) Transmission Status (Frame Transmitted) message is received;
- d) there are no pending Tx Frame messages for a destination SAS address with which a PL\_PM state machine has established a connection; and
- e) the last Persistent Connection Established confirmation received from the link layer associated with the PL\_PM state machine contained an Enabled argument,

then this state shall send a No Pending Tx Frames request to the link layer:

- a) associated with the PL\_PM state machine; and
- b) from which the last Persistent Connection Established (Enabled) confirmation was received.

If:

- a) persistent connections are supported (see 4.1.13.2);
- b) this port is an SSP port;
- c) there are no pending Tx Frame messages for a destination SAS address with which a PL\_PM state machine has established a connection;
- d) the connection was established by a message from this state; and
- e) the last Persistent Connection Established confirmation received contained a Disabled argument (i.e., a Persistent Connection Established (Disabled) confirmation was received from the PL\_PM state machine that established the connection),

then this state should send a Close Connection message to the PL\_PM state machine.

If:

- a) persistent connections are supported (see 4.1.13.2);
- b) this port is an SSP port;
- c) has no pending Tx Frame messages for a destination SAS address with which a PL\_PM state machine has established a connection;
- d) the connection was established by the destination; and

- e) the last Persistent Connection Established confirmation received contained a Disabled argument (i.e., a Persistent Connection Established (Disabled) confirmation was received from the PL\_PM state machine that established the connection),

then this state:

- 1) may wait a vendor specific time; and
- 2) shall send a Close Connection message to the PL\_PM state machine, if the last Persistent Connection Established confirmation received did not contain an Enabled argument.

If this state has received a Disable Tx Frames message from a PL\_PM state machine, then this state should send a Close Connection message to the PL\_PM state machine.

NOTE 53 - The PL\_PM state machine sends a Close Connection request to the link layer upon receipt of a Close Connection message or on expiration of the Bus Inactivity Time Limit timer (see 7.2.3.4.1).

#### 7.2.2.3.10 PL\_OC2:Overall\_Control state frame transmission

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

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

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

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

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

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

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

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

Once this state has sent a Tx Frame message containing a Non-Interlocked argument to a PL\_PM state machine, this state shall not send a Tx Frame message containing a Non-Interlocked argument with the same I\_T\_L nexus and command identifier combination to another PL\_PM state machine until this state has received one of the following messages for each Tx Frame message containing a Non-Interlocked argument for the same I\_T\_L nexus and command identifier combination:

- a) Transmission Status (ACK Received);
- b) Transmission Status (NAK Received);

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

For a bidirectional command, frames with the Non-Interlocked argument for an I\_T\_L nexus and command identifier combination may be transmitted on one phy at the same time as frames with the Non-Interlocked argument for the same I\_T\_L nexus and command identifier combination are received on the same phy or on a different phy.

If this port is an SMP initiator port, then this state shall send the Tx Frame message containing the SMP REQUEST frame to the PL\_PM state machine on which the connection was established for the Tx Open message. If this port is an SMP target port, then this state shall send the Tx Frame message containing the SMP\_RESPONSE frame to the PL\_PM state machine on which the connection was established for the Tx Open message. See 6.22 for additional information about SMP connections.

Characteristics of STP connections are defined by SATA (also see 6.21).

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

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

A Balance Not Required argument shall only be included if:

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

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

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

#### **7.2.2.3.11 PL\_OC2:Overall\_Control state frame transmission cancellations**

Cancel requests cause this state to cancel previous Transmit Frame requests. A Cancel request includes the following arguments:

- a) Destination SAS Address; and
- b) Initiator Port Transfer Tag.

If this state receives a Cancel request and has not already sent a Tx Frame message for the Transmit Frame request to a PL\_PM state machine for the Transmit Frame request specified by the Cancel request, then this state shall:

- a) discard all Transmit Frame requests for the specified destination SAS address and initiator port transfer tag; and
- b) send a Transmission Status (Cancel Acknowledge) confirmation to the transport layer.

If this state receives a Cancel request and has already sent a Tx Frame message to a PL\_PM state machine for the Transmit Frame request specified by the Cancel request, then this state shall send a Cancel message to the PL\_PM state machine to which the Tx Frame message was sent. The Cancel message shall include the initiator port transfer tag.

#### **7.2.2.3.12 Transition PL\_OC2:Overall\_Control to PL\_OC1:Idle**

This transition shall occur after:

- a) sending a HARD\_RESET Received confirmation to the transport layer;
- b) a No Phys In Port confirmation is sent to the transport layer; or
- c) sending a NOTIFY Received (Power Loss Expected) confirmation to the transport layer.

### 7.2.3 PL\_PM (port layer phy manager) state machine

#### 7.2.3.1 PL\_PM state machine overview

A PL\_PM state machine:

- a) receives messages from the PL\_OC state machine;
- b) sends requests to the link layer;
- c) receives confirmations from the link layer;
- d) sends confirmations to the transport layer;
- e) sends messages to PL\_OC state machine;
- f) has an Arbitration Wait Time timer;
- g) may have a Bus Inactivity Time Limit timer; and
- h) may have Maximum Connect Time Limit timer.

This state machine consist of the following states:

- a) PL\_PM1:Idle (see 7.2.3.2) (initial state);
- b) PL\_PM2:Req\_Wait (see 7.2.3.3);
- c) PL\_PM3:Connected (see 7.2.3.4); and
- d) PL\_PM4:Wait\_For\_Close (see 7.2.3.5).

This state machine shall start in the PL\_PM1:Idle state after power on.

This state machine shall maintain the timers listed in table 204.

**Table 204 – PL\_PM state machine timers**

Timer	Initial value
Arbitration Wait Time timer	The arbitration wait time argument from a Retry Open message (see 7.2.2.3.1).
Bus Inactivity Time Limit timer	Depending on the protocol used by the port: <ul style="list-style-type: none"> <li>a) for SSP target ports, the value in the BUS INACTIVITY LIMIT field in the Disconnect-Reconnect mode page (see 9.2.7.2);</li> <li>b) for STP target ports, the value in the STP BUS INACTIVITY LIMIT field in the SMP REPORT GENERAL response (see 9.4.3.4);</li> <li>c) for SSP initiator ports, the value in the BUS INACTIVITY LIMIT field in the Disconnect-Reconnect mode page (see 9.2.7.2) of the destination SSP target port; or</li> <li>d) for STP initiator ports, the value in the STP BUS INACTIVITY LIMIT field in the SMP REPORT GENERAL response (see 9.4.3.4) of the destination STP target port.</li> </ul>
Maximum Connect Time Limit timer	Depending on the protocol used by the port: <ul style="list-style-type: none"> <li>a) for an SSP target port, the value in the CONNECT TIME LIMIT field in the Disconnect-Reconnect mode page (see 9.2.7.2);</li> <li>b) for an STP target port, the value in the STP CONNECT TIME LIMIT field in the SMP REPORT GENERAL response;</li> <li>c) for an SMP target port, 2 ms;</li> <li>d) for an SSP initiator port, the value in the CONNECT TIME LIMIT field in the Disconnect-Reconnect mode page (see 9.2.7.2) of the destination SSP target port; or</li> <li>e) for an STP initiator port, the value in the STP CONNECT TIME LIMIT field in the SMP REPORT GENERAL response of the destination STP target port.</li> </ul>

Figure 198 shows part 1 of the PL\_PM state machine.

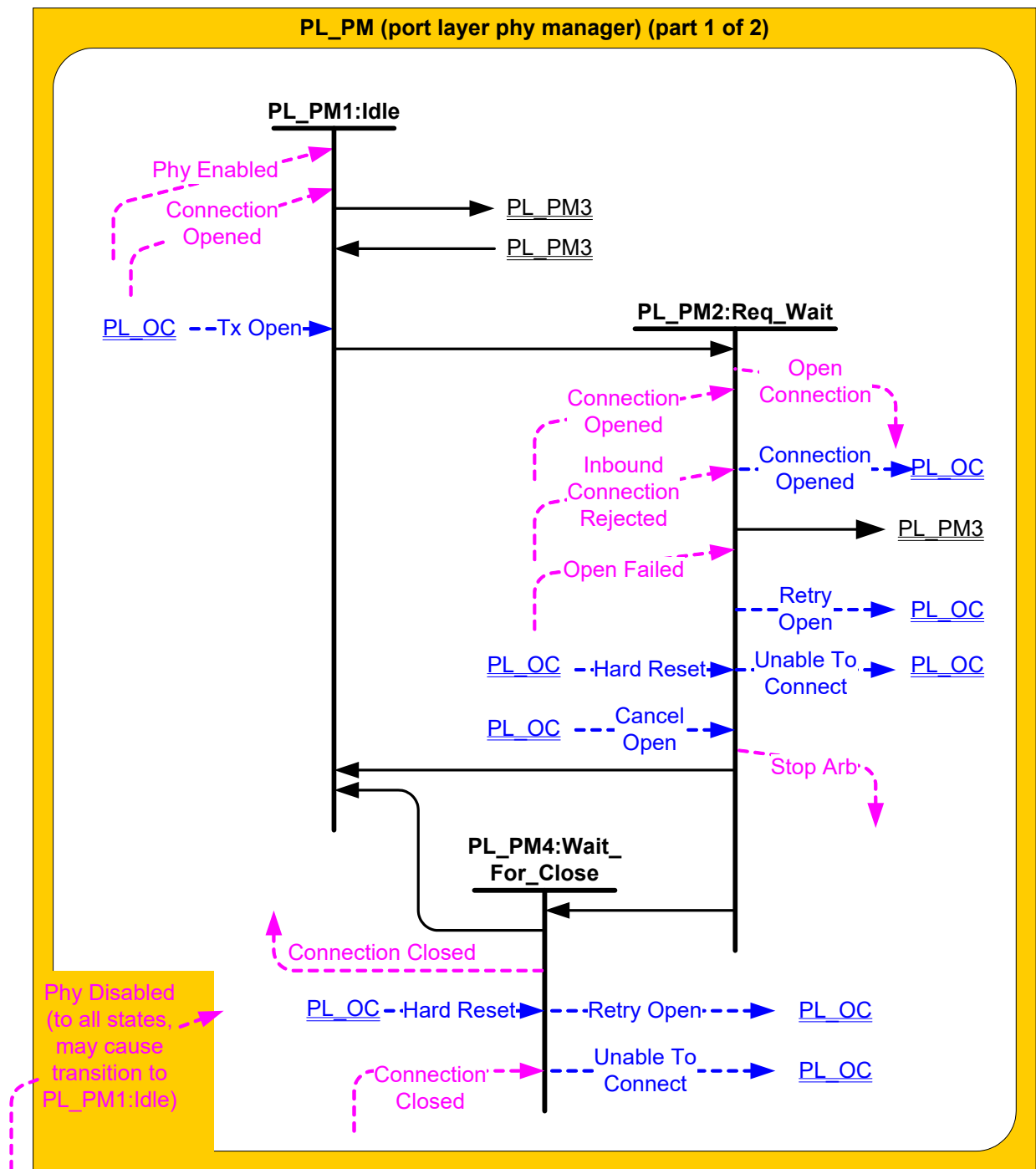


Figure 198 – PL\_PM (port layer phy manager) state machine (1 of 2)



[illegible]

**Figure 199 – PL\_PM (port layer phy manager) state machine (2 of 2)**

### **7.2.3.2 PL\_PM1:Idle state**

#### **7.2.3.2.1 PL\_PM1:Idle state description**

This is the initial state of the PL\_PM state machine.

#### **7.2.3.2.2 Transition PL\_PM1:Idle to PL\_PM2:Req\_Wait**

This transition shall occur after:

- a) a Phy Enabled confirmation is received;
- b) a Tx Open message is received; and
- c) a Connection Opened confirmation has not been received.

This transition shall include:

- a) the received Tx Open arguments.

#### **7.2.3.2.3 Transition PL\_PM1:Idle to PL\_PM3:Connected**

This transition shall occur after a Connection Opened confirmation is received. The transition shall include the received OPEN address frame.

### **7.2.3.3 PL\_PM2:Req\_Wait state**

#### **7.2.3.3.1 PL\_PM2:Req\_Wait state overview**

This state sends an Open Connection request to the link layer and waits for a confirmation. This state sends and receives connection management messages to and from the PL\_OC state machine.

If this state receives a Hard Reset message, then this state shall terminate all operations.

#### **7.2.3.3.2 PL\_PM2:Req\_Wait establishing a connection**

Upon entry into this state, this state shall:

- 1) create an Arbitration Wait Time timer;
- 2) initialize the Arbitration Wait Time timer to the arbitration wait time argument received with the Tx Open message;
- 3) start the Arbitration Wait Time timer; and
- 4) send an Open Connection request to the link layer.

The Open Connection request shall contain the following arguments from the Tx Open message to be used in an OPEN address frame:

- a) Initiator Port Bit;
- b) Protocol;
- c) Connection Rate;
- d) Initiator Connection Tag;
- e) Destination SAS Address;
- f) Source SAS Address;
- g) Pathway Blocked Count; and
- h) Arbitration Wait Time.

If persistent connections are supported (see 4.1.13.2), then the Open Connection request shall contain the following additional argument from the Tx Open message to be used in an OPEN address frame:

- a) Send Extend Bit.

If credit advance is implemented (see 4.1.14), then the Open Connection request shall contain the following additional argument from the Tx Open message to be used in an OPEN address frame:

- a) Credit Advance Bit.

#### **7.2.3.3.3 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 SAS 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 to the PL\_OC state machine. 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 to the PL\_OC state machine.

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

- a) Initiator Port 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 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.

If persistent connections are supported (see 4.1.13.2), then a Retry Open (Opened By Destination) or Retry Open (Opened By Other) message shall contain the following additional argument:

- a) Send Extend Bit set to the value received with the Tx Open message.

If credit advance is implemented (see 4.1.14), then a Retry Open (Opened By Destination) or Retry Open (Opened By Other) message shall contain the following additional argument:

- a) Credit Advance Bit set to the value received with the Tx Open message.

#### **7.2.3.3.4 PL\_PM2:Req\_Wait unable to establish a connection**

If this state receives one of the Open Failed confirmations listed in table 205, then this state shall send either a Retry Open message or an Unable To Connect message to the PL\_OC state machine.

Table 205 defines the message to be sent to the PL\_OC state machine for each Open Failed confirmation.

**Table 205 – Messages from Open Failed confirmations**

Confirmation received	Message sent to the PL_OC state machine
Open Failed (No Destination)	Retry Open (No Destination)
Open Failed (Pathway Blocked)	Retry Open (Pathway Blocked)
Open Failed (Reserved Continue 0)	Retry Open (Retry)
Open Failed (Reserved Continue 1)	Retry Open (Retry)
Open Failed (Reserved Initialize 0)	Retry Open (No Destination)
Open Failed (Reserved Initialize 1)	Retry Open (No Destination)
Open Failed (Reserved Stop 0)	Retry Open (Pathway Blocked)
Open Failed (Reserved Stop 1)	Retry Open (Pathway Blocked)
Open Failed (Retry)	Retry Open (Retry)
Open Failed (Low Phy Power Condition)	Retry Open (Low Phy Power Condition)
Open Failed (Bad Destination)	Unable To Connect (Bad Destination)
Open Failed (Connection Rate Not Supported)	Unable To Connect (Connection Rate Not Supported)
Open Failed (Protocol Not Supported)	Unable To Connect (Protocol Not Supported)
Open Failed (Reserved Abandon 1)	Unable To Connect (Reserved Abandon 1)
Open Failed (Reserved Abandon 2)	Unable To Connect (Reserved Abandon 2)
Open Failed (Reserved Abandon 3)	Unable To Connect (Reserved Abandon 3)
Open Failed (STP Resources Busy)	Unable To Connect (STP Resources Busy)
Open Failed (Wrong Destination)	Unable To Connect (Wrong Destination)
Open Failed (Zone Violation)	Unable To Connect (Zone Violation)

If this state receives an Inbound 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 message shall include the following arguments:

- a) Initiator Port 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.

If persistent connections are supported (see 4.1.13.2), then the Retry Open message shall include the following additional argument:

- a) Send Extend Bit set to the value received with the Tx Open message.

If credit advance is implemented (see 4.1.14), then a Retry Open message shall include the following additional argument:

- a) Credit Advance Bit set to the value received with the Tx Open message.

An Unable To Connect message shall include the following arguments:

- a) Initiator Connection Tag set to the value received with the Tx Open message;
- b) Destination SAS Address set to the value received with the Tx Open message; and
- c) Source SAS Address set to the value received with the Tx Open message.

#### **7.2.3.3.5 PL\_PM2:Req\_Wait connection management**

If this state receives a Cancel Open message and a Connection Opened confirmation has not been received, then this state shall send a Stop Arb request to the link layer.

#### **7.2.3.3.6 Transition PL\_PM2:Req\_Wait to PL\_PM1:Idle**

If a Connection Opened confirmation has not been received, then this transition shall occur after:

- a) a Retry Open message is sent to the PL\_OC state machine;
- b) an Unable To Connect message is sent to the PL\_OC state machine;
- c) all operations have been terminated after receiving a Hard Reset message; or
- d) a Phy Disabled confirmation is received.

#### **7.2.3.3.7 Transition PL\_PM2:Req\_Wait to PL\_PM3:Connected**

This transition shall occur:

- a) after a Connection Opened confirmation is received.

#### **7.2.3.3.8 Transition PL\_PM2:Req\_Wait to PL\_PM4:Wait\_For\_Close**

This transition shall occur after one of the following confirmations is received:

- a) an Open Failed (Open Timeout Occurred);
- b) an Open Failed (Break Received); or
- c) an Open Failed (Arb Stopped).

### **7.2.3.4 PL\_PM3:Connected state**

#### **7.2.3.4.1 PL\_PM3:Connected state description**

If this state was entered from the PL\_PM1:Idle state, then this state shall send a Connection Opened message to the PL\_OC state machine that includes as an argument in the received OPEN address frame.

If:

- a) the protocol for the connection is SSP, the port is an SSP target port, the Disconnect-Reconnect mode page is implemented, and the CONNECT TIME LIMIT field in the Disconnect-Reconnect mode page (see 9.2.7.2) is not set to zero;
- b) the protocol for the connection is SMP and the port is an SMP target port; or
- c) the protocol for the connection is STP, the port is an STP target port, and the STP CONNECT TIME LIMIT field is not set to zero in the SMP REPORT GENERAL response (see 9.4.3.4),

then, upon entry into this state, this state shall:

- 1) create a Maximum Connect Time Limit timer;

- 2) initialize the Maximum Connect Time Limit timer as specified in table 204 (see 7.2.3.1); and
- 3) start the Maximum Connect Time Limit timer.

If:

- a) the protocol for the connection is SSP, the port is an SSP initiator port, and the CONNECT TIME LIMIT field in the Disconnect-Reconnect mode page (see 9.2.7.2) for the destination SSP target port is not set to zero; or
- b) the protocol for the connection is STP, the port is an STP initiator port, and the STP CONNECT TIME LIMIT field is not set to zero in the SMP REPORT GENERAL response (see 9.4.3.4) for the destination STP target port,

then, upon entry into this state, this state may:

- 1) create a Maximum Connect Time Limit timer;
- 2) initialize the Maximum Connect Time Limit timer as specified in table 204 (see 7.2.3.1); and
- 3) start the Maximum Connect Time Limit timer.

If:

- a) the protocol for the connection is SSP, the port is an SSP target port, and the BUS INACTIVITY LIMIT field is set to a non-zero value in the Disconnect-Reconnect mode page (see 9.2.7.2); or
- b) the protocol for the connection is STP, the port is an STP initiator port, and the STP BUS INACTIVITY LIMIT field is not set to zero in the SMP REPORT GENERAL response for the destination STP target port,

then, upon entry into this state, this state shall:

- 1) create a Bus Inactivity Time Limit timer;
- 2) initialize the Bus Inactivity Time Limit timer as specified in table 204; and
- 3) start the Bus Inactivity Time Limit timer.

If a Bus Inactivity Time Limit timer has been created and the connection is:

- a) SSP and this state receives a Tx Frame message; or
- b) STP and the phy is not both transmitting and receiving SATA\_SYNC,

then this state shall:

- 1) stop the Bus Inactivity Time Limit timer, if it is running;
- 2) initialize the Bus Inactivity Time Limit timer as specified in table 204; and
- 3) start the Bus Inactivity Time Limit timer.

If this state receives a Tx Frame message, then this state shall send a Tx Frame request to the link layer. The following arguments from the Tx Frame message shall be included with the Tx Frame request:

- a) the frame to be transmitted; and
- b) if this port is an SSP port, Balance Required or Balance Not Required.

For STP connections, this state connects the STP transport layer to the STP link layer.

If a Bus Inactivity Time Limit timer expires, then:

- a) if persistent connections are not supported (see 4.1.13) and the connection is SSP and there is no Tx Frame request outstanding (i.e., this state is not waiting for an ACK Received or NAK Received confirmation), then this state shall send a Close Connection request to the link layer;
- b) if persistent connections are not supported and the connection is SSP and there is a Tx Frame request outstanding (i.e., this state is waiting for an ACK Received or NAK Received confirmation), then this state shall send a Close Connection request to the link layer after receiving an ACK Received or NAK Received confirmation; or
- c) if the connection is STP, then this state shall send a Close Connection request to the link layer.

If:

- a) the connection is SSP;
- b) persistent connections are supported (see 4.1.13.2);

- c) the Bus Inactivity Time Limit timer expires; and
- d) the last Persistent Connection Established confirmation received contained a Disabled argument,

then:

- a) if there is no Tx Frame request outstanding (i.e., this state is not waiting for an ACK Received or NAK Received confirmation), then this state shall send a Close Connection request to the link layer; or
- b) if there is a Tx Frame request outstanding (i.e., this state is waiting for an ACK Received or NAK Received confirmation), then this state shall send a Close Connection request to the link layer after receiving an ACK Received or NAK Received confirmation.

If:

- a) the connection is SSP;
- b) persistent connections are supported (see 4.1.13.2);
- c) the Bus Inactivity Time Limit timer (see table 204) expires; and
- d) the last Persistent Connection Established confirmation received contained an Enabled argument,

then this state shall:

- 1) initialize the Bus Inactivity Time Limit timer as specified in table 204; and
- 2) start the Bus Inactivity Time Limit timer.

If a Maximum Connect Time Limit timer (see table 204) expires, then:

- a) if persistent connections are not supported (see 4.1.13) and the connection is SSP and there is no Tx Frame request outstanding (i.e., this state is not waiting for an ACK Received or NAK Received confirmation), then this state shall send a Close Connection request to the link layer;
- b) if persistent connections are not supported and the connection is SSP and there is a Tx Frame request outstanding (i.e., this state is waiting for an ACK Received or NAK Received confirmation), then this state shall send a Close Connection request to the link layer after receiving an ACK Received or NAK Received confirmation;
- c) if the connection is SMP, then this state shall send an SMP Transmit Break request to the link layer; or
- d) if the connection is STP, then this state shall send a Close Connection request to the link layer after the phy is both transmitting and receiving SATA\_SYNC.

If:

- a) the connection is SSP;
- b) persistent connections are supported (see 4.1.13.2);
- c) the Maximum Connect Time Limit timer (see table 204) expires; and
- d) the last Persistent Connection Established confirmation received contained a Disabled argument,

then:

- a) if there is no Tx Frame request outstanding (i.e., this state is not waiting for an ACK Received or NAK Received confirmation), then this state shall send a Close Connection request to the link layer; or
- b) if there is a Tx Frame request outstanding (i.e., this state is waiting for an ACK Received or NAK Received confirmation), then this state shall send a Close Connection request to the link layer after receiving an ACK Received or NAK Received confirmation.

If:

- a) the connection is SSP;
- b) persistent connections are supported;
- c) the Maximum Connect Time Limit timer expires; and
- d) the last Persistent Connection Established confirmation received contained an Enabled argument,

then this state shall:

- 1) initialize the Maximum Connect Time Limit timer as specified in table 204; and
- 2) start the Maximum Connect Time Limit timer.

If this state receives a Tx Frame message after sending a Close Connection request but before receiving a Connection Closed confirmation, then this state shall send a Retry Frame message to the PL\_OC state machine.

If this state receives a Frame Received confirmation, then this state shall send a Frame Received confirmation to the transport layer. The confirmation shall include the arguments received with the confirmation (e.g., the frame).

If this state receives an ACK Transmitted confirmation, then this state shall send an ACK Transmitted confirmation to the transport layer including the initiator port transfer tag of the frame that was ACKed.

If this state receives a Frame Transmitted confirmation, then this state shall send:

- a) a Transmission Status (Frame Transmitted) confirmation to the transport layer; and
- b) a Transmission Status (Frame Transmitted) message to the PL\_OC state machine.

If this state receives an ACK Received confirmation, then this state shall send:

- a) a Transmission Status (ACK Received) confirmation to the transport layer; and
- b) a Transmission Status (ACK Received) message to the PL\_OC state machine.

If this state receives a NAK Received confirmation, then this state shall send:

- a) a Transmission Status (NAK Received) confirmation to the transport layer; and
- b) a Transmission Status (NAK Received) message to the PL\_OC state machine.

If this state receives an ACK/NAK Timeout confirmation, then this state shall send:

- a) a Transmission Status (ACK/NAK Timeout) confirmation to the transport layer; and
- b) a Transmission Status (ACK/NAK Timeout) message to the PL\_OC state machine.

If this state receives a Cancel message, then this state shall:

- a) discard all Tx Frame requests for the specified initiator port transfer tag;
- b) send a Transmission Status (Cancel Acknowledge) confirmation to the transport layer including the destination SAS address and the initiator port transfer tag as arguments; and
- c) discard any subsequent confirmations for previous Tx Frame requests sent for the initiator port transfer tag.

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 one of the following:

- a) a Connection Closed (Normal) confirmation;
- b) a Connection Closed (Close Timeout) confirmation;
- c) a Connection Closed (Break Requested) confirmation;
- d) a Connection Closed (Break Received) confirmation; or
- e) a Connection Closed (Transition to Idle) confirmation,

then this state shall send a Connection Closed message to the PL\_OC state machine including the argument received with the confirmation.

If this state receives a Connection Closed (Transition to Idle) confirmation after receiving one of the following:

- a) a Connection Closed (Break Received) confirmation; or
- b) a Connection Closed (Break Requested) confirmation,

then this state shall send a Transmission Status (Break Received) confirmation to the transport layer.

If this state receives a Connection Closed (Normal) confirmation, a Connection Closed (Transition to Idle) confirmation, or a Phy Disabled confirmation after sending a Transmission Status (Frame Transmitted) confirmation, before receiving an ACK Received or NAK Received confirmation, then this state shall send:

- a) a Transmission Status (Connection Lost Without ACK/NAK) confirmation to the transport layer; and
- b) a Transmission Status (Connection Lost Without ACK/NAK) message to the PL\_OC state machine.



If this state receives a Connection Closed (Normal) confirmation, a Connection Closed (Transition to Idle) confirmation, or a Phy Disabled confirmation after sending a Tx Frame request but before receiving a Frame Transmitted confirmation, then this state shall send a Retry Frame message to the PL\_OC state machine.

If this state receives a Connection Closed confirmation during an SMP connection, then this state shall send a Connection Closed confirmation to the transport layer.

If this state receives a Credit Timeout confirmation, then this state shall send a Retry Frame message to the PL\_OC state machine.

A Retry Frame message shall include the following arguments from the Tx Frame message:

- a) Initiator Port Bit;
- b) Protocol;
- c) Connection Rate;
- d) Initiator Connection Tag;
- e) Destination SAS Address;
- f) Source SAS Address; and
- g) Frame.

After this state receives a DONE Received (Normal) confirmation or DONE Received (Credit Timeout) confirmation, if this state does not receive a Tx Frame message within 1 ms, then this state shall send a Disable Tx Frames message to the PL\_OC state machine.

If this state receives a DONE Received (ACK/NAK Timeout) confirmation or DONE Transmitted confirmation, then this state shall send a Disable Tx Frames message to the PL\_OC state machine.

If this state receives an SMP Transmit Break message, then this state shall send an SMP Transmit Break request to the link layer.

If this state receives a Hard Reset message, then this state shall terminate all operations.

#### **7.2.3.4.2 Transition PL\_PM3:Connected to PL\_PM1:Idle**

This transition shall occur after:

- a) a Connection Closed (Transition to Idle) message is sent to the PL\_OC state machine; or
- b) all operations are terminated after receiving a Hard Reset message.

#### **7.2.3.5 PL\_PM4:Wait\_For\_Close state**

##### **7.2.3.5.1 PL\_PM4:Wait\_For\_Close state description**

After receiving a Connection Closed (Transition to Idle) confirmation, if this state was entered as the result of the PL\_PM2:Req\_Wait state receiving:

- a) an Open Failed (Open Timeout Occurred) confirmation, then this state shall send a Retry Open (Open Timeout Occurred) message to the PL\_OC state machine; or
- b) an Open Failed (Break Received) confirmation, then this state shall send a Retry Open (Break Received) message to the PL\_OC state machine.

The Retry Open message shall include the following arguments:

- a) Initiator Port 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 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.

If persistent connections are supported (see 4.1.13.2), then the Retry Open message shall include the following additional argument:

- a) Send Extend Bit set to the value received with the Tx Open message.

If this state receives a Connection Closed confirmation and the connection request was for an SMP connection, then this state shall send a Connection Closed confirmation to the transport layer.

After receiving a Connection Closed (Transition to Idle) confirmation, if this state was entered after the PL\_PM2:Req\_Wait state received an Open Failed (Arb Stopped) confirmation (i.e., as the result of the PL\_PM2:Req\_Wait state sending a Stop Arb request), then this state shall send an Unable To Connect (Arb Stopped) message to the PL\_OC state machine.

If this state receives a Hard Reset message, then this state shall terminate all operations.

#### **7.2.3.5.2 Transition PL\_PM4:Wait\_For\_Close to PL\_PM1:Idle**

This transition shall occur after:

- a) a Retry Open or Unable To Connect message is sent to the PL\_OC state machine; or
- b) all operations are terminated after receiving a Hard Reset message.

## 8 Transport layer

### 8.1 Transport layer overview

The transport layer defines frame formats and how frames are processed by this layer. Transport layer state machines interface to the application layer and port layer and construct and parse frame contents. For SSP, the transport layer only receives frames from the port layer for which an ACK is going to be transmitted by the link layer.

## 8.2 SSP transport layer

### 8.2.1 SSP frame format

Table 206 defines the SSP frame format.

**Table 206 – SSP frame format**

Byte\Bit	7	6	5	4	3	2	1	0
0	FRAME TYPE							
1	(MSB)	HASHED DESTINATION SAS ADDRESS						
...								
3								
4	Reserved							
5	(MSB)	HASHED SOURCE SAS ADDRESS						
...								
7								
8	Reserved							
9	Reserved							
10	Reserved			TLR CONTROL		RETRY DATA FRAMES	RETRANSMIT	CHANGING DATA POINTER
11	Reserved						NUMBER OF FILL BYTES	
12	Reserved							
13	Reserved							
...								
15								
16	(MSB)	INITIATOR PORT TRANSFER TAG						
17	(LSB)							
18	(MSB)	TARGET PORT TRANSFER TAG						
19	(LSB)							
20	(MSB)	DATA OFFSET						
...								
23	(LSB)							
24	INFORMATION UNIT (e.g., see table 209, table 211, table 213, table 214, or table 215)							
...								
m								
	Fill bytes, if needed							
n - 3	(MSB)	CRC						
...								
n	(LSB)							

Table 207 defines the FRAME TYPE field, which defines the format of the INFORMATION UNIT field.

**Table 207 – FRAME TYPE field**

Code	Name of frame	Type of information unit	Originator	Information unit size (bytes)	Reference
01h	DATA frame (i.e., write DATA frame or read DATA frame)	Data information unit (i.e., write Data information unit or read Data information unit)	SSP initiator port or SSP target port	1 to 1 024	8.2.2.4
05h	XFER_RDY frame	Transfer Ready information unit	SSP target port	12	8.2.2.3
06h	COMMAND frame	Command information unit	SSP initiator port	28 to 280	8.2.2.1
07h	RESPONSE frame	Response information unit	SSP target port	24 to 1 024	8.2.2.5
16h	TASK frame	Task Management Function information unit	SSP initiator port	28	8.2.2.2
F0h to FFh	Vendor specific				
All others	Reserved				

The HASHED DESTINATION SAS ADDRESS field contains the hashed value of the destination SAS address (see 4.2.5). See 8.2.6.2.2 and 8.2.6.3.2 for transport layer requirements on checking this field.

The HASHED SOURCE SAS ADDRESS field contains the hashed value of the source SAS address (see 4.2.5). See 8.2.6.2.2 and 8.2.6.3.2 for transport layer requirements on checking this field.

Table 208 defines the TLR CONTROL field for COMMAND frames. The TLR CONTROL field is reserved for all other frame types.

**Table 208 – TLR CONTROL field for COMMAND frames**

Code <sup>a</sup>	Description
00b or 11b	The SSP target port shall use the TRANSPORT LAYER RETRIES bit in the Protocol Specific Logical Unit mode page (see 9.2.7.3) to enable or disable transport layer retries for this command as follows if the TRANSPORT LAYER RETRIES bit is set to: a) one, then the SSP target port shall set the RETRY DATA FRAMES bit to one in any XFER_RDY frames that the SSP target port transmits for this command; or b) zero, then the SSP target port shall set the RETRY DATA FRAMES bit to zero in any XFER_RDY frames that the SSP target port transmits for this command.
01b	The SSP target port may enable transport layer retries for this command.  If the SSP target port enables transport layer retries, then it shall set the RETRY DATA FRAMES bit to one in any XFER_RDY frames that it transmits for this command.  If the SSP target port does not enable transport layer retries, then it shall set the RETRY DATA FRAMES bit to zero in any XFER_RDY frames that it transmits for this command.
10b	The SSP target port shall: a) disable transport layer retries for this command; and b) set the RETRY DATA FRAMES bit to zero in any XFER_RDY frames that it transmits for this command.
<sup>a</sup> If the SSP target port receives a non-zero value in the TLR CONTROL field and does not support non-zero values in the TLR CONTROL field, then the SSP target port shall reply with a RESPONSE frame with the DATAPRES field set to RESPONSE_DATA and the RESPONSE CODE field set to 02h (i.e., INVALID FRAME).	

If an SSP initiator port supports transport layer retries, then it shall set the TLR CONTROL field to 01b in each COMMAND frame that it sends unless it has determined that the I\_T\_L nexus does not support the TLR CONTROL field.

If an SSP initiator port does not support transport layer retries, then it shall set the TLR CONTROL field to 10b in each COMMAND frame that it sends unless it has determined that the I\_T\_L nexus does not support the TLR CONTROL field.

An SSP initiator port determines that an I\_T\_L nexus does not support the TLR CONTROL field if the SSP initiator port sends a COMMAND frame with the TLR CONTROL field set to 01b or 10b and receives a RESPONSE frame with the DATAPRES field set to RESPONSE\_DATA and the RESPONSE CODE field set to 02h (i.e., INVALID FRAME). After determining that an I\_T\_L nexus does not support the TLR CONTROL field, the SSP initiator port shall set the TLR CONTROL field to 00b for subsequent COMMAND frames for that I\_T\_L nexus.

NOTE 54 - Initiator ports compliant with SAS-1.1 always set the TLR CONTROL field to 00b.

NOTE 55 - The TLR CONTROL SUPPORTED field in the Protocol Specific Logical Unit Information VPD page (see 9.2.11.3) indicates if the SSP target port supports the TLR CONTROL field set to a non-zero value.

An SSP target port sets the RETRY DATA FRAMES bit in an XFER\_RDY frame based on the TLR CONTROL field received in the COMMAND frame for the command (see table 208) and the TRANSPORT LAYER RETRIES bit in the Protocol Specific Logical Unit mode page (see 9.2.7.3).

A RETRY DATA FRAMES bit set to one in an XFER\_RDY frame specifies that the SSP initiator port shall enable transport layer retries for write DATA transfers related to this XFER\_RDY.

A RETRY DATA FRAMES bit set to zero in an XFER\_RDY frame specifies that the SSP initiator port shall disable transport layer retries for write DATA transfers related to this XFER\_RDY.

The RETRY DATA FRAMES bit is reserved for frames other than XFER\_RDY frames.

A RETRANSMIT bit set to one specifies that the frame is a retransmission after the SSP port failed in its previous attempt to transmit the frame (e.g., the SSP port received a NAK for the frame transmitted in the previous attempt). The RETRANSMIT bit is set to one for TASK frames, RESPONSE frames, and XFER\_RDY frames under the conditions defined in 8.2.4 and shall be set to zero for all other frame types.

A CHANGING DATA POINTER bit set to one specifies that the frame is a retransmission after the SSP port failed in its previous attempt to transmit the frame or a subsequent frame and the DATA OFFSET field of the frame may not be sequentially increased from that of the previous frame. The CHANGING DATA POINTER bit is set to one for DATA frames under the conditions defined in 8.2.4.5 and shall be set to zero for all other frame types.

The NUMBER OF FILL BYTES field specifies the number of fill bytes between the INFORMATION UNIT field and the CRC field. The NUMBER OF FILL BYTES field shall be set to 00b for all frame types except DATA frames as specified in 8.2.2.4 and RESPONSE frames as specified in 8.2.2.5 (i.e., all other frame types are four-byte aligned).

The INITIATOR PORT TRANSFER TAG field contains a value that allows the SSP initiator port to establish a context for commands and task management functions.

For COMMAND frames and TASK frames, the SSP initiator port shall set the INITIATOR PORT TRANSFER TAG field to a value that is unique for the I\_T nexus established by the connection (see 6.16). An SSP initiator port shall not reuse the same initiator port transfer tag when transmitting COMMAND frames or TASK frames to different LUNs in the same SSP target port. An SSP initiator port may reuse an initiator port transfer tag when transmitting frames to different SSP target ports. An SSP initiator port does not reuse an initiator port transfer tag until it receives indication from the SSP target port that the initiator port transfer tag is no longer in use (see 8.2.4, 8.2.5, and 9.2.2).

The INITIATOR PORT TRANSFER TAG field in a COMMAND frame contains the command identifier defined in SAM-5. The INITIATOR PORT TRANSFER TAG field in a TASK frame is the association between a Send Task Management Request (see 9.2.1.12) and a Received Task Management Function Executed (see 9.2.1.15). The number space used in the INITIATOR PORT TRANSFER TAG fields is shared across COMMAND frames and TASK frames (e.g., if an initiator port transfer tag is used for a COMMAND frame, then it is not also used for a concurrent TASK frame).

For DATA, XFER\_RDY, and RESPONSE frames, the SSP target port shall set the INITIATOR PORT TRANSFER TAG field to the initiator port transfer tag of the command or task management function to which the frame pertains.

The TARGET PORT TRANSFER TAG field provides an optional method for an SSP target port to establish the write data context when receiving a write DATA frame (i.e., determine the command to which the write data corresponds). Unlike the INITIATOR PORT TRANSFER TAG field, which was assigned by the SSP initiator port, the TARGET PORT TRANSFER TAG field in a write DATA frame contains a value assigned by the SSP target port that was delivered to the SSP initiator port in the XFER\_RDY frame requesting the write data.

NOTE 56 - The TARGET PORT TRANSFER TAG field is useful if an SSP target port has more than one XFER\_RDY frame outstanding (i.e., the SSP target port has transmitted an XFER\_RDY frame for each of two or more commands and has not yet received all the write data for them).

SSP target ports may set the TARGET PORT TRANSFER TAG field to any value when transmitting any SSP frame. SSP target ports that use this field should set the TARGET PORT TRANSFER TAG field in every XFER\_RDY frame to a value that is unique for the I\_T\_L nexus and command identifier combination (i.e., that is unique for every XFER\_RDY that is outstanding from the SSP target port).

SSP initiator ports shall set the TARGET PORT TRANSFER TAG field as follows:

- a) for each write DATA frame that is sent in response to an XFER\_RDY frame, the SSP initiator port shall set the TARGET PORT TRANSFER TAG field to the value that was in the corresponding XFER\_RDY frame;

- b) for each write DATA frame that is sent containing first burst data (see 8.2.2.4), the SSP initiator port shall set the TARGET PORT TRANSFER TAG field to FFFFh; and
- c) for frames other than write DATA frames, the SSP initiator port shall set the TARGET PORT TRANSFER TAG field to FFFFh.

For DATA frames, the DATA OFFSET field is described in 8.2.2.4. For all other frame types, the DATA OFFSET field shall be ignored.

The INFORMATION UNIT field contains the information unit, the format of which is defined by the FRAME TYPE field (see table 207). The maximum size of the INFORMATION UNIT field is 1 024 bytes, making the maximum size of the frame 1 052 bytes (i.e., 24 bytes of header + 1 024 bytes of data + 4 bytes of CRC).

Fill bytes shall be included after the INFORMATION UNIT field so the CRC field is aligned on a four byte boundary. The number of fill bytes are specified by the NUMBER OF FILL BYTES field. The contents of the fill bytes are vendor specific.

The CRC field contains a CRC value (see 6.7) that is computed over the entire SSP frame prior to the CRC field including the fill bytes (i.e., all data dwords between the SOF and the CRC field). The CRC field is checked by the link layer (see 6.20), not the transport layer.

## 8.2.2 Information units

### 8.2.2.1 COMMAND frame - Command information unit

A COMMAND frame is sent by an SSP initiator port to request that a command be processed by the SCSI device server in a logical unit (see 8.2.3.3, 8.2.3.4, 8.2.3.5, and 8.2.3.6).

Table 209 defines the Command information unit used in a COMMAND frame.

**Table 209 – COMMAND frame - Command information unit**

Byte\Bit	7	6	5	4	3	2	1	0
0	LOGICAL UNIT NUMBER							
...								
7								
8	Reserved							
9	ENABLE FIRST BURST	COMMAND PRIORITY				TASK ATTRIBUTE		
10	Reserved							
11	ADDITIONAL CDB LENGTH (n dwords)						Reserved	
12	CDB							
...								
27								
28	ADDITIONAL CDB BYTES							
...								
27+n×4								

The LOGICAL UNIT NUMBER field specifies the logical unit number of the logical unit to which the task router shall route the command. The structure of the LOGICAL UNIT NUMBER field shall be as defined in SAM-5. If the addressed logical unit does not exist, then the task router shall follow the rules for selection of incorrect logical units defined in SAM-5.



An ENABLE FIRST BURST bit set to one specifies that the SSP target port shall expect first burst data for the command as defined by the FIRST BURST SIZE field in the Disconnect-Reconnect mode page (see 9.2.7.2). An ENABLE FIRST BURST bit set to zero specifies that the SSP target port shall not expect first burst data for the command (i.e., the FIRST BURST SIZE field in the Disconnect-Reconnect mode page shall be ignored). SCSI application clients shall only set the ENABLE FIRST BURST bit to one if:

- a) the FIRST BURST SIZE field in the Disconnect-Reconnect mode page is set to a value other than 0000h; and
- b) the logical unit and SSP target port comply with this standard (e.g., as reported in the standard INQUIRY data version descriptors (see SPC-4)).

The COMMAND PRIORITY field specifies the relative scheduling importance of a command with the TASK ATTRIBUTE field set to 000b (i.e., SIMPLE) in relation to other commands already in the task set with SIMPLE task attributes (see SAM-5).

The TASK ATTRIBUTE field is defined in table 210.

**Table 210 – TASK ATTRIBUTE field**

Code	Task attribute	Description
000b	SIMPLE	Specifies that the command be managed according to the rules for a SIMPLE task attribute (see SAM-5).
001b	HEAD OF QUEUE	Specifies that the command be managed according to the rules for a HEAD OF QUEUE task attribute (see SAM-5).
010b	ORDERED	Specifies that the command be managed according to the rules for an ORDERED task attribute (see SAM-5).
011b	Reserved	
100b	ACA	Specifies that the command be managed according to the rules for an ACA task attribute (see SAM-5).
101b to 111b	Reserved	

The ADDITIONAL CDB LENGTH field contains the length in dwords (i.e., four bytes) of the ADDITIONAL CDB BYTES field.

The CDB field and ADDITIONAL CDB BYTES field together contain the CDB to be interpreted by the device server in the addressed logical unit. Any bytes after the end of the actual CDB within the two fields shall be ignored (e.g., a six-byte CDB occupies the first six bytes of the CDB field, the remaining ten bytes of the CDB field are ignored, and the ADDITIONAL CDB BYTES field is not present).

The contents of the CDB are defined in the SCSI command standards (e.g., SPC-4).

#### **8.2.2.2 TASK frame - Task Management Function information unit**

A TASK frame is sent by an SSP initiator port to request that a task management function be processed by the task manager in a logical unit (see 8.2.3.2).

Table 211 defines the Task Management Function information unit used in a TASK frame.

**Table 211 – TASK frame - Task Management Function information unit**

Byte\Bit	7	6	5	4	3	2	1	0						
0	LOGICAL UNIT NUMBER													
...														
7														
8	Reserved													
9	Reserved													
10	TASK MANAGEMENT FUNCTION													
11	Reserved													
12	(MSB)	INITIATOR PORT TRANSFER TAG TO MANAGE												
13								(LSB)						
14	Reserved													
...														
27														

The LOGICAL UNIT NUMBER field specifies the logical unit number of the logical unit, if any, to which the task router shall route the task management function. The structure of the LOGICAL UNIT NUMBER field shall be as defined in SAM-5. If the addressed logical unit does not exist, then the task router shall return a RESPONSE frame with the DATAPRES field set to RESPONSE\_DATA and its RESPONSE CODE field set to INCORRECT LOGICAL UNIT NUMBER.

Table 212 defines the TASK MANAGEMENT FUNCTION field.

**Table 212 – TASK MANAGEMENT FUNCTION field** (part 1 of 2)

Code	Task management function	Support <sup>a</sup>	Uses LOGICAL UNIT NUMBER field	Uses INITIATOR PORT TRANSFER TAG TO MANAGE field	Description <sup>b</sup>
01h	ABORT TASK	M	yes	yes	The task manager shall perform the ABORT TASK task management function with the L of the I_T_L nexus set to the value of the LOGICAL UNIT NUMBER field and the command identifier set to the value of the INITIATOR PORT TRANSFER TAG TO MANAGE field (see SAM-5).
02h	ABORT TASK SET	M	yes	no	The task manager shall perform the ABORT TASK SET task management function with the L of the I_T_L nexus set to the value of the LOGICAL UNIT NUMBER field (see SAM-5).
04h	CLEAR TASK SET	M	yes	no	The task manager shall perform the CLEAR TASK SET task management function with the L of the I_T_L nexus set to the value of the LOGICAL UNIT NUMBER field (see SAM-5).
08h	LOGICAL UNIT RESET	M	yes	no	The task manager for the selected logical unit shall perform the LOGICAL UNIT RESET task management function with the L of the I_T_L nexus set to the value of the LOGICAL UNIT NUMBER field (see SAM-5).
10h	I_T NEXUS RESET	M	no	no	The task manager shall perform the I_T NEXUS RESET task management function (see SAM-5).
20h	Reserved				
40h	CLEAR ACA	X	yes	no	The task manager shall perform the CLEAR ACA task management function with the L of the I_T_L nexus set to the value of the LOGICAL UNIT NUMBER field (see SAM-5).
<sup>a</sup> M = implementation is mandatory, X = implementation requirements are specified by SAM-5. <sup>b</sup> The task manager or device server shall perform the specified task management function with the I and T arguments set to the SSP initiator port and SSP target port involved in the connection used to deliver the TASK frame.					

Table 212 – TASK MANAGEMENT FUNCTION field (part 2 of 2)

Code	Task management function	Support <sup>a</sup>	Uses LOGICAL UNIT NUMBER field	Uses INITIATOR PORT TRANSFER TAG TO MANAGE field	Description <sup>b</sup>
80h	QUERY TASK	M	yes	yes	The task manager shall perform the QUERY TASK task management function with L set to the value of the LOGICAL UNIT NUMBER field and command identifier set to the value of the INITIATOR PORT TRANSFER TAG TO MANAGE field (see SAM-5).
81h	QUERY TASK SET	M	yes	no	The task manager shall perform the QUERY TASK SET task management function with L set to the value of the LOGICAL UNIT NUMBER field (see SAM-5).
82h	QUERY ASYNCHRONOUS EVENT	M	yes	no	The task manager shall perform the QUERY ASYNCHRONOUS EVENT task management function with L set to the value of the LOGICAL UNIT NUMBER field (see SAM-5).
All others	Reserved				
<sup>a</sup> M = implementation is mandatory, X = implementation requirements are specified by SAM-5. <sup>b</sup> The task manager or device server shall perform the specified task management function with the I and T arguments set to the SSP initiator port and SSP target port involved in the connection used to deliver the TASK frame.					

If the TASK MANAGEMENT FUNCTION field contains a reserved or unsupported value, then the task manager shall return a RESPONSE frame with the DATAPRES field set to RESPONSE\_DATA and its RESPONSE CODE field set to TASK MANAGEMENT FUNCTION NOT SUPPORTED.

If the TASK MANAGEMENT FUNCTION field is set to ABORT TASK or QUERY TASK, then the INITIATOR PORT TRANSFER TAG TO MANAGE field specifies the INITIATOR PORT TRANSFER TAG value from the COMMAND frame that contained the command to be aborted or checked. For all other task management functions, the INITIATOR PORT TRANSFER TAG TO MANAGE field shall be ignored.

### 8.2.2.3 XFER\_RDY frame - Transfer Ready information unit

An XFER\_RDY frame is sent by an SSP target port to request write data from the SSP initiator port during a write command or a bidirectional command (see 8.2.3.4 and 8.2.3.6).

Table 213 defines the Transfer Ready information unit used in an XFER\_RDY frame.

**Table 213 – XFER\_RDY frame - Transfer Ready information unit**

Byte\Bit	7	6	5	4	3	2	1	0
0	(MSB) REQUESTED OFFSET (LSB)							
...								
3								
4	(MSB) WRITE DATA LENGTH (LSB)							
...								
7								
8	Reserved							
...								
11								

The REQUESTED OFFSET field contains the application client buffer offset of the segment of write data in the data-out buffer that the SSP initiator port may transmit using write DATA frames. The requested offset shall be a multiple of four (i.e., each write DATA frame shall begin transferring data on a dword boundary).

The REQUESTED OFFSET field shall be set to 00000000h for the first XFER\_RDY frame of a command unless:

- a) the ENABLE FIRST BURST bit was set to one in the COMMAND frame (see 8.2.2.1); and
- b) the FIRST BURST SIZE field is set to a value other than 0000h in the Disconnect-Reconnect mode page (see 9.2.7.2.5).

If the ENABLE FIRST BURST bit was set to one in the COMMAND frame (see 8.2.2.1), then in the initial XFER\_RDY frame for the command, the SSP target port shall set the REQUESTED OFFSET field to the application client buffer offset of the segment of write data following the first burst data defined by the FIRST BURST SIZE field in the Disconnect-Reconnect mode page (see 9.2.7.2.5).

If any additional XFER\_RDY frames are required for the command and transport layer retries are not being used, then the REQUESTED OFFSET field shall be set to the sum of the requested offset and write data length of the previous XFER\_RDY frame.

The WRITE DATA LENGTH field contains the number of bytes of write data the SSP initiator port may transmit using write DATA frames from the application client data-out buffer starting at the requested offset. The SSP target port shall set the WRITE DATA LENGTH field to a value greater than or equal to 00000001h. If the MAXIMUM BURST SIZE field is not set to 0000h in the Disconnect-Reconnect mode page, and a persistent connection has not been established (see 4.1.13), then the SSP target port shall set the WRITE DATA LENGTH field to a value less than or equal to the number of bytes specified by the MAXIMUM BURST SIZE field (see 9.2.7.2.5).

If an SSP target port transmits an XFER\_RDY frame containing a WRITE DATA LENGTH field set to a value that is not divisible by four, then the SSP target port shall not transmit any subsequent XFER\_RDY frames for that command (i.e., only the last XFER\_RDY for a command may request a non-dword multiple write data length).

The value in the REQUESTED OFFSET field plus the value of the WRITE DATA LENGTH field shall not be greater than 1\_00000000h (i.e., a SCSI command shall not transfer more than  $2^{32}$  bytes of write data).

#### 8.2.2.4 DATA frame - Data information unit

During a write command or a bidirectional command (see 8.2.3.4 and 8.2.3.6), one or more write DATA frames are sent by an SSP initiator port to deliver write data.

During a read command or a bidirectional command (see 8.2.3.5 and 8.2.3.6), one or more read DATA frames are sent by an SSP target port to deliver read data.

Table 214 defines the Data information unit used in a DATA frame.

**Table 214 – DATA frame - Data information unit**

Byte\Bit	7	6	5	4	3	2	1	0
0	DATA							
...								
n								

The DATA field contains the read data (i.e., data to the application client's data-in buffer) or write data (i.e., data from the application client's data-out buffer).

The size of the DATA field (i.e., the data length) is determined by subtracting the following values from the DATA frame size (i.e., the number of bytes between SOF and EOF (see 6.20.3)):

- a) the number of bytes in frame header (i.e., 24);
- b) the number of bytes in the CRC field (i.e., 4); and
- c) the number of fill bytes, specified by the NUMBER OF FILL BYTES field in the frame header (see 8.2.1).

The maximum size of the Data information unit (i.e., the DATA field) is the maximum size of any information unit in an SSP frame (see 8.2.1). The minimum size of the Data information unit is one byte.

An SSP initiator port shall only transmit a write DATA frame:

- a) in response to an XFER\_RDY frame; or
- b) after transmitting a COMMAND frame if the ENABLE FIRST BURST bit was set to one in the COMMAND frame (see 8.2.2.1) and the FIRST BURST SIZE field in the Disconnect-Reconnect mode page is set to a value other than 0000h (see 9.2.7.2.5).

If the MAXIMUM BURST SIZE field is not set to 0000h in the Disconnect-Reconnect mode page and a persistent connection has not been established (see 4.1.13), then the maximum amount of data that is transferred at one time by an SSP target port per I\_T\_L nexus and command identifier combination is limited by the value in the MAXIMUM BURST SIZE field (see 9.2.7.2.5).

A write DATA frame shall only contain write data for a single XFER\_RDY frame.

An SSP initiator port shall set the NUMBER OF FILL BYTES field to 00b in the frame header (see 8.2.1) in all write DATA frames that it transmits in response to an XFER\_RDY frame except the last write DATA frame for that XFER\_RDY frame. An SSP initiator port may set the NUMBER OF FILL BYTES field to a non-zero value in the last DATA frame that it transmits in response to an XFER\_RDY.

An SSP target port shall set the NUMBER OF FILL BYTES field to 00b in the frame header (see 8.2.1) in all read DATA frames for a command except the last read DATA frame for that command. The SSP target port may set the NUMBER OF FILL BYTES field to a non-zero value in the last read DATA frame for a command (i.e., only the last read DATA frame for a command may contain data with a length that is not a multiple of four).

An SSP initiator port shall not transmit a write DATA frame for a given I\_T\_L nexus and command identifier combination after the SSP initiator port has sent a TASK frame that terminates that command (e.g., an ABORT TASK).

The DATA OFFSET field in the frame header (see 8.2.1) contains the application client buffer offset as described by SAM-5. For read DATA frames, this is the offset into the application client's data-in buffer. For write DATA frames, this is the offset into the application client's data-out buffer. The data offset shall be a multiple of four (i.e., each DATA frame shall transfer data beginning on a dword boundary).

The DATA OFFSET field shall be set to 00000000h in the initial read DATA frame for a command with the ENABLE FIRST BURST bit set to zero (see 8.2.2.1). If any additional read DATA frames are required for the command and transport layer retries are not being used, then the DATA OFFSET field shall be set to the sum of the data offset and data length of the previous read DATA frame.

The DATA OFFSET field shall be set to 00000000h in the initial write DATA frame for a command. If any additional write DATA frames are required for the command and transport layer retries are not being used, then the DATA OFFSET field shall be set to the sum of the data offset and data length of the previous write DATA frame.

The value in the DATA OFFSET field plus the size of the DATA field shall not be greater than 1\_00000000h (i.e., a SCSI command shall not transfer more than  $2^{32}$  bytes of write data and/or more than  $2^{32}$  bytes of read data).

### 8.2.2.5 RESPONSE frame - Response information unit

#### 8.2.2.5.1 RESPONSE frame - Response information unit overview

A RESPONSE frame is sent by an SSP target port to deliver:

- a) a service response, SCSI status (e.g., GOOD or CHECK CONDITION), and sense data, if any, for a command (see 8.2.3.3, 8.2.3.4, 8.2.3.5, and 8.2.3.6);
- b) a service response for a task management function (see 8.2.3.2); or
- c) an SSP-specific response (e.g., invalid frame format).

Table 215 defines the Response information unit used in a RESPONSE frame.

**Table 215 – RESPONSE frame - Response information unit**

Byte\Bit	7	6	5	4	3	2	1	0
0	Reserved							
...								
7								
8	STATUS QUALIFIER							
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)						
24	RESPONSE DATA (see table 217 in 8.2.2.5.3) (if any)							
...								
23+m								
24+m	SENSE DATA (if any)							
...								
23+m+n								

Table 216 defines the DATAPRES field, which specifies the format and content of the STATUS field, the STATUS QUALIFIER field, the SENSE DATA LENGTH field, the RESPONSE DATA LENGTH field, the RESPONSE DATA field, and the SENSE DATA field.

**Table 216 – DATAPRES field**

Code	Name	Description	Reference
00b	NO_DATA	No response data or sense data present	8.2.2.5.2
01b	RESPONSE_DATA	Response data present	8.2.2.5.3
10b	SENSE_DATA	Sense data present	8.2.2.5.4
11b	Reserved		

An SSP target port shall set the DATAPRES field to NO\_DATA in a RESPONSE frame sent for a command that completes without response data or sense data.

An SSP target port shall set the DATAPRES field to RESPONSE\_DATA in a RESPONSE frame sent for:

- a) every TASK frame; and
- b) every COMMAND frame for which errors occur while the transport layer is processing the frame (see 8.2.5.3).

An SSP target port shall set the DATAPRES field to SENSE\_DATA in a RESPONSE frame sent for a command that 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.

#### **8.2.2.5.2 Response information unit - NO\_DATA format**

If the DATAPRES field is set to NO\_DATA, then:

- a) the SSP target port shall set the STATUS field to the status code for a completed command (see SAM-5 for a list of status codes);
- b) the SSP target port shall set the STATUS QUALIFIER field to the status qualifier for the command (see SAM-5);
- c) the SSP target port shall set the SENSE DATA LENGTH field to 00000000h and the RESPONSE DATA LENGTH field to 00000000h;
- d) the SSP initiator port shall ignore the SENSE DATA LENGTH field and the RESPONSE DATA LENGTH field; and
- e) the SSP target port shall not include the SENSE DATA field and the RESPONSE DATA field.

#### **8.2.2.5.3 Response information unit - RESPONSE\_DATA format**

If the DATAPRES field is set to RESPONSE\_DATA, then:

- a) the SSP target port shall set the STATUS field to 00h, the STATUS QUALIFIER field to 0000h, and the SENSE DATA LENGTH field to 00000000h;
- b) the SSP initiator port shall ignore the STATUS field, the STATUS QUALIFIER field, and the SENSE DATA LENGTH field;
- c) the SSP target port shall not include the SENSE DATA field;
- d) the SSP target port shall set the RESPONSE DATA LENGTH field to 00000004h; and
- e) the SSP target port shall include the RESPONSE DATA field.



Table 217 defines the RESPONSE DATA field. The RESPONSE DATA field shall be present if the SSP target port detects any of the conditions described by a non-zero value in the RESPONSE CODE field and shall be present for a RESPONSE frame sent in response to a TASK frame.

**Table 217 – RESPONSE DATA field**

Byte\Bit	7	6	5	4	3	2	1	0
0	ADDITIONAL RESPONSE INFORMATION							
...								
2								
3	RESPONSE CODE							

The ADDITIONAL RESPONSE INFORMATION field contains additional response information for certain task management functions (e.g., QUERY ASYNCHRONOUS EVENT) as defined in SAM-5. If the task management function does not define additional response information or the logical unit does not support additional response information, then the SSP target port shall set the ADDITIONAL RESPONSE INFORMATION field to 000000h.

Table 218 defines the RESPONSE CODE field, which specifies the error condition or the completion status of a task management function. See 9.2.1.5 and 9.2.1.15 for the mapping of these response codes to SCSI service responses.

**Table 218 – RESPONSE CODE field**

Code	Description
00h	TASK MANAGEMENT FUNCTION COMPLETE <sup>a</sup>
02h	INVALID FRAME
04h	TASK MANAGEMENT FUNCTION NOT SUPPORTED <sup>a</sup>
05h	TASK MANAGEMENT FUNCTION FAILED <sup>a</sup>
08h	TASK MANAGEMENT FUNCTION SUCCEEDED <sup>a</sup>
09h	INCORRECT LOGICAL UNIT NUMBER <sup>a</sup>
0Ah	OVERLAPPED INITIATOR PORT TRANSFER TAG ATTEMPTED <sup>b</sup>
All others	Reserved
<sup>a</sup> Only valid when responding to a TASK frame. <sup>b</sup> Returned in case of command/task management function or task management function/task management function initiator port transfer tag conflicts.	

#### 8.2.2.5.4 Response information unit - SENSE\_DATA format

If the DATAPRES field is set to SENSE\_DATA, then:

- the SSP target port shall set the STATUS field to the status code for a completed command (see SAM-5 for a list of status codes);
- the SSP target port shall set the STATUS QUALIFIER field to the status qualifier for the command (see SAM-5);
- the SSP target port shall set the RESPONSE DATA LENGTH field to 00000000h;
- the SSP initiator port shall ignore the RESPONSE DATA LENGTH field;

- e) the SSP target port shall not include the RESPONSE DATA field;
- f) the SSP target port shall set the SENSE DATA LENGTH field to a non-zero value indicating the number of bytes in the SENSE DATA field. The value in the SENSE DATA LENGTH field shall not be greater than 1 000 (see table 207 in 8.2.1); and
- g) the SSP target port shall set the SENSE DATA field to the sense data (see SAM-5).

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

### 8.2.3 Sequences of SSP frames

#### 8.2.3.1 Sequences of SSP frames overview

Table 219 lists the sequences of SSP frames supporting the SCSI transport protocol services described in 9.2.1.

**Table 219 – Sequences of SSP frames**

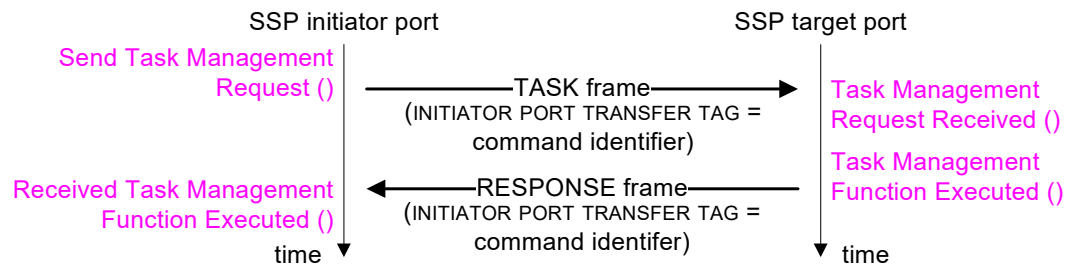
Sequence	Reference
Task management function	8.2.3.2
Non-data command	8.2.3.3
Write command	8.2.3.4
Read command	8.2.3.5
Bidirectional command	8.2.3.6

When multiple commands and/or task management functions are outstanding, frames from each of the individual sequences may be interleaved in any order. RESPONSE frames may be returned in any order (i.e., the order in which TASK frames and COMMAND frames are sent has no effect on the order that RESPONSE frames are returned).

Frames in a sequence may be transmitted during one or more connections (see 6.16) (e.g., for a write command using a single XFER\_RDY frame, the COMMAND frame may be transmitted in a connection originated by the SSP initiator port, the XFER\_RDY frame in a connection originated by the SSP target port, the DATA frames in one or more connections originated by the SSP initiator port, and the RESPONSE frame in a connection originated by the SSP target port. Alternatively, all the frames may be transmitted in one connection).

### 8.2.3.2 Task management function sequence of SSP frames

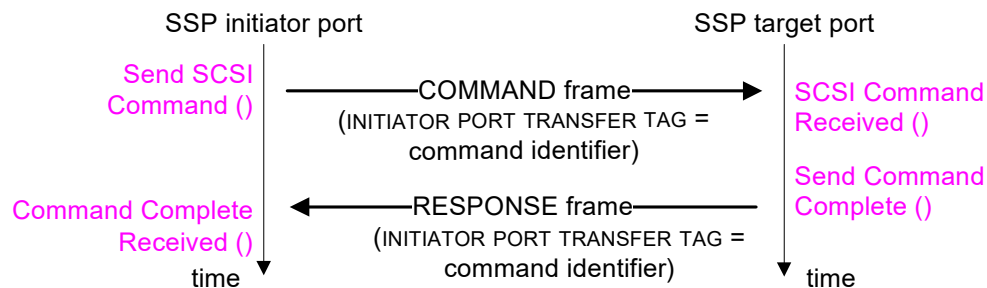
Figure 200 shows the sequence of SSP frames for a task management function (e.g., ABORT TASK (see SAM-5)), including the transport protocol services (see 9.2.1) invoked by the SCSI application layer.



**Figure 200 – Task management function sequence of SSP frames**

### 8.2.3.3 Non-data command sequence of SSP frames

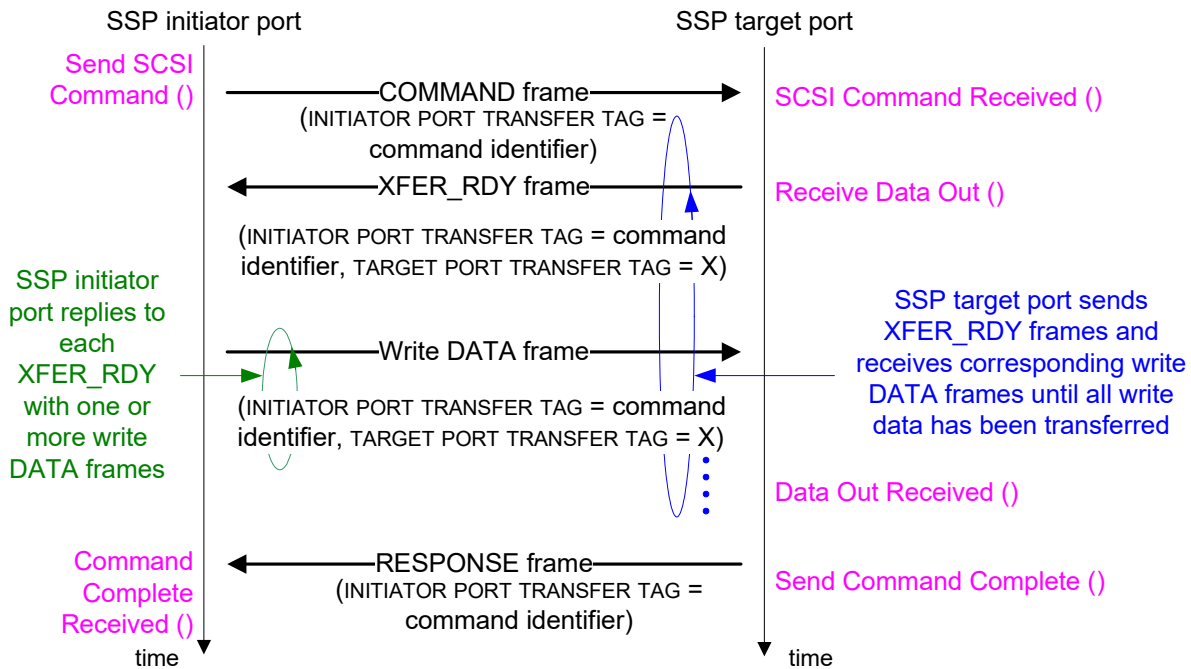
Figure 201 shows the sequence of SSP frames for a non-data command (e.g., TEST UNIT READY (see SPC-4)), including the transport protocol services (see 9.2.1) invoked by the SCSI application layer.



**Figure 201 – Non-data command sequence of SSP frames**

### 8.2.3.4 Write command sequence of SSP frames

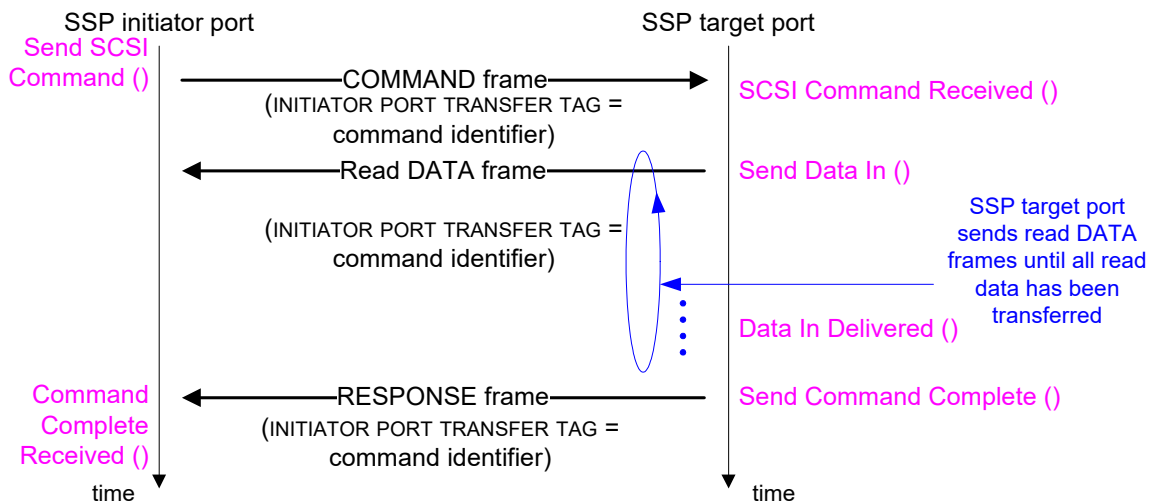
Figure 202 shows the sequence of SSP frames for a write command (e.g., MODE SELECT (see SPC-4)), including the transport protocol services (see 9.2.1) invoked by the SCSI application layer.



**Figure 202 – Write command sequence of SSP frames**

### 8.2.3.5 Read command sequence of SSP frames

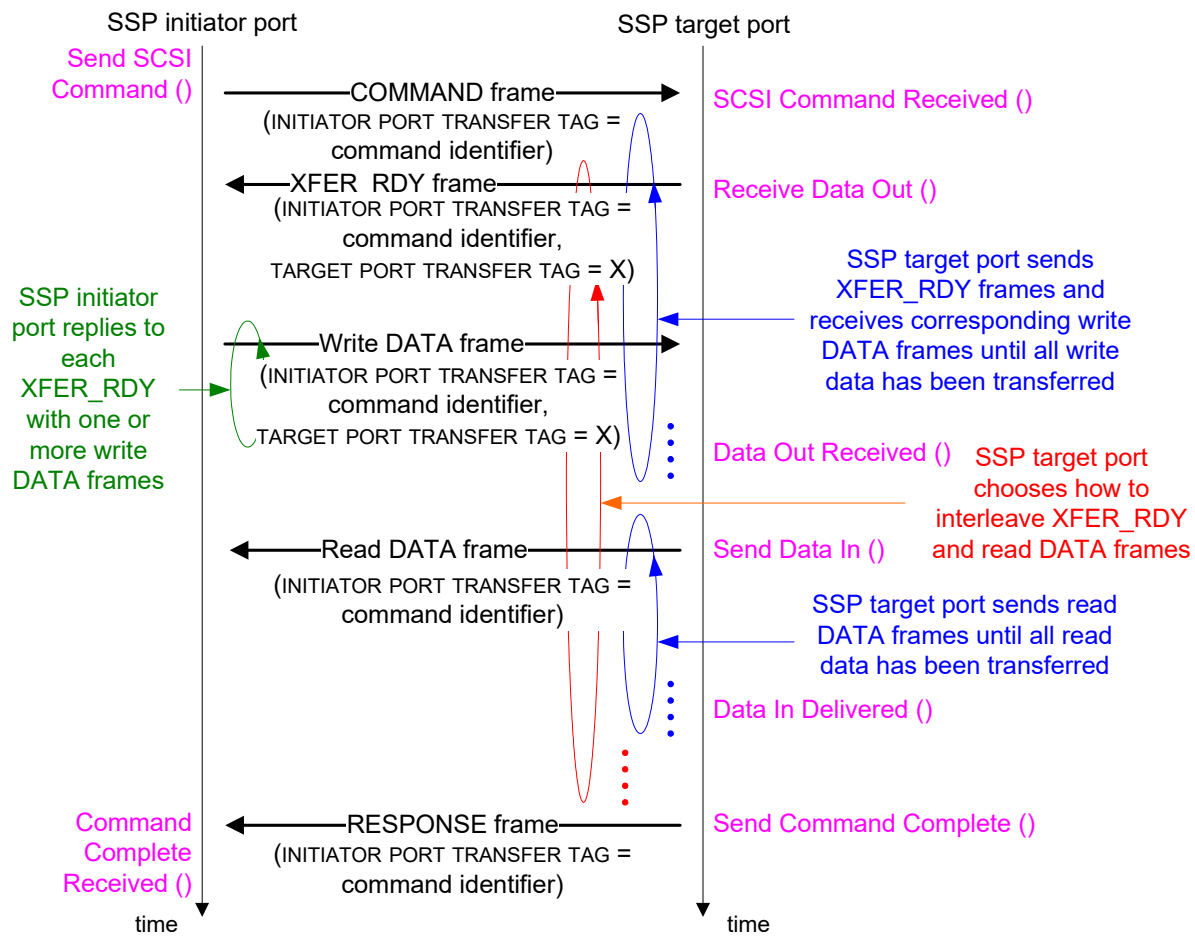
Figure 203 shows the sequence of SSP frames for a read command (e.g., INQUIRY, REPORT LUNS, or MODE SENSE (see SPC-4)), including the transport protocol services (see 9.2.1) invoked by the SCSI application layer.



**Figure 203 – Read command sequence of SSP frames**

### 8.2.3.6 Bidirectional command sequence of SSP frames

Figure 204 shows the sequence of SSP frames for a bidirectional command (e.g., XDWRITEREAD (see SBC-3)), including the transport protocol services (see 9.2.1) invoked by the SCSI application layer.



**Figure 204 – Bidirectional command sequence of SSP frames**

An SSP target port may transmit read DATA frames for a bidirectional command at the same time it is receiving write DATA frames for the same bidirectional command.

### 8.2.4 SSP transport layer handling of link layer errors

#### 8.2.4.1 SSP transport layer handling of link layer errors overview

The transport layer, sometimes assisted by the SCSI application layer, handles some link layer errors (e.g., NAKs and ACK/NAK timeouts). See 8.2.5 for transport layer handling of transport layer errors (e.g., invalid frame contents).

Link layer errors that occur when transmitting XFER\_RDY and DATA frames are processed based on the values in the TLR CONTROL field in the COMMAND frame header (see 8.2.1) and the TRANSPORT LAYER RETRIES bit in the Protocol Specific Logical Unit mode page (see 9.2.7.3) of the logical unit that is the source of the frame.

If transport layer retries are disabled, then the SSP target port:

- sets the RETRY DATA FRAMES bit to zero in each XFER\_RDY frame;

- b) may or may not select a different value for the TARGET PORT TRANSFER TAG field in each XFER\_RDY frame than that used in the previous XFER\_RDY frame for that I\_T\_L nexus and command identifier combination;
- c) processes XFER\_RDY frame link layer errors as described in 8.2.4.4.3;
- d) processes read DATA frame link layer errors as described in 8.2.4.5.3; and
- e) processes write DATA frame link layer errors as described in 8.2.4.6.3.

If transport layer retries are enabled, then the SSP target port:

- a) sets the RETRY DATA FRAMES bit to one in each XFER\_RDY frame;
- b) selects a different value for the TARGET PORT TRANSFER TAG field in each XFER\_RDY frame than that used in the previous XFER\_RDY frame for that I\_T\_L nexus and command identifier combination;
- c) processes XFER\_RDY frame link layer errors as described in 8.2.4.4.2;
- d) processes read DATA frame link layer errors as described in 8.2.4.5.2; and
- e) processes write DATA frame link layer errors as described in 8.2.4.6.2.

#### 8.2.4.2 COMMAND frame - handling of link layer errors

If an SSP initiator port transmits a COMMAND frame and receives a NAK for that frame, then the COMMAND frame was not received. The SSP initiator port should retransmit, in the same or in a new connection, the COMMAND frame at least one time (see 8.2.6.2.3.3). The SSP initiator port may reuse the initiator port transfer tag from the frame for which the NAK was received.

If an SSP initiator port transmits a COMMAND frame and does not receive an ACK or NAK for that frame (e.g., times out or the connection is broken), then:

- 1) the SSP\_TF state machine closes the connection with DONE (ACK/NAK TIMEOUT) (see 6.20.9.6.5);
- 2) to determine whether the command was received, the SCSI application client calls Send Task Management Request () (see 9.2.2) with:
  - A) Nexus set to the I\_T\_L nexus and the command identifier of the COMMAND frame; and
  - B) Function Identifier set to QUERY TASK;
 and
- 3) the SSP initiator port transmits the TASK frame in a new connection to the SSP target port.

If the command is a write command or a bidirectional command and the SSP initiator port receives an XFER\_RDY frame for the I\_T\_L nexus and command identifier combination of the command before the RESPONSE frame for the QUERY TASK, then the COMMAND frame was received and is being processed by the SSP target port, and the XFER\_RDY frame is valid.

If the command is a read command or a bidirectional command and the SSP initiator port receives a read DATA frame for the I\_T\_L nexus and command identifier combination of the command before the RESPONSE frame for the QUERY TASK, then the COMMAND frame was received and is being processed by the SSP target port, and the read DATA frame is valid.

If the SSP initiator port receives a RESPONSE frame for the I\_T\_L nexus and command identifier combination of the command before the RESPONSE frame for the QUERY TASK, then the COMMAND frame was received by the SSP target port, the RESPONSE frame is valid, and the command processing is complete. The SSP initiator port may reuse the initiator port transfer tag of the COMMAND frame.

If the SSP initiator port receives a RESPONSE frame for the QUERY TASK with a response code of TASK MANAGEMENT FUNCTION SUCCEEDED, then the COMMAND frame was received by the SSP target port (i.e., an ACK was transmitted by the SSP target port for the COMMAND frame) and the command is being processed.

If the SSP initiator port receives a RESPONSE frame for the QUERY TASK with a response code of TASK MANAGEMENT FUNCTION COMPLETE, then the COMMAND frame is not being processed. If neither an XFER\_RDY frame, a read DATA frame, nor a RESPONSE frame has been received for the I\_T\_L nexus and command identifier combination of the command, then the COMMAND frame was not received. The SSP initiator port should retransmit the COMMAND frame at least one time. The SSP initiator port may reuse the initiator port transfer tag of the COMMAND frame.

### 8.2.4.3 TASK frame - handling of link layer errors

If an SSP initiator port transmits a TASK frame and receives a NAK for that frame, then the TASK frame was not received. The SSP initiator port should retransmit, in the same or in a new connection, the TASK frame at least one time with the RETRANSMIT bit set to one (see 8.2.6.2.2.2). The SSP initiator port may reuse the initiator port transfer tag from the frame for which the NAK was received.

If an SSP initiator port transmits a TASK frame and does not receive an ACK or NAK for that frame (e.g., an ACK/NAK timeout occurs or the connection is broken), then:

- 1) the SSP\_TF state machine closes the connection with DONE (ACK/NAK TIMEOUT) (see 6.20.9.6.5);
- 2) the SCSI application client calls Send Task Management Request () using the same initiator port transfer tag (see 9.2.2); and
- 3) the SSP initiator port transmits the TASK frame with the RETRANSMIT bit set to one in a new connection to the SSP target port (see 8.2.6.2.2.2).

If the SSP initiator port receives a RESPONSE frame for the TASK frame that arrives before the ACK or NAK for the TASK frame, then the TASK frame was received by the SSP target port (i.e., an ACK was transmitted by the SSP target port for the TASK frame), the RESPONSE frame is valid, and the task management function is complete (see 8.2.6.2.2.3). The initiator port may reuse the initiator port transfer tag of the TASK frame.

### 8.2.4.4 XFER\_RDY frame - handling of link layer errors

#### 8.2.4.4.1 XFER\_RDY frame overview

If transport layer retries are enabled, then the SSP target port processes link layer errors that occur while transmitting XFER\_RDY frames as described in 8.2.4.4.2.

If transport layer retries are disabled, then the SSP target port processes link layer errors that occur while transmitting XFER\_RDY frames as described in 8.2.4.4.3.

#### 8.2.4.4.2 XFER\_RDY frame with transport layer retries enabled

If an SSP target port transmits an XFER\_RDY frame and receives a NAK for that frame, then the SSP target port retransmits, in the same or a new connection, the XFER\_RDY frame at least one time with:

- a) the TARGET PORT TRANSFER TAG field set to a different value than in the original XFER\_RDY frame;
- b) the RETRANSMIT bit set to one; and
- c) the other fields set to the same values as in the original XFER\_RDY frame (see 8.2.6.3.3.3).

If an SSP target port transmits an XFER\_RDY frame and does not receive an ACK or NAK for that frame (e.g., times out or the connection is broken), then:

- 1) the SSP\_TF state machine closes the connection with DONE (ACK/NAK TIMEOUT) (see 6.20.9.6.5); and
- 2) the SSP target port retransmits, in a new connection, the XFER\_RDY frame with:
  - A) the TARGET PORT TRANSFER TAG field set to a different value than in the original XFER\_RDY frame;
  - B) the RETRANSMIT bit set to one; and
  - C) the other fields set to the same values as in the original XFER\_RDY frame (see 8.2.6.3.3.3).

If an SSP initiator port is processing an XFER\_RDY frame for an I\_T\_L nexus and command identifier combination (e.g., transmitting write DATA frames for the command) and the SSP initiator port receives a new XFER\_RDY frame for that I\_T\_L nexus and command identifier combination with the RETRANSMIT bit set to one in the frame, then the ST\_ITS state machine stops processing the original XFER\_RDY frame (i.e., stops transmitting write DATA frames) and starts servicing the new XFER\_RDY frame (see 8.2.6.2.3). The ST\_ITS state machine does not transmit any write DATA frames for the original XFER\_RDY frame after transmitting a write DATA frame for the new XFER\_RDY frame.

An SSP target port may reuse the value in the TARGET PORT TRANSFER TAG field from an XFER\_RDY frame for an I\_T\_L nexus and command identifier combination after the SSP target port receives a write DATA frame for a subsequent XFER\_RDY frame for that I\_T\_L nexus and command identifier combination.

An SSP target port retransmits each XFER\_RDY frame that does not receive an ACK or NAK at least one time.

The number of times an SSP target port retransmits each XFER\_RDY frame is vendor specific. When the SSP target port reaches its vendor specific limit, the SSP target port follows the procedure for transport layer retries disabled described in 8.2.4.4.3.

#### **8.2.4.4.3 XFER\_RDY frame with transport layer retries disabled**

If an SSP target port transmits an XFER\_RDY frame and receives a NAK for that frame, then:

- 1) the SCSI device server calls Send Command Complete () to return CHECK CONDITION status for that command with the sense key set to ABORTED COMMAND and the additional sense code set to NAK RECEIVED (see 9.2.3); and
- 2) the SSP target port transmits the RESPONSE frame in the same or a new connection.

If an SSP target port transmits an XFER\_RDY frame and does not receive an ACK or NAK for that frame (e.g., times out or the connection is broken), then:

- 1) the SSP\_TF state machine closes the connection with DONE (ACK/NAK TIMEOUT) (see 6.20.9.6.5);
- 2) the SCSI device server calls Send Command Complete () to return CHECK CONDITION status for that command with the sense key set to ABORTED COMMAND and the additional sense code set to ACK/NAK TIMEOUT or CONNECTION LOST (see 9.2.3); and
- 3) the SSP target port transmits the RESPONSE frame in a new connection.

#### **8.2.4.5 Read DATA frame - handling of link layer errors**

##### **8.2.4.5.1 Read DATA frame overview**

If an SSP target port transmits a read DATA frame for a command with transport layer retries enabled, then the SSP target port processes link layer errors that occur while transmitting read DATA frames as described in 8.2.4.5.2.

If an SSP target port transmits a read DATA frame for a command with transport layer retries disabled, then the SSP target port processes link layer errors that occur while transmitting read DATA frames as described in 8.2.4.5.3.

##### **8.2.4.5.2 Read DATA frame with transport layer retries enabled**

If an SSP target port transmits a read DATA frame and receives a NAK for that frame, then the read DATA frame was not received. The SSP target port retransmits, in the same or in a new connection, all the read DATA frames for that I\_T\_L nexus and command identifier combination since a previous time when ACK/NAK balance occurred at least one time (see 8.2.6.3.3.4).

If an SSP target port transmits a read DATA frame and does not receive an ACK or NAK for that frame (e.g., an ACK/NAK timeout occurs or the connection is broken), then:

- 1) the SSP\_TF state machine closes the connection with DONE (ACK/NAK TIMEOUT) (see 6.20.9.6.5); and
- 2) the ST\_TTS state machine retransmits, in a new connection, all the read DATA frames for that I\_T\_L nexus and command identifier combination since a previous time when ACK/NAK balance occurred at least one time (see 8.2.6.3.3.4).

The CHANGING DATA POINTER bit is set to one in the first retransmitted read DATA frame and the CHANGING DATA POINTER bit is set to zero in subsequent read DATA frames.

The ST\_TTS state machine retransmits each read DATA frame that does not receive an ACK at least one time (see 8.2.6.3.3).

The number of times an SSP target port retransmits each read DATA frame is vendor specific. When an SSP target port reaches its vendor specific limit for retransmitting read DATA frames, the SSP target port follows the procedure for transport layer retries disabled described in 8.2.4.5.3.



### 8.2.4.5.3 Read DATA frame with transport layer retries disabled

If an SSP target port transmits a read DATA frame and receives a NAK for that frame, then:

- 1) the SCSI device server calls Send Command Complete () to return CHECK CONDITION status for that command with the sense key set to ABORTED COMMAND and the additional sense code set to NAK RECEIVED (see 9.2.3); and
- 2) the SSP target port transmits the RESPONSE frame in the same or a new connection.

If an SSP target port transmits a read DATA frame and does not receive an ACK or NAK for that frame (e.g., an ACK/NAK timeout occurs), then:

- 1) the SSP\_TF state machine closes the connection with DONE (ACK/NAK TIMEOUT) (see 6.20.9.6.5);
- 2) the SCSI device server calls Send Command Complete () to return CHECK CONDITION status for that command with the sense key set to ABORTED COMMAND and the additional sense code set to ACK/NAK TIMEOUT (see 9.2.3); and
- 3) the SSP target port transmits the RESPONSE frame in a new connection.

### 8.2.4.6 Write DATA frame - handling of link layer errors

#### 8.2.4.6.1 Write DATA frame overview

An SSP initiator port processes link layer errors that occur while transmitting write DATA frames transmitted in response to an XFER\_RDY frame that has its RETRY DATA FRAMES bit set to one as described in 8.2.4.6.2.

An SSP initiator port processes link layer errors that occur while transmitting write DATA frames in response to an XFER\_RDY frame that has its RETRY DATA FRAMES bit set to zero as described in 8.2.4.6.3.

#### 8.2.4.6.2 Write DATA frame with transport layer retries enabled

If an SSP initiator port transmits a write DATA frame and receives a NAK for that frame, then the write DATA frame was not received. The SSP\_ITS state machine retransmits, in the same or in a new connection, all the write DATA frames for the previous XFER\_RDY frame (see 8.2.6.2.3.3.2).

If an SSP initiator port transmits a write DATA frame and does not receive an ACK or NAK for that frame (e.g., an ACK/NAK timeout occurs or the connection is broken), then:

- 1) the SSP\_TF state machine closes the connection with DONE (ACK/NAK TIMEOUT) (see 6.20.9.6.5); and
- 2) the ST\_ITS state machine retransmits, in a new connection, all the write DATA frames for the previous XFER\_RDY frame (see 8.2.6.2.3.3.2).

If that SSP initiator port receives a new XFER\_RDY frame or a RESPONSE frame for the command while retransmitting or preparing to retransmit the write DATA frames, then the ST\_IFR state machine and ST\_ITS state machine process the XFER\_RDY frame or RESPONSE frame and stop retransmitting the write DATA frames (see 8.2.6.2.2 and 8.2.6.2.3). The ST\_ITS state machine does not transmit a write DATA frame for the previous XFER\_RDY frame after transmitting a write DATA frame in response to the new XFER\_RDY frame.

The CHANGING DATA POINTER bit is set to one in the first retransmitted write DATA frame and the CHANGING DATA POINTER bit is set to zero in subsequent write DATA frames.

The ST\_ITS state machine retransmits each write DATA frame that does not receive an ACK at least one time (see 8.2.6.2.3).

The number of times an SSP initiator port retransmits each write DATA frame is vendor specific. When an SSP initiator port reaches its vendor specific limit for retransmitting write DATA frames, the SSP initiator port follows the procedure for transport layer retries disabled described in 8.2.4.6.3.

#### 8.2.4.6.3 Write DATA frame with transport layer retries disabled

If an SSP initiator port transmits a write DATA frame and does not receive an ACK or NAK for that frame (e.g., an ACK/NAK timeout occurs or the connection is broken), then:

- 1) the SSP\_TF state machine closes the connection with DONE (ACK/NAK TIMEOUT) (see 6.20.9.6.5); and
- 2) the SCSI application client aborts the command (see 9.2.2).

If an SSP initiator port transmits a write DATA frame and receives a NAK for that frame, then the SCSI application client aborts the command (see 9.2.2).

#### 8.2.4.7 RESPONSE frame - handling of link layer errors

If an SSP target port transmits a RESPONSE frame and receives a NAK for that frame, then the SSP target port retransmits, in the same or a new connection, the RESPONSE frame at least one time with the RETRANSMIT bit set to one and with the other fields set to the same values as in the original RESPONSE frame (see 8.2.6.3.3.3).

If an SSP target port transmits a RESPONSE frame and does not receive an ACK or NAK for that frame (e.g., an ACK/NAK timeout occurs or the connection is broken), then:

- 1) the SSP\_TF state machine closes the connection with DONE (ACK/NAK TIMEOUT) (see 6.20.9.6.5); and
- 2) the SSP target port retransmits, in a new connection, the RESPONSE frame with:
  - A) the RETRANSMIT bit set to one; and
  - B) the other fields set to the same values as in the original RESPONSE frame (see 8.2.6.3.3.3).

The ST\_TTS state machine retransmits each RESPONSE frame that does not receive an ACK at least one time (see 8.2.6.3.3). The number of times an SSP target port retransmits each RESPONSE frame is vendor specific.

If an SSP initiator port receives a RESPONSE frame with a RETRANSMIT bit set to one and the SSP initiator port has previously received a RESPONSE frame for the same I\_T\_L nexus and command identifier combination, then the ST\_IFR state machine discards the extra RESPONSE frame (see 8.2.6.2.2.3). If the ST\_IFR state machine has not previously received a RESPONSE frame for the I\_T\_L nexus and command identifier combination, then the state machine processes the RESPONSE frame.

### 8.2.5 SSP transport layer error handling summary

#### 8.2.5.1 SSP transport layer error handling summary introduction

This subclause contains a summary of how SSP ports process transport layer errors. This summary does not include every error case. See 8.2.4 for transport layer handling of link layer errors (e.g., using transport layer retries).

#### 8.2.5.2 SSP initiator port transport layer error handling summary

If an SSP initiator port receives a COMMAND frame, TASK frame, or an unsupported frame type, then the ST\_IFR state machine discards the frame (see 8.2.6.2.2.3).

If an SSP initiator port receives an XFER\_RDY frame, read DATA frame, or RESPONSE frame with an unknown INITIATOR PORT TRANSFER TAG field value, then the ST\_IFR state machine discards the frame (see 8.2.6.2.3.7). The SCSI application client may then abort the command with that initiator port transfer tag.

If an SSP initiator port receives an XFER\_RDY frame with a Transfer Ready information unit that is not 12 bytes long, then the ST\_IFR state machine discards the frame (see 8.2.6.2.3.7). The SCSI application client may then abort the command.

If an SSP initiator port receives an XFER\_RDY frame in response to a command that does not specify write data, then the ST\_IFR state machine discards the frame (see 8.2.6.2.2.3) and the SCSI application client aborts the command (see 9.2.2).

If an SSP initiator port receives an XFER\_RDY frame requesting more write data than expected, then the ST\_ITS state machine discards the frame (see 8.2.6.2.3.3) and the SCSI application client aborts the command (see 9.2.2).

If an SSP initiator port receives an XFER\_RDY frame requesting zero bytes, then the ST\_ITS state machine discards the frame (see 8.2.6.2.3.3) and the SCSI application client aborts the command (see 9.2.2).

If transport layer retries are disabled and an SSP initiator port receives an XFER\_RDY frame with a requested offset that was not expected, then the ST\_ITS state machine discards the frame (see 8.2.6.2.3.3) and the SCSI application client aborts the command (see 9.2.2).

If an SSP initiator port receives a read DATA frame in response to a command with no read data, then the ST\_IFR state machine discards the frame (see 8.2.6.2.2.3) and the SCSI application client aborts the command (see 9.2.2).

If an SSP initiator port receives a read DATA frame with more read data than expected, then the ST\_ITS state machine discards the frame (see 8.2.6.2.3.3) and the SCSI application client aborts the command (see 9.2.2). The SSP initiator port may receive a RESPONSE frame for the command before being able to abort the command.

If an SSP initiator port receives a read DATA frame with zero bytes, then the ST\_ITS state machine discards the frame (see 8.2.6.2.3.3) and the SCSI application client aborts the command (see 9.2.2). The SSP initiator port may receive a RESPONSE frame for the command before being able to abort the command.

If transport layer retries are disabled and an SSP initiator port receives a read DATA frame with a data offset that was not expected, then:

- a) the ST\_ITS state machine discards that frame and any subsequent read DATA frames received for that command (see 8.2.6.2.3.7); and
- b) the SCSI application client aborts the command (see 9.2.2).

The SSP initiator port may receive a RESPONSE frame for the command before being able to abort the command.

If an SSP initiator port receives a RESPONSE frame that is not the correct length, then the ST\_IFR state machine considers the command or task management function completed with an error and discards the frame (see 8.2.6.2.2.3).

### 8.2.5.3 SSP target port transport layer error handling summary

If an SSP target port receives an XFER\_RDY or RESPONSE frame or another unsupported frame type, then the ST\_TFR state machine discards the frame (see 8.2.6.3.2.2).

If an SSP target port receives a COMMAND frame and:

- a) the frame is too short to contain a LOGICAL UNIT NUMBER field;
- b) the frame is too short to contain a CDB;
- c) the ADDITIONAL CDB LENGTH field specifies that the frame should be a different length; or
- d) the TLR CONTROL field is set to a non-zero value and non-zero values are not supported,

then the ST\_TTS state machine returns a RESPONSE frame with the DATAPRES field set to RESPONSE\_DATA and the RESPONSE CODE field set to INVALID FRAME (see 8.2.6.3.2.2).

If an SSP target port receives a TASK frame that is too short, then the ST\_TTS state machine returns a RESPONSE frame with the DATAPRES field set to RESPONSE\_DATA and the RESPONSE CODE field set to INVALID FRAME (see 8.2.6.3.2.2).

If an SSP target port receives a COMMAND frame with an initiator port transfer tag that is already in use for another command and the SSP target port implements initiator port transfer tag checking, then the task router and task managers return CHECK CONDITION status with the sense key set to ABORTED COMMAND and the additional sense code set to OVERLAPPED COMMANDS ATTEMPTED (see 9.2.4).

If an SSP target port receives:

- a) a COMMAND frame with an initiator port transfer tag that is already in use for a task management function; or
- b) a TASK frame with an initiator port transfer tag that is already in use for a command or another task management function,

then the task router and task managers return a RESPONSE frame with the RESPONSE CODE field set to OVERLAPPED INITIATOR PORT TRANSFER TAG ATTEMPTED (see 9.2.4).

If an SSP target port receives a write DATA frame with an unknown initiator port transfer tag, then the ST\_TFR state machine discards the frame (see 8.2.6.3.2).

If an SSP target port receives a write DATA frame that does not contain first burst data and for which there is no XFER\_RDY frame outstanding (i.e., it has received all requested write data), then the ST\_TFR state machine discards the frame (see 8.2.6.3.2.2).

If an SSP target port receives a TASK frame with an unknown logical unit number, then the ST\_TFR state machine returns a RESPONSE frame with the DATAPRES field set to RESPONSE\_DATA and the RESPONSE CODE field set to INCORRECT LOGICAL UNIT NUMBER (see 8.2.6.3.2.2).

If an SSP target port receives a COMMAND frame or TASK frame with a TARGET PORT TRANSFER TAG field set to a value other than FFFFh, then the ST\_TFR state machine may return a RESPONSE frame with the DATAPRES field set to RESPONSE\_DATA and the RESPONSE CODE field set to INVALID FRAME (see 8.2.6.3.2.2).

If an SSP target port is using target port transfer tags and receives a write DATA frame with an unknown target port transfer tag, then the ST\_TFR state machine discards the frame (see 8.2.6.3.3).

If transport layer retries are disabled and an SSP target port receives a write DATA frame with a data offset that was not expected, then:

- a) the ST\_TTS state machine discards the frame (see 8.2.6.3.3.6.1); and
- b) the SCSI device server terminates the command with CHECK CONDITION status with the sense key set to ABORTED COMMAND and the additional sense code set to DATA OFFSET ERROR (see 9.2.3).

If an SSP target port receives a write DATA frame with more write data than expected (i.e., the write DATA frame contains data in excess of that requested by an XFER\_RDY frame or, for first burst data, indicated by the FIRST BURST LENGTH field in the Disconnect-Reconnect mode page), then:

- a) the ST\_TTS state machine discards the frame (see 8.2.6.3.3.6.1); and
- b) the SCSI device server terminates the command with CHECK CONDITION status with the sense key set to ABORTED COMMAND and the additional sense code set to TOO MUCH WRITE DATA (see 9.2.3).

If an SSP target port receives a write DATA frame with zero bytes, then:

- a) the ST\_TTS state machine discards the frame (see 8.2.6.3.3.6.1); and
- b) the SCSI device server terminates the command with CHECK CONDITION status with the sense key set to ABORTED COMMAND and the additional sense code set to INFORMATION UNIT TOO SHORT (see 9.2.3).

## 8.2.6 ST (transport layer for SSP ports) state machines

### 8.2.6.1 ST state machines overview

The ST state machines perform the following functions:

- a) receive and process transport protocol service requests and transport protocol service responses from the SCSI application layer;
- b) receive and process other SAS connection management requests from the SCSI application layer;
- c) send transport protocol service indications and transport protocol service confirmations to the SCSI application layer;
- d) send requests to the port layer to transmit frames and manage SAS connections; and
- e) receive confirmations from the port layer.

The following confirmations between the ST state machines and the port layer:

- a) Transmission Status; and
- b) Frame Received,

include the following as arguments:

- a) initiator port transfer tag;
- b) Destination SAS Address; and
- c) Source SAS Address,

and are used to route the confirmations to the correct ST state machines.

NOTE 57 - Although allowed by this standard, the ST state machines do not handle bidirectional commands that result in concurrent write DATA frames and read DATA frames.

## 8.2.6.2 ST\_I (transport layer for SSP initiator ports) state machines

### 8.2.6.2.1 ST\_I state machines overview

The ST\_I state machines are as follows:

- a) ST\_IFR (initiator frame router) state machine (see 8.2.6.2.2); and
- b) ST\_ITS (initiator transport server) state machine (see 8.2.6.2.3).

Each SAS initiator port includes:

- a) one ST\_IFR state machine; and
- b) one ST\_ITS state machine for each possible command and task management function (i.e., for each initiator port transfer tag).

Figure 205 shows the ST\_I state machines.

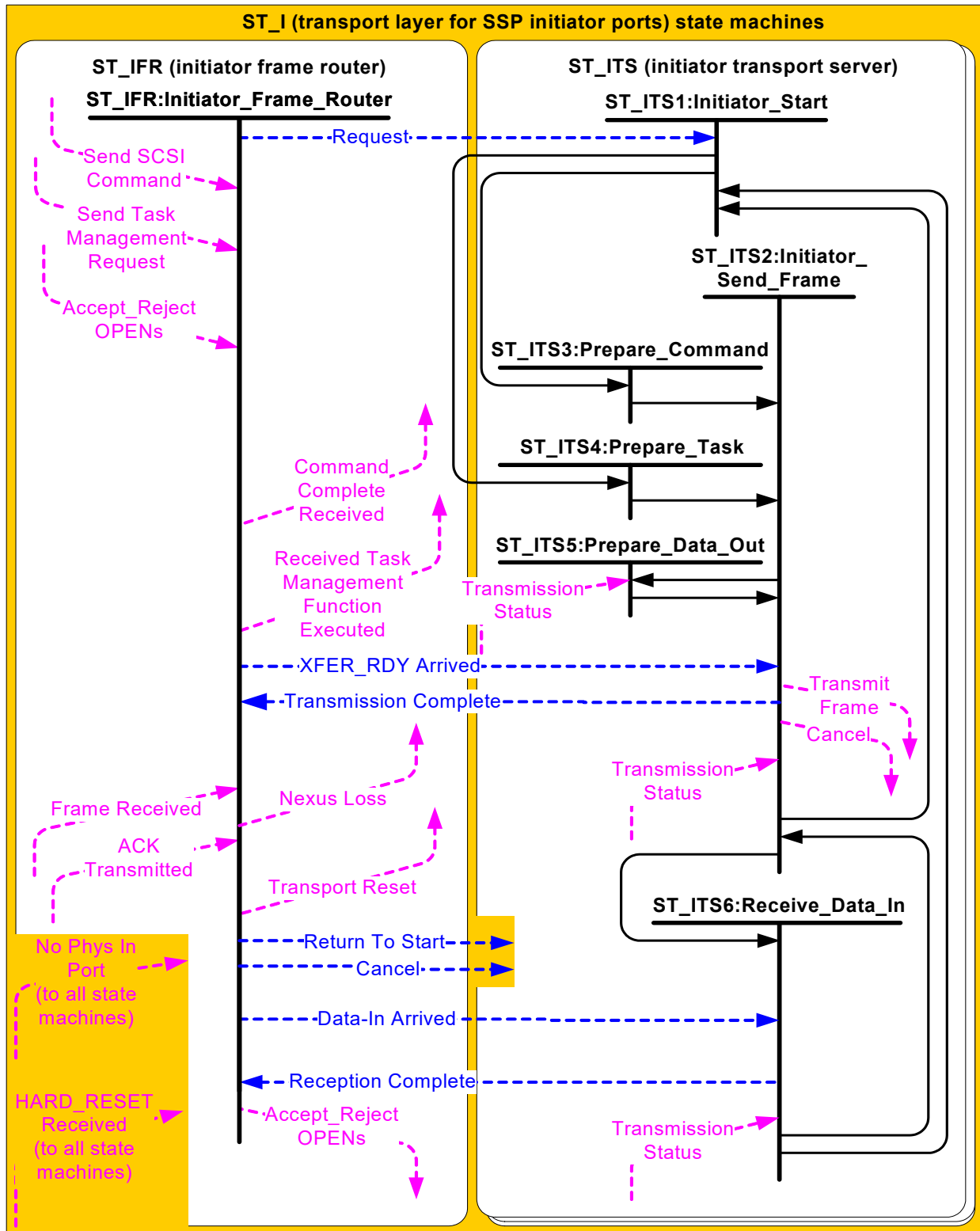


Figure 205 – ST\_I (transport layer for SSP initiator ports) state machines

## 8.2.6.2.2 ST\_IFR (initiator frame router) state machine

### 8.2.6.2.2.1 ST\_IFR state machine overview

The ST\_IFR state machine performs the following functions:

- a) receives Send SCSI Command and Send Task Management transport protocol service requests from the SCSI application layer;
- b) sends messages to the ST\_ITS state machine;
- c) receives messages from the ST\_ITS state machine;
- d) receives confirmations from the port layer;
- e) sends transport protocol service confirmations to the SCSI application layer;
- f) receives vendor specific requests from the SCSI application layer;
- g) sends vendor specific confirmations to the SCSI application layer;
- h) receives Accept\_Reject OPENs requests from the SCSI application layer;
- i) sends Accept\_Reject OPENs requests to the port layer;
- j) sends Nexus Loss event notifications to the SCSI application layer; and
- k) sends Transport Reset event notifications to the SCSI application layer.

This state machine consists of one state.

This state machine shall be started after power on.

### 8.2.6.2.2.2 Processing transport protocol service requests

If this state machine receives a Send SCSI Command transport protocol service request, then this state machine shall send a Request (Send Command) message with Command arguments and Application Client Buffer arguments to the ST\_ITS state machine for the specified initiator port transfer tag.

The following is the list of Command arguments:

- a) connection rate;
- b) initiator connection tag;
- c) Destination SAS Address;
- d) Source SAS Address set to the SAS address of the SSP initiator port;
- e) initiator port transfer tag;
- f) logical unit number;
- g) command priority;
- h) task attribute;
- i) additional CDB length;
- j) CDB;
- k) additional CDB bytes, if any;
- l) first burst enabled; and
- m) request fence.

The following is the list of Application Client Buffer arguments:

- a) data-in buffer size;
- b) data-out buffer; and
- c) data-out buffer size.

If the command specifies a write operation and the Send SCSI Command transport service request contains a First Burst Enabled argument, then the Request (Send Command) message shall also include the Enable First Burst argument and the number of bytes for the First Burst Size argument.

If this state machine receives a Send Task Management Request transport protocol service request, then this state machine shall send a Request (Send Task) message with the Task arguments to the ST\_ITS state machine for the specified initiator port transfer tag.

The following is the list of Task arguments:

- a) connection rate;

- b) initiator connection tag;
- c) Source SAS Address set to the SAS address of the SSP initiator port;
- d) Destination SAS Address;
- e) retransmit bit;
- f) initiator port transfer tag;
- g) logical unit number;
- h) task management function;
- i) initiator port transfer tag to manage; and
- j) request fence.

If the ST\_ITS state machine for the initiator port transfer tag specified in the Send Task Management Request is currently in use, then this state machine shall set the retransmit bit argument to one. If the ST\_ITS state machine for the initiator port transfer tag specified in the Send Task Management Request is not currently in use, then this state machine shall set the retransmit bit argument to zero.

#### 8.2.6.2.2.3 Processing Frame Received confirmations

If this state machine receives a Frame Received (ACK/NAK Balanced) confirmation or Frame Received (ACK/NAK Not Balanced) confirmation, then this state machine shall compare the frame type of the frame received with the received confirmation (see table 207 in 8.2.1). If the confirmation was Frame Received (ACK/NAK Balanced) and the frame type is not XFER\_RDY, RESPONSE, or DATA, then this state machine shall discard the frame. If the confirmation was Frame Received (ACK/NAK Not Balanced) and the frame type is not DATA, then this state machine shall discard the frame.

If the frame type is correct relative to the Frame Received confirmation, then this state machine may check that the hashed source SAS address matches the SAS address of the SAS port that transmitted the frame and that the hashed destination SAS address matches the SAS address of the SAS port that received the frame based on the connection information. If this state machine checks these SAS addresses and they do not match, then this state machine:

- a) shall discard the frame; and
- b) may send a vendor specific confirmation to the SCSI application layer to cause the command using that initiator port transfer tag to be aborted.

If the frame type is XFER\_RDY, then this state machine shall check the length of the information unit. If the length of the information unit is not correct, then this state machine:

- a) shall discard the frame; and
- b) may send a vendor specific confirmation to the SCSI application layer to cause the command using that initiator port transfer tag to be aborted.

If the frame type is XFER\_RDY and the initiator port transfer tag is for a command with no write data, then this state machine shall:

- a) discard the frame;
- b) send a Command Complete Received transport protocol service confirmation with the Service Response argument set to Service Delivery or Target Failure - XFER\_RDY Not Expected to the SCSI application layer; and
- c) if there is an ST\_ITS state machine for the initiator port transfer tag, then send a Return To Start message to that state machine.

If the frame type is DATA and the initiator port transfer tag is for a command with no read data, then this state machine shall:

- a) discard the frame;
- b) send a Command Complete Received transport protocol service confirmation with the Service Response argument set to Service Delivery or Target Failure - DATA Not Expected to the SCSI application layer; and
- c) if there is an ST\_ITS state machine for the initiator port transfer tag, then send a Return To Start message to that state machine.



If the frame type is RESPONSE, then this state machine shall check the length of the information unit. If the length of the information unit is not correct and the RESPONSE frame was for a command, then this state machine shall discard the frame and send a Command Complete Received transport protocol service confirmation to the SCSI application layer with the Service Response argument set to Service Delivery or Target Failure - RESPONSE Incorrect Length. If the length of the information unit is not correct and the RESPONSE frame was for a task management function, then this state machine shall discard the frame and send a Received Task Management Function Executed transport protocol service confirmation to the SCSI application layer with the Service Response argument set to Service Delivery or Target Failure - RESPONSE Incorrect Length.

If the frame type is correct relative to the Frame Received confirmation, then this state machine shall check the initiator port transfer tag. If the initiator port transfer tag does not specify a valid ST\_ITS state machine, then this state machine shall discard the frame and may send a vendor specific confirmation to the SCSI application layer to cause the command using that initiator port transfer tag to be aborted.

If the frame type is RESPONSE and this state machine has previously received a RESPONSE frame for the I\_T\_L nexus and command identifier combination, then this state machine shall discard the frame.

If the frame type is RESPONSE, the fields checked in the frame are correct, and this state machine has not previously received a RESPONSE frame for this I\_T\_L nexus and command identifier combination, then this state machine shall send a Return To Start message to the ST\_ITS state machine for the specified initiator port transfer tag and:

- a) if the RESPONSE frame was for a command, then this state machine shall send a Command Complete Received transport protocol service confirmation to the SCSI application layer with the arguments set as specified in table 245; or
- b) if the RESPONSE frame was for a task management request, then this state machine shall send a Received Task Management Function Executed transport protocol service confirmation to the SCSI application layer with the arguments set as specified in table 255.

If the frame type is XFER\_RDY and the fields checked in the frame are correct, then this state machine shall wait to receive an ACK Transmitted confirmation.

If this state machine receives an ACK Transmitted confirmation for an XFER\_RDY frame, then it shall send an XFER\_RDY Arrived message to the ST\_ITS state machine specified by the initiator port transfer tag. The message shall include the following Xfer\_Rdy arguments:

- a) retry data frames;
- b) retransmit bit;
- c) target port transfer tag;
- d) requested offset; and
- e) write data length.

If the frame type is DATA and the fields checked in the frame are correct, then this state machine shall send a Data-In Arrived message to the ST\_ITS state machine specified by the initiator port transfer tag. The message shall include the following Read Data arguments:

- a) changing data pointer;
- b) number of fill bytes;
- c) data offset; and
- d) data.

#### 8.2.6.2.2.4 Processing Transmission Complete and Reception Complete messages

If this state machine receives a Transmission Complete (I\_T Nexus Loss) message, then this state machine shall send a Nexus Loss event notification to the SCSI application layer.

Table 220 defines the transport protocol service confirmation and Delivery Result argument generated as a result of receiving a Transmission Complete message or a Reception Complete message indicating that an error occurred during the transmission or reception of a frame.

**Table 220 – Confirmations sent to the SCSI application layer if a frame transmission error or reception error occurs**

<b>Message received from ST_ITS state machine</b>	<b>Transport protocol service confirmation and Delivery Result argument sent to the SCSI application layer</b>
Transmission Complete (Command Failed, ACK/NAK Timeout)	Command Complete Received (Service Delivery or Target Failure - ACK/NAK Timeout)
Transmission Complete (Command Failed, Connection Failed)	Command Complete Received (Service Delivery or Target Failure - Connection Failed)
Transmission Complete (Command Failed, NAK Received)	Command Complete Received (Service Delivery or Target Failure - NAK Received)
Transmission Complete (Task Failed, ACK/NAK Timeout)	Received Task Management Function Executed (Service Delivery or Target Failure - ACK/NAK Timeout)
Transmission Complete (Task Failed, Connection Failed)	Received Task Management Function Executed (Service Delivery or Target Failure - Connection Failed)
Transmission Complete (Task Failed, NAK Received)	Received Task Management Function Executed (Service Delivery or Target Failure - NAK Received)
Transmission Complete (XFER_RDY Incorrect Write Data Length)	Command Complete Received (Service Delivery or Target Failure - XFER_RDY Incorrect Write Data Length)
Transmission Complete (XFER_RDY Requested Offset Error)	Command Complete Received (Service Delivery or Target Failure - XFER_RDY Requested Offset Error)
Transmission Complete (Cancel Acknowledged)	Command Complete Received (Service Delivery or Target Failure - Cancel Acknowledged)
Reception Complete (Command Failed, ACK/NAK Timeout)	Command Complete Received (Service Delivery or Target Failure - ACK/NAK Timeout)
Reception Complete (Data Offset Error)	Command Complete Received (Service Delivery or Target Failure - DATA Data Offset Error)
Reception Complete (Too Much Read Data)	Command Complete Received (Service Delivery or Target Failure - DATA Too Much Read Data)
Reception Complete (Incorrect Data Length)	Command Complete Received (Service Delivery or Target Failure - DATA Incorrect Data Length)
Reception Complete (Cancel Acknowledged)	Command Complete Received (Service Delivery or Target Failure - Cancel Acknowledged)

The transport protocol service confirmation shall include the initiator port transfer tag as an argument.

#### **8.2.6.2.2.5 Processing miscellaneous requests**

If this state machine receives an Accept\_Reject OPENs (Accept SSP) request or an Accept\_Reject OPENs (Reject SSP) request, then this state machine shall send an Accept\_Reject OPENs request with the same arguments to the port layer.

If this state machine receives a HARD\_RESET Received confirmation, then this state machine shall send a Transport Reset event notification to the SCSI application layer.

If this state machine receives a No Phys In Port confirmation, then this state machine shall send a Command Complete Received (Service Delivery or Target Failure - Connection Failed) or Received Task Management Function Executed (Service Delivery or Target Failure - Connection Failed) confirmation to the SCSI application layer for each ST\_ITS state machine that is not in the ST\_ITS1:Initiator\_Start state.

This state machine may receive vendor specific requests from the SCSI application layer that cause this state machine to send a Cancel message to an ST\_ITS state machine.

### 8.2.6.2.3 ST\_ITS (initiator transport server) state machine

#### 8.2.6.2.3.1 ST\_ITS state machine overview

The ST\_ITS state machine performs the following functions:

- a) receives and processes messages from the ST\_IFR state machine;
- b) sends messages to the ST\_IFR state machine;
- c) sends request to the port layer regarding frame transmission;
- d) receives confirmations from the port layer regarding frame transmission; and
- e) receives HARD\_RESET Received confirmations and No Phys In Port confirmations from the port layer.

This state machine consists of the following states:

- a) ST\_ITS1:Initiator\_Start state (see 8.2.6.2.3.2) (initial state);
- b) ST\_ITS2:Initiator\_Send\_Frame state (see 8.2.6.2.3.3);
- c) ST\_ITS3:Prepare\_Command state (see 8.2.6.2.3.4);
- d) ST\_ITS4:Prepare\_Task state (see 8.2.6.2.3.5);
- e) ST\_ITS5:Prepare\_Data\_Out state (see 8.2.6.2.3.6); and
- f) ST\_ITS6:Receive\_Data\_In state (see 8.2.6.2.3.7).

This state machine shall start in the ST\_ITS1:Initiator\_Start state after power on.

If this state machine receives a HARD\_RESET Received confirmation or a No Phys In Port confirmation, then this state machine shall transition to the ST\_ITS1:Initiator\_Start state.

This state machine shall maintain the state machine variables defined in table 221.

**Table 221 – ST\_ITS state machine variables**

State machine variable	Description
Data-In Buffer Offset	Current offset in the application client's data-in buffer (i.e., the application client buffer for read data)
Data-Out Buffer Offset	Current offset in the application client's data-out buffer (i.e., the application client buffer for write data)
Previous Requested Offset	Offset in the application client's data-out buffer (i.e., the application client buffer for write data) from the last XFER_RDY frame received
Previous Write Data Length	Write data length from the last XFER_RDY frame received

This state machine shall maintain the state machine arguments defined in table 222.

**Table 222 – ST\_ITS state machine arguments**

State machine argument	Description
Command	Consists of the Command arguments received in the Request (Send Command) message
Task	Consists of the arguments received in the Request (Send Task) message
Xfer_Rdy	Consists of the arguments received in the XFER_RDY Arrived message
Data-Out Buffer	The location of the application client's data-out buffer (i.e., the application client buffer for write data)
Data-Out Buffer Size	The size in bytes of the application client's data-out buffer (i.e., the application client buffer for write data)
Data-In Buffer Size	The size in bytes of the application client's data-in buffer (i.e., the application client buffer for read data)

#### **8.2.6.2.3.2 ST\_ITS1:Initiator\_Start state**

##### **8.2.6.2.3.2.1 State description**

This state is the initial state of the ST\_ITS state machine.

Upon entry into this state, this state shall set the Data-In Buffer Offset state machine variable to zero.

Upon entry into this state, this state shall set the Data-Out Buffer Offset state machine variable to zero.

##### **8.2.6.2.3.2.2 Transition ST\_ITS1:Initiator\_Start to ST\_ITS3:Prepare\_Command**

This transition shall occur after receiving a Request (Send Command) message.

##### **8.2.6.2.3.2.3 Transition ST\_ITS1:Initiator\_Start to ST\_ITS4:Prepare\_Task**

This transition shall occur after receiving a Request (Send Task) message.

#### **8.2.6.2.3.3 ST\_ITS2:Initiator\_Send\_Frame state**

If this state is entered from the ST\_ITS3:Prepare\_Command state for transmission of a COMMAND frame, then this state shall send a Transmit Frame (Interlocked) request to the port layer.

If this state is entered from the ST\_ITS6:Receive\_Data\_In state and the vendor specific number of retries has not been reached for the COMMAND frame requesting a read operation, then this state shall send a Transmit Frame (Interlocked) request to the port layer.

If this state is entered from the ST\_ITS4:Prepare\_Task state for transmission of a TASK frame, then this state shall send a Transmit Frame (Interlocked) request to the port layer.

If this state is entered from the ST\_ITS5:Prepare\_Data\_Out state for transmission of a write DATA frame, then this state shall send a Transmit Frame (Non-Interlocked) request to the port layer after this state has received an XFER\_RDY Arrived message.

If this state is entered from the ST\_ITS5:Prepare\_Data\_Out state for transmission of a write DATA frame and first burst is enabled, then this state shall send a Transmit Frame (Non-Interlocked) request to the port layer after this state has received a Transmission Status (Frame Transmitted) confirmation and a Transmission Status (ACK Received) confirmation for the COMMAND frame.

A Transmit Frame request shall include the COMMAND frame from the ST\_ITS3:Prepare\_Command state or from the ST\_ITS6:Receive\_Data\_In state, the TASK frame from the ST\_ITS4:Prepare\_Task state, or the write DATA frame from the ST\_ITS5:Prepare\_Data\_Out state and the following arguments to be used for any OPEN address frame:

- a) Initiator Port Bit set to one;
- b) Protocol set to SSP;
- c) Connection Rate;
- d) Initiator Connection Tag;
- e) Destination SAS Address; and
- f) Source SAS Address.

If persistent connections are supported (see 4.1.13.2), then the Transmit Frame request shall include the following additional argument to be used for any OPEN address frame:

- a) Send Extend Bit.

If credit advance is implemented (see 4.1.14), then the Transmit Frame request shall include the following additional argument to be used for an OPEN address frame:

- a) Credit Advance Bit.

After sending a Transmit Frame request this state shall wait to receive a Transmission Status confirmation.

If the confirmation is Transmission Status (I\_T Nexus Loss), then this state shall send a Transmission Complete (I\_T Nexus Loss) message to the ST\_IFR state machine. This Transmission Complete message shall include the initiator port transfer tag as an argument.

If the confirmation is not Transmission Status (Frame Transmitted) or Transmission Status (I\_T Nexus Loss) (see table 203), and the Transmit Frame request was for a COMMAND frame or a DATA frame, then this state shall send a Transmission Complete (Command Failed, Connection Failed) message to the ST\_IFR state machine. The message shall include the initiator port transfer tag.

If the confirmation is not Transmission Status (Frame Transmitted) or Transmission Status (I\_T Nexus Loss) (see table 203), and the Transmit Frame request was for a TASK frame, then this state shall send a Transmission Complete (Task Failed, Connection Failed) message to the ST\_IFR state machine. The message shall include the initiator port transfer tag.

If the confirmation is Transmission Status (Frame Transmitted) and:

- a) the Transmit Frame request was for a COMMAND frame not requesting a read operation;
- b) a COMMAND frame not requesting a write operation;
- c) a TASK frame; or
- d) a write DATA frame where the number of data bytes that have been transmitted equal the Data-Out Buffer Size state machine argument,

then this state shall wait to receive one of the following confirmations:

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

If the confirmation is Transmission Status (Frame Transmitted) and the Transmit Frame request was for a COMMAND frame requesting a write operation, or a write DATA frame where the number of data bytes that have been transmitted is less than the Data-Out Buffer Size state machine argument and the write data length from the previous XFER\_RDY frame, then this state shall wait to receive one of the following:

- a) a Transmission Status (ACK Received) confirmation;
- b) a Transmission Status (NAK Received) confirmation;
- c) a Transmission Status (ACK/NAK Timeout) confirmation;
- d) a Transmission Status (Connection Lost Without ACK/NAK) confirmation; or
- e) an XFER\_RDY Arrived message.

If an XFER\_RDY Arrived message is received, then the ST\_ITS shall respond to the XFER\_RDY frame as if a Transmission Status (ACK Received) confirmation was received.

If the number of data bytes requested to be transmitted for the Send SCSI Command transport protocol service request are fewer than the number of bytes in the service request, then this state may send additional Transmit Frame requests for write DATA frames for the transport protocol service request before receiving a Transmission Status (ACK Received), Transmission Status (NAK Received), Transmission Status (ACK/NAK Timeout), or Transmission Status (Connection Lost Without ACK/NAK) confirmation for Transmit Frame requests for previous write DATA frames sent for the I\_T\_L nexus and command identifier combination.

After a Transmission Status (Frame Transmitted) confirmation is received, if a Transmission Status (NAK Received) confirmation is received, the Transmit Frame request was for a COMMAND frame, and the vendor specific number of retries has not been reached, then this state shall send a Transmit Frame (Interlocked) request to the port layer (i.e., the last COMMAND frame is retransmitted).

After a Transmission Status (Frame Transmitted) confirmation is received, if a Transmission Status (NAK Received) confirmation is received, the Transmit Frame request was for a TASK frame, and the vendor specific number of retries has not been reached, then this state shall send a Transmit Frame (Interlocked) request to the port layer (i.e., the last TASK frame is retransmitted).

Table 223 defines the messages that this state shall send to the ST\_IFR state machine upon receipt of the listed confirmations, based on the conditions under which each confirmation was received.

**Table 223 – Messages sent to the ST\_IFR state machine**

Confirmation received from the port layer	Conditions under which confirmation was received	Message sent to the ST_IFR state machine
Transmission Status (ACK/NAK Timeout) or Transmission Status (Connection Lost Without ACK/NAK)	The Transmit Frame request was for a COMMAND frame.	Transmission Complete (Command Failed, ACK/NAK Timeout)
	The Transmit Frame request was for a TASK frame.	Transmission Complete (Task Failed, ACK/NAK Timeout)
Transmission Status (NAK Received)	The Transmit Frame request was for a COMMAND frame and the vendor specific number of retries has been reached.	Transmission Complete (Command Failed, NAK Received)
	The Transmit Frame request was for a TASK frame and the vendor specific number of retries has been reached.	Transmission Complete (Task Failed, NAK Received)
Transmission Status (NAK Received)	The Transmit Frame request was for a write DATA frame and:	Transmission Complete (Data-Out Failed, NAK Received)
Transmission Status (ACK/NAK Timeout) or Transmission Status (Connection Lost Without ACK/NAK)	a) the RETRY DATA FRAMES bit was set to zero in the XFER_RDY frame requesting the data; or	Transmission Complete (Data-Out Failed, ACK/NAK Timeout)
	b) the RETRY DATA FRAMES bit was set to one in the XFER_RDY frame requesting the data and the vendor specific number of retries has been reached.	

After this state sends a Transmission Complete (Command Failed, ACK/NAK Timeout) message this state shall continue processing messages and confirmations.

If this state receives a Return to Start message or a Return to Start argument, and this state has not received confirmations for all Transmit Frame requests sent to the port layer, then this state shall send a Cancel request to the port layer. This state may also send a Cancel request to the port layer to cancel a previous Transmit Frame request.

If this state receives a Cancel message or a Cancel argument, and this state has received confirmations for all Transmit Frame requests sent to the port layer, then this state shall send a Transmission Complete (Cancel Acknowledged) message to the ST\_IFR state machine.

If this state receives a Cancel message or a Cancel argument, and this state has not received confirmations for all Transmit Frame requests sent to the port layer, then this state shall send a Cancel request to the port layer. This state may also send a Cancel request to the port layer to cancel a previous Transmit Frame request. The Cancel request shall include the following arguments:

- a) Destination SAS Address; and
- b) Initiator Port Transfer Tag.

NOTE 58 - The Cancel message results from a vendor specific request from the SCSI application layer after the SCSI application layer has used a task management function to determine that the SSP target port did not receive the COMMAND frame.

If this state receives a Transmission Status (Cancel Acknowledged) confirmation, then this state shall send a Transmission Complete (Cancel Acknowledged) message to the ST\_IFR state machine.

If this state receives an XFER\_RDY Arrived message, then this state shall verify the Xfer\_Rdy state machine argument as specified in table 224. If the verification fails, then this state shall send the Transmission Complete message specified in table 224 to the ST\_IFR state machine.

**Table 224 – Transmission Complete messages for XFER\_RDY frame verification failures**

Message sent to the ST_IFR state machine <sup>a</sup>	Condition
Transmission Complete (XFER_RDY Incorrect Write Data Length)	The Write Data Length Xfer_Rdy state machine argument is set to zero.
	The Requested Offset Xfer_Rdy state machine argument plus the Write Data Length Xfer_Rdy state machine argument is greater than the Data-Out Buffer Size state machine argument.
Transmission Complete (XFER_RDY Requested Offset Error)	First burst is disabled, this is the first XFER_RDY frame for a command, and the value in the Requested Offset Xfer_Rdy state machine argument is not set to zero.
	First burst is enabled, this is the first XFER_RDY frame for a command, and the value in the Requested Offset Xfer_Rdy state machine argument is not equal to the value indicated by the FIRST BURST SIZE field in the Disconnect-Reconnect mode page (see 9.2.7.2.5).
	Transport layer retries are disabled and the Requested Offset Xfer_Rdy state machine argument is not equal to the Previous Requested Offset state machine variable plus the Previous Write Data Length Field state machine variable.
	Transport layer retries are enabled, the Retransmit Bit Xfer_Rdy state machine argument is set to zero, and the Requested Offset Xfer_Rdy state machine argument is not equal to the Previous Requested Offset state machine variable plus the Previous Write Data Length state machine variable.
	Transport layer retries are enabled, this is not the first XFER_RDY frame for the command, the Retransmit Bit Xfer_Rdy state machine argument is set to one, and the Requested Offset Xfer_Rdy state machine argument is not equal to the Previous Requested Offset state machine variable.
<sup>a</sup> If more than one condition is true, then this state shall send the Transmission Complete (XFER_RDY Incorrect Write Data Length) message to the ST_IFR state machine.	

After this state verifies an XFER\_RDY frame, this state shall:

- set the Data-Out Buffer Offset state machine variable to the Requested Offset Xfer\_Rdy state machine argument;
- set the Previous Requested Offset state machine variable to the Requested Offset Xfer\_Rdy state machine argument; and
- set the Previous Write Data Length state machine variable to the Write Data Length Xfer\_Rdy state machine argument.

#### 8.2.6.2.3.3.1 Transition ST\_ITS2:Initiator\_Send\_Frame to ST\_ITS1:Initiator\_Start

This transition shall occur after:

- this state has sent one of the following to the ST\_IFR state machine:
  - a Transmission Complete (Command Failed, ACK/NAK Timeout) message and the command was for a non-data operation;
  - a Transmission Complete (Command Failed, Connection Failed) message;



- C) a Transmission Complete (Command Failed, NAK Received) message;
- D) a Transmission Complete (Task Failed, ACK/NAK Timeout) message;
- E) a Transmission Complete (Task Failed, Connection Failed) message;
- F) a Transmission Complete (Task Failed, NAK Received) message;
- G) a Transmission Complete (Data-Out Failed, ACK/NAK Timeout) message;
- H) a Transmission Complete (Data-Out Failed, NAK Received) message;
- I) a Transmission Complete (XFER\_RDY Incorrect Write Data Length) message;
- J) a Transmission Complete (XFER\_RDY Requested Offset Error) message; or
- K) a Transmission Complete (Cancel Acknowledged) message;

or

- b) this state has received a Return To Start message or Return To Start argument, and has received:
  - A) confirmations for all Transmit Frame requests sent to the port layer; or
  - B) a Transmission Status (Cancel Acknowledged) confirmation.

#### 8.2.6.2.3.3.2 Transition ST\_ITS2:Initiator\_Send\_Frame to ST\_ITS5:Prepare\_Data\_Out

If first burst is enabled, then this transition shall occur and include the First Burst argument after receiving a Transmission Status (Frame Transmitted) confirmation:

- a) followed by a Transmission Status (ACK Received) confirmation for a COMMAND frame requesting a write operation; or
- b) for a Transmit Frame (Non-interlocked) request if the Data-Out Buffer Offset state machine variable is less than the first burst size.

This transition shall occur after receiving:

- a) an XFER\_RDY Arrived message; or
- b) a Transmission Status (Frame Transmitted) confirmation for a Transmit Frame (Non-interlocked) request if the Data-Out Buffer Offset state machine variable is less than the Requested Offset Xfer\_Rdy state machine argument plus the Write Data Length Xfer\_Rdy state machine argument.

NOTE 59 - This transition occurs even if this state has not received a Transmission Status (ACK Received) confirmation for the write DATA frame.

This transition shall include a Retry argument and occur after:

- a) this state receives one of the following confirmations or arguments for a write DATA frame:
  - A) Transmission Status (NAK Received);
  - B) Transmission Status (ACK/NAK Timeout); or
  - C) Transmission Status (Connection Lost without ACK/NAK);
- b) the RETRY DATA FRAMES bit is set to one in the XFER\_RDY frame for the write operation;
- c) the Data-Out Buffer Offset state machine variable is set to the Requested Offset Xfer\_Rdy state machine argument;
- d) all write DATA frames that have received a Transmission Status (Frame Transmitted) confirmation have received a Transmission Status confirmation; and
- e) the vendor specific number of retries, if any, for the write DATA frame has not been reached.

#### 8.2.6.2.3.3.3 Transition ST\_ITS2:Initiator\_Send\_Frame to ST\_ITS6:Receive\_Data\_In

This transition shall occur:

- a) after receiving a Transmission Status (Frame Transmitted) confirmation for a COMMAND frame for a command requesting a read operation.

NOTE 60 - This transition occurs even if this state has not received a Transmission Status (ACK Received) for the COMMAND frame.

**8.2.6.2.3.4 ST\_ITS3:Prepare\_Command state****8.2.6.2.3.4.1 State description**

This state shall construct a COMMAND frame using the Command arguments:

- a) set the FRAME TYPE field to 06h (i.e., COMMAND frame);
- b) set the HASHED DESTINATION SAS ADDRESS field to the hashed value of the Destination SAS Address Commands argument;
- c) set the HASHED SOURCE SAS ADDRESS field to the hashed value of the SSP initiator port's SAS address;
- d) set the RETRY DATA FRAMES bit to zero;
- e) set the RETRANSMIT bit to zero;
- f) set the CHANGING DATA POINTER bit to zero;
- g) set the NUMBER OF FILL BYTES field to 00b;
- h) set the INITIATOR PORT TRANSFER TAG field to the Initiator Port Transfer Tag Command argument;
- i) TARGET PORT TRANSFER TAG field to FFFFh;
- j) set the DATA OFFSET field to 00000000h;
- k) in the information unit, set the LOGICAL UNIT NUMBER field to the Logical Unit Number Command argument;
- l) in the information unit, set the ENABLE FIRST BURST bit to the Enable First Burst Command argument;
- m) in the information unit, set the COMMAND PRIORITY field to the Command Priority Command argument;
- n) in the information unit, set the TASK ATTRIBUTE field to the Task Attribute Command argument;
- o) in the information unit, set the ADDITIONAL CDB LENGTH field to the Additional CDB Length Command argument;
- p) in the information unit, set the CDB field to the CDB Command argument;
- q) in the information unit, set the ADDITIONAL CDB BYTES field to the Additional CDB Bytes Command argument, if any; and
- r) no fill bytes.

**8.2.6.2.3.4.2 Transition ST\_ITS3:Prepare\_Command to ST\_ITS2:Initiator\_Send\_Frame**

This transition shall occur after this state:

- a) constructs a COMMAND frame;
- b) receives a Cancel message; or
- c) receives a Return To Start message.

This transition shall include:

- a) if neither a Cancel message nor a Return to Start message was received, then the COMMAND frame as an argument;
- b) if a Cancel message was received, then a Cancel argument; or
- c) if a Return To Start message was received, then a Return To Start argument.

**8.2.6.2.3.5 ST\_ITS4:Prepare\_Task state****8.2.6.2.3.5.1 State description**

This state shall construct a TASK frame using the Task arguments:

- a) set the FRAME TYPE field to 16h (i.e., TASK frame);
- b) set the HASHED DESTINATION SAS ADDRESS field to the hashed value of the Destination SAS Address Task argument;
- c) set the HASHED SOURCE SAS ADDRESS field to the hashed value of the SSP initiator port's SAS address;
- d) set the RETRY DATA FRAMES bit to zero;
- e) set the RETRANSMIT bit to the Retransmit Bit Task argument;
- f) set the CHANGING DATA POINTER bit to zero;
- g) set the NUMBER OF FILL BYTES field to 00b;
- h) set the INITIATOR PORT TRANSFER TAG field to the Initiator Port Transfer Tag Task argument;

- i) set the TARGET PORT TRANSFER TAG field to FFFFh;
- j) set the DATA OFFSET field to 00000000h;
- k) in the information unit, set the LOGICAL UNIT NUMBER field to the Logical Unit Number Task argument;
- l) in the information unit, set the TASK MANAGEMENT FUNCTION field to the Task Management Function Task argument;
- m) in the information unit, set the INITIATOR PORT TRANSFER TAG TO MANAGE field to the Initiator Port Transfer Tag Task argument of the command to be managed; and
- n) no fill bytes.

#### 8.2.6.2.3.5.2 Transition ST\_ITS4:Prepare\_Task to ST\_ITS2:Initiator\_Send\_Frame

This transition shall occur after this state:

- a) constructs a TASK frame;
- b) receives a Cancel message; or
- c) receives a Return To Start message.

This transition shall include:

- a) if neither a Cancel message nor a Return to Start message was received, then the TASK frame as an argument;
- b) if a Cancel message was received, then a Cancel argument; or
- c) if a Return To Start message was received, then a Return To Start argument.

#### 8.2.6.2.3.6 ST\_ITS5:Prepare\_Data\_Out state

##### 8.2.6.2.3.6.1 State description

This state shall construct a write DATA frame using the following Xfer\_Rdy state machine arguments and Command state machine arguments:

- a) set the FRAME TYPE field to 01h (i.e., DATA frame);
- b) set the HASHED DESTINATION SAS ADDRESS field to the hashed value of the Destination SAS Address Command argument;
- c) set the HASHED SOURCE SAS ADDRESS field to the hashed value of the SSP initiator port's SAS address;
- d) set the RETRY DATA FRAMES bit to zero;
- e) set the RETRANSMIT bit to zero;
- f) set the CHANGING DATA POINTER bit as specified in this subclause;
- g) set the NUMBER OF FILL BYTES field to the number of fill bytes, based on the length of the specified data;
- h) set the INITIATOR PORT TRANSFER TAG field to the Initiator Port Transfer Tag Command argument;
- i) set the TARGET PORT TRANSFER TAG field to FFFFh if this state received a First Burst argument or the Target Port Transfer Tag Xfer\_Rdy argument if this state did not receive a First Burst argument;
- j) set the DATA OFFSET field to the Data-Out Buffer Offset state machine variable;
- k) in the information unit, set the DATA field to the information that starts at the location in the Data-Out Buffer state machine argument pointed to by the Data-Out Buffer Offset state machine variable. If the number of bytes remaining to be transferred as defined by the following calculation:

bytes remaining to be transferred = Write Data Length Xfer\_Rdy state machine argument - (Data-Out Buffer Offset state machine argument - Requested Offset Xfer\_Rdy state machine argument)

is equal to the maximum size of the write Data information unit, then the amount of data shall be the maximum size of the write Data information unit, otherwise, the amount of data shall be the lesser of:

- A) the bytes remaining to be transferred; and
- B) the maximum size of the Write information unit;

and

- l) fill bytes, if any.

If this state is entered without a Retry argument, then this state shall set the CHANGING DATA POINTER bit to zero.

If this state is entered with a Retry argument, then this state shall set the CHANGING DATA POINTER bit to one.

After constructing the write DATA frame, this state shall set the Data-Out Buffer Offset state machine variable to the value of the DATA OFFSET field plus the number of bytes in the DATA field in the write Data information unit.

#### 8.2.6.2.3.6.2 Transition ST\_ITS5:Prepare\_Data\_Out to ST\_ITS2:Initiator\_Send\_Frame

This transition shall occur after this state:

- a) constructs a write DATA frame;
- b) receives a Cancel message; or
- c) receives a Return To Start message.

This transition shall include the received Transmission Status, if any, as an argument and:

- a) if neither a Cancel message nor a Return to Start message was received, then the write DATA frame as an argument;
- b) if a Cancel message was received, then a Cancel argument; or
- c) if a Return To Start message was received, then a Return To Start argument.

#### 8.2.6.2.3.7 ST\_ITS6:Receive\_Data\_In state

##### 8.2.6.2.3.7.1 State description

If this state receives a Data-In Arrived message, then this state shall verify the values in the read DATA frame received with the message as defined in table 225.

If the verification fails, then this state sends the Reception Complete message specified in table 225 to the ST\_IFR state machine.

**Table 225 – Reception Complete messages for read DATA frame verification failures**

Message sent to the ST_IFR state machine <sup>a</sup>	Condition
Reception Complete (Data Offset Error)	Transport layer retries are disabled and the DATA OFFSET field in the read DATA frame is not equal to the Data-In Buffer Offset state machine variable.
	The DATA OFFSET field in the read DATA frame is greater than the Data-In Buffer Size state machine argument.
Reception Complete (Too Much Read Data)	The number of bytes in the DATA field in the read Data information unit plus the Data-In Buffer Offset state machine variable is greater than the Data-In Buffer Size state machine argument.
Reception Complete (Information Unit Too Short)	Either: a) the number of bytes in the DATA field in the read Data information unit is zero; or b) this is not the last read DATA frame for the command and the NUMBER OF FILL BYTES field is not set to 00b.
<sup>a</sup> If more than one condition is true, then this state shall select which message to send to the ST_IFR state machine using the following order: 1) Reception Complete (Data Offset Error); 2) Reception Complete (Too Much Read Data); or 3) Reception Complete (Information Unit Too Short).	

If:

- a) transport layer retries are enabled;
- b) the CHANGING DATA POINTER bit is set to zero;
- c) the DATA OFFSET field is not set to the Data-In Buffer Offset state machine variable;
- d) the DATA OFFSET field is less than the Data-In Buffer Size state machine argument; and
- e) the DATA OFFSET field plus the number of bytes in the DATA field in the read Data information unit is less than or equal to the Data-In Buffer Size state machine argument,

then this state should discard all Data-In Arrived messages until a read DATA frame is received in which the CHANGING DATA POINTER bit is set to one. This state shall resume processing additional Data-In Arrived messages when it receives a Data-In Arrived message with the CHANGING DATA POINTER bit set to one.

If the read DATA frame verification is successful or after this state resumes processing Data-In Arrived messages, then this state shall process the data received in the read DATA frame and set the Data-In Buffer Offset state machine variable to the DATA OFFSET field plus the number of bytes in the DATA field in the read Data information unit.

If data received in the read DATA frame overlaps data previously received and verified as having no errors, then this state may either discard the overlapping data or replace the previously received data with the new data.

If this state receives a Transmission Status (ACK/NAK Timeout) confirmation or a Transmission Status (Connection Lost Without ACK/NAK) confirmation, then this state shall send a Reception Complete (Command Failed, ACK/NAK Timeout) message to the ST\_IFR state machine.

After this state sends a Reception Complete (Command Failed, ACK/NAK Timeout) message, this state shall continue processing messages and confirmations.

If this state receives a Cancel message, then this state shall send a Reception Complete (Cancel Acknowledged) message to the ST\_IFR state machine. The Reception Complete message shall include the initiator port transfer tag as an argument.

NOTE 61 - The Cancel message results from a vendor specific request from the SCSI application layer after the SCSI application layer has used a task management function to determine that the SSP target port did not receive the COMMAND frame.

#### 8.2.6.2.3.7.2 Transition ST\_ITS6:Receive\_Data\_In to ST\_ITS1:Initiator\_Start

This transition shall occur after this state:

- a) sends one of the following to the ST\_IFR state machine:
  - A) a Reception Complete (Data Offset Error) message;
  - B) a Reception Complete (Too Much Read Data) message;
  - C) a Reception Complete (Incorrect Data Length) message; or
  - D) a Reception Complete (Cancel Acknowledged) message;
- or
- b) receives a Return To Start message.

#### 8.2.6.2.3.7.3 Transition ST\_ITS6:Receive\_Data\_In to ST\_ITS2:Initiator\_Send\_Frame

This transition shall occur:

- a) after this state receives a Transmission Status (NAK Received) confirmation for a COMMAND frame for a command requesting a read operation.

### 8.2.6.3 ST\_T (transport layer for SSP target ports) state machines

#### 8.2.6.3.1 ST\_T state machines overview

The ST\_T state machines are as follows:

- a) ST\_TFR (target frame router) state machine (see 8.2.6.3.2); and
- b) ST\_TTS (target transport server) state machine (see 8.2.6.3.3).

The SAS target port includes:

- a) one ST\_TFR state machine; and
- b) one ST\_TTS state machine for each possible command and task management function (i.e., for each initiator port transfer tag).

This state machine may maintain the timers listed in table 226.

**Table 226 – ST\_T state machine timers**

Timer	Initial value
Initiator Response Timeout	The value in the INITIATOR RESPONSE TIMEOUT field in the Protocol Specific Port mode page (see 9.2.7.4).

Figure 206 shows the ST\_T state machines.

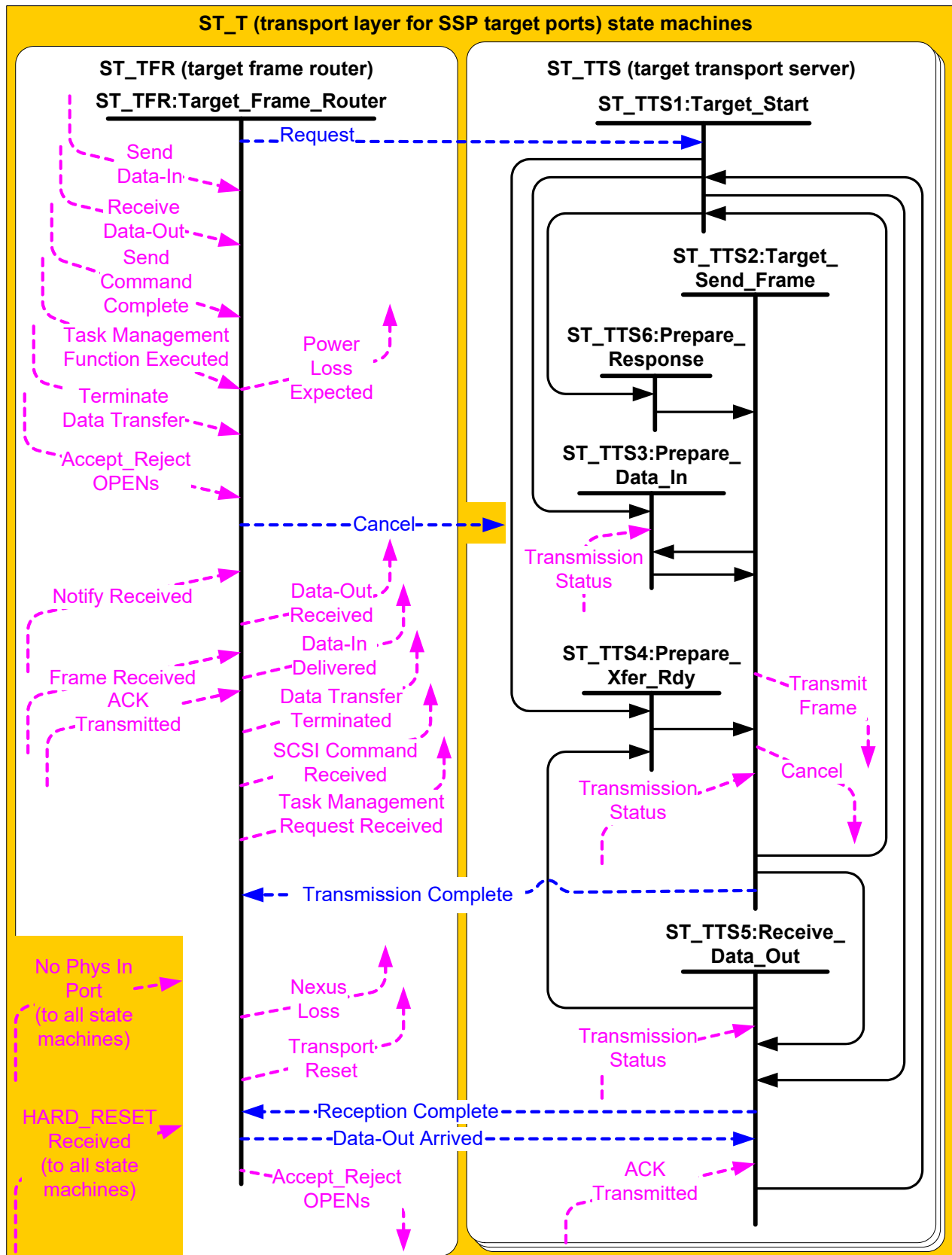


Figure 206 – ST\_T (transport layer for SSP target ports) state machines

### 8.2.6.3.2 ST\_TFR (target frame router) state machine

#### 8.2.6.3.2.1 ST\_TFR state machine overview

The ST\_TFR state machine performs the following functions:

- a) receives confirmations from the port layer;
- b) receives transport protocol service requests from the SCSI application layer;
- c) sends transport protocol service indications to the SCSI application layer;
- d) sends messages to the ST\_TTS state machine;
- e) receives messages from the ST\_TTS state machine;
- f) receives Accept\_Reject OPENs requests from the SCSI application layer;
- g) sends Accept\_Reject OPENs requests to the port layer;
- h) sends Nexus Loss event notifications to the SCSI application layer;
- i) sends Transport Reset event notifications to the SCSI application layer; and
- j) sends Power Loss Expected event notifications to the SCSI application layer.

This state machine consists of one state.

This state machine shall be started after power on.

If this state receives a NOTIFY Received (Power Loss Expected) confirmation, then this state shall:

- a) send a Cancel message to all the ST\_TTS state machines; and
- b) send a Power Loss Expected confirmation to the SCSI application layer.

#### 8.2.6.3.2.2 Processing Frame Received confirmations

If this state machine receives a Frame Received (ACK/NAK Balanced) or Frame Received (ACK/NAK Not Balanced) confirmation, then this state machine shall check the frame type in the received frame (see table 207 in 8.2.1). If the frame type is not COMMAND, TASK, or DATA, then this state machine shall discard the frame. If the confirmation was Frame Received (ACK/NAK Not Balanced) and the frame type is not DATA, then this state machine shall discard the frame.

This state machine may check that reserved fields in the received frame are zero. If non-zero values are not supported in the TLR CONTROL field in a COMMAND frame, then the TLR CONTROL field shall be treated as a reserved field. If any reserved fields are checked and they are not set to zero, then this state machine shall send the following to an ST\_TTS state machine that does not have an active command or task management function and discard the frame:

- a) a Request (Send Transport Response) message with the Transport Response arguments;
- b) the Destination SAS Address argument set to the SAS address from which the invalid frame was received; and
- c) the Service Response argument set to Invalid Frame.

The check of reserved fields within the frame shall not apply to the reserved fields within the CDB in a COMMAND frame. Checking of reserved fields in a CDB is described in SPC-4.

The following is the list of Transport Response arguments:

- a) Connection Rate;
- b) Initiator Connection Tag;
- c) Destination SAS Address (i.e., the SAS address to which the RESPONSE frame is to be transmitted);
- d) Source SAS Address set to the SAS address of the SAS port containing the state machine;
- e) Initiator Port Transfer Tag; and
- f) Service Response.

The Response Fence argument is not included in the Transport Response arguments.



If the frame type is correct relative to the Frame Received confirmation, then this state machine may check that the hashed source SAS address matches the SAS address of the SAS port that transmitted the frame and that the hashed destination SAS address matches the SAS address of the SAS port that received the frame based on the connection information. If this state machine checks these SAS addresses and they do not match, then this state machine shall discard the frame.

If the frame type is COMMAND or TASK, then this state machine shall check the length of the information unit. If the length of the information unit is not correct, then this state machine shall send the following to an ST\_TTS state machine that does not have an active command or task management function and discard the frame:

- a) a Request (Send Transport Response) message with the Transport Response arguments;
- b) the Destination SAS Address argument set to the SAS address from which the invalid frame was received; and
- c) the Service Response argument set to Invalid Frame.

If the frame type is TASK, this state machine checks initiator port transfer tags, the RETRANSMIT bit is set to one in the new TASK frame, and the initiator port transfer tag for the new TASK frame is the same as the initiator port transfer tag for a previous TASK frame where the task management function for the previous TASK frame is not complete, then this state machine shall discard the new TASK frame and not send a Task Management Request Received confirmation to the SCSI application layer.

If the frame type is TASK and this state machine does not check initiator port transfer tags, then this state machine shall ignore the RETRANSMIT bit.

If the frame type is COMMAND or TASK, then this state machine may check the target port transfer tag. If this state checks the target port transfer tag and it is set to a value other than FFFFh, then this state machine shall send the following to an ST\_TTS state machine that does not have an active command or task management function and discard the frame:

- a) a Request (Send Transport Response) message with the Transport Response arguments;
- b) the Destination SAS Address argument set to the SAS address from which the invalid frame was received; and
- c) the Service Response argument set to Invalid Frame.

If the frame type is TASK, then this state machine shall check the logical unit number. If the logical unit number is unknown, then this state machine shall send the following to an ST\_TTS state machine that does not have an active command or task management function and discard the frame:

- a) a Request (Send Transport Response) message with the Transport Response arguments;
- b) the Destination SAS Address argument set to the SAS address from which the invalid frame was received; and
- c) the Service Response argument set to Incorrect Logical Unit Number.

If:

- a) the frame type is DATA and this frame is for first burst data; or
- b) this state machine did not assign a target port transfer tag for the data transfer,

then this state machine may check the target port transfer tag. If the target port transfer tag is set to a value other than FFFFh, then this state machine shall send the following to an ST\_TTS state machine that does not have an active command or task management function and discard the frame:

- a) a Request (Send Transport Response) message with the Transport Response arguments;
- b) the Destination SAS Address argument set to the SAS address from which the invalid frame was received; and
- c) the Service Response argument set to Invalid Frame.

If the frame type is COMMAND or TASK and the fields checked in the frame are correct, then this state machine shall wait to receive an ACK Transmitted confirmation.

If the frame type is COMMAND, the fields checked in the frame are correct, and this state machine receives an ACK Transmitted confirmation, then this state machine shall send a SCSI Command Received transport protocol service indication with the following arguments to the SCSI application layer:

- a) Source SAS Address (i.e., the SAS address that transmitted the COMMAND frame);
- b) Initiator Port Transfer Tag;
- c) Logical Unit Number;
- d) Task Attribute;
- e) Command Priority;
- f) CDB; and
- g) Additional CDB Bytes, if any.

If the frame type is TASK, the fields checked in the frame are correct, and this state machine receives an ACK Transmitted confirmation, then this state machine shall send a Task Management Request Received transport protocol service indication with the following arguments to the SCSI application layer:

- a) Source SAS Address (i.e., the SAS address that transmitted the TASK frame);
- b) Initiator Port Transfer Tag;
- c) Logical Unit Number;
- d) Task Management Function; and
- e) Initiator Port Transfer Tag To Manage.

If the frame type is DATA and the initiator port transfer tag does not match an initiator port transfer tag for an outstanding command performing write operations, then this state machine shall discard the frame.

If the frame type is DATA and the initiator port transfer tag matches an initiator port transfer tag for an outstanding command performing write operations when first burst is disabled or for which no Transmission Complete (Xfer\_Rdy Delivered) message has been received from an ST\_TTS state machine, then this state machine shall discard the frame.

If the frame type is DATA and a target port transfer tag was received in a Transmission Complete (Xfer\_Rdy Delivered) message, then this state machine shall check the target port transfer tag. If the target port transfer tag received in the DATA frame does not match the Target Port Transfer Tag argument in the Transmission Complete (Xfer\_Rdy Delivered) message, then this state machine shall discard the frame.

If the frame type is DATA, the fields checked in the frame are correct, and first burst is enabled or this state machine has received a Transmission Complete (Xfer\_Rdy Delivered) from the ST\_TTS state machine for the request, then this state machine shall send a Data-Out Arrived message to the ST\_TTS state machine specified by the initiator port transfer tag in the frame. The message shall include the content of the write DATA frame.

#### 8.2.6.3.2.3 Processing transport protocol service requests and responses

If this state machine receives a Send Data-In transport protocol service request from the SCSI application layer, then this state machine shall send a Request (Send Data-In) message to an ST\_TTS state machine that does not have an active command or task management function. The message shall include the following Data-In arguments:

- a) Connection Rate;
- b) Initiator Connection Tag;
- c) Destination SAS Address (i.e., the SAS address to which the read DATA frame is to be transmitted);
- d) Source SAS Address set to the SAS address of the SSP target port;
- e) Initiator Port Transfer Tag;
- f) Device Server Buffer;
- g) Request Byte Count; and
- h) Application Client Buffer Offset.

If this state machine receives a Receive Data-Out transport protocol service request from the SCSI application layer, then this state machine shall send a Request (Receive Data-Out) message to an ST\_TTS state machine that does not have an active command or task management function. The message shall include the following Data-Out state machine arguments:

- a) Connection Rate;
- b) Initiator Connection Tag;
- c) Destination SAS Address (i.e., the SAS address to which the XFER\_RDY frame is to be transmitted);
- d) Source SAS Address set to the SAS address of the SSP target port;
- e) Initiator Port Transfer Tag;
- f) Device Server Buffer;
- g) Request Byte Count;
- h) Application Client Buffer Offset; and
- i) Target Port Transfer Tag.

If first burst is enabled, then the Request (Receive Data\_Out) message shall also include the Enable First Burst argument and First Burst Size argument. The First Burst Size argument shall be set to the first burst size from the Disconnect-Reconnect mode page (see 9.2.7.2.5).

If this state machine receives a Send Command Complete transport protocol service response from the SCSI application layer with the Service Response argument set to TASK COMPLETE, then this state machine shall send a Request (Send Application Response) message to the ST\_TTS state machine specified by the initiator port transfer tag. The message shall include the following Application Response arguments:

- a) Connection Rate;
- b) Initiator Connection Tag;
- c) Destination SAS Address (i.e., the SAS address to which the RESPONSE frame is to be transmitted);
- d) Source SAS Address set to the SAS address of the SSP target port;
- e) Initiator Port Transfer Tag;
- f) Status;
- g) Status Qualifier, if any;
- h) Sense Data, if any; and
- i) Response Fence.

If this state machine receives a Task Management Function Executed transport protocol service response from the SCSI application layer, then this state machine shall send the following to the ST\_TTS state machine specified by the initiator port transfer tag:

- a) a Request (Send Transport Response) message with the Transport Response arguments;
- b) the Service Response argument set as specified in table 227; and
- c) the Response Fence argument set to the Task Management Function Executed transport protocol service response Response Fence argument.

Table 227 specifies which argument to send with the Request (Send Transport Response) message based on the Service Response argument that was received.

**Table 227 – Task Management Function Executed Service Response argument mapping to Request (Send Transport Response) Service Response argument**

<b>Task Management Function Executed transport protocol service response Service Response argument received</b>	<b>Request (Send Transport Response) message Service Response argument</b>
FUNCTION COMPLETE	Task Management Function Complete
FUNCTION SUCCEEDED	Task Management Function Succeeded
FUNCTION REJECTED	Task Management Function Not Supported
INCORRECT LOGICAL UNIT NUMBER	Incorrect Logical Unit Number
SERVICE DELIVERY OR TARGET FAILURE - Overlapped Initiator Port Transfer Tag Attempted	Overlapped Initiator Port Transfer Tag Attempted

If this state machine receives a Terminate Data Transfer transport protocol service request from the SCSI application layer and this state machine has not sent a Request message to an ST\_TTS state machine for the Send Data-In or Receive Data-Out transport protocol service request to which the Terminate Data Transfer request applies, then this state machine shall:

- 1) discard the Terminate Data Transfer transport protocol service request and any corresponding Send Data-In or Receive Data-Out transport protocol service request; and
- 2) send a Data Transfer Terminated transport protocol service confirmation to the SCSI application layer.

If this state machine receives a Terminate Data Transfer transport protocol service request from the SCSI application layer and this state machine has sent a Request message to an ST\_TTS state machine for the Send Data-In transport protocol service request to which the Terminate Data Transfer request applies, then this state machine shall send a Cancel message to the ST\_TTS state machine specified by the initiator port transfer tag and the Send Data-In transport protocol service request.

If this state machine receives a Terminate Data Transfer transport protocol service request from the SCSI application layer and this state machine has sent a Request message to an ST\_TTS state machine for the Receive Data-Out transport protocol service request to which the Terminate Data Transfer request applies, then this state machine shall send a Cancel message to the ST\_TTS state machine specified by the initiator port transfer tag and the Receive Data-Out transport protocol service request.

This state machine receives Transmission Complete messages and Reception Complete messages that may result in this state machine sending a Nexus Loss event notification or a transport protocol service confirmation to the SCSI application layer.

If this state machine receives:

- a) a Transmission Complete (I\_T Nexus Loss) message, then this state machine shall send a Nexus Loss event notification to the SCSI application layer; or
- b) a Transmission Complete (Break Occurred) message or a Reception Complete (Break Occurred) message, then this state machine shall send a Break Occurred event notification to the SCSI application layer.

Table 228 defines messages received from ST\_TTS state machines and the corresponding transport protocol service confirmations, if any, that shall be sent upon receipt of the message.

**Table 228 – Confirmations sent to the SCSI application layer**

<b>Message received from ST_TTS state machine</b>	<b>Transport protocol service confirmation sent to SCSI application layer</b>
Transmission Complete (Xfer_Rdy Delivered)	None
Transmission Complete (Response Delivered)	None
Transmission Complete (Response Failed) <sup>a</sup>	None
Transmission Complete (Data Transfer Terminated)	Data Transfer Terminated
Transmission Complete (Data-In Delivered)	Data-In Delivered with the Delivery Result argument set to DELIVERY SUCCESSFUL
Transmission Complete (Xfer_Rdy Failed, NAK Received)	Data-Out Received with the Delivery Result argument set to DELIVERY FAILURE - NAK RECEIVED
Transmission Complete (Xfer_Rdy Failed, Connection Failed)	Data-Out Received with the Delivery Result argument set to DELIVERY FAILURE - CONNECTION FAILED
Transmission Complete (Data-In Failed, NAK Received)	Data-In Delivered with the Delivery Result argument set to DELIVERY FAILURE - NAK RECEIVED
Transmission Complete (Data-In Failed, ACK/NAK Timeout)	Data-In Delivered with the Delivery Result argument set to DELIVERY FAILURE - ACK/NAK TIMEOUT
Transmission Complete (Data-In Failed, Connection Lost without ACK/NAK)	Data-In Delivered with the Delivery Result argument set to DELIVERY FAILURE - CONNECTION FAILED
Reception Complete (Data-Out Received)	Data-Out Received with the Delivery Result argument set to DELIVERY SUCCESSFUL
Reception Complete (Data Offset Error)	Data-Out Received with the Delivery Result argument set to DELIVERY FAILURE - DATA OFFSET ERROR
Reception Complete (Too Much Write Data)	Data-Out Received with the Delivery Result argument set to DELIVERY FAILURE - TOO MUCH WRITE DATA
Reception Complete (Information Unit Too Short)	Data-Out Received with the Delivery Result argument set to DELIVERY FAILURE - INFORMATION UNIT TOO SHORT
Reception Complete (Initiator Response Timeout)	Data-Out Received with the Delivery Result argument set to DELIVERY FAILURE - INITIATOR RESPONSE TIMEOUT
Reception Complete (Data Transfer Terminated)	Data Transfer Terminated
<sup>a</sup> SAM-5 does not define a mechanism for the SCSI device server to determine the result of its Send Command Complete and Task Management Function Executed transport protocol service response calls.	

Each transport protocol service confirmation shall include the following arguments:

- a) the initiator port transfer tag; and
- a) the I\_T nexus or I\_T\_L nexus identifying the scope of the data transfer.

#### 8.2.6.3.2.4 Processing miscellaneous requests and confirmations

If this state machine receives an Accept\_Reject OPENs (Accept SSP) request or an Accept\_Reject OPENs (Reject SSP) request, then this state machine shall send an Accept\_Reject OPENs request with the same arguments to the port layer.

If this state machine receives a HARD\_RESET Received confirmation, then this state shall send a Transport Reset event notification to the SCSI application layer.

If this state machine receives a No Phys In Port confirmation, then this state shall send a Nexus Loss event notification to the SCSI application layer.

#### 8.2.6.3.3 ST\_TTS (target transport server) state machine

##### 8.2.6.3.3.1 ST\_TTS state machine overview

The ST\_TTS state machine performs the following functions:

- a) receives and processes messages from the ST\_TFR state machine;
- b) sends messages to the ST\_TFR state machine;
- c) communicates with the port layer using requests and confirmations regarding frame transmission;  
and
- d) receives HARD\_RESET Received confirmations and No Phys In Port confirmations from the port layer.

This state machine consists of the following states:

- a) ST\_TTS1:Target\_Start (see 8.2.6.3.3.2) (initial state);
- b) ST\_TTS2:Target\_Send\_Frame (see 8.2.6.3.3.3);
- c) ST\_TTS3:Prepare\_Data\_In (see 8.2.6.3.3.4);
- d) ST\_TTS4:Prepare\_Xfer\_Rdy (see 8.2.6.3.3.5);
- e) ST\_TTS5:Receive\_Data\_Out (see 8.2.6.3.3.6); and
- f) ST\_TTS6:Prepare\_Response (see 8.2.6.3.3.7).

This state machine shall start in the ST\_TTS1:Target\_Start state after power on.

If this state machine receives a HARD\_RESET Received confirmation or a No Phys In Port confirmation, then this state machine shall transition to the ST\_TTS1:Target\_Start state.

The state machine shall maintain the state machine variables defined in table 229.

**Table 229 – ST\_TTS state machine variables**

State machine variable	Description
Read Data Offset	Offset into the application client's data-in buffer (i.e., the application client buffer for read data)
Balance Point Read Data Offset	Offset into the application client's data-in buffer (i.e., the application client buffer for read data) of last point at which the number of Transmission Status (ACK Received) confirmations or arguments was equal to the number of transmitted read DATA frames
Read Data Frames Transmitted	The number of Transmission Status (Frame Transmitted) confirmations received for read DATA frames
Read Data Frames ACKed	The number of Transmission Status (ACK Received) confirmations received for read DATA frames
Read Data Buffer End	One greater than the offset into the application client's data-in buffer (i.e., the application client buffer for read data) of the last location into which read data is to be placed
Requested Write Data Offset	SCSI device server requested offset in the application client buffer for write data
Requested Write Data Length	Amount of write data requested by the SCSI device server from the application client buffer
Write Data Offset	Offset into the application client's data-out buffer (i.e., the application client buffer containing write data)

This state machine shall maintain the state machine arguments defined in table 230.

**Table 230 – ST\_TTS state machine arguments**

State machine argument	Description
Data-In	The Data-In arguments received in the Request (Send Data-In) message (see 8.2.6.3.2.3)
Data-Out	The Data-Out arguments received in the Request (Receive Data-Out) message (see 8.2.6.3.2.3)

#### **8.2.6.3.3.2 ST\_TTS1:Target\_Start state**

##### **8.2.6.3.3.2.1 State description**

This state is the initial state of the ST\_TTS state machine.

Upon entry into this state, this state shall:

- set the Read Data Offset state machine variable to the Application Client Buffer Offset Data-In state machine argument;
- set the Balance Point Read Data Offset state machine variable to the Application Client Buffer Offset Data-In state machine argument;
- set the Read Data Frames Transmitted state machine variable to zero;

- d) set the Read Data Frames ACKed state machine variable to zero;
- e) set the Read Data Buffer End state machine variable to the Application Client Buffer Offset Data-In state machine argument plus the Request Byte Count Data-In state machine argument; and
- f) set the Requested Write Data Offset state machine variable to the Application Client Buffer Offset Data-Out state machine argument.

If this state was entered without an Enable First Burst Data-Out state machine argument, then the Requested Write Data Length state machine variable shall be set to the Request Byte Count Data-Out state machine argument.

If this state was entered with an Enable First Burst Data-Out state machine argument, then the Requested Write Data Length state machine variable shall be set to the First Burst Size Data-Out state machine argument.

#### **8.2.6.3.3.2.2 Transition ST\_TTS1:Target\_Start to ST\_TTS3:Prepare\_Data\_In**

This transition shall occur:

- a) after receiving a Request (Send Data-In) message.

#### **8.2.6.3.3.2.3 Transition ST\_TTS1:Target\_Start to ST\_TTS4:Prepare\_Xfer\_Rdy**

If this state was entered without an Enable First Burst Data-Out state machine argument, then this transition shall occur:

- a) after a Request (Receive Data-Out) message is received.

#### **8.2.6.3.3.2.4 Transition ST\_TTS1:Target\_Start to ST\_TTS5:Receive\_Data\_Out**

If this state was entered with an Enable First Burst Data-Out state machine argument, then this transition shall occur:

- a) after a Request (Receive Data-Out) message is received.

#### **8.2.6.3.3.2.5 Transition ST\_TTS1:Target\_Start to ST\_TTS6:Prepare\_Response**

This transition shall occur:

- a) after receiving a Request (Send Transport Response) message.

The transition shall include:

- a) the Transport Response arguments.

#### **8.2.6.3.3.3 ST\_TTS2:Target\_Send\_Frame state**

##### **8.2.6.3.3.3.1 State description**

If this state is entered from the ST\_TTS3:Prepare\_Data\_In state for transmission of a read DATA frame, then this state shall send a Transmit Frame (Non-Interlocked) request to the port layer.

If this state is entered from the ST\_TTS4:Prepare\_Xfer\_Rdy state for transmission of an XFER\_RDY frame, then this state shall send a Transmit Frame (Interlocked) request to the port layer.

If this state is entered from the ST\_TTS6:Prepare\_Response state for transmission of a RESPONSE frame, then this state shall send a Transmit Frame (Interlocked) request to the port layer.

All Transmit Frame requests from this state shall include the read DATA frame from the ST\_TTS3:Prepare\_Data\_In state, the XFER\_RDY frame from the ST\_TTS4:Prepare\_Xfer\_Rdy state, or the RESPONSE frame from the ST\_TTS6:Prepare\_Response state and the following arguments to be used for any OPEN address frame:

- a) Initiator Port Bit set to zero;



- b) Protocol set to SSP;
- c) Connection Rate;
- d) Initiator Connection Tag;
- e) Destination SAS Address; and
- f) Source SAS Address.

If persistent connections are supported (see 4.1.13.2), then the Transmit Frame request shall include the following additional argument to be used for any OPEN address frame:

- a) Send Extend Bit set to zero.

If credit advance is implemented (see 4.1.14), then the Transmit Frame request shall include the following additional argument to be used for an OPEN address frame:

- a) Credit Advance Bit.

After sending a Transmit Frame request, this state shall wait to receive a Transmission Status confirmation.

If the confirmation or argument is Transmission Status (I\_T Nexus Loss), then this state shall send a Transmission Complete (I\_T Nexus Loss) message to the ST\_TFR state machine. The Transmission Complete message shall include the initiator port transfer tag as an argument.

If the confirmation or argument is not Transmission Status (Frame Transmitted) or Transmission Status (I\_T Nexus Loss), then this state shall send the Transmission Complete message defined in table 231 to the ST\_TFR state machine. The message shall include the following arguments:

- a) Initiator Port Transfer Tag; and
- b) arguments received with the Transmission Status confirmation.

If the confirmation is Transmission Status (Frame Transmitted) and the Transmit Frame request was for:

- a) an XFER\_RDY frame; or
- b) a RESPONSE frame,

then this state shall wait to receive one of the following confirmations:

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

If the confirmation is Transmission Status (Frame Transmitted) and the Transmit Frame request was for a read DATA frame, then this state shall:

- a) increment the Read Data Frames Transmitted state machine variable by one; and
- b) set the Read Data Offset state machine variable to the current Read Data Offset state machine variable plus the number of read data bytes transmitted in the DATA frame associated with the Transmission Status (Frame Transmitted) confirmation.

If the confirmation is Transmission Status (ACK Received) and the Transmit Frame request was for a read DATA frame, then this state shall increment the Read Data Frames ACKed state machine variable by one.

If the confirmation is Transmission Status (Frame Transmitted), the Transmit Frame request was for a read DATA frame, and the Read Data Offset state machine variable is equal to the Read Data Buffer End state machine variable, then this state shall wait to receive:

- a) Transmission Status (ACK Received) confirmations or arguments for each outstanding read DATA frame (i.e., Read Data Frames Transmitted state machine variable equals the Read Data Frames ACKed state machine variable); or
- b) one of the following confirmations:
  - A) Transmission Status (NAK Received);
  - B) Transmission Status (ACK/NAK Timeout); or
  - C) Transmission Status (Connection Lost Without ACK/NAK).

NOTE 62 - If the number of data bytes that have been transmitted for a Request (Send Data-In) message are fewer than the Request Byte Count Data-In state machine argument, then this state transitions to the ST\_TTS3:Prepare\_Data\_In state to construct the additional read DATA frames for the request before receiving a Transmission Status (ACK Received), Transmission Status (NAK Received), Transmission Status (ACK/NAK Timeout), or Transmission Status (Connection Lost Without ACK/NAK) confirmation.

When the Read Data Frames Transmitted state machine variable equals the Read Data Frames ACKed state machine variable and the Transmit Frame request was for a read DATA frame, this state shall:

- a) not modify the Balance Point Read Data Offset state machine variable (i.e., the balance point remains at the last point at which balance occurred); or
- b) set the Balance Point Read Data Offset state machine variable to the current Read Data Offset state machine variable.

If the Transmit Frame request was for a RESPONSE frame, the vendor specific number of retries has not been reached, and this state receives one of the following confirmations:

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

then this state shall:

- a) set the RETRANSMIT bit to one;
- b) set the other fields to the same values as contained in the failed RESPONSE frame; and
- c) resend a Transmit Frame (Interlocked) request to the port layer for the failed RESPONSE frame.

If transport layer retries are enabled, the Transmit Frame request was for an XFER\_RDY frame, the vendor specific number of retries has not been reached, and this state receives one of the following confirmations:

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

then this state shall:

- a) set the RETRANSMIT bit to one;
- b) set the TARGET PORT TRANSFER TAG field to a value that is different than the target port transfer tag in the previous XFER\_RDY frame associated with the Data-Out state machine arguments and is different than any other target port transfer tag currently in use. If write data is received for a subsequent XFER\_RDY frame for a command, then all target port transfer tags used for previous XFER\_RDY frames for the command are no longer in use;
- c) set the other fields to the same values contained in the failed XFER\_RDY frame; and
- d) resend a Transmit Frame (Interlocked) request to the port layer for the failed XFER\_RDY frame.

Table 231 defines messages that this state shall send to the ST\_TFR state machine upon receipt of the listed confirmations and arguments, based on the conditions under which each confirmation or argument was received.

**Table 231 – Messages sent to the ST\_TFR state machine**

Confirmation received from the port layer or argument received from ST_TTS3:Prepare_Data_In	Conditions under which confirmation was received	Message sent to the ST_TFR state machine
Transmission Status (ACK Received)	The Transmit Frame request was for an XFER_RDY frame.	Transmission Complete (Xfer_Rdy Delivered) with a Target Port Transfer Tag argument
	The Transmit Frame request was for a RESPONSE frame.	Transmission Complete (Response Delivered)
	The Transmit Frame request was for a read DATA frame and the Read Data Offset state machine variable is equal to: a) the Read Data Buffer End state machine variable; and b) the Balance Point Read Data Offset state machine variable.	Transmission Complete (Data-In Delivered)
Transmission Status (NAK Received), Transmission Status (ACK/NAK Timeout), or Transmission Status (Connection Lost Without ACK/NAK)	The Transmit Frame request was for a RESPONSE frame and the vendor specific number of retries has been reached.	Transmission Complete (Response Failed)
Transmission Status (NAK Received)	The Transmit Frame request was for an XFER_RDY frame and if transport layer retries are: a) disabled; or b) enabled and the vendor specific number of retries has been reached.	Transmission Complete (Xfer_Rdy Failed, NAK Received)
Transmission Status (ACK/NAK Timeout) or Transmission Status (Connection Lost Without ACK/NAK)		Transmission Complete (Xfer_Rdy Failed, Connection Failed)
Transmission Status (NAK Received)	The Transmit Frame request was for a read DATA frame and if transport layer retries are: a) disabled; or b) enabled and the vendor specific number of retries has been reached.	Transmission Complete (Data-In Failed, NAK Received)
Transmission Status (ACK/NAK Timeout)		Transmission Complete (Data-In Failed, ACK/NAK Timeout)
Transmission Status (Connection Lost Without ACK/NAK)		Transmission Complete (Data-In Failed, Connection Lost Without ACK/NAK)

Table 232 defines messages that this state shall send to the ST\_TFR state machine upon receipt of the listed confirmations and arguments.

**Table 232 – Additional messages sent to the ST\_TFR state machine**

Confirmation received from the port layer or argument received from ST_TTS3:Prepare_Data_In	Message sent to the ST_TFR state machine
Transmission Status (Break Received)	Transmission Complete (Break Occurred)

If this state receives a Cancel message or a Cancel argument and this state has received confirmations for all Transmit Frame requests sent to the port layer, then this state shall send a Transmission Complete (Data Transfer Terminated) message to the ST\_TFR state machine.

If this state receives a Cancel message or a Cancel argument and this state has not received confirmations for all Transmit Frame requests sent to the port layer, then this state shall send a Cancel request to the port layer to cancel previous Transmit Frame requests. The Cancel request shall include the following arguments:

- a) Destination SAS Address; and
- b) Initiator Port Transfer Tag.

Upon receipt of a Transmission Status (Cancel Acknowledged) confirmation or argument this state shall send a Transmission Complete (Data Transfer Terminated) message to the ST\_TFR state machine.

A Transmission Complete message to the ST\_TFR state machine shall include the following arguments:

- a) Destination SAS Address; and
- b) Initiator Port Transfer Tag.

#### **8.2.6.3.3.2 Transition ST\_TTS2:Target\_Send\_Frame to ST\_TTS1:Target\_Start**

This transition shall occur:

- a) after sending a Transmission Complete message other than the Transmission Complete (Xfer\_Rdy Delivered) message to the ST\_TFR state machine.

#### **8.2.6.3.3.3 Transition ST\_TTS2:Target\_Send\_Frame to ST\_TTS3:Prepare\_Data\_In**

This transition shall occur:

- a) after receiving a Transmission Status (Frame Transmitted) confirmation for a read DATA frame if the Read Data Offset state machine variable is less than the Read Data Buffer End state machine variable (i.e., there is more read data to transfer).

If transport layer retries are enabled and the vendor specific number of retries, if any, for the read DATA frame has not been reached, then this transition shall occur and include a Retry argument after receiving one of the following confirmations for a read DATA frame:

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

#### **8.2.6.3.3.4 Transition ST\_TTS2:Target\_Send\_Frame to ST\_TTS5:Receive\_Data\_Out**

This transition shall occur:

- a) after sending a Transmission Complete (Xfer\_Rdy Delivered) message to the ST\_TFR state machine.

**8.2.6.3.3.4 ST\_TTS3:Prepare\_Data\_In state****8.2.6.3.3.4.1 State description**

This state retrieves the data from the Device Server Buffer Data-In state machine argument and constructs a read DATA frame.

This state shall construct a read DATA frame using the Data-In state machine arguments as follows:

- a) set the FRAME TYPE field to 01h (i.e., DATA frame);
- b) set the HASHED DESTINATION SAS ADDRESS field to the hashed value of the Destination SAS Address Data-In state machine argument;
- c) set the HASHED SOURCE SAS ADDRESS field to the hashed value of the SSP target port's SAS address;
- d) set the RETRY DATA FRAMES bit to zero;
- e) set the RETRANSMIT bit to zero;
- f) set the CHANGING DATA POINTER as specified in this subclause;
- g) set the NUMBER OF FILL BYTES field to the number of fill bytes to be used in the DATA frame for the specified read data;
- h) set the INITIATOR PORT TRANSFER TAG field to the Initiator Port Transfer Tag Data-In state machine argument;
- i) set the TARGET PORT TRANSFER TAG field to a vendor specific value;
- j) set the DATA OFFSET field as specified in this subclause;
- k) in the information unit, set the DATA field as specified in this subclause; and
- l) fill bytes, if required.

If this state is entered without a Retry argument, then this state shall:

- a) set the CHANGING DATA POINTER bit to zero;
- b) set the DATA OFFSET field to the Read Data Offset state machine variable; and
- c) in the information unit, set the DATA field to the information in the Device Server Buffer argument that corresponds to the read data to be transferred. If the Read Data Buffer End state machine variable minus the Read Data Offset state machine variable is equal to the maximum size of the read Data information unit, then the amount of data shall be the maximum size of the read Data information unit, otherwise the amount of data shall be the lesser of:
  - A) the Read Data Buffer End state machine variable minus the Read Data Offset state machine variable; and
  - B) the maximum size of the read Data information unit for this Send Data-In request.

If this state is entered with a Retry argument, then this state shall either:

- a) set the CHANGING DATA POINTER bit to one;
- b) set the DATA OFFSET field to the Balance Point Read Data Offset state machine variable;
- c) set the Read Data Offset state machine variable to the Balance Point Read Data Offset state machine variable;
- d) set the Read Data Frames Transmitted state machine variable to zero;
- e) set the Read Data Frames ACKed state machine variable to zero; and
- f) in the information unit, set the DATA field to the information in the Device Server Buffer argument that corresponds to the read data to be transferred. If the Read Data Buffer End state machine variable minus the Read Data Offset state machine variable is equal to the maximum size of the read Data information unit, then the amount of data shall be the maximum size of the read Data information unit, otherwise the amount of data shall be the lesser of:
  - A) the Read Data Buffer End state machine variable minus the Balance Point Read Data Offset state machine variable; and
  - B) the maximum size of the read Data information unit for this Send Data-In request

or:

- a) set the CHANGING DATA POINTER bit to one;
- b) set the DATA OFFSET field to the Application Client Buffer Offset Data-In state machine argument;
- c) set the Read Data Offset state machine variable to the Application Client Buffer Offset Data-In state machine argument;
- d) set the Read Data Frames Transmitted state machine variable to zero;

- e) set the Read Data Frames ACKed state machine variable to zero; and
- f) in the information unit, set the DATA field to the information in the Device Server Buffer argument that corresponds to the read data to be transferred. If the Request Byte Count Data-In state machine argument is equal to the maximum size of the read Data information unit, then the amount of data shall be the maximum size of the read Data information unit, otherwise the amount of data shall be the lesser of:
  - A) the Request Byte Count Data-In state machine argument; and
  - B) the maximum size of the read Data information unit for this Send Data-In request.

#### 8.2.6.3.3.4.2 Transition ST\_TTS3:Prepare\_Data\_In to ST\_TTS2:Target\_Send\_Frame

This transition shall occur after this state:

- a) constructs a read DATA frame; or
- b) receives a Cancel message.

This transition shall include the received Transmission Status, if any, as an argument and if a Cancel message was:

- a) not received, then the read DATA frame as an argument; or
- b) received, then a Cancel argument.

#### 8.2.6.3.3.5 ST\_TTS4:Prepare\_Xfer\_Rdy state

##### 8.2.6.3.3.5.1 State description

This state shall construct an XFER\_RDY frame using the Data-Out state machine arguments:

- a) set the FRAME TYPE field to 05h (i.e., XFER\_RDY frame);
- b) set the HASHED DESTINATION SAS ADDRESS field to the hashed value of the Destination SAS Address Data-Out state machine argument;
- c) set the HASHED SOURCE SAS ADDRESS field to the hashed value of the SSP target port's SAS address;
- d) set the RETRY DATA FRAMES bit to one if transport layer retries are enabled and zero if transport layer retries are disabled;
- e) set the RETRANSMIT bit to zero;
- f) set the CHANGING DATA POINTER bit to zero;
- g) set the NUMBER OF FILL BYTES field to 00b;
- h) set the INITIATOR PORT TRANSFER TAG field to the Initiator Port Transfer Tag Data-Out state machine argument;
- i) if transport layer retries are disabled, then set the TARGET PORT TRANSFER TAG field to a vendor specific value;
- j) if transport layer retries are enabled, then set the TARGET PORT TRANSFER TAG field to a vendor specific value that is different from:
  - A) the target port transfer tag in the previous XFER\_RDY frame associated with the Data-Out state machine arguments; and
  - B) any other target port transfer tag currently in use.

If write data is received for a subsequent XFER\_RDY frame for a command, then all target port transfer tags used for previous XFER\_RDY frames for the command are no longer in use;
- k) set the DATA OFFSET field to 00000000h;
- l) in the information unit, set the REQUESTED OFFSET field to the Requested Write Data Offset state machine variable;
- m) in the information unit, set the WRITE DATA LENGTH field as specified in this subclause; and
- n) no fill bytes.

If the SSP target port has the resources available to receive all of the write data as indicated by the Requested Write Data Length state machine variable, then this state shall set the WRITE DATA LENGTH field in the XFER\_RDY information unit to the Requested Write Data Length state machine variable.

If the SSP target port does not have the resources available to receive all of the write data as indicated by the Requested Write Data Length state machine variable (e.g., the SSP target port has a vendor specific limit as to how much write data may be received during one operation), then this state shall set the WRITE DATA LENGTH field in the XFER\_RDY information unit and the Requested Write Data Length state machine variable to a value representing the amount of write data for which the SSP target port has available resources to receive.

#### 8.2.6.3.3.5.2 Transition ST\_TTS4:Prepare\_Xfer\_Rdy to ST\_TTS2:Target\_Send\_Frame

This transition shall occur after this state:

- a) constructs an XFER\_RDY frame; or
- b) receives a Cancel message.

This transition shall include if a Cancel message was:

- a) not received, then the XFER\_RDY frame as an argument; or
- b) received, then a Cancel argument.

#### 8.2.6.3.3.6 ST\_TTS5:Receive\_Data\_Out state

##### 8.2.6.3.3.6.1 State description

Upon entry into this state, the Write Data Offset state machine variable is set to the Requested Write Data Offset state machine variable.

If this state receives a Data-Out Arrived message, then this state shall verify the write DATA frame received with the Data-Out Arrived values as specified in table 233. If the verification test fails, then this state sends the message specified in table 233 to the ST\_TFR state machine.

**Table 233 – Reception Complete message for write DATA frame verification failures**

Message sent to the ST_TFR state machine <sup>a</sup>	Condition
Reception Complete (Data Offset Error)	Transport layer retries are disabled and the DATA OFFSET field is not equal to the Write Data Offset state machine variable.  The DATA OFFSET field is: a) less than the Requested Write Data Offset state machine variable; or b) greater than or equal to the Requested Write Data Offset state machine variable plus the Requested Write Data Length state machine variable.
Reception Complete (Too Much Write Data)	The number of bytes in the DATA field in the write Data information unit plus the Write Data Offset state machine variable is greater than the Request Byte Count Data-Out state machine argument.
Reception Complete (Information Unit Too Short)	Either: a) the number of bytes in the DATA field in the write Data information unit is zero; or b) this is not the last write DATA frame for the command and the NUMBER OF FILL BYTES field for the frame is not set to 00b.
<sup>a</sup> If more than one condition is true, then this state shall select which message to send to the ST_TFR state machine using the following order: 1) Reception Complete (Data Offset Error); 2) Reception Complete (Too Much Write Data); or 3) Reception Complete (Information Unit Too Short).	

If:

- a) transport layer retries are enabled;
- b) the CHANGING DATA POINTER bit is set to zero; and
- c) the value in the DATA OFFSET field is not equal to the Write Data Offset state machine variable,

then this state should discard all Data-Out Arrived messages until the CHANGING DATA POINTER bit is set to one. This state shall resume processing additional Data-Out Arrived messages when it receives a Data-Out Arrived message with the CHANGING DATA POINTER bit set to one.

If the WRITE data frame verification is successful and the Data-Out Arrived message is not discarded, then this state shall:

- a) process the write data as indicated in the Data-Out state machine arguments using the Device Server Buffer (e.g., logical block address) to which the write data is to be transferred; and
- b) set the Write Data Offset state machine variable to the current Write Data Offset state machine variable plus the number of bytes received in the DATA field of the write Data information unit.

If the WRITE data frame verification is successful and the CHANGING DATA POINTER bit set to one, then this state shall:

- a) set the Write Data Offset state machine variable to the Requested Write Data Offset state machine variable plus the number of bytes received in the DATA field of the write Data information unit; and
- b) process the write data as indicated in the Data-Out state machine arguments using the Device Server Buffer (e.g., logical block address) to which the write data is to be transferred.

If data received in the write DATA frame overlaps data previously received and verified to have no errors, then this state may either discard the overlapping data or replace the previously received data with the new data.

If the Initiator Response Timeout timer is implemented, then this state shall initialize and start the Initiator Response Timeout timer:

- a) upon entry into this state; and
- b) when this state receives and verifies the write DATA frame received with the Data-Out Arrived values (i.e., Data-Out data was received and processed).

If the Initiator Response Timeout timer is running, then this state shall stop the timer before transitioning from this state.

If the Initiator Response Timeout timer expires, then this state shall send a Reception Complete (Initiator Response Timeout) message to the ST\_TFR state machine.

If the Write Data Offset state machine variable equals the Request Byte Count Data-Out state machine argument plus the Application Client Buffer Offset Data-Out state machine argument, then this state shall send a Reception Complete (Data-Out Received) message to the ST\_TFR state machine after an ACK Transmitted confirmation is received for each write DATA frame received.

If this state receives a Cancel message, then this state shall send a Reception Complete (Data Transfer Terminated) message to the ST\_TFR state machine.

If this state receives Transmission Status (Break Received) confirmation, then this state shall send a Reception Complete (Break Occurred) message to the ST\_TFR state machine.

The Reception Complete message, if any, shall include the initiator port transfer tag as an argument.

#### **8.2.6.3.3.6.2 Transition ST\_TTS5:Receive\_Data\_Out to ST\_TTS1:Target\_Start**

This transition shall occur after sending a Reception Complete message to the ST\_TFR state machine.

#### **8.2.6.3.3.6.3 Transition ST\_TTS5:Receive\_Data\_Out to ST\_TTS4:Prepare\_Xfer\_Rdy**

This transition shall occur:

- 1) if the Write Data Offset state machine variable is less than Request Byte Count Data-Out state machine argument plus the Application Client Buffer Offset Data-Out state machine argument and



equal to the Requested Write Data Offset state machine variable plus the Requested Write Data Length state machine variable;

- 2) after an ACK Transmitted confirmation is received for each write DATA frame received;
- 3) after determining the amount of write data already transferred by subtracting the Application Client Buffer Offset Data-Out state machine argument from the Write Data Offset state machine variable;
- 4) after setting the Requested Write Data Length state machine variable to the Request Byte Count Data-Out state machine argument minus the amount of write data already transferred; and
- 5) after setting the Requested Write Data Offset state machine variable to the Write Data Offset state machine variable.

#### **8.2.6.3.3.7 ST\_TTS6:Prepare\_Response state**

##### **8.2.6.3.3.7.1 State description**

This state shall construct a RESPONSE frame using the received Application Response arguments or the received Transport Response arguments as follows:

- a) set the FRAME TYPE field to 07h (i.e., RESPONSE frame);
- b) set the HASHED DESTINATION SAS ADDRESS field to the hashed value of the Application Response or Transport Response Destination SAS Address argument;
- c) set the HASHED SOURCE SAS ADDRESS field to the hashed value of the SSP target port's SAS address;
- d) set the RETRY DATA FRAMES bit to zero;
- e) set the RETRANSMIT bit to zero;
- f) set the CHANGING DATA POINTER bit to zero;
- g) set the INITIATOR PORT TRANSFER TAG field to the Initiator Port Transfer Tag Application Response argument or the Initiator Port Transfer Tag Transport Response argument;
- h) set the TARGET PORT TRANSFER TAG field to a vendor specific value;
- i) set the DATA OFFSET field to 00000000h;
- j) set the information unit as specified in this subclause; and
- k) fill bytes, if needed as specified in this subclause.

If this state was entered with the Transport Response arguments, then this state shall set the fields as follows:

- a) set the NUMBER OF FILL BYTES field to the number of fill bytes, based on the length of the response data, if any;
- b) in the information unit, set the DATAPRES field to RESPONSE DATA;
- c) in the information unit, set the STATUS field to 00h;
- d) in the information unit, set the STATUS QUALIFIER field to 0000h;
- e) in the information unit, set the SENSE DATA LENGTH field to 00000000h;
- f) in the information unit, set the RESPONSE DATA LENGTH field to 00000004h;
- g) in the information unit, set the RESPONSE DATA field as specified in table 234; and
- h) in the information unit, do not include the SENSE DATA field.

Table 234 defines how the RESPONSE DATA field shall be set based on the arguments received with the Request (Send Transport Response) message.

**Table 234 – Request (Send Transport Response) message Service Response argument to RESPONSE frame RESPONSE DATA field mapping**

<b>Request (Send Transport Response) message Service Response argument</b>	<b>RESPONSE frame RESPONSE DATA field</b>
Invalid Frame	INVALID FRAME
Task Management Function Complete	TASK MANAGEMENT FUNCTION COMPLETE
Task Management Function Succeeded	TASK MANAGEMENT FUNCTION SUCCEEDED
Task Management Function Not Supported	TASK MANAGEMENT FUNCTION NOT SUPPORTED
Task Management Function Failed	TASK MANAGEMENT FUNCTION FAILED
Incorrect Logical Unit Number	INCORRECT LOGICAL UNIT NUMBER
Overlapped Initiator Port Transfer Tag Attempted	OVERLAPPED INITIATOR PORT TRANSFER TAG ATTEMPTED

If this state was entered with the Application Response arguments, then this state shall set the fields as follows:

- in the information unit, set the DATAPRES field to SENSE\_DATA if sense data is to be included in the information unit or NO\_DATA if sense data is not to be included in the information unit;
- in the information unit, set the STATUS field to the status;
- in the information unit, set the STATUS QUALIFIER field to the status qualifier, if any;
- in the information unit, set the SENSE DATA LENGTH field to the length of the sense data, if any;
- in the information unit, set the RESPONSE DATA LENGTH field to 00000000h;
- in the information unit, do not include the RESPONSE DATA field;
- in the information unit, set the SENSE DATA field to the sense data, if any; and
- NUMBER OF FILL BYTES field set to the number of fill bytes, based on the length of the sense data, if needed.

#### 8.2.6.3.3.7.2 Transition ST\_TTS6:Prepare\_Response to ST\_TTS2:Target\_Send\_Frame

This transition shall occur:

- after this state constructs a RESPONSE frame.

This transition shall include, if a Cancel message was:

- not received, then the RESPONSE frame as an argument; or
- received, then a Cancel argument.

### 8.3 STP transport layer

#### 8.3.1 Initial FIS

A SATA device phy transmits a Register - Device to Host FIS after completing the link reset sequence, except for the case described in G.5. The expander device shall update a set of shadow registers with the contents of this FIS and shall not deliver this FIS to any STP initiator port. SMP initiator ports may read the shadow register contents using the SMP REPORT PHY SATA function (see 9.4.3.12). The expander device originates a Broadcast (Change) after receiving the Register - Device to Host FIS (see 6.15).

### 8.3.2 BIST Activate FIS

STP initiator ports and STP target ports shall not generate BIST Activate FISes and shall process any BIST Activate FISes received as frames having invalid FIS types (i.e., have the link layer generate SATA\_R\_ERR in response).

### 8.3.3 TT (transport layer for STP ports) state machines

The STP transport layer uses the transport layer state machines defined in SATA, modified to communicate with the port layer rather than directly with the link layer. These modifications are not described in this standard.

## 8.4 SMP transport layer

### 8.4.1 SMP transport layer overview

Table 235 defines the SMP frame format.

**Table 235 – SMP frame format**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE							
1	Frame-type dependent bytes							
...								
n - 4								
n - 3	(MSB)							
...	CRC							
n								

Table 236 defines the SMP FRAME TYPE field, which defines the format of the frame-type dependent bytes.

**Table 236 – SMP FRAME TYPE field**

Code	Name	Frame type	Originator	Reference
40h	SMP_REQUEST	SMP function request	SMP initiator port	8.4.2
41h	SMP_RESPONSE	SMP function response	SMP target port	8.4.3
All others	Reserved			

The number of frame-type dependent bytes shall be either:

- a) three bytes; or
- b) three bytes plus an integer multiple of four bytes,

so the CRC field is aligned on a four byte boundary.

The CRC field contains a CRC value (see 6.7) that is computed over the entire SMP frame prior to the CRC field and shall begin on a four-byte boundary. The CRC field is checked by the SMP link layer (see 6.22).

#### 8.4.2 SMP\_REQUEST frame

The SMP\_REQUEST frame is sent by an SMP initiator port to request an SMP function be performed by a management device server. Table 237 defines the SMP\_REQUEST frame format.

**Table 237 – SMP\_REQUEST frame format**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (40h)							
1	Request bytes							
...								
n - 4								
n - 3	(MSB)	CRC						(LSB)
...								
n								

The SMP FRAME TYPE field shall be set as shown in table 237 for the SMP\_REQUEST frame format.

The format and length of the request bytes is defined by the SMP function description (see 9.4.3.2).

The number of request bytes are either:

- a) three bytes; or
- b) three bytes plus an integer multiple of four bytes,

so the CRC field is aligned on a four byte boundary.

The maximum number of request bytes is 1 023, making the maximum size of the frame 1 028 bytes (i.e., 1 byte header + 1 023 request bytes + 4 bytes of CRC).

NOTE 63 - If a management application client compliant with SAS-1.1 sends a vendor specific SMP request frame containing 1 027 request bytes, then SMP\_TP state machine discards that SMP request frame as it that exceeds the maximum allowed request size of 1 023 bytes (see 6.22.6.4.2.2). SMP request frames defined in SAS-1.1 do not have more than 39 request bytes.

The CRC field is defined in 8.4.1.

#### 8.4.3 SMP\_RESPONSE frame

The SMP\_RESPONSE frame is sent by an SMP target port in response to an SMP\_REQUEST frame. Table 238 defines the SMP\_RESPONSE frame format.

**Table 238 – SMP\_RESPONSE frame format**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (41h)							
1	Response bytes							
...								
n - 4								
n - 3	(MSB)	CRC						(LSB)
...								
n								

The SMP FRAME TYPE field shall be set as shown in table 238 for the SMP\_RESPONSE frame format.

The format and length of the request bytes is defined by the SMP function description (see 9.4.3.3).

The number of response bytes are either:

- a) three bytes; or
- b) three bytes plus an integer multiple of four bytes,

so the CRC field is aligned on a four byte boundary.

The maximum number of response bytes is 1 023, making the maximum size of the frame 1 028 bytes (i.e., 1 byte header + 1 023 request bytes + 4 bytes of CRC).

NOTE 64 - If a management device server compliant with SAS-1.1 sends a vendor specific SMP response frame containing 1 027 response bytes, then the SMP\_IP state machine discards that SMP response frame as it exceeds the maximum allowed request size of 1 023 bytes (see 6.22.6.3.4). SMP response frames defined in SAS-1.1 do not have more than 59 request bytes.

The CRC field is defined in 8.4.1.

#### 8.4.4 Sequence of SMP frames

Inside an SMP connection, the SMP initiator port transmits a single SMP\_REQUEST frame and the SMP target port replies with a single SMP\_RESPONSE frame.

Figure 207 shows the sequence of SMP frames.

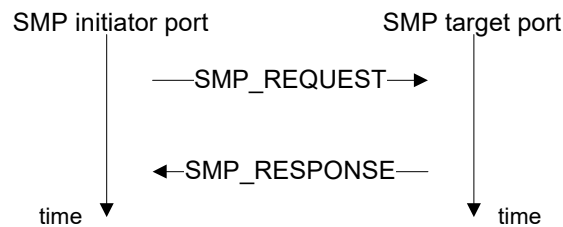


Figure 207 – Sequence of SMP frames

#### 8.4.5 MT (transport layer for SMP ports) state machines

##### 8.4.5.1 SMP transport layer state machines overview

The SMP transport layer contains state machines that process requests from the management application layer and return confirmations to the management application layer. The SMP transport state machines are as follows:

- a) MT\_IP (transport layer for SMP initiator ports) state machine (see 8.4.5.2); and
- b) MT\_TP (transport layer for SMP target ports) state machine (see 8.4.5.3).

##### 8.4.5.2 MT\_IP (transport layer for SMP initiator ports) state machine

###### 8.4.5.2.1 MT\_IP state machine overview

The MT\_IP state machine processes requests from the management application layer. These management requests are sent to the port layer, and the resulting SMP frame or error condition is sent to the management application layer as a confirmation.

This state machine consists of the following states:

- a) MT\_IP1:Idle (see 8.4.5.2.2) (initial state);

- b) MT\_IP2:Send (see 8.4.5.2.3); and
- c) MT\_IP3:Receive (see 8.4.5.2.4).

This state machine shall start in the MT\_IP1:Idle state.

Figure 208 shows the MT\_IP state machine.

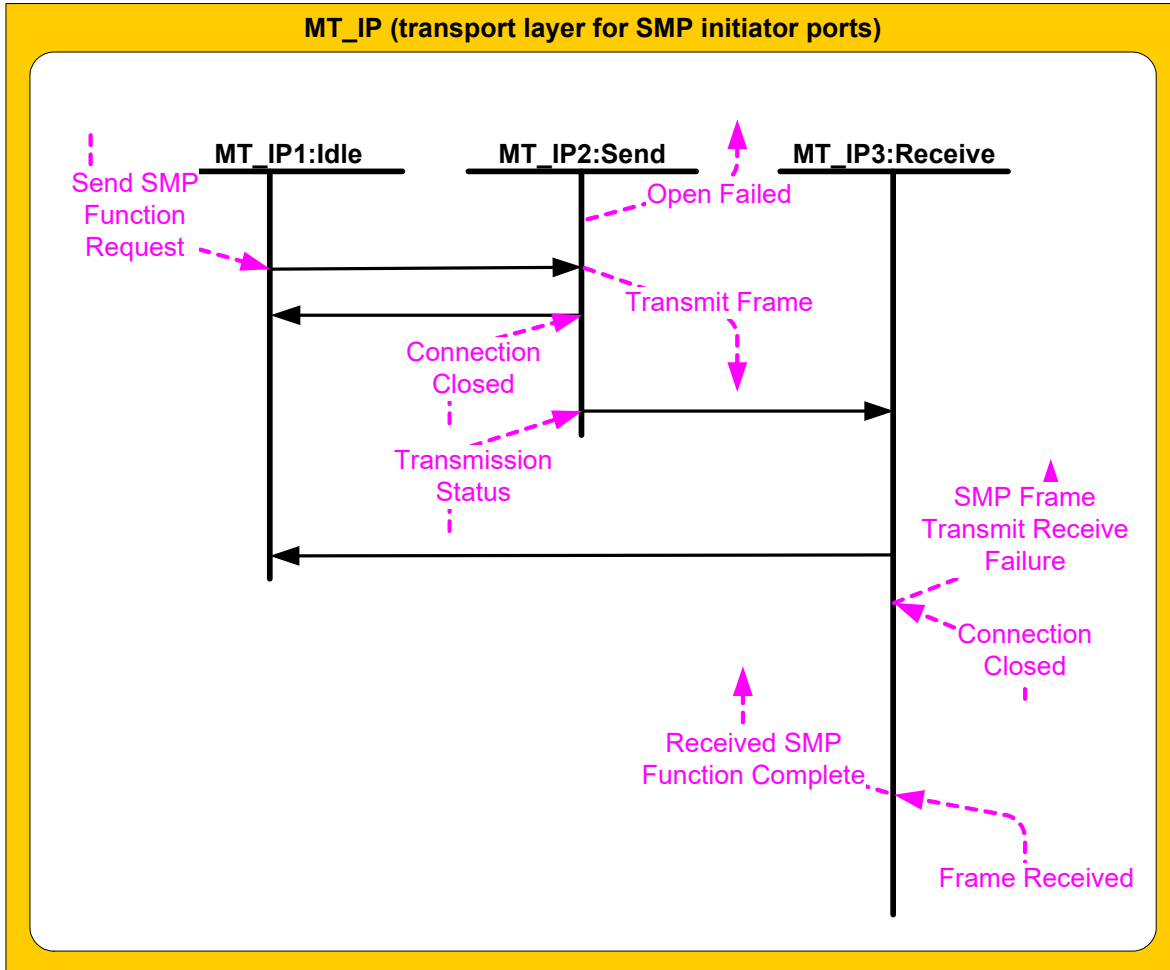


Figure 208 – MT\_IP (transport layer for SMP initiator ports) state machine

#### 8.4.5.2.2 MT\_IP1:Idle state

##### 8.4.5.2.2.1 State description

This state is the initial state of the MT\_IP state machine.

This state waits for a Send SMP Function Request request, which includes the following arguments:

- a) Connection Rate;
- b) Destination SAS Address; and
- c) Request Bytes.

##### 8.4.5.2.2.2 Transition MT\_IP1:Idle to MT\_IP2:Send

This transition shall occur:

- a) after a Send SMP Function Request request is received.

This transition shall include the following arguments:

- a) Connection Rate;
- b) Destination SAS Address; and
- c) Request Bytes.

#### **8.4.5.2.3 MT\_IP2:Send state**

##### **8.4.5.2.3.1 State description**

This state constructs an SMP\_REQUEST frame using the following arguments received in the transition into this state:

- a) Request Bytes

and sends a Transmit Frame request to the port layer with the following arguments:

- a) Initiator Port bit set to one;
- b) Protocol set to SMP;
- c) Connection Rate;
- d) Initiator Connection Tag set to FFFFh;
- e) Destination SAS Address;
- f) Source SAS Address set to the SAS address of the SMP initiator port; and
- g) Request Bytes.

##### **8.4.5.2.3.2 Transition MT\_IP2:Send to MT\_IP1:Idle**

This transition shall occur after:

- a) receiving either a Connection Closed confirmation or a Transmission Status confirmation other than a Transmission Status (Frame Transmitted) confirmation; and
- b) sending an Open Failed confirmation to the management application layer.

##### **8.4.5.2.3.3 Transition MT\_IP2:Send to MT\_IP3:Receive**

This transition shall occur:

- a) after receiving a Transmission Status (Frame Transmitted) confirmation.

#### **8.4.5.2.4 MT\_IP3:Receive state**

##### **8.4.5.2.4.1 State description**

This state waits for a confirmation from the port layer that either an SMP frame has been received or a failure occurred.

If a Frame Received (SMP Successful) confirmation is received and the SMP frame type is equal to 41h, then this state shall send a Received SMP Function Complete confirmation to the management application layer.

If a Frame Received (SMP Successful) confirmation is received and the SMP frame type is not equal to 41h, then this state shall send an SMP Frame Transmit Receive Failure confirmation to the management application layer.

If a Connection Closed or Frame Received (SMP Unsuccessful) confirmation is received, then this state shall send an SMP Frame Transmit Receive Failure confirmation to the management application layer.

##### **8.4.5.2.4.2 Transition MT\_IP3:Receive to MT\_IP1:Idle**

This transition shall occur after one of the following:

- a) sending a Received SMP Function Complete confirmation; or
- b) sending an SMP Frame Transmit Receive Failure confirmation.

### 8.4.5.3 MT\_TP (transport layer for SMP target ports) state machine

#### 8.4.5.3.1 MT\_TP state machine overview

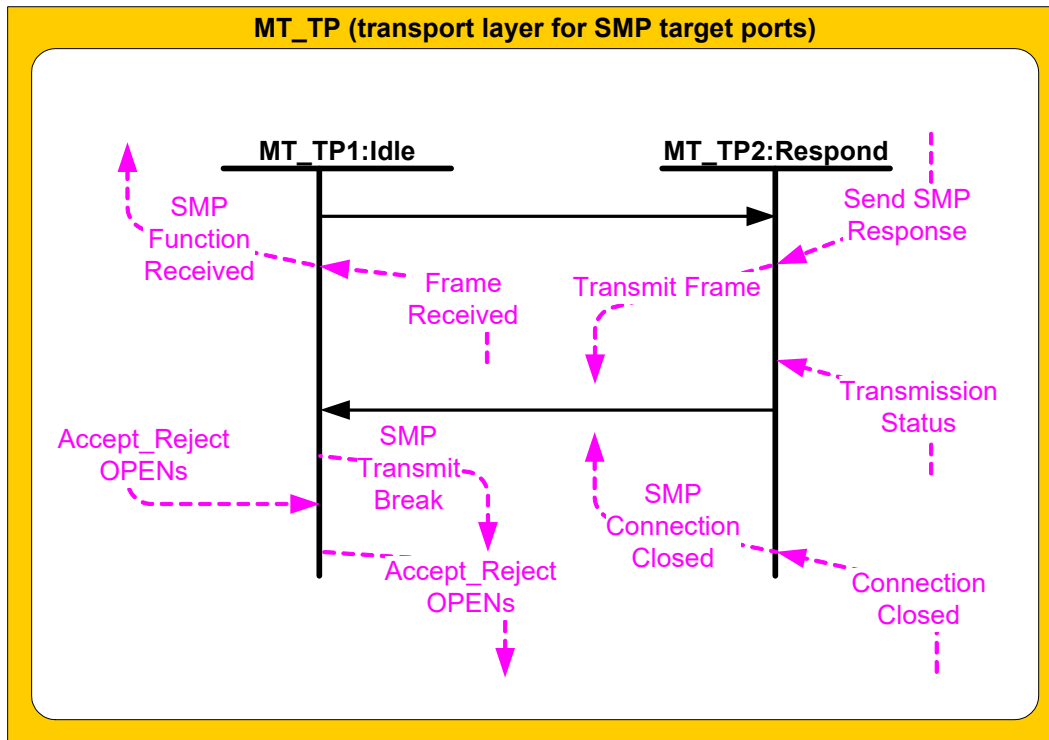
The MT\_TP state machine informs the management application layer of the receipt of an SMP frame and sends the resulting SMP frame to the port layer.

This state machine consists of the following states:

- MT\_TP1:Idle (see 8.4.5.3.2) (initial state); and
- MT\_TP2:Respond (see 8.4.5.3.3).

This state machine shall start in the MT\_TP1:Idle state.

Figure 209 shows the MT\_TP state machine.



**Figure 209 – MT\_TP (transport layer for SMP target ports) state machine**

The MT\_TP state machine shall comply with the time limits listed in table 239.

**Table 239 – MT\_TP time limits**

Time limit	Value	Description
SMP Response time limit	1 900 $\mu$ s	Maximum time from receiving an SMP_REQUEST frame to transmitting an SMP_RESPONSE frame

#### 8.4.5.3.2 MT\_TP1:Idle state

##### 8.4.5.3.2.1 State description

This state is the initial state of the MT\_TP state machine.



This state waits for a Frame Received (SMP Successful) confirmation. If the SMP frame type is not equal to 40h, then this state shall discard the frame and send an SMP Transmit Break request to the port layer, otherwise this state shall send an SMP Function Received confirmation to the management application layer.

If an Accept\_Reject OPENs (Accept SMP) request or an Accept\_Reject OPENs (Reject SMP) request is received, then this state shall send an Accept\_Reject OPENs request with the same arguments to the port layer.

#### **8.4.5.3.2 Transition MT\_TP1:Idle to MT\_TP2:Respond**

This transition shall occur:

- a) after sending an SMP Function Received confirmation.

#### **8.4.5.3.3 MT\_TP2:Respond state**

##### **8.4.5.3.3.1 State description**

This state waits for a Send SMP Response request, which includes the following argument:

- a) Response Bytes.

After receiving a Send SMP Response request, this state shall construct an SMP\_RESPONSE frame using the arguments from the Send SMP Response request and send a Transmit Frame request to the port layer within the SMP Response time limit specified in table 239 (see 8.4.5.3.1).

If this state receives a Connection Closed confirmation, then this state shall send an SMP Connection Closed confirmation to the management application layer.

##### **8.4.5.3.3.2 Transition MT\_TP2:Respond to MT\_TP1:Idle**

This transition shall occur after one of the following:

- a) receiving a Transmission Status (Frame Transmitted) confirmation; or
- b) sending an SMP Connection Closed confirmation.

## 9 Application layer

### 9.1 Application layer overview

The application layer defines SCSI, ATA, and management specific features.

### 9.2 SCSI application layer

#### 9.2.1 SCSI transport protocol services

##### 9.2.1.1 SCSI transport protocol services overview

A SCSI application client requests the processing of a SCSI command by invoking SCSI transport protocol services, the collective operation of which is conceptually modeled in the following procedure call (see SAM-5):

**Service response = Execute Command (IN (I\_T\_L Nexus, Command Identifier, CDB, Task Attribute, [Data-In Buffer Size], [Data-Out Buffer], [Data-Out Buffer Size], [Command Priority]), OUT ([Data-In Buffer], [Sense Data], [Sense Data Length], Status, [Status Qualifier]))**

This standard defines the transport protocol services required by SAM-5 in support of this procedure call. Table 240 describes the mapping of the **Execute Command** procedure call to transport protocol services and the SSP implementation of each transport protocol service.

**Table 240 – Execute Command procedure call transport protocol services**

Transport protocol service	I/T <sup>a</sup>	SSP implementation	Reference
<b>Command and status</b>			
<b>Send SCSI Command request</b>	I	COMMAND frame	9.2.1.2
<b>SCSI Command Received indication</b>	T	Receipt of the COMMAND frame	9.2.1.3
<b>Send Command Complete response</b>	T	RESPONSE frame	9.2.1.4
<b>Command Complete Received confirmation</b>	I	Receipt of the RESPONSE frame or problem transmitting the COMMAND frame	9.2.1.5
<b>Data-In delivery <sup>b</sup></b>			
<b>Send Data-In request</b>	T	Read DATA frames	9.2.1.6
<b>Data-In Delivered confirmation</b>	T	Receipt of ACKs for the read DATA frames	9.2.1.7
<b>Data-Out delivery <sup>b</sup></b>			
<b>Receive Data-Out request</b>	T	XFER_RDY frame	9.2.1.8
<b>Data-Out Received confirmation</b>	T	Receipt of write DATA frames	9.2.1.9
<b>Terminate Data Transfer <sup>b</sup></b>			
<b>Terminate Data Transfer request</b>	T		9.2.1.10
<b>Data Transfer Terminated confirmation</b>	T		9.2.1.11
<sup>a</sup> I/T indicates whether the SSP initiator port (I) or the SSP target port (T) implements the transport protocol service. <sup>b</sup> Data transfer transport protocol services for SCSI initiator ports are not specified by SAM-5.			

A SCSI application client requests the processing of a SCSI task management function by invoking SCSI transport protocol services, the collective operation of which is conceptually modeled in the following procedure calls (see SAM-5):

- a) **Service Response = ABORT TASK (IN (Nexus));**
- b) **Service Response = ABORT TASK SET (IN (Nexus));**
- c) **Service Response = CLEAR ACA (IN (Nexus));**
- d) **Service Response = CLEAR TASK SET (IN (Nexus));**
- e) **Service Response = I\_T NEXUS RESET (IN (Nexus));**
- f) **Service Response = LOGICAL UNIT RESET (IN (Nexus));**
- g) **Service Response = QUERY TASK (IN (Nexus));**
- h) **Service Response = QUERY TASK SET (IN (Nexus), OUT ([Additional Response Information]));**  
and
- i) **Service Response = QUERY ASYNCHRONOUS EVENT (IN (Nexus), OUT ([Additional Response Information]));**

This standard defines the transport protocol services required by SAM-5 in support of these procedure calls. Table 241 describes the mapping of these procedure calls to transport protocol services and the SSP implementation of each transport protocol service.

**Table 241 – Task management function procedure call transport protocol services**

Transport protocol service	I/T <sup>a</sup>	SSP implementation	Reference
Task management			
<b>Send Task Management Request request</b>	I	TASK frame	9.2.1.12
<b>Task Management Request Received indication</b>	T	Receipt of the TASK frame	9.2.1.13
<b>Task Management Function Executed response</b>	T	RESPONSE frame	9.2.1.14
<b>Received Task Management Function Executed confirmation</b>	I	Receipt of the RESPONSE frame or problem transmitting the TASK frame	9.2.1.15
<sup>a</sup> I/T indicates whether the SSP initiator port (I) or the SSP target port (T) implements the transport protocol service.			

Transport protocol services are used as the requests and confirmations to the SSP transport layer state machines (see 8.2.6) from the SCSI application layer.

#### 9.2.1.2 Send SCSI Command SCSI transport protocol service

A SCSI application client invokes the **Send SCSI Command** SCSI transport protocol service request to request that an SSP initiator port transmit a COMMAND frame.

**Send SCSI Command (IN (I\_T\_L Nexus, Command Identifier, CDB, Task Attribute, [Data-In Buffer Size], [Data-Out Buffer], [Data-Out Buffer Size], [Command Priority], [CRN], [First Burst Enabled], [Request Fence]))**

Table 242 shows how the arguments to the **Send SCSI Command** SCSI transport protocol service are used.

**Table 242 – Send SCSI Command SCSI transport protocol service arguments**

Argument	SAS SSP implementation
I_T_L nexus	I_T_L nexus, where: a) I specifies the SSP initiator port to send the COMMAND frame; b) T specifies the SSP target port to which the COMMAND frame is to be sent; and c) L specifies the LOGICAL UNIT NUMBER field in the COMMAND frame header.
Command Identifier	Specifies the INITIATOR PORT TRANSFER TAG field in the COMMAND frame header.
CDB	Specifies the CDB field in the COMMAND frame.
Task Attribute	Specifies the TASK ATTRIBUTE field in the COMMAND frame.
[Data-In Buffer Size]	Maximum of $2^{32}$ bytes. <sup>a</sup>
[Data-Out Buffer]	Internal to the SSP initiator port.
[Data-Out Buffer Size]	Maximum of $2^{32}$ bytes. <sup>a</sup>
[CRN]	Ignored
[Command Priority]	Specifies the COMMAND PRIORITY field in the COMMAND frame.
[First Burst Enabled]	Specifies the ENABLE FIRST BURST bit in the COMMAND frame and causes the SSP initiator port to transmit the number of bytes indicated by the FIRST BURST SIZE field in the Disconnect-Reconnect mode page (see 9.2.7.2.5) for the SSP target port without waiting for an XFER_RDY frame.
[Request Fence]	If included, specifies an I_T nexus, I_T_L nexus, or I_T_L nexus and command identifier combination for which the COMMAND frame is fenced.
<sup>a</sup> See the restrictions on the REQUESTED OFFSET field and the WRITE DATA LENGTH field in the SSP XFER_RDY frame (see 8.2.2.3) and the DATA OFFSET field in the SSP DATA frame (see 8.2.1).	

A SCSI application client shall set the Request Fence argument to the nexus containing any commands or task management functions that the command affects (e.g., for a PERSISTENT RESERVE OUT command with a PREEMPT AND ABORT service action, the SCSI application client sets the Response Fence argument to the I\_T\_L nexus) or upon which the command depends (e.g., when the Task Attribute argument is set to Ordered, the SCSI application client sets the Response Fence argument to the I\_T\_L nexus and command identifier combination of the previous command). If the SCSI application client is not able to determine the nexus affected by the command or upon which the command depends, then the SCSI application client should set the Request Fence argument to the I\_T nexus.

### 9.2.1.3 SCSI Command Received SCSI transport protocol service

An SSP target port invokes the **SCSI Command Received** SCSI transport protocol service indication to notify a task manager that the SSP target port has received a COMMAND frame.

**SCSI Command Received (IN (I\_T\_L Nexus, Command Identifier, CDB, Task Attribute, [Command Priority], [CRN], [First Burst Enabled]))**

Table 243 shows how the arguments to the **SCSI Command Received** SCSI transport protocol service are determined.

**Table 243 – SCSI Command Received SCSI transport protocol service arguments**

Argument	SAS SSP implementation
I_T_L nexus	I_T_L nexus, where: a) I indicates the SSP initiator port that sent the COMMAND frame; b) T indicates the SSP target port that received the COMMAND frame; and c) L indicates the value of the LOGICAL UNIT NUMBER field in the COMMAND frame header.
Command Identifier	Indicates the value of the INITIATOR PORT TRANSFER TAG field in the COMMAND frame header.
CDB	Indicates the value of the CDB field and the ADDITIONAL CDB BYTES field, if any, in the COMMAND frame.
Task Attribute	Indicates the value of the TASK ATTRIBUTE field in the COMMAND frame.
[CRN]	Ignored
[Command Priority]	Indicates the value of the COMMAND PRIORITY field in the COMMAND frame.
[First Burst Enabled]	Indicates that first burst data is being delivered based on the ENABLE FIRST BURST bit in the COMMAND frame and the FIRST BURST SIZE field in the Disconnect-Reconnect mode page (see 9.2.7.2.5).

#### 9.2.1.4 Send Command Complete SCSI transport protocol service

A SCSI device server invokes the **Send Command Complete** SCSI transport protocol service response to request that an SSP target port transmit a RESPONSE frame.

**Send Command Complete (IN (I\_T\_L Nexus, Command Identifier, [Sense Data], [Sense Data Length], Status, [Status Qualifier], Service Response, [Response Fence]))**

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

A SCSI device server shall not call **Send Command Complete** () for a given I\_T\_L nexus and command identifier combination until the SCSI device server has:

- responded to all outstanding **Receive Data-Out** () calls for that I\_T\_L nexus and command identifier combination with **Data-Out Received** (); and
- responded to all outstanding **Send Data-In** () calls for that I\_T\_L nexus and command identifier combination with **Data-In Delivered** ().

Table 244 shows how the arguments to the **Send Command Complete** SCSI transport protocol service are used.

**Table 244 – Send Command Complete SCSI transport protocol service arguments**

Argument	SAS SSP implementation
I_T_L nexus	I_T_L nexus, where: a) I specifies the SSP initiator port to which the RESPONSE frame is to be sent; b) T specifies the SSP target port to send the RESPONSE frame; and c) L specifies the LOGICAL UNIT NUMBER field in the RESPONSE frame header.
Command Identifier	Specifies the INITIATOR PORT TRANSFER TAG field in the RESPONSE frame header.
[Sense Data]	Specifies the SENSE DATA field in the RESPONSE frame.
[Sense Data Length]	Specifies the SENSE DATA LENGTH field in the RESPONSE frame.
Status	Specifies the STATUS field in the RESPONSE frame.
[Status Qualifier]	Specifies the STATUS QUALIFIER field in the RESPONSE frame.
Service Response	Specifies the DATAPRES field and STATUS field in the RESPONSE frame: a) COMMAND COMPLETE: The DATAPRES field is set to NO_DATA or SENSE_DATA; or b) SERVICE DELIVERY OR TARGET FAILURE - Overlapped Initiator Port Transfer Tag Attempted: The DATAPRES field is set to RESPONSE_DATA and the RESPONSE CODE field is set to OVERLAPPED INITIATOR PORT TRANSFER TAG ATTEMPTED.
[Response Fence]	If included, specifies an I_T nexus, I_T_L nexus, or I_T_L nexus and command identifier combination for which the RESPONSE frame is fenced.

A SCSI device server shall set the Response Fence argument to the nexus containing any commands or task management functions that the command affects (e.g., for a PERSISTENT RESERVE OUT command with a PREEMPT AND ABORT service action, the SCSI device server sets the Response Fence argument to the I\_T\_L nexus) or upon which the command completion depends (e.g., when returning a unit attention condition with the additional sense code set to COMMANDS CLEARED BY ANOTHER INITIATOR, the SCSI device server sets the Response Fence argument to the I\_T\_L nexus). If the SCSI device server is not able to determine the nexus affected by the command or upon which the command depends, then the SCSI device server should set the Response Fence argument to the I\_T nexus.

#### 9.2.1.5 Command Complete Received SCSI transport protocol service

An SSP initiator port invokes the **Command Complete Received** SCSI transport protocol service confirmation to notify a SCSI application client that the SSP initiator port has received a response for its COMMAND frame (e.g., a RESPONSE frame or a NAK) or terminated a command because of an error.

**Command Complete Received (IN (I\_T\_L Nexus, Command Identifier, [Data-In Buffer], [Sense Data], [Sense Data Length], Status, [Status Qualifier], Service Response))**

Table 245 shows how the arguments to the **Command Complete Received** SCSI transport protocol service are determined.

**Table 245 – Command Complete Received SCSI transport protocol service arguments**

Argument	SAS SSP implementation
I_T_L nexus	I_T_L nexus, where: a) I indicates the SSP initiator port that received the RESPONSE frame; b) T indicates the SSP target port that sent the RESPONSE frame; and c) L indicates the value of the LOGICAL UNIT NUMBER field in the RESPONSE frame header or COMMAND frame header.
Command Identifier	Indicates the value of the INITIATOR PORT TRANSFER TAG field in the RESPONSE frame header or COMMAND frame header.
[Data-In Buffer]	Internal to the SSP initiator port.
[Sense Data]	Indicates the value of the SENSE DATA field in the RESPONSE frame.
[Sense Data Length]	The smaller of the value of the SENSE DATA LENGTH field in the RESPONSE frame and the actual number of sense data bytes received by the SSP initiator port.
Status	Indicates the value of the STATUS field in the RESPONSE frame.
[Status Qualifier]	Indicates the value of the STATUS QUALIFIER field in the RESPONSE frame.
Service Response	Either: a) COMMAND COMPLETE: The RESPONSE frame contains a DATAPRES field set to NO_DATA or SENSE_DATA; or b) SERVICE DELIVERY OR TARGET FAILURE: Either: A) the RESPONSE frame contains a DATAPRES field set to RESPONSE_DATA and a RESPONSE CODE field set to INVALID FRAME or OVERLAPPED INITIATOR PORT TRANSFER TAG ATTEMPTED; or B) the ST_IFR state machine detects an error as described in 8.2.6.2.2.3 and 8.2.6.2.2.4 (e.g., a NAK was received for the COMMAND frame or the length of the RESPONSE frame is incorrect).

#### 9.2.1.6 Send Data-In SCSI transport protocol service

A SCSI device server invokes the **Send Data-In** SCSI transport protocol service request to request that an SSP target port transmit a read DATA frame.

**Send Data-In (IN (I\_T\_L Nexus, Command Identifier, Device Server Buffer, Application Client Buffer Offset, Request Byte Count))**

A SCSI device server shall only call **Send Data-In** () during a read command or bidirectional command.

A SCSI device server shall not call **Send Data-In** () for a given I\_T\_L nexus and command identifier combination after the SCSI device server has called **Send Command Complete** () for that I\_T\_L nexus and command identifier combination (e.g., a RESPONSE frame for that I\_T\_L nexus and command identifier combination has been transmitted) or called Task Management Function Executed () for a task management function that terminates that command (e.g., an ABORT TASK).



Table 246 shows how the arguments to the **Send Data-In** SCSI transport protocol service are used.

**Table 246 – Send Data-In SCSI transport protocol service arguments**

Argument	SAS SSP implementation
I_T_L nexus	I_T_L nexus, where: a) I specifies the SSP initiator port to which the read DATA frame is to be sent; b) T specifies the SSP target port to send the read DATA frame; and c) L specifies the LOGICAL UNIT NUMBER field in the read DATA frame header.
Command Identifier	Specifies the INITIATOR PORT TRANSFER TAG field in the read DATA frame header.
Device Server Buffer	Internal to the SCSI device server.
Application Client Buffer Offset	Specifies the DATA OFFSET field in the read DATA frame.
Request Byte Count	Specifies the size of the read DATA frame.

#### 9.2.1.7 Data-In Delivered SCSI transport protocol service

An SSP target port invokes the **Data-In Delivered** SCSI transport protocol service indication to notify a SCSI device server of the results of transmitting a read DATA frame.

##### **Data-In Delivered (IN (I\_T\_L Nexus, Command Identifier, Delivery Result))**

Table 247 shows how the arguments to the **Data-In Delivered** SCSI transport protocol service are determined.

**Table 247 – Data-In Delivered SCSI transport protocol service arguments**

Argument	SAS SSP implementation
I_T_L nexus	I_T_L nexus, where: a) I indicates the SSP initiator port that sent the read DATA frame; b) T indicates the SSP target port that received the read DATA frame; and c) L indicates the value of the LOGICAL UNIT NUMBER field in the read DATA frame header.
Command Identifier	Indicates the value of the INITIATOR PORT TRANSFER TAG field in the read DATA frame header.
Delivery Result	From the response to the outgoing read DATA frame: a) DELIVERY SUCCESSFUL: The read DATA frame received an ACK; or b) DELIVERY FAILURE: The read DATA frame received a NAK or no response.

#### 9.2.1.8 Receive Data-Out SCSI transport protocol service

A SCSI device server invokes the **Receive Data-Out** SCSI transport protocol service request to request that an SSP target port transmit an XFER\_RDY frame.

##### **Receive Data-Out (IN (I\_T\_L Nexus, Command Identifier, Application Client Buffer Offset, Request Byte Count, Device Server Buffer))**

A SCSI device server shall only call **Receive Data-Out** () during a write command or bidirectional command.

A SCSI device server shall not call **Receive Data-Out** () for a given I\_T\_L nexus and command identifier combination until the **Data-Out Received** () has completed without error for the previous **Receive Data-Out** () call for that I\_T\_L nexus and command identifier combination (i.e., no XFER\_RDY frame shall be transmitted until all write DATA frames for the previous XFER\_RDY frame, if any, have been received, and the link layer has provided acknowledgement for all of the previous write DATA frames for that I\_T\_L nexus and command identifier combination).

A SCSI device server shall not call **Receive Data-Out** () for a given I\_T\_L nexus and command identifier combination after a **Send Command Complete** () has been called for that I\_T\_L nexus and command identifier combination or after a **Task Management Function Executed** () has been called for a task management function that terminates that command (e.g., an ABORT TASK).

If the Protocol Specific Port mode page (see 9.2.7.4) is supported and the value in the MAXIMUM ALLOWED XFER\_RDY field is not set to zero, then a SCSI device server shall not call **Receive Data-Out** () for a given I\_T\_L nexus more than the number of times specified in the MAXIMUM ALLOWED XFER\_RDY field in the Protocol Specific Port mode page, until a **Data-Out Received** () has completed without error for one of the previous **Receive Data-Out** () calls for that I\_T\_L nexus. For each **Data-Out Received** () that completes without error for one of the previous **Receive Data-Out** () calls for that I\_T\_L nexus, the device server may call **Receive Data-Out** () for that I\_T\_L nexus.

Table 248 shows how the arguments to the **Receive Data-Out** SCSI transport protocol service are used.

**Table 248 – Receive Data-Out SCSI transport protocol service arguments**

Argument	SAS SSP implementation
I_T_L nexus	I_T_L nexus, where: a) I specifies the SSP initiator port to which the XFER_RDY frame is to be sent; b) T specifies the SSP target port to send the XFER_RDY frame; and c) L specifies the LOGICAL UNIT NUMBER field in the XFER_RDY frame header.
Command Identifier	Specifies the INITIATOR PORT TRANSFER TAG field in the XFER_RDY frame header.
Application Client Buffer Offset	Specifies the REQUESTED OFFSET field in the XFER_RDY frame.
Request Byte Count	Specifies WRITE DATA LENGTH field in the XFER_RDY frame.
Device Server Buffer	Internal to the SCSI device server.

#### 9.2.1.9 Data-Out Received SCSI transport protocol service

An SSP target port invokes the **Data-Out Received** SCSI transport protocol service indication to notify a SCSI device server of the result of transmitting an XFER\_RDY frame (e.g., receiving write DATA frames in response).

**Data-Out Received (IN (I\_T\_L Nexus, Command Identifier, Delivery Result))**

Table 249 shows how the arguments to the **Data-Out Received** SCSI transport protocol service are determined.

**Table 249 – Data-Out Received SCSI transport protocol service arguments**

Argument	SAS SSP implementation
I_T_L nexus	I_T_L nexus, where: a) I indicates the SSP initiator port to which the XFER_RDY frame was sent; b) T indicates the SSP target port that sent the XFER_RDY frame; and c) L indicates the value of the LOGICAL UNIT NUMBER field in the XFER_RDY frame header.
Command Identifier	Indicates the value of the INITIATOR PORT TRANSFER TAG field in the XFER_RDY frame header.
Delivery Result	From the response to the XFER_RDY: a) DELIVERY SUCCESSFUL: The XFER_RDY frame was transmitted without error and all the write DATA frames for the requested write data were received; or b) DELIVERY FAILURE: The XFER_RDY frame received a NAK or no response.

#### 9.2.1.10 Terminate Data Transfer SCSI transport protocol service request

A SCSI device server invokes the **Terminate Data Transfer** SCSI transport protocol service request to request that an SSP target port terminate any **Send Data-In** () or **Receive Data-Out** () SCSI transport protocol services transport protocol services, if any, being processed using the specified nexus.

**Terminate Data Transfer** SCSI transport protocol service request:

**Terminate Data Transfer (IN (Nexus, [Command Identifier]))**

Table 250 shows how the arguments to the **Terminate Data Transfer** SCSI transport protocol service are used.

**Table 250 – Terminate Data Transfer SCSI transport protocol service arguments**

Argument	SAS SSP implementation
Nexus	I_T nexus or I_T_L nexus identifying the scope of the data transfers to terminate.
Command Identifier	Identifies the command associated with the data transfer being terminated.

#### 9.2.1.11 Data Transfer Terminated SCSI transport protocol service confirmation

An SSP target port invokes the **Data Transfer Terminated** SCSI transport protocol service confirmation to notify a SCSI device server that all data transfers for the indicated nexus have been terminated.

**Data Transfer Terminated** SCSI transport protocol service confirmation:

**Data Transfer Terminated (IN (Nexus), [Command Identifier])**

Table 251 shows how the arguments to the **Data Transfer Terminated** SCSI transport protocol service are determined.

**Table 251 – Data Transfer Terminated SCSI transport protocol service arguments**

Argument	SAS SSP implementation
Nexus	I_T nexus or I_T_L nexus identified by the preceding <b>Terminate Data Transfer</b> () call.
Command Identifier	Identifies the command associated with the data transfer being terminated.

#### 9.2.1.12 Send Task Management Request SCSI transport protocol service

A SCSI application client invokes the **Send Task Management Request** SCSI transport protocol service request to request that an SSP initiator port transmit a TASK frame.

##### **Send Task Management Request (IN (Nexus, [Command Identifier], Function Identifier, Task Management Tag, [Request Fence]))**

Table 252 shows how the arguments to the **Send Task Management Request** SCSI transport protocol service are used.

**Table 252 – Send Task Management Request SCSI transport protocol service arguments**

Argument	SAS SSP implementation
Nexus	I_T nexus or I_T_L nexus (depending on the Function Identifier), where: a) I specifies the SSP initiator port to send the TASK frame; b) T specifies the SSP target port to which the TASK frame is sent; and c) L specifies the LOGICAL UNIT NUMBER field in the TASK frame header, if any.
Command Identifier	Specifies the INITIATOR PORT TRANSFER TAG TO MANAGE field in the TASK frame header.
Function Identifier	Specifies the TASK MANAGEMENT FUNCTION field in the TASK frame. Only these task management functions are supported: a) ABORT TASK (Nexus argument specifies an I_T_L Nexus and a command identifier); b) ABORT TASK SET (Nexus argument specifies an I_T_L Nexus); c) CLEAR ACA (Nexus argument specifies an I_T_L Nexus); d) CLEAR TASK SET (Nexus argument specifies an I_T_L Nexus); e) I_T NEXUS RESET (Nexus argument specifies an I_T Nexus); f) LOGICAL UNIT RESET (Nexus argument specifies an I_T_L Nexus); g) QUERY TASK (Nexus argument specifies an I_T_L Nexus and a command identifier); h) QUERY TASK SET (Nexus argument specifies an I_T_L Nexus); and i) QUERY ASYNCHRONOUS EVENT (Nexus argument specifies an I_T_L Nexus).
Task Management Tag	Specifies the INITIATOR PORT TRANSFER TAG field in the TASK frame header.
[Request Fence]	If included, specifies an I_T nexus, I_T_L nexus, or I_T_L nexus and command identifier combination for which the TASK frame is fenced.

A SCSI application client shall set the Request Fence argument to the Nexus argument.

### 9.2.1.13 Task Management Request Received SCSI transport protocol service

An SSP target port invokes the **Task Management Request Received** SCSI transport protocol service indication to notify a task manager that the SSP target port has received a TASK frame.

**Task Management Request Received (IN (Nexus, [Command Identifier], Function Identifier, Task Management Tag))**

Table 253 shows how the arguments to the **Task Management Request Received** SCSI transport protocol service are determined.

**Table 253 – Task Management Request Received SCSI transport protocol service arguments**

Argument	SAS SSP implementation
Nexus	I_T nexus or I_T_L nexus (depending on the Function Identifier), where: a) I indicates the SSP initiator port that sent the TASK frame; b) T indicates the SSP target port that received the TASK frame; and c) L indicates the LOGICAL UNIT NUMBER field in the TASK frame header, if any.
Command Identifier	Indicates the value of the INITIATOR PORT TRANSFER TAG TO MANAGE field in the TASK frame header.
Function Identifier	Indicates the value of the TASK MANAGEMENT FUNCTION field in the TASK frame. Only these task management functions are supported: a) ABORT TASK (Nexus argument indicates an I_T_L Nexus and a command identifier); b) ABORT TASK SET (Nexus argument indicates an I_T_L Nexus); c) CLEAR ACA (Nexus argument indicates an I_T_L Nexus); d) CLEAR TASK SET (Nexus argument indicates an I_T_L Nexus); e) I_T NEXUS RESET (Nexus argument indicates an I_T Nexus); f) LOGICAL UNIT RESET (Nexus argument indicates an I_T_L Nexus); g) QUERY TASK (Nexus argument indicates an I_T_L Nexus and a command identifier); h) QUERY TASK SET (Nexus argument indicates an I_T_L Nexus); and i) QUERY ASYNCHRONOUS EVENT (Nexus argument indicates an I_T_L Nexus).
Task Management Tag	Indicates the value of the INITIATOR PORT TRANSFER TAG field in the TASK frame header.

### 9.2.1.14 Task Management Function Executed SCSI transport protocol service

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

**Task Management Function Executed (IN (Nexus, [Command Identifier], Service Response, [Additional Response Information], Task Management Tag, [Response Fence]))**

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

Table 254 shows how the arguments to the **Task Management Function Executed** SCSI transport protocol service are used.

**Table 254 – Task Management Function Executed SCSI transport protocol service arguments**

Argument	SAS SSP implementation
Nexus	I_T_L nexus, where: a) I specifies the SSP initiator port to which the RESPONSE frame is sent; b) T specifies the SSP target port to send the RESPONSE frame; and c) L specifies the logical unit that is sending the response frame, if any.
Command Identifier	Specifies the command that was managed by the task management function.
Service Response	Specifies the DATAPRES field and RESPONSE CODE field in the RESPONSE frame: a) FUNCTION COMPLETE: The DATAPRES field is set to RESPONSE_DATA and the RESPONSE CODE field is set to TASK MANAGEMENT FUNCTION COMPLETE; b) FUNCTION SUCCEEDED: The DATAPRES field is set to RESPONSE_DATA and the RESPONSE CODE field is set to TASK MANAGEMENT FUNCTION SUCCEEDED; c) FUNCTION REJECTED: The DATAPRES field is set to RESPONSE_DATA and the RESPONSE CODE field is set to TASK MANAGEMENT FUNCTION NOT SUPPORTED; d) INCORRECT LOGICAL UNIT NUMBER: The DATAPRES field is set to RESPONSE_DATA and the RESPONSE CODE field is set to INCORRECT LOGICAL UNIT NUMBER; e) SERVICE DELIVERY OR TARGET FAILURE: The DATAPRES field is set to RESPONSE_DATA and the RESPONSE CODE field is set to TASK MANAGEMENT FUNCTION FAILED; or f) SERVICE DELIVERY OR TARGET FAILURE - Overlapped Initiator Port Transfer Tag Attempted: The DATAPRES field is set to RESPONSE_DATA and the RESPONSE CODE field is set to OVERLAPPED INITIATOR PORT TRANSFER TAG ATTEMPTED.
[Additional Response Information]	Specifies the ADDITIONAL RESPONSE INFORMATION field in the RESPONSE frame.
Task Management Tag	Specifies the INITIATOR PORT TRANSFER TAG field in the RESPONSE frame header.
[Response Fence]	If included, specifies an I_T nexus, I_T_L nexus, or I_T_L nexus and command identifier combination for which the RESPONSE frame is fenced.

A SCSI device server shall set the Response Fence argument to the Nexus argument.

#### 9.2.1.15 Received Task Management Function Executed SCSI transport protocol service

An SSP initiator port invokes the **Received Task Management Function Executed** SCSI transport protocol service confirmation to notify a SCSI application client that the SSP initiator port has received a response to a TASK frame (e.g., received a RESPONSE frame or a NAK).

**Received Task Management Function Executed (IN (Nexus, [Command Identifier], Service Response, [Additional Response Information], Task Management Tag))**

Table 255 shows how the arguments to the **Received Task Management Function Executed** SCSI transport protocol service are determined.

**Table 255 – Received Task Management Function Executed SCSI transport protocol service arguments**

Argument	SAS SSP implementation
Nexus	I_T nexus or I_T_L nexus (depending on the function), where: a) I indicates the SSP initiator port that received the RESPONSE frame; b) T indicates the SSP target port that sent the RESPONSE frame; and c) L, if any, indicates the logical unit that sent the response frame and is indicated by the LOGICAL UNIT NUMBER field of the TASK frame with an INITIATOR PORT TRANSFER TAG field equal to the INITIATOR PORT TRANSFER TAG field in the RESPONSE frame header.
Command Identifier	Indicates the command that was managed by the task management function, (i.e., the Command Identifier is set to the contents of the INITIATOR PORT TRANSFER TAG TO MANAGE field of the TASK frame that contained an INITIATOR PORT TRANSFER TAG field
Service Response	Indicates the response to the TASK frame: a) FUNCTION COMPLETE: The RESPONSE frame contains a DATAPRES field set to RESPONSE_DATA and a RESPONSE CODE field set to TASK MANAGEMENT FUNCTION COMPLETE; b) FUNCTION SUCCEEDED: The RESPONSE frame contains a DATAPRES field set to RESPONSE_DATA and a RESPONSE CODE field set to TASK MANAGEMENT FUNCTION SUCCEEDED; c) FUNCTION REJECTED: The RESPONSE frame contains a DATAPRES field set to RESPONSE_DATA and a RESPONSE CODE field set to TASK MANAGEMENT FUNCTION NOT SUPPORTED; d) INCORRECT LOGICAL UNIT NUMBER: The RESPONSE frame contains a DATAPRES field set to RESPONSE_DATA and a RESPONSE CODE field set to INCORRECT LOGICAL UNIT NUMBER; or e) SERVICE DELIVERY OR TARGET FAILURE: The ST_IFR state machine detects an error as described in 8.2.6.2.2.3 and 8.2.6.2.2.4 (e.g., a NAK was received for the COMMAND frame or the length of the RESPONSE frame is incorrect), or the RESPONSE frame contains a DATAPRES field set to RESPONSE_DATA and a RESPONSE CODE field set to: A) INVALID FRAME; B) TASK MANAGEMENT FUNCTION FAILED; or C) OVERLAPPED INITIATOR PORT TRANSFER TAG ATTEMPTED.
[Additional Response Information]	Indicates the ADDITIONAL RESPONSE INFORMATION field in the RESPONSE frame.
Task Management Tag	Indicates the INITIATOR PORT TRANSFER TAG field in the RESPONSE frame header or the TASK frame header.

### 9.2.2 SCSI application client error handling

If a SCSI application client processes **Command Complete Received** () with a Service Response of:

- a) Service Delivery or Target Failure - XFER\_RDY Incorrect Write Data Length;
- b) Service Delivery or Target Failure - XFER\_RDY Requested Offset Error;
- c) Service Delivery or Target Failure - XFER\_RDY Not Expected;

- d) Service Delivery or Target Failure - DATA Incorrect Data Length;
- e) Service Delivery or Target Failure - DATA Too Much Read Data;
- f) Service Delivery or Target Failure - DATA Data Offset Error;
- g) Service Delivery or Target Failure - DATA Not Expected;
- h) Service Delivery or Target Failure - RESPONSE Incorrect Length;
- i) Service Delivery or Target Failure - NAK Received; or
- j) Service Delivery or Target Failure - Connection Failed,

then the SCSI application client shall abort the command (e.g., by sending an ABORT TASK task management function).

After a SCSI application client calls **Send SCSI Command ()**, if **Command Complete Received ()** returns a Service Response of Service Delivery or Target Failure - ACK/NAK Timeout, then the SCSI application client shall send a QUERY TASK task management function with **Send Task Management Request ()** to determine whether the command was received with no error. If **Received Task Management Function Executed ()** returns a Service Response of:

- a) FUNCTION SUCCEEDED, then the SCSI application client shall assume the command was delivered with no error; or
- b) FUNCTION COMPLETE and **Command Complete Received ()** has not yet been invoked a second time for the command in question (e.g., indicating a RESPONSE frame arrived for the command before the QUERY TASK was processed), then the SCSI application client shall assume the command was not delivered and may reuse the initiator port transfer tag. The SCSI application client should call **Send SCSI Command ()** again with identical arguments.

After a **Received Task Management Function Executed ()** call with a Service Response of Service Delivery or Target Failure - ACK/NAK Timeout, a SCSI application client should call **Send Task Management Request ()** with identical arguments, including the same initiator port transfer tag.

After a **Command Complete Received ()** or **Received Task Management Function Executed ()** call returns a Service Response other than Service Delivery or Target Failure - ACK/NAK Timeout, a SCSI application client shall not reuse the initiator port transfer tag until it determines the initiator port transfer tag is no longer in use by the logical unit (e.g., the ACK for the RESPONSE frame was seen by the SSP target port). Examples of ways the SCSI application client may determine that an initiator port transfer tag may be reused are:

- a) receiving another frame in the same connection;
- b) receiving a DONE (NORMAL) or DONE (CREDIT TIMEOUT) in the same connection; or
- c) receiving a DONE (ACK/NAK TIMEOUT) in the same connection, then running a QUERY TASK task management function to confirm that the initiator port transfer tag is no longer active in the logical unit.

### 9.2.3 SCSI device server error handling

#### 9.2.3.1 SCSI Command Received () error handling

If a SCSI device server processes **SCSI Command Received ()** and the CDB argument does not contain all the bytes of the CDB (see SPC-4), then the SCSI device server shall terminate the command with CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN COMMAND INFORMATION UNIT.



### 9.2.3.2 Data-Out Received () error handling

If a SCSI device server processes **Data-Out Received ()** with a Delivery Result set to a value in table 256, then the SCSI device server shall terminate the command with CHECK CONDITION status with the sense key set to ABORTED COMMAND and the additional sense code set as indicated in table 256.

**Table 256 – Delivery Result to additional sense code mapping**

Delivery Result	Additional sense code
DELIVERY FAILURE - DATA OFFSET ERROR	DATA OFFSET ERROR
DELIVERY FAILURE - TOO MUCH WRITE DATA	TOO MUCH WRITE DATA
DELIVERY FAILURE - INFORMATION UNIT TOO SHORT	INFORMATION UNIT TOO SHORT
DELIVERY FAILURE - ACK/NAK TIMEOUT	ACK/NAK TIMEOUT
DELIVERY FAILURE - CONNECTION FAILED	Should be CONNECTION LOST May be ACK/NAK TIMEOUT
DELIVERY FAILURE - NAK RECEIVED	NAK RECEIVED
DELIVERY FAILURE - INITIATOR RESPONSE TIMEOUT	INITIATOR RESPONSE TIMEOUT

### 9.2.4 Task router and task manager error handling

If a SCSI target device performs initiator port transfer tag checking and a task router or task manager processes **SCSI Command Received ()** with an initiator port transfer tag already in use by another command (i.e., an overlapped command (see SAM-5)) in any logical unit, then the task router or task managers shall:

- a) abort all task management functions received on that I\_T nexus; and
- b) respond to the overlapped command as defined in SAM-5.

If a SCSI target device performs initiator port transfer tag checking and:

- a) a task router or task manager processes **SCSI Command Received ()** with an initiator port transfer tag already in use by a task management function in any logical unit; or
- b) a task router or task manager processes **Task Management Request Received ()** with an initiator port transfer tag already in use by a command or task management function in any logical unit,

then the task router or task manager shall:

- a) abort all commands received on that I\_T nexus;
- b) abort all task management functions received on that I\_T nexus; and
- c) call **Task Management Function Executed ()** with the Service Response set to SERVICE DELIVERY OR TARGET FAILURE - Overlapped Initiator Port Transfer Tag Attempted (i.e., requesting that the SSP target port set the DATAPRES field to RESPONSE\_DATA and the RESPONSE CODE field to OVERLAPPED INITIATOR PORT TRANSFER TAG ATTEMPTED).

### 9.2.5 SCSI transport protocol services for event notifications

The SCSI transport protocol services are used by:

- a) an SSP initiator port to deliver an indication of an event to a SCSI application client; and
- b) an SSP target port to deliver an indication of an event to a task manager and a SCSI device server.

Table 257 lists the SCSI transport protocol services for event notifications supported by this standard.

**Table 257 – SCSI transport protocol events**

Delivered transport protocol service indication	I/T <sup>a</sup>	SSP implementation	Reference
<b>Transport Reset (IN ( SCSI Port ))</b> <sup>b</sup>	I/T	Transport Reset	4.4.2
<b>Nexus Loss (IN ( I_T Nexus ))</b> <sup>c</sup>	I/T	Nexus Loss	4.4.3
<b>Power Loss Expected (IN ( SCSI Port ))</b> <sup>d</sup>	T	Power Loss Expected	6.2.5.3.3
<b>Break Occurred (IN (Nexus, [Command Identifier] ))</b> <sup>e f g</sup>	T	Break Occurred	8.2.6.3.2.3
<sup>a</sup> I/T indicates whether the SSP initiator port (I) or the SSP target port (T) implements the transport protocol service. <sup>b</sup> The specific SCSI port in the SCSI device for which a transport reset was detected. <sup>c</sup> The specific I_T nexus that has been detected as lost. <sup>d</sup> The specific SCSI port in the SCSI device for which power loss expected was detected. <sup>e</sup> The specific I_T nexus or I_T_L nexus and command associated with the break. <sup>f</sup> The Break Occurred SCSI transport protocol event is not specified by SAM-5. <sup>g</sup> After a Break Occurred confirmation has been received, <b>Data-In Delivered</b> SCSI transport protocol service confirmations and <b>Data-Out Received</b> SCSI transport protocol service confirmations shall not be sent for the command specified by the command identifier			

## 9.2.6 SCSI commands

### 9.2.6.1 INQUIRY command

SAS-specific vital product data accessed with the INQUIRY command (see SPC-4) is described in 9.2.11.

### 9.2.6.2 LOG SELECT and LOG SENSE commands

SAS-specific log pages accessed with the LOG SELECT command and LOG SENSE command (see SPC-4) are described in 9.2.8.

### 9.2.6.3 MODE SELECT and MODE SENSE commands

SAS-specific mode pages accessed with the MODE SELECT command and MODE SENSE command (see SPC-4) are described in 9.2.7.

### 9.2.6.4 SEND DIAGNOSTIC and RECEIVE DIAGNOSTIC RESULTS commands

SAS-specific diagnostic pages accessed with the SEND DIAGNOSTIC command and RECEIVE DIAGNOSTIC RESULTS command (see SPC-4) are described in 9.2.9.

Zoning (see 4.8) is applied to SES-3 diagnostic pages as described in 9.2.9.

### 9.2.6.5 START STOP UNIT command

The power condition states controlled by the START STOP UNIT command (see SBC-3) for a SAS device are described in 9.2.10.

## 9.2.7 SCSI mode parameters

### 9.2.7.1 SCSI mode parameters overview

Table 258 defines mode pages supported by logical units in SCSI target devices in SAS domains (i.e., with SSP target ports) that support the MODE SELECT command or MODE SENSE command.

**Table 258 – SSP target port mode pages**

Mode page code	Subpage code	Description	Reference
02h	00h	Disconnect-Reconnect mode page	9.2.7.2
18h	00h	Protocol Specific Logical Unit mode page	9.2.7.3
	01h to DFh	Reserved	
	E0h to FEh	Vendor specific	
	FFh	Return all subpages for this mode page code	SPC-4
19h	00h	Protocol Specific Port mode page	9.2.7.4
	01h	Phy Control And Discover mode page	9.2.7.5
	02h	Shared Port Control mode page	9.2.7.6
	03h	Enhanced Phy Control mode page	9.2.7.7
	04h to DFh	Reserved	
	E0h to FEh	Vendor specific	
	FFh	Return all subpages for this mode page code	SPC-4

If any field in an implemented mode page is not implemented, then the value of the field shall be assumed to be zero (i.e., as if the field is set to zero) (see SPC-4).

If a mode page defined by this standard is not implemented, then the value of each field in that mode page that is:

- a) allowed by this standard to be changeable (e.g., not defined as a read only field); and
- b) not used solely to define the mode page structure (e.g., the NUMBER OF PHYS field in the Phy Control And Discover mode page) or coordinate access to the mode page (e.g., the GENERATION CODE field in the Phy Control And Discover mode page),

shall be assumed to be zero (i.e., as if the mode page is implemented and the field is set to zero).

### 9.2.7.2 Disconnect-Reconnect mode page

#### 9.2.7.2.1 Disconnect-Reconnect mode page overview

The Disconnect-Reconnect mode page (see SPC-4) provides the SCSI application client the means to tune the performance of a service delivery subsystem. Table 259 defines the parameters that are applicable to SAS SSP.

The SCSI application client sends the values in the fields to be used by the SCSI device server to control the SSP connections by means of a MODE SELECT command. The SCSI device server shall then communicate the field values to the SSP target port. The field values are communicated from the SCSI device server to the SSP target port in a vendor specific manner.

SAS devices shall only use the parameter fields defined in table 259. If any other fields within the Disconnect-Reconnect mode page of the MODE SELECT command contain a non-zero value, then the SCSI device server shall terminate the MODE SELECT command with CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

**Table 259 – Disconnect-Reconnect mode page for SAS SSP**

Byte\Bit	7	6	5	4	3	2	1	0
0	PS	SPF (0b)	PAGE CODE (02h)					
1	PAGE LENGTH (0Eh)							
2	Reserved							
3	Reserved							
4	(MSB)	BUS INACTIVITY LIMIT						
5								(LSB)
6		Reserved						
7								
8	(MSB)	CONNECT TIME LIMIT						
9								(LSB)
10	(MSB)	MAXIMUM BURST SIZE						
11								(LSB)
12		Reserved						
13								
14	(MSB)	FIRST BURST SIZE						
15								(LSB)

The parameters saveable (PS) bit is defined in SPC-4.

The subpage format (SPF) bit is defined in SPC-4 and shall be set as shown in table 259 for the Disconnect-Reconnect mode page for SAS SSP.

The PAGE CODE field is defined in SPC-4 and shall be set as shown in table 259 for the Disconnect-Reconnect mode page for SAS SSP.

The PAGE LENGTH field is defined in SPC-4 and shall be set as shown in table 259 for the Disconnect-Reconnect mode page for SAS SSP.

The BUS INACTIVITY LIMIT field is defined in SPC-4 and 9.2.7.2.2.

The CONNECT TIME LIMIT field is defined in SPC-4 and 9.2.7.2.3.

The MAXIMUM BURST SIZE field is defined in SPC-4 and 9.2.7.2.4.

The FIRST BURST SIZE field is defined in SPC-4 and 9.2.7.2.5.

#### **9.2.7.2.2 BUS INACTIVITY LIMIT field**

The BUS INACTIVITY LIMIT field contains the maximum time, in 100  $\mu$ s increments, that an SSP target port is permitted to maintain a connection (see 4.1.12) without transferring a frame to the SSP initiator port. If this time is exceeded and a persistent connection has not been established (see 4.1.13), then the SSP target port shall prepare to close the connection (i.e., by requesting to have the link layer transmit DONE). This value may be rounded as defined in SPC-4. A value of 0000h in this field specifies that there is no bus inactivity time limit. The bus inactivity time limit is enforced by the port layer (see 7.2.3).

#### **9.2.7.2.3 CONNECT TIME LIMIT field**

The CONNECT TIME LIMIT field contains the maximum duration of a connection (see 4.1.12) in 100  $\mu$ s increments (e.g., a value of 0001h in this field means that the time is less than or equal to 100  $\mu$ s and a value of 0002h in this field means that the time is less than or equal to 200  $\mu$ s). If this time is exceeded and a persistent connection has not been established (see 4.1.13), then the SSP target port shall prepare to close the connection. If an SSP target port is transferring a frame when the maximum connection time limit is exceeded, then the SSP target port shall complete transfer of the frame before preparing to close the connection. This value may be rounded as defined in SPC-4. A value of 0000h in this field specifies that there is no maximum connection time limit. The maximum connection time limit is enforced by the port layer (see 7.2.3).

#### **9.2.7.2.4 MAXIMUM BURST SIZE field**

If a persistent connection has been established (see 4.1.13), then the MAXIMUM BURST SIZE field shall be ignored.

If a persistent connection has not been established (see 4.1.13), then:

- a) for read data, the MAXIMUM BURST SIZE field contains the maximum amount of data in 512-byte increments that is transferred during a connection by an SSP target port per I\_T\_L nexus and command identifier combination without transferring at least one frame for a different I\_T\_L nexus and command identifier combination.

If the SSP target port:

- A) has read data to transfer for only one I\_T\_L nexus and command identifier combination; and
- B) has no requests to transfer write data for any I\_T\_L nexus and command identifier combination,

then the SSP target port shall prepare to close the connection after the amount of data specified by the MAXIMUM BURST SIZE field is transferred to the SSP initiator port; and

- b) for write data, the MAXIMUM BURST SIZE field shall specify the maximum amount of data that an SSP target port requests via a single XFER\_RDY frame (see 8.2.2.3).

If a persistent connection has not been established (see 4.1.13), then:

- a) the MAXIMUM BURST SIZE field is specified in 512-byte increments (e.g., a value of 0001h in this field means that the number of bytes transferred to the SSP initiator port for the nexus is less than or equal to 512, and a value of 0002h in this field means that the number of bytes transferred to the SSP initiator port for the nexus is less than or equal to 1 024). A value of 0000h in this field specifies that there is no maximum burst size; and

- b) in terms of the SCSI transport protocol services (see 9.2.1), the SCSI device server shall limit the Request Byte Count argument to the Receive Data-Out transport protocol service and the Send Data-In transport protocol service to the amount specified in this field.

#### 9.2.7.2.5 FIRST BURST SIZE field

If the ENABLE FIRST BURST bit is set to zero in the COMMAND frame, then the FIRST BURST SIZE field is ignored.

If the ENABLE FIRST BURST bit is set to one in the COMMAND frame, then the FIRST BURST SIZE field contains the maximum amount of write data in 512-byte increments that may be sent by the SSP initiator port to the SSP target port without having to receive an XFER\_RDY frame (see 8.2.2.3) from the SSP target port (e.g., a value of 0001h in this field means that the number of bytes transferred by the SSP initiator port is less than or equal to 512 and a value of 0002h in this field means that the number of bytes transferred by the SSP initiator port is less than or equal to 1 024).

Specifying a non-zero value in the FIRST BURST SIZE field is equivalent to an implicit XFER\_RDY frame for each command requiring write data where the WRITE DATA LENGTH field of the XFER\_RDY frame is set to 512 times the value of the FIRST BURST SIZE field.

The rules for data transferred using the value in the FIRST BURST SIZE field are the same as those used for data transferred for an XFER\_RDY frame (i.e., the number of bytes transferred using the value in the FIRST BURST SIZE field is as if that number of bytes was requested by an XFER\_RDY frame).

If the amount of data to be transferred for the command is less than the amount of data specified by the FIRST BURST SIZE field, then the SSP target port shall not transmit an XFER\_RDY frame for the command. If the amount of data to be transferred for the command is greater than the amount of data specified by the FIRST BURST SIZE field, then the SSP target port shall transmit an XFER\_RDY frame after it has received all of the data specified by the FIRST BURST SIZE field from the SSP initiator port. All data for the command is not required to be transferred during the same connection in which the command is transferred.

A value of 0000h in this field specifies that there is no first burst size (i.e., an SSP initiator port transmits no write DATA frames to the SSP target port before receiving an XFER\_RDY frame).

The first burst size is handled by the SCSI transport protocol services (see 9.2.1) and the SSP transport layer (see 8.2.6).

#### 9.2.7.3 Protocol Specific Logical Unit mode page

The Protocol Specific Logical Unit mode page (see SPC-4) contains parameters that affect SSP target port operation on behalf of the logical unit.

The mode page policy (see SPC-4) for this mode page shall be either shared or per target port. If the SAS target device has multiple SSP target ports, then the mode page policy should be per target port.

Parameters in this mode page shall affect all phys in:

- a) the SSP target port if the mode page policy is per target port; or
- b) all SSP target ports in the SAS target device if the mode page policy is shared.

Table 260 defines the format of the page for SAS SSP.

**Table 260 – Protocol Specific Logical Unit mode page for SAS SSP**

Byte\Bit	7	6	5	4	3	2	1	0
0	PS	SPF (0b)	PAGE CODE (18h)					
1	PAGE LENGTH (06h)							
2	Reserved			TRANSPORT LAYER RETRIES	PROTOCOL IDENTIFIER (6h)			
3	Reserved							
...								
7								

The parameters saveable (PS) bit is defined in SPC-4.

The subpage format (SPF) bit is defined in SPC-4 and shall be set as shown in table 260 for the Protocol Specific Logical Unit mode page for SAS SSP.

The PAGE CODE field is defined in SPC-4 and shall be set as shown in table 260 for the Protocol Specific Logical Unit mode page for SAS SSP.

The PAGE LENGTH field is defined in SPC-4 and shall be set as shown in table 260 for the Protocol Specific Logical Unit mode page for SAS SSP.

A TRANSPORT LAYER RETRIES bit set to one specifies that, for commands received in COMMAND frames with the TLR CONTROL field set to 00b or 11b (see 8.2.1), the SSP target port shall support transport layer retries for XFER\_RDY and DATA frames for the logical unit as described in 8.2.4 (i.e., transport layer retries are enabled). A TRANSPORT LAYER RETRIES bit set to zero specifies that, for commands received in COMMAND frames with the TLR CONTROL field set to 00b or 11b (see 8.2.1), transport layer retries shall not be used (i.e., transport layer retries are disabled).

The PROTOCOL IDENTIFIER field is defined in SPC-4 and shall be set as shown in table 260 for the Protocol Specific Logical Unit mode page for SAS SSP indicating that this is a SAS SSP specific mode page.

#### 9.2.7.4 Protocol Specific Port mode page

The Protocol Specific Port mode page (see SPC-4) contains parameters that affect SSP target port operation. If the mode page is implemented by one logical unit in a SCSI target device, then it shall be implemented by all logical units in the SCSI target device that support the MODE SELECT or MODE SENSE commands.

The mode page policy (see SPC-4) for this mode page shall be either shared or per target port. If a SAS target device has multiple SSP target ports, then the mode page policy should be per target port.

Parameters in this mode page shall affect all phys in:

- the SSP target port if the mode page policy is per target port; or
- all SSP target ports in the SAS target device if the mode page policy is shared.

Table 261 defines the format of the page for SAS SSP.

**Table 261 – Protocol Specific Port mode page for SAS SSP**

Byte\Bit	7	6	5	4	3	2	1	0
0	PS	SPF (0b)	PAGE CODE (19h)					
1	PAGE LENGTH (0Eh)							
2	Reserved	CONTINUE AWT	BROADCAST ASYNCHRONOUS EVENT	READY LED MEANING	PROTOCOL IDENTIFIER (6h)			
3	Reserved							
4	(MSB) I_T NEXUS LOSS TIME (LSB)							
5								
6	(MSB) INITIATOR RESPONSE TIMEOUT (LSB)							
7								
8	(MSB) REJECT TO OPEN LIMIT (LSB)							
9								
10	MAXIMUM ALLOWED XFER_RDY							
11	Reserved							
...								
15								

The parameters saveable (PS) bit is defined in SPC-4.

The subpage format (SPF) bit is defined in SPC-4 and shall be set as shown in table 261 for the Protocol Specific Port mode page for SAS SSP.

The PAGE CODE field is defined in SPC-4 and shall be set as shown in table 261 for the Protocol Specific Port mode page for SAS SSP.

The PAGE LENGTH field is defined in SPC-4 and shall be set as shown in table 261 for the Protocol Specific Port mode page for SAS SSP.

A CONTINUE AWT bit set to one specifies that the SAS port shall not stop the Arbitration Wait Time timer and shall not set the Arbitration Wait Time timer to zero when the SAS port receives an OPEN\_REJECT (RETRY). A CONTINUE AWT bit set to zero specifies that the SAS port shall stop the Arbitration Wait Time timer and set the Arbitration Wait Time timer to zero when the SAS port receives an OPEN\_REJECT (RETRY).

A BROADCAST ASYNCHRONOUS EVENT bit set to one specifies that the SCSI device server shall enable origination of Broadcast (Asynchronous Event) (see 4.1.15). A BROADCAST ASYNCHRONOUS EVENT bit set to zero specifies that the SCSI device server shall disable origination of Broadcast (Asynchronous Event).



The READY LED MEANING bit specifies the READY LED signal behavior (see 9.4.1). Regardless of the mode page policy (see SPC-4) for this mode page, the shared mode page policy shall be applied to the READY LED MEANING bit.

The PROTOCOL IDENTIFIER field is defined in SPC-4 and shall be set as shown in table 261 for the Protocol Specific Port mode page for SAS SSP indicating that this is a SAS SSP specific mode page.

The I\_T NEXUS LOSS TIME field contains the minimum time that the SSP target port shall retry connection requests to an SSP initiator port that are rejected with certain responses indicating that the SSP initiator port may no longer be present (see 7.2.2) before recognizing an I\_T nexus loss (see 4.4.3).

An SSP initiator port should retry connection requests for at least the time indicated by the I\_T NEXUS LOSS TIME field in the Protocol Specific Port mode page for the SSP target port to which it is trying to establish a connection (see 4.4.3).

Table 262 defines the values of the I\_T NEXUS LOSS TIME field. This value is enforced by the port layer (see 7.2.2).

**Table 262 – I\_T NEXUS LOSS TIME field**

Code <sup>a</sup>	Description
0000h	Vendor specific amount of time.
0001h to FFFh	Time in one millisecond increments.
FFFFh	The SSP target port shall never recognize an I_T nexus loss (i.e., it shall retry the connection requests forever).
<sup>a</sup> If this mode page is implemented, then the default value of the I_T NEXUS LOSS TIME field should be non-zero. It is recommended that this value be 07D0h (i.e., 2 000 ms).	

The INITIATOR RESPONSE TIMEOUT field contains the minimum time in one millisecond increments that the SSP target port shall wait for the receipt of a frame (e.g., a write DATA frame) before aborting the command associated with that frame. An INITIATOR RESPONSE TIMEOUT field set to 0000h indicates that the SSP target port shall disable the initiator response timeout timer. This value is enforced by the transport layer (see 8.2.6.3).

The REJECT TO OPEN LIMIT field contains the minimum time, in 10  $\mu$ s increments, that the SSP target port shall wait to establish a connection request with an initiator port on an I\_T nexus after receiving an OPEN\_REJECT (RETRY), OPEN\_REJECT (RESERVED CONTINUE 0), or OPEN\_REJECT (RESERVED CONTINUE 1). This value may be rounded as defined in SPC-4. A REJECT TO OPEN LIMIT field set to 0000h indicates that the minimum time is vendor specific. This minimum time is enforced by the port layer (see 7.2.3).

The MAXIMUM ALLOWED XFER\_RDY field specifies the maximum number of times a device server may call the Receive Data-Out transport protocol service as described in 9.2.1.8. A MAXIMUM ALLOWED XFER\_RDY field set to zero specifies that there is no limit to the number of times a device server may call the Receive Data-Out transport protocol service.

#### 9.2.7.5 Phy Control And Discover mode page

The Phy Control And Discover mode page contains parameters that affect SSP target phy operation. If the mode page is implemented by one logical unit in a SCSI target device, then it shall be implemented by all logical units in the SCSI target device that support the MODE SELECT command or MODE SENSE command.

The mode page policy (see SPC-4) for this mode page shall be shared for all SSP target ports. Parameters in this mode page shall affect only the referenced phy.

Table 263 defines the format of this mode page.

**Table 263 – Phy Control And Discover mode page**

Byte\Bit	7	6	5	4	3	2	1	0
0	PS	SPF (1b)	PAGE CODE (19h)					
1	SUBPAGE CODE (01h)							
2	(MSB) PAGE LENGTH (n - 3) (LSB)							
3								
4	Reserved							
5	Reserved				PROTOCOL IDENTIFIER (6h)			
6	GENERATION CODE							
7	NUMBER OF PHYS							
SAS phy mode descriptor list								
8	SAS phy mode descriptor (first) (see table 264)							
...								
55								
...	...							
n - 47	SAS phy mode descriptor (last) (see table 264)							
...								
n								

The parameters saveable (PS) bit is defined in SPC-4.

The subpage format (SPF) bit is defined in SPC-4 and shall be set as shown in table 263 for the Phy Control And Discover mode page.

The PAGE CODE field is defined in SPC-4 and shall be set as shown in table 263 for the Phy Control And Discover mode page.

The SUBPAGE CODE field is defined in SPC-4 and shall be set as shown in table 263 for the Phy Control And Discover mode page.

The PAGE LENGTH field is defined in SPC-4 and shall be set as shown in table 263 for the Phy Control And Discover mode page (i.e.,  $4 + ((\text{the value of the NUMBER OF PHYS field}) \times (\text{the length in bytes of the SAS phy mode descriptor}))$ ).

The PROTOCOL IDENTIFIER field is defined in SPC-4 and shall be set as shown in table 263 for the Phy Control And Discover mode page indicating that this is a SAS SSP specific mode page.

The GENERATION CODE field is a one-byte counter that shall be incremented by one by the SCSI device server every time the values in this mode page or the Enhanced Phy Control mode page (see 9.2.7.7) are changed. A GENERATION CODE field set to 00h indicates the generation code is unknown. The SCSI device server shall wrap this field to 01h as the next increment after the generation code reaches its maximum value (i.e., FFh). The GENERATION CODE field is also contained in the Enhanced Phy Control mode page (see 9.2.7.7) and the Protocol Specific Port log page (see 9.2.8.1) and may be used to correlate phy settings across mode page and log page accesses.

NOTE 65 - SCSI device servers compliant with SAS-1.1 set the GENERATION CODE field to 00h.

The NUMBER OF PHYS field contains the number of phys in the SAS target device and indicates the number of SAS phy mode descriptors in the SAS phy mode descriptor list. This field shall not be changeable with the MODE SELECT command.

The SAS phy mode descriptor list contains a SAS phy mode descriptor for each phy in the SAS target device, not just the SAS target port, starting with the lowest numbered phy and ending with the highest numbered phy as determined by the value in the PHY IDENTIFIER field in the SAS phy mode descriptor.

Table 264 defines the SAS phy mode descriptor.

**Table 264 – SAS phy mode descriptor**

Byte\Bit	7	6	5	4	3	2	1	0
0	Reserved							
1	PHY IDENTIFIER							
2	Reserved							
3								
4	Reserved	ATTACHED SAS DEVICE TYPE			ATTACHED REASON			
5	REASON				NEGOTIATED LOGICAL LINK RATE			
6	Reserved				ATTACHED SSP INITIATOR PORT	ATTACHED STP INITIATOR PORT	ATTACHED SMP INITIATOR PORT	Reserved
7	Reserved				ATTACHED SSP TARGET PORT	ATTACHED STP TARGET PORT	ATTACHED SMP TARGET PORT	Reserved
8	SAS ADDRESS							
...								
15								
16	ATTACHED SAS ADDRESS							
...								
23								
24	ATTACHED PHY IDENTIFIER							
25	ATTACHED PERSISTENT CAPABLE	ATTACHED POWER CAPABLE		ATTACHED SLUMBER CAPABLE	ATTACHED PARTIAL CAPABLE	ATTACHED INSIDE ZPSDS PERSISTENT	ATTACHED REQUESTED INSIDE ZPSDS	ATTACHED BREAK_REPLY CAPABLE
26	Reserved					ATTACHED APTA CAPABLE	ATTACHED SMP PRIORITY CAPABLE	ATTACHED PWR_DIS CAPABLE
27	Reserved for IDENTIFY frame related fields							
...								
31								
32	PROGRAMMED MINIMUM PHYSICAL LINK RATE				HARDWARE MINIMUM PHYSICAL LINK RATE			
33	PROGRAMMED MAXIMUM PHYSICAL LINK RATE				HARDWARE MAXIMUM PHYSICAL LINK RATE			
34	Reserved							
...								
41								
42	Vendor specific							
...								
43								
44	Reserved							
...								
47								

The PROGRAMMED MINIMUM PHYSICAL LINK RATE field and PROGRAMMED MAXIMUM PHYSICAL LINK RATE field are defined in the SMP PHY CONTROL function (see 9.4.3.28) for accesses with MODE SELECT commands and in the SMP DISCOVER function (see 9.4.3.10) for accesses with MODE SENSE commands.

The fields in the SAS phy mode descriptor not defined in this subclause are defined in the SMP DISCOVER response (see 9.4.3.10). These fields shall not be changeable with the MODE SELECT command.

### 9.2.7.6 Shared Port Control mode page

The Shared Port Control mode page contains parameters that affect SSP target port operation. If the mode page is implemented by one logical unit in a SCSI target device, then it shall be implemented by all logical units in the SCSI target device that support the MODE SELECT command or MODE SENSE command.

The mode page policy (see SPC-4) for this mode page shall be shared for all SSP target ports.

Table 265 defines the format of this mode page.

**Table 265 – Shared Port Control mode page**

Byte\Bit	7	6	5	4	3	2	1	0
0	PS	SPF (1b)	PAGE CODE (19h)					
1	SUBPAGE CODE (02h)							
2	(MSB) PAGE LENGTH (000Ch) (LSB)							
3								
4	Reserved							
5	Reserved				PROTOCOL IDENTIFIER (6h)			
6	(MSB) POWER LOSS TIMEOUT (LSB)							
7								
8	Reserved							
9	POWER GRANT TIMEOUT							
10	Reserved							
...								
15								

The parameters saveable (PS) bit is defined in SPC-4.

The subpage format (SPF) bit is defined in SPC-4 and shall be set as shown in table 265 for the Shared Port Control mode page.

The PAGE CODE field is defined in SPC-4 and shall be set as shown in table 265 for the Shared Port Control mode page.

The SUBPAGE CODE field is defined in SPC-4 and shall be set as shown in table 265 for the Shared Port Control mode page.

The PAGE LENGTH field is defined in SPC-4 and shall be set as shown in table 265 for the Shared Port Control mode page.

The PROTOCOL IDENTIFIER field is defined in SPC-4 and shall be set as shown in table 265 for the Shared Port Control mode page indicating that this is a SAS SSP specific mode page.

The POWER LOSS TIMEOUT field contains the maximum time, in one millisecond increments, that a target port shall respond to connection requests with OPEN\_REJECT (RETRY) after receiving NOTIFY (POWER LOSS EXPECTED) (see 6.2.5.3.3). A POWER LOSS TIMEOUT field set to 0000h specifies that the maximum time is vendor specific. The power loss timeout shall be restarted on each NOTIFY (POWER LOSS EXPECTED) that is received.

The POWER GRANT TIMEOUT field contains the minimum time, in one second increments, that a SAS target device shall wait to receive a PWR\_GRANT (see 6.14.5.4) from a power source device (see 6.14.1). A POWER GRANT TIMEOUT field set to 00h specifies that the time limit is vendor specific.

#### **9.2.7.7 Enhanced Phy Control mode page**

The Enhanced Phy Control mode page contains parameters that affect SSP target phy operation. If the mode page is implemented by one logical unit in a SCSI target device, then it shall be implemented by all logical units in the SCSI target device that support the MODE SELECT command or MODE SENSE command.

The mode page policy (see SPC-4) for this mode page shall be shared for all SSP target ports.

Table 266 defines the format of this mode page.

**Table 266 – Enhanced Phy Control mode page**

Byte\Bit	7	6	5	4	3	2	1	0
0	PS	SPF (1b)	PAGE CODE (19h)					
1	SUBPAGE CODE (03h)							
2	(MSB) PAGE LENGTH (n - 3) (LSB)							
3								
4	Reserved							
5	Reserved				PROTOCOL IDENTIFIER (6h)			
6	GENERATION CODE							
7	NUMBER OF PHYS							
Enhanced phy control mode descriptor list								
8	Enhanced phy control mode descriptor (first) (see table 267)							
...								
27								
...	...							
n - 19	Enhanced phy control mode descriptor (last) (see table 267)							
...								
n								

The parameters saveable (PS) bit is defined in SPC-4.

The subpage format (SPF) bit is defined in SPC-4 and shall be set as shown in table 266 for the Enhanced Phy Control mode page.

The PAGE CODE field is defined in SPC-4 and shall be set as shown in table 266 for the Enhanced Phy Control mode page.

The SUBPAGE CODE field is defined in SPC-4 and shall be set as shown in table 266 for the Enhanced Phy Control mode page.

The PAGE LENGTH field is defined in SPC-4 and shall be set as shown in table 266 for the Enhanced Phy Control mode page (i.e.,  $4 + ((\text{the value of the NUMBER OF PHYS field}) \times (\text{the length in bytes of the enhanced phy control mode descriptor}))$ ).

The PROTOCOL IDENTIFIER field is defined in SPC-4 and shall be set as shown in table 266 for the Enhanced Phy Control mode page indicating that this is a SAS SSP specific mode page.

The GENERATION CODE field is defined in the Phy Control and Discover mode page (see 9.2.7.5).

The NUMBER OF PHYS field contains the number of phys in the SAS target device and indicates the number of enhanced phy control mode descriptors in the enhanced phy control mode descriptor list. This field shall not be changeable with the MODE SELECT command.

The enhanced phy control mode descriptor list contains an enhanced phy control mode descriptor for each phy in the SAS target device, not just the SAS target port, starting with the lowest numbered phy and ending with the highest numbered phy as determined by the value in the PHY IDENTIFIER field in the enhanced phy control mode descriptor.

Table 267 defines the enhanced phy control mode descriptor.

**Table 267 – Enhanced phy control mode descriptor**

Byte\Bit	7	6	5	4	3	2	1	0
0	Reserved							
1	PHY IDENTIFIER							
2	(MSB) _____ DESCRIPTOR LENGTH (0010h) _____ (LSB)							
3								
4	PROGRAMMED PHY CAPABILITIES _____							
...								
7								
8	CURRENT PHY CAPABILITIES _____							
...								
11								
12	ATTACHED PHY CAPABILITIES _____							
...								
15								
16	Reserved							
17								
18	Reserved		OPTICAL MODE ENABLED	NEGOTIATED SSC	NEGOTIATED PHYSICAL LINK RATE			
19	Reserved					ENABLE SLUMBER	ENABLE PARTIAL	HARDWARE MUXING SUPPORTED

The DESCRIPTOR LENGTH field contains the length in bytes that follow in the descriptor and shall be set as shown in table 267 for the enhanced phy control mode descriptor.



An ENABLE SLUMBER bit set to one specifies that the SCSI device server shall enable the management application layer to control the slumber phy power condition (see 4.10.1.4) on the phy specified by the PHY IDENTIFIER field. An ENABLE SLUMBER bit set to zero specifies that the SCSI device server shall disable control of the slumber phy power condition by the management application layer on the phy specified by the PHY IDENTIFIER field.

An ENABLE PARTIAL bit set to one specifies that the SCSI device server shall enable the management application layer to control the partial phy power condition (see 4.10.1.3) on the phy specified by the PHY IDENTIFIER field. An ENABLE PARTIAL bit set to zero specifies that the SCSI device server shall disable control of the partial phy power condition by the management application layer on the phy specified by the PHY IDENTIFIER field.

The fields in the enhanced phy control mode descriptor not defined in this subclause are defined in the SMP DISCOVER response (see 9.4.3.10). These fields shall not be changeable with the MODE SELECT command.

## 9.2.8 SCSI log parameters

### 9.2.8.1 Protocol Specific Port log page

The Protocol Specific Port log page for SAS SSP defined in table 269 provides the SCSI application client a means to determine information about phy events concerning the SAS target device's phys. The parameter codes for the Protocol Specific Port log page are listed in table 268.

**Table 268 – Protocol Specific Port log parameters**

Parameter code	Description	Resettable or Changeable <sup>a</sup>	Reference	Support Required
0000h	Reserved			
0001h to FFFFh	Protocol Specific Port log parameter for SAS target ports	Never	9.2.8.2	Optional
<sup>a</sup> The keywords in this column – Always, Reset Only, and Never – are defined in SPC-45.				

The Protocol Specific Port log page for SAS SSP has the format shown in table 269.

**Table 269 – Protocol Specific Port log page for SAS SSP**

Byte\Bit	7	6	5	4	3	2	1	0
0	DS	SPF (0b)	PAGE CODE (18h)					
1	SUBPAGE CODE (00h)							
2	(MSB)PAGE LENGTH (n - 3)							
3	(LSB)							
Protocol Specific Port log parameter list								
4	Protocol Specific Port log parameter (first) (see table 270)							
...								
...	...							
	Protocol Specific Port log parameter (last) (see table 270)							
...								
n								

The disable save (DS) bit, subpage format (SPF) bit, PAGE CODE field, SUBPAGE CODE field, and PAGE LENGTH field are described in SPC-4. The SPF bit, PAGE CODE field, and SUBPAGE CODE field shall be set as shown in table 269 for the Protocol Specific Port log page for SAS SSP.

The Protocol Specific Port log parameter list contains a Protocol Specific Port log parameter for each SCSI port in the SAS target device.

### 9.2.8.2 Protocol Specific Port log parameter for SAS target ports

Table 270 defines the format for the Protocol Specific Port log parameter for SAS target ports. The SAS log parameter is a list parameter (i.e., not a data counter) and only has cumulative (i.e., not threshold) values (see SPC-4).

**Table 270 – Protocol Specific Port log parameter for SAS target ports**

Byte\Bit	7	6	5	4	3	2	1	0
0	(MSB) _____							
1	PARAMETER CODE (relative target port identifier)							(LSB)
2	Parameter control byte - binary format list log parameter (see SPC-5)							
	DU	Obsolete	TSD	Obsolete		FORMAT AND LINKING		
3	PARAMETER LENGTH (y - 3)							
4	Reserved				PROTOCOL IDENTIFIER (6h)			
5	Reserved							
6	GENERATION CODE							
7	NUMBER OF PHYS							
SAS phy log descriptor list								
8	SAS phy log descriptor (first) (see table 271)							
...								
8 + m	...							
...								
y - m	SAS phy log descriptor (last) (see table 271)							
...								
y								

The PARAMETER CODE field is defined in SPC-5 and contains the relative target port identifier (see SPC-5) of the SSP target port that the log parameter describes.

The DU bit, TSD bit, and FORMAT AND LINKING field are described in SPC-5. These fields shall be set as described for a binary format list log parameter (see SPC-5) for the Protocol Specific Port log parameter for SAS target ports.

The PARAMETER LENGTH field is defined in SPC-5.

The PROTOCOL IDENTIFIER field is defined in SPC-5 and shall be set as shown in table 270 for the Protocol Specific Port log page parameter for SAS target ports.

The GENERATION CODE field is defined in the Phy Control and Discover mode page (see 9.2.7.5).

The NUMBER OF PHYS field contains the number of phys in the SAS target port (not in the entire SAS target device) and indicates the number of SAS phy log descriptors in the SAS phy log descriptor list.

The SAS phy log descriptor list contains SAS phy log descriptors.

Table 271 defines the SAS phy log descriptor.

**Table 271 – SAS phy log descriptor (part 1 of 2)**

Byte\Bit	7	6	5	4	3	2	1	0
0	Reserved							
1	PHY IDENTIFIER							
2	Reserved							
3	SAS PHY LOG DESCRIPTOR LENGTH (m - 3)							
4	Reserved	ATTACHED SAS DEVICE TYPE			ATTACHED REASON			
5	REASON				NEGOTIATED LOGICAL LINK RATE			
6	Reserved				ATTACHED SSP INITIATOR PORT	ATTACHED STP INITIATOR PORT	ATTACHED SMP INITIATOR PORT	Reserved
7	Reserved				ATTACHED SSP TARGET PORT	ATTACHED STP TARGET PORT	ATTACHED SMP TARGET PORT	Reserved
8	SAS ADDRESS							
...								
15								
16	ATTACHED SAS ADDRESS							
...								
23								
24	ATTACHED PHY IDENTIFIER							
25	Reserved							
...								
31								

Table 271 – SAS phy log descriptor (part 2 of 2)

Byte\Bit	7	6	5	4	3	2	1	0
32	(MSB)							
...	INVALID DWORD COUNT							
35	(LSB)							
36	(MSB)							
...	RUNNING DISPARITY ERROR COUNT							
39	(LSB)							
40	(MSB)							
...	LOSS OF DWORD SYNCHRONIZATION COUNT							
43	(LSB)							
44	(MSB)							
...	PHY RESET PROBLEM COUNT							
47	(LSB)							
48	Reserved							
49								
50	PHY EVENT DESCRIPTOR LENGTH							
51	NUMBER OF PHY EVENT DESCRIPTORS							
Phy event descriptor list								
52								
...	Phy event descriptor (first) (see table 343 in 9.4.3.14.4)							
63								
...	...							
m - 11								
...	Phy event descriptor (last) (see table 343 in 9.4.3.14.4)							
m								

The SAS PHY LOG DESCRIPTOR LENGTH field indicates the number of bytes that follow in the SAS phy log descriptor. A SAS PHY LOG DESCRIPTOR LENGTH field set to 00h indicates that there are 44 additional bytes.

NOTE 66 - Logical units compliant with SAS and SAS-1.1 only support a 48 byte SAS phy log descriptor.

The INVALID DWORD COUNT field, RUNNING DISPARITY ERROR COUNT field, LOSS OF DWORD SYNCHRONIZATION COUNT field, and PHY RESET PROBLEM COUNT field are defined in the SMP REPORT PHY ERROR LOG response (see 9.4.3.11).

For the INVALID DWORD COUNT field, RUNNING DISPARITY ERROR COUNT field, LOSS OF DWORD SYNCHRONIZATION COUNT field, and PHY RESET PROBLEM COUNT field, the phy should support a 32-bit counter, however the phy may support a counter size less than 32-bits. If it reaches its maximum value, then the counter shall stop and the SCSI device server shall set the field to FFFFFFFFh in the SAS phy log descriptor.

The PHY EVENT DESCRIPTOR LENGTH field indicates the number of bytes in the phy event descriptor (see 9.4.3.14.4).

The NUMBER OF PHY EVENT DESCRIPTORS field indicates the number of phy event descriptors in the phy event descriptor list.

Each phy event descriptor uses the format defined for the SMP REPORT PHY EVENT function in table 343 (see 9.4.3.14.4).

The fields in the SAS phy log descriptor not defined in this subclause are defined in the SMP DISCOVER response (see 9.4.3.10). These fields shall not be changeable with the LOG SELECT command.

## 9.2.9 SCSI diagnostic parameters

### 9.2.9.1 SCSI diagnostic parameters overview

Table 272 defines diagnostic pages supported by logical units in SCSI target devices in SAS domains (i.e., with SSP target ports) that support the SEND DIAGNOSTIC or RECEIVE DIAGNOSTIC RESULTS commands.

**Table 272 – SSP target port diagnostic pages**

Diagnostic page code	Description	Reference
3Fh	Protocol Specific diagnostic page	9.2.9.2

An enclosure services process (see SES-3) describing elements in a SAS domain that are attached to a zoning expander device with zoning enabled (see 4.8) shall apply the zone permission table when providing access to those elements. Element types that may be subject to zoning include:

- a) Device Slot element;
- b) Array Device Slot element;
- c) Enclosure Services Controller Electronics element;
- d) SCC Controller Electronics element;
- e) SCSI Port/Transceiver element;
- f) SCSI Target Port element;
- g) SCSI Initiator Port element;
- h) SAS Expander element; and
- i) SAS Connector element.

Table 273 defines SCSI enclosure services diagnostic pages supported by logical units in SCSI target devices in SAS domains (e.g., with SSP target ports) that are affected by zoning.

**Table 273 – Diagnostic pages affected by zoning**

Diagnostic page code	Description	Reference
02h	Enclosure Control diagnostic page	SES-3 and 9.2.9.3
	Enclosure Status diagnostic page	SES-3 and 9.2.9.4
0Ah	Additional Element Status diagnostic page	SES-3 and 9.2.9.5

#### 9.2.9.2 Protocol Specific diagnostic page

The Protocol Specific diagnostic page provides a method for a SCSI application client to enable and disable phy test functions (see 4.11) for selected phys. The diagnostic page format is specified in SPC-4.

The Protocol Specific diagnostic page is sent by a SCSI application client using the SEND DIAGNOSTIC command. If the SCSI device server receives a RECEIVE DIAGNOSTIC RESULTS command with the PAGE CODE field set to 3Fh, then the SCSI device server shall terminate the command with CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN CDB.

Table 274 defines the Protocol Specific diagnostic page for SAS SSP.

**Table 274 – Protocol Specific diagnostic page for SAS SSP**

Byte\Bit	7	6	5	4	3	2	1	0
0	PAGE CODE (3Fh)							
1	Reserved				PROTOCOL IDENTIFIER (6h)			
2	(MSB) PAGE LENGTH (001Ch)							
3	(LSB)							
4	PHY IDENTIFIER							
5	PHY TEST FUNCTION							
6	PHY TEST PATTERN							
7	Reserved	PHY TEST FUNCTION SATA	PHY TEST FUNCTION SSC		PHY TEST FUNCTION PHYSICAL LINK RATE			
8	Reserved							
...								
10								
11	PHY TEST PATTERN DWORDS CONTROL							
12	PHY TEST PATTERN DWORDS							
...								
19								
20	Reserved							
...								
31								

The PAGE CODE field is defined in SPC-4 and shall be set as shown in table 274 for the Protocol Specific diagnostic page for SAS SSP.

The PROTOCOL IDENTIFIER field is defined in SPC-4 and shall be set as shown in table 274 for the Protocol Specific diagnostic page for SAS SSP indicating this is a SAS SSP specific diagnostic page.

The PAGE LENGTH field is defined in SPC-4 and shall be set as shown in table 274 for the Protocol Specific diagnostic page for SAS SSP.



The PHY IDENTIFIER field specifies the phy identifier (see 4.2.10) of the phy that is to perform or to stop performing a phy test function (i.e., the selected phy). If the PHY IDENTIFIER field specifies a phy that does not exist, then the SCSI device server shall terminate the SEND DIAGNOSTIC command with CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

The PHY TEST FUNCTION field specifies the phy test function to be performed and is defined in table 275. If the PHY TEST FUNCTION field specifies a phy test function that is not supported, then the SCSI device server shall terminate the SEND DIAGNOSTIC command with CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

**Table 275 – PHY TEST FUNCTION field**

Code	Name	Description
00h	STOP	<p>If the selected phy is performing a phy test function, then the selected phy shall stop performing the phy test function and originate a link reset sequence.</p> <p>If the selected phy is not performing a phy test function, then this function has no effect on the selected phy. <sup>a</sup></p>
01h	TRANSMIT PATTERN	<p>If the selected phy is not performing a phy test function, then the selected phy shall perform the transmit pattern phy test function (see 4.11.2) using the phy test pattern specified by the PHY TEST PATTERN field and the physical link rate specified by the PHY TEST FUNCTION PHYSICAL LINK RATE field.</p> <p>If the selected phy is performing a phy test function, then the SCSI device server shall terminate the SEND DIAGNOSTIC command with CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to PHY TEST FUNCTION IN PROGRESS. <sup>a</sup></p>
02h to EFh	Reserved	
F0h to FFh	Vendor specific	
<sup>a</sup> If there is no SSP target port available to receive a SEND DIAGNOSTIC command to stop a phy from performing a phy test function, then a power on may be required to cause the phy to stop performing the function and originate a phy reset sequence.		

If the PHY TEST FUNCTION field is set to 01h (i.e., TRANSMIT\_PATTERN), then the PHY TEST PATTERN field specifies the phy test pattern to be transmitted as defined by table 276. If the PHY TEST PATTERN field specifies a phy test pattern that is not supported by the specified SAS phy, then the SCSI device server shall terminate the SEND DIAGNOSTIC command with CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

**Table 276 – PHY TEST PATTERN field**

Code	Name	Description
00h	Reserved	
01h	JTPAT	The selected phy shall repeatedly transmit JTPAT for RD+ and RD- (see A.1).
02h	CJTPAT	The selected phy shall repeatedly transmit CJTPAT (see A.2).
03h	PRBS9	The selected phy shall repeatedly transmit PRBS9 (see SAS-4)
04h	PRBS15	The selected phy shall repeatedly transmit PRBS15 (see SAS-4)
05h to 0Fh	Reserved	
10h	TRAIN	The selected phy shall repeatedly transmit the TRAIN pattern (see 5.11.4.2.3.5).
11h	TRAIN_DONE	The selected phy shall repeatedly transmit the TRAIN_DONE pattern (see 5.11.4.2.3.5).
12h	IDLE	The selected phy shall repeatedly transmit idle dwords (see 6.6).
13h	SCRAMBLED_0	The selected phy shall repeatedly transmit a repeating pattern of at least 58 dwords (i.e., 2 320 bits on the physical link) set to 00000000h that are transmitted scrambled and 8b10b encoded (see 6.6). The scrambler shall be reinitialized at the beginning of each pattern. See table F.2 in F.1.4.
14h to 3Fh	Reserved	
40h	TWO_DWORDS	<p>The selected phy shall repeatedly transmit the dwords specified by the PHY TEST PATTERN DWORDS CONTROL field and the PHY TEST PATTERN DWORDS field without scrambling.</p> <p>This pattern is only for use for characterization of the transmitter device and the passive interconnect. Phys are not required to support all patterns that may be specified.</p>
41h to EFh	Reserved	
F0h to FFh	Vendor specific	

A PHY TEST FUNCTION SATA bit set to one specifies that the phy shall transmit as a SATA phy during the phy test function. A PHY TEST FUNCTION SATA bit set to zero specifies that the phy shall transmit as a SAS phy during the phy test function. If the PHY TEST FUNCTION SATA bit is set to one and the phy does not support SATA, then the SCSI device server shall terminate the SEND DIAGNOSTIC command with CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

The PHY TEST FUNCTION SSC field specifies the SSC modulation type (see SAS-4) that the phy shall use during transmission during the phy test function and is defined in table 277. If the SSC modulation type specified by the PHY TEST FUNCTION SSC field is not supported (e.g., if the phy is a SAS phy that does not support center-spreading SSC, then it only supports no-spreading and down-spreading SSC), then the SCSI device server shall terminate the SEND DIAGNOSTIC command with CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

**Table 277 – PHY TEST FUNCTION SSC field**

Code	Description
00b	No-spreading
01b	Center-spreading SSC <sup>a</sup>
10b	Down-spreading SSC
11b	Reserved
<sup>a</sup> If the PHY TEST FUNCTION SATA bit is set to one (i.e., a SATA phy is requested to transmit center-spreading), then the SCSI device server shall terminate the SEND DIAGNOSTIC command with CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN PARAMETER LIST.	

The PHY TEST FUNCTION PHYSICAL LINK RATE field specifies the physical link rate at which the phy test function shall be performed and is defined in table 278. If the physical link rate specified by the PHY TEST FUNCTION PHYSICAL LINK RATE field is less than the hardware minimum physical link rate or greater than the hardware maximum physical link rate, then the SCSI device server shall terminate the SEND DIAGNOSTIC command with CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

**Table 278 – PHY TEST FUNCTION PHYSICAL LINK RATE field**

Code	Description
0h to 7h	Reserved
8h	1.5 Gbit/s
9h	3 Gbit/s
Ah	6 Gbit/s
Bh	12 Gbit/s
Ch	22.5 Gbit/s
Dh to Fh	Reserved

The PHY TEST PATTERN DWORDS CONTROL field and PHY TEST PATTERN DWORDS field are only used if the PHY TEST PATTERN field is set to 40h (i.e., TWO\_DWORDS) (see table 276).

The PHY TEST PATTERN DWORDS CONTROL field defined in table 279 controls whether the bytes in the PHY TEST PATTERN DWORDS field are sent as control characters or data characters.

**Table 279 – PHY TEST PATTERN DWORDS CONTROL field**

Code	Description
00h	Each byte in the PHY TEST PATTERN DWORDS field shall be sent as a data character (i.e., Dxx.y) (see 5.3.6) without scrambling.
08h	The fifth byte in the PHY TEST PATTERN DWORDS field shall be sent as a control character (i.e., Kxx.y) (see 5.3.7). Each other byte shall be sent as a data character without scrambling.
80h	The first byte in the PHY TEST PATTERN DWORDS field shall be sent as a control character. Each other byte shall be sent as a data character without scrambling.
88h	The first and fifth bytes in the PHY TEST PATTERN DWORDS field shall each be sent as a control character. Each other byte shall be sent as a data character without scrambling.
All others	Reserved

The PHY TEST PATTERN DWORDS field contains the two dwords that are sent during a TWO\_DWORDS test pattern. Whether each byte in the dwords is sent as a control character or a data character is specified by the PHY TEST PATTERN DWORDS CONTROL field. A byte specifying a control character shall only specify a control character that is used in this standard (see table 51 in 5.3.7) and is supported by the phy (i.e., all phys support K28.5 (i.e., BCh), but only phys supporting STP support K28.3 (i.e., 7Ch) or K28.6 (i.e., DCh)).

The SCSI device server shall terminate a SEND DIAGNOSTIC command specifying any unsupported combination with CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

Table 280 lists some examples of TWO\_DWORDS phy test patterns.

**Table 280 – TWO\_DWORDS phy test pattern examples**

PHY TEST PATTERN DWORDS CONTROL field	PHY TEST PATTERN DWORDS field	Description
00h	4A4A4A4A 4A4A4A4Ah	D10.2 characters (see table 49 in 5.3.6). This pattern contains 01b repeating and has the highest possible frequency. This pattern may be used for measuring intra-pair skew, rise time, fall time, and RJ (see SAS-4).
00h	B5B5B5B5 B5B5B5B5h	D21.5 characters (see table 49 in 5.3.6). This pattern contains 10b repeating and has the highest possible frequency. This pattern may be used for measuring intra-pair skew, rise time, fall time, and RJ (see SAS-4).
00h	78787878 78787878h	D24.3 characters (see table 49 in 5.3.6). This pattern contains 0011b or 1100b repeating (depending on starting disparity) and has half the highest possible frequency. This pattern may be used for calibrating the JTF, calibrating the reference transmitter test load, and measuring transmitter device S-parameters (see SAS-4).
00h	D926D926 D926D926h	Pairs of D25.6 and D6.1 characters (see table 49 in 5.3.6). This pattern contains 1001b repeating and has half the highest possible frequency.
00h	7E7E7E7E 7E7E7E7Eh	D30.3 characters (see table 49 in 5.3.6). This pattern contains four bits of one polarity, three bits of the other polarity, and three bits of the first polarity (e.g., 11 11000111b), followed by the inverse (e.g., 00 00111000b). This pattern may be used for measuring transmitter equalization and SSC-induced jitter (see SAS-4).
88h	BC4A4A7B BC4A4A7Bh	ALIGN (0) primitives (see table 125 in 6.2.3). This pattern may appear during OOB bursts (SAS-4), the SATA speed negotiation sequence (see 5.11.2.2), and the SAS speed negotiation sequence (see 5.11.4.2).
88h	BC070707 BC070707h	ALIGN (1) primitives (see table 125 in 6.2.3). This pattern may appear during the SAS speed negotiation sequences (see 5.11.4.2).
80h	BC4A4A7B 4A787E7Eh	Pairs of an ALIGN (0) (see table 125 in 6.2.3) and a dword containing D10.2, D24.3, D30.3, and D30.3 characters (see table 49 in 5.3.6).

### 9.2.9.3 Enclosure Control diagnostic page

If the SELECT bit (see SES-3) is set to one for any element that represents a SAS device attached to an expander phy for which the SAS initiator port performing the SEND DIAGNOSTIC command does not have access according to the zone permission table, then the enclosure services process shall terminate the SEND DIAGNOSTIC command with CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

#### 9.2.9.4 Enclosure Status diagnostic page

The enclosure services process shall set the ELEMENT STATUS CODE field (see SES-3) to 8h (i.e., No Access Allowed) for each element that represents a SAS device attached to an expander phy for which the SAS initiator port performing the RECEIVE DIAGNOSTIC RESULTS command does not have access according to the zone permission table.

#### 9.2.9.5 Additional Element Status diagnostic page

The enclosure services process shall set the INVALID bit to one in the Additional Element Status descriptor (see SES-3) for each element that represents a SAS target device attached to an expander phy for which the SAS initiator port performing the RECEIVE DIAGNOSTIC RESULTS command does not have access according to the zone permission table.

### 9.2.10 SCSI power conditions

#### 9.2.10.1 SCSI power conditions overview

The logical unit power condition states (see 9.2.10.2) are controlled by the Power Condition mode page (see SPC-4) and START STOP UNIT command (see SBC-3), if implemented, and shall interact with the SL\_P\_C state machine (see 6.14.5) to control temporary consumption of additional power (e.g., to spin up rotating media) as described in this subclause.

The device server in the logical unit sends requests to the SL\_P\_C state machine and receives confirmations from the SL\_P\_C state machine to delay:

- a) initial temporary consumption of additional power after power on;
- b) temporary consumption of additional power requested by START STOP UNIT commands; and
- c) temporary consumption of additional power while making a change from a standby power condition to a higher power condition.

#### 9.2.10.2 SA\_PC (SCSI application layer power condition) state machine

##### 9.2.10.2.1 SA\_PC state machine overview

The SA\_PC (SCSI application layer power condition) state machine describes how the SAS target device processes logical unit power condition state change requests.

Logical units with device types other than direct-access block device (e.g., sequential-access devices) that implement an additional power condition, not defined by this standard, that consumes more peak power during a change from that additional power condition to a higher power condition than the typical peak power consumption in the active power condition shall use the SL\_P\_C state machine (see 6.14.5) to delay consumption of additional power when making the change to a higher power condition.

NOTE 67 - This state machine is an enhanced version of the logical unit power condition state machines described in SPC-4 and SBC-3 that adds the interactions with the SL\_P\_C state machine.

This state machine consists of the states shown in table 281.

**Table 281 – Summary of states in the SA\_PC state machine**

State	Reference	Modified <sup>a</sup>	States that contribute to definition	
			SPC-4	SBC-3
SA_PC_0:Powered_On <sup>b</sup>	9.2.10.2.2	No	PC0:Powered_On	SSU_PC0:Powered_On
SA_PC_1:Active	9.2.10.2.3	No	PC1:Active	SSU_PC1:Active
SA_PC_2:Idle	9.2.10.2.4	No	PC2:Idle	SSU_PC2:Idle
SA_PC_3:Standby	9.2.10.2.5	No	PC3:Standby	SSU_PC3:Standby
SA_PC_4:Active_Wait	9.2.10.2.6	Yes	PC4:Active_Wait	SSU_PC4:Active_Wait
SA_PC_5:Wait_Idle	9.2.10.2.7	No	PC5:Wait_Idle	SSU_PC5:Wait_Idle
SA_PC_6:Wait_Standby	9.2.10.2.8	No	PC6:Wait_Standby	SSU_PC6:Wait_Standby
SA_PC_7:Idle_Wait	9.2.10.2.9	Yes		SSU_PC7:Idle_Wait
SA_PC_8:Stopped	9.2.10.2.10	No		SSU_PC8:Stopped
SA_PC_9:Standby_Wait	9.2.10.2.11	No		SSU_PC9:Standby_Wait
SA_PC_10:Wait_Stopped	9.2.10.2.12	No		SSU_PC10:Wait_Stopped
<sup>a</sup> Yes indicates that this standard adds requirements to a state. No indicates that this standard does not alter or enhance the requirements defined in SPC-4 and SBC-3. <sup>b</sup> SA_PC_0:Powered_On is the initial state.				

While in the following SA\_PC states the logical unit may be increasing power usage to enter a higher power condition:

- a) SA\_PC\_4:Active\_Wait;
- b) SA\_PC\_7:Idle\_Wait; or
- c) SA\_PC\_9:Standby\_Wait.

While in the following SA\_PC states the logical unit may be decreasing power usage to enter a lower power condition:

- a) SA\_PC\_5:Wait\_Idle;
- b) SA\_PC\_6:Wait\_Standby; or
- c) SA\_PC\_10:Wait\_Stopped.

Any command causing a state machine transition (e.g., a START STOP UNIT command with the IMMED bit set to zero) shall not complete with GOOD status until this state machine reaches the state (i.e., power condition) required by the command.

This state machine shall start in the SA\_PC\_0:Powered\_On state after power on. For direct-access block devices, the SA\_PC state machine shall be configured to transition to the SA\_PC\_8:Stopped state or the SA\_PC\_4:Active\_Wait state after power on by a mechanism outside the scope of this standard.

This state machine receives the following confirmations from the SL\_P\_C link layer state machine:

- a) Power Use Granted; and
- b) Power Request Failed.

This state machine sends the following requests to the SL\_P\_C link layer state machine:

- a) Request Additional Power; and
- b) Power Use Complete.

This state machine sends the following message to the management application layer:

- a) Phy Power Condition Status.

This state machine uses the following timers that are controlled by the Power Condition mode page (see SPC-4):

- a) the idle condition timers; and
- b) the standby condition timers.

If the SCSI device server processes a START STOP UNIT command with the IMMED bit set to one, then the SCSI device server shall complete the command before completing the transition, if any, specified by a START STOP UNIT command.



Figure 210 shows the SA\_PC state machine.

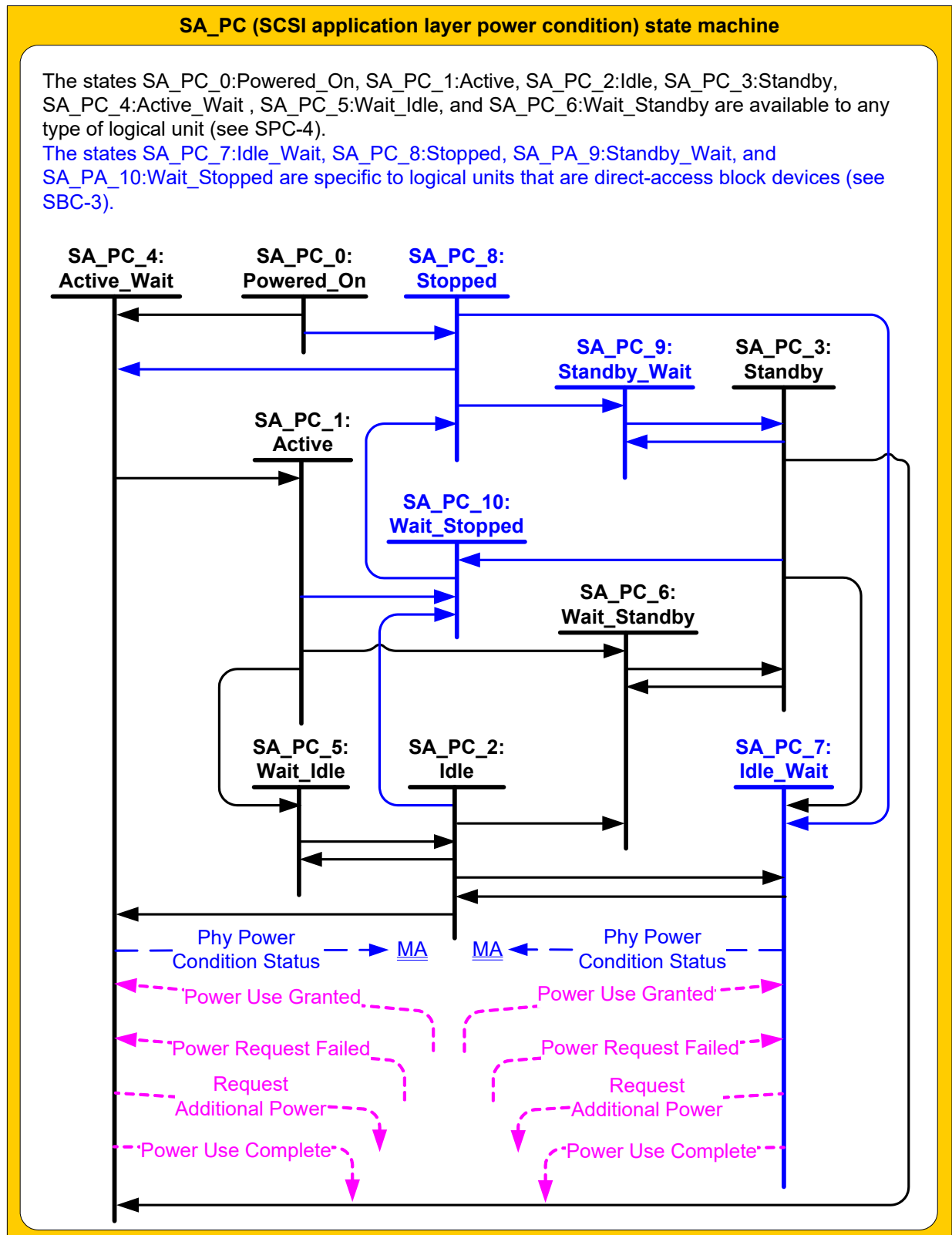


Figure 210 – SA\_PC (SCSI application layer power condition) state machine for SAS

**9.2.10.2.2 SA\_PC\_0:Powered\_On state****9.2.10.2.2.1 State description**

See the PC0:Power\_On state in SPC-4 for details about this state.

**9.2.10.2.2.2 Transition SA\_PC\_0:Powered\_On to SA\_PC\_4:Active\_Wait**

For SAS target devices that are not direct-access block devices, see the PC0:Power\_On to PC4:Active\_Wait transition in SPC-4 for details about this transition.

For direct-access block devices, see the SSU\_PC0:Power\_On to SSU\_PC4:Active\_Wait transition in SBC-3 for details about this transition.

**9.2.10.2.2.3 Transition SA\_PC\_0:Powered\_On to SA\_PC\_8:Stopped**

This transition is only implemented in logical units that are direct-access block devices. See the SSU\_PC0:Power\_On to SSU\_PC8:Stopped transition in SBC-3 for details about this transition.

**9.2.10.2.3 SA\_PC\_1:Active state****9.2.10.2.3.1 State description**

See the PC1:Active state in SPC-4 for details about this state.

**9.2.10.2.3.2 Transition SA\_PC\_1:Active to SA\_PC\_5:Wait\_Idle**

For SAS target devices that are not direct-access block devices, see the PC1:Active to PC5:Wait\_Idle transition in SPC-4 for details about this transition.

For direct-access block devices, see the SSU\_PC1:Active to SSU\_PC5:Wait\_Idle transition in SBC-3 for details about this transition.

**9.2.10.2.3.3 Transition SA\_PC\_1:Active to SA\_PC\_6:Wait\_Standby**

For SAS target devices that are not direct-access block devices, see the PC1:Active to PC6:Wait\_Standby transition in SPC-4 for details about this transition.

For direct-access block devices, see the SSU\_PC1:Active to SSU\_PC6:Wait\_Standby transition in SBC-3 for details about this transition.

**9.2.10.2.3.4 Transition SA\_PC\_1:Active to SA\_PC\_10:Wait\_Stopped**

This transition is only implemented in logical units that are direct-access block devices. See the SSU\_PC1:Active to SSU\_PC10:Wait\_Stopped transition in SBC-3 for details about this transition.

**9.2.10.2.4 SA\_PC\_2:Idle state****9.2.10.2.4.1 State description**

See the PC2:Idle state in SPC-4 for details about this state.

**9.2.10.2.4.2 Transition SA\_PC\_2:Idle to SA\_PC\_4:Active\_Wait**

For SAS target devices that are not direct-access block devices, see the PC2:Idle to PC4:Active\_Wait transition in SPC-4 for details about this transition.

For direct access block devices, see the SSU\_PC2:Idle to SSU\_PC4:Active\_Wait transition in SBC-3 for details about this transition.

**9.2.10.2.4.3 Transition SA\_PC\_2:Idle to SA\_PC\_5:Wait\_Idle**

For SAS target devices that are not direct-access block devices, see the PC2:Idle to PC5:Wait\_Idle transition in SPC-4 for details about this transition.

For direct-access block devices, see the SSU\_PC2:Idle to SSU\_PC5:Wait\_Idle transition in SBC-3 for details about this transition.

**9.2.10.2.4.4 Transition SA\_PC\_2:Idle to SA\_PC\_6:Wait\_Standby**

For devices that are not direct-access block device, see the PC2:Idle to PC6:Wait\_Standby transition in SPC-4 for details about this transition.

For direct-access block devices, see the SSU\_PC2:Idles to SSU\_PC6:Wait\_Standby transition in SBC-3 for details about this transition.

**9.2.10.2.4.5 Transition SA\_PC\_2:Idle to SA\_PC\_7:Idle\_Wait**

This transition is only implemented in logical units that are direct-access block devices. See the SSU\_PC2:Idle to SSU\_PC7:Idle\_Wait transition in SBC-3 for details about this transition.

**9.2.10.2.4.6 Transition SA\_PC\_2:Idle to SA\_PC\_10:Wait\_Stopped**

This transition is only implemented in logical units that are direct-access block devices. See the SSU\_PC2:Idle to SSU\_PC10:Wait\_Stopped transition in SBC-3 for details about this transition.

**9.2.10.2.5 SA\_PC\_3:Standby state****9.2.10.2.5.1 State description**

See the PC3:Standby state in SPC-4 for details about this state.

**9.2.10.2.5.2 Transition SA\_PC\_3:Standby to SA\_PC\_4:Active\_Wait**

For SAS target devices that are not direct-access block devices, see the PC3:Standby to PC4:Active\_Wait transition in SPC-4 for details about this transition.

For direct-access block devices, see the SSU\_PC3:Standby to SSU\_PC4:Active\_Wait transition in SBC-3 for details about this transition.

**9.2.10.2.5.3 Transition SA\_PC\_3:Standby to SA\_PC\_6:Wait\_Standby**

For SAS target devices that are not direct-access block devices, see the PC3:Standby to PC6:Wait\_Standby transition in SPC-4 for details about this transition.

For direct-access block devices, see the SSU\_PC3:Standby to SSU\_PC6:Wait\_Standby transition in SBC-3 for details about this transition.

**9.2.10.2.5.4 Transition SA\_PC\_3:Standby to SA\_PC\_7:Idle\_Wait**

This transition is only implemented in logical units that are direct-access block devices. See the SSU\_PC3:Standby to SSU\_PC7:Idle\_Wait transition in SBC-3 for details about this transition.

**9.2.10.2.5.5 Transition SA\_PC\_3:Standby to SA\_PC\_9: Standby\_Wait**

This transition is only implemented in logical units that are direct-access block devices. See the SSU\_PC3:Standby to SSU\_PC9:Standby\_Wait transition in SBC-3 for details about this transition.

**9.2.10.2.5.6 Transition SA\_PC\_3:Standby to SA\_PC\_10:Wait\_Stopped**

This transition is only implemented in logical units that are direct-access block devices. See the SSU\_PC3:Standby to SSU\_PC10:Wait\_Stopped transition in SBC-3 for details about this transition.

**9.2.10.2.6 SA\_PC\_4:Active\_Wait state****9.2.10.2.6.1 State description**

If this state was entered with a Transitioning From Power On argument, Transitioning From Standby argument, or Transitioning From Stopped argument, then this state shall:

- 1) send a Phy Power Condition Status (Disable Low Phy Power Conditions) message to the management application layer; and
- 2) if power control is enabled and the consumption of additional power is required, then send a Request Additional Power request to the SL\_P\_C state machine (see 6.14.5).

While in this state:

- a) each idle condition timer that is enabled and not expired is running;
- b) each standby condition timer that is enabled and not expired is running;
- c) the SCSI device server shall provide pollable sense data (see SPC-4) with the sense key set to NO SENSE and the additional sense code set to LOGICAL UNIT TRANSITIONING TO ANOTHER POWER CONDITION;
- d) if a Power Request Failed (Grant Timeout) confirmation is received, then this state shall send a Request Additional Power request to the SL\_P\_C state machine on any phy;
- e) if a Power Request Failed (ACK Timeout) confirmation or a Power Request Failed (Phy Disabled) confirmation is received, then this state should send a Request Additional Power request to the SL\_P\_C state machine on a different phy from the one that Power Request Failed confirmation was received; and
- f) the logical unit is performing the operations required for it to be in the SA\_PC\_1:Active state (e.g., a hard disk drive spins up its media).

If this state was entered with a Transitioning From Idle argument, then:

- a) the SCSI device server is capable of processing and completing the same commands, except a START STOP UNIT command with the IMMED bit set to zero (see SBC-3), that the SCSI device server is able to process and complete while in the SA\_PC\_2:Idle state;
- b) the peak power consumed in this state shall be no more than the typical peak power consumed in the SA\_PC\_1: Active state; and
- c) if:
  - A) this state was entered with a Transitioning From Idle\_c argument; and
  - B) the CCF IDLE field in the Power Conditions mode page (see SPC-4) is set to 10b (i.e., enabled),
 then the SCSI device server shall terminate any command, except START STOP UNIT command, that requires the logical unit be in the SA\_PC\_1:Active state to continue processing, with CHECK CONDITION status, with the sense key set to NOT READY and the additional sense code set to LOGICAL UNIT IS IN PROCESS OF BECOMING READY.

If this state was entered with a Transitioning From Standby argument, then:

- a) the SCSI device server is capable of processing and completing the same commands, except a START STOP UNIT command with the IMMED bit set to zero, that the SCSI device server is able to process and complete while in the SA\_PC\_3:Standby state;
- b) if the CCF STANDBY field in the Power Conditions mode page (see SPC-4) is set to 10b (i.e., enabled), then the SCSI device server shall terminate any command, except a START STOP UNIT command, that requires the logical unit be in the SA\_PC\_1:Active state or SA\_PC\_2:Idle state to continue processing, with CHECK CONDITION status, with the sense key set to NOT READY and:
  - A) if power control is enabled (see 6.14) and the Power Use Granted confirmation has not been received, then the additional sense code set to LOGICAL UNIT NOT READY, ADDITIONAL POWER USE NOT YET GRANTED;

- B) if power control is not enabled (see 6.14) and the Power Use Granted confirmation has not been received, then the additional sense code set to LOGICAL UNIT NOT READY, NOTIFY (ENABLE SPINUP) REQUIRED; or
- C) if the Power Use Granted confirmation has been received, then the additional sense code set to LOGICAL UNIT IN THE PROCESS OF BECOMING READY;
- c) if a Power Use Granted confirmation has not been received, then the peak power consumption shall be no more than the typical peak power consumed in the SA\_PC\_1: Active state; and
- d) if a Power Use Granted confirmation has been received, then the peak power consumption is not limited by this standard.

If this state was entered with a Transitioning From Stopped argument, then:

- a) the SCSI device server is capable of processing and completing the same commands, except a START STOP UNIT command with the IMMED bit set to zero, that the SCSI device server is able to process and complete while in the SA\_PC\_8:Stopped state;
- b) if the CCF STOPPED field in the Power Conditions mode page (see SPC-4) is set to 10b (i.e., enabled), then the SCSI device server shall terminate any TEST UNIT READY command or media access command, with CHECK CONDITION status, with the sense key set to NOT READY and:
  - A) if power control is enabled (see 6.14) and the Power Use Granted confirmation has not been received, then the additional sense code set to LOGICAL UNIT NOT READY, ADDITIONAL POWER USE NOT YET GRANTED;
  - B) if power control is not enabled (see 6.14) and the Power Use Granted confirmation has not been received, then the additional sense code set to LOGICAL UNIT NOT READY, NOTIFY (ENABLE SPINUP) REQUIRED; or
  - C) if the Power Use Granted confirmation has been received, then the additional sense code set to LOGICAL UNIT IN THE PROCESS OF BECOMING READY;
- c) if a Power Use Granted confirmation has not been received, then the peak power consumption shall be no more than the typical peak power consumed in the SA\_PC\_1: Active state; and
- d) if a Power Use Granted confirmation has been received, then the peak power consumption is not limited by this standard.

If this state was entered with a Transitioning From Powered On argument, then:

- a) the SCSI device server is capable of processing and completing the same commands, except a START STOP UNIT command with the IMMED bit set to zero, that the SCSI device server is able to process and complete while in the SA\_PC\_8:Stopped state for direct-access devices;
- b) the SCSI device server shall terminate any TEST UNIT READY command or media access command with CHECK CONDITION status with the sense key set to NOT READY and:
  - A) if power control is enabled (see 6.14) and the Power Use Granted confirmation has not been received, then the additional sense code set to LOGICAL UNIT NOT READY, ADDITIONAL POWER USE NOT YET GRANTED;
  - B) if power control is not enabled (see 6.14) and the Power Use Granted confirmation has not been received, then the additional sense code set to LOGICAL UNIT NOT READY, NOTIFY (ENABLE SPINUP) REQUIRED; or
  - C) if the Power Use Granted confirmation has been received, then the additional sense code shall be set to LOGICAL UNIT IN THE PROCESS OF BECOMING READY;
- c) if a Power Use Granted confirmation has not been received, then the peak power consumption shall be no more than the typical peak power consumed in the SA\_PC\_1: Active state; and
- d) if a Power Use Granted confirmation has been received, then the peak power consumption is not limited by this standard.

If an idle condition timer or a standby condition timer is enabled and expires, then that timer shall be ignored.

When the SCSI device server meets the requirements for the logical unit being in the SA\_PC\_1:Active state this state shall:

- 1) if this state sent a Request Additional Power request to the SL\_P\_C state machine (see 6.14.5), then send a Power Use Complete request to the SL\_P\_C state machine (see 6.14.5); and
- 2) send a Phy Power Condition Status (Enable Low Phy Power Conditions) message to the management application layer.

**9.2.10.2.6.2 Transition SA\_PC\_4:Active\_Wait to SA\_PC\_1:Active**

This transition shall occur:

- a) after sending a Phy Power Condition Status (Enable Low Phy Power Conditions) message to the management application layer.

**9.2.10.2.7 SA\_PC\_5:Wait\_Idle state****9.2.10.2.7.1 SA\_PC\_5:Wait\_Idle state description**

See the PC5:Wait\_Idle state in SPC-4 for details about this state.

**9.2.10.2.7.2 Transition SA\_PC\_5:Wait\_Idle to SA\_PC\_2:Idle**

See the PC5:Wait\_Idle to PC2:Idle transition in SPC-4 for details about this transition.

**9.2.10.2.8 SA\_PC\_6:Wait\_Standby state****9.2.10.2.8.1 SA\_PC\_6:Wait\_Standby state description**

See the PC6:Wait\_Standby state in SPC-4 for details about this state.

**9.2.10.2.8.2 Transition SA\_PC\_6:Wait\_Standby to SA\_PC\_3:Standby**

See the PC6:Wait\_Standby to PC3:Standby transition in SPC-4 for details about this transition.

**9.2.10.2.9 SA\_PC\_7:Idle\_Wait state****9.2.10.2.9.1 State description**

If this state was entered with a Transitioning From Standby argument or Transitioning From Stopped argument, then this state shall:

- 1) send a Phy Power Condition Status (Disable Low Phy Power Conditions) message to the management application layer; and
- 2) if power control is enabled and the consumption of additional power is required, then send a Request Additional Power request to the SL\_P\_C state machine (see 6.14.5).

While in this state:

- a) each idle condition timer that is enabled and not expired is running;
- b) each standby condition timer that is enabled and not expired is running;
- c) the SCSI device server shall provide pollable sense data (see SPC-4) with the sense key set to NO SENSE and the additional sense code set to LOGICAL UNIT TRANSITIONING TO ANOTHER POWER CONDITION;
- d) if a Power Request Failed (Grant Timeout) confirmation is received, then this state shall send a Request Additional Power request to the SL\_P\_C state machine on any phy;
- e) if a Power Request Failed (ACK Timeout) confirmation or a Power Request Failed (Phy Disabled) confirmation is received, then this state should send a Request Additional Power request to the SL\_P\_C state machine on a different phy from the one that Power Request Failed confirmation was received; and
- f) the logical unit is performing the operations required for it to be in the SA\_PC\_2:Idle state (e.g., a hard disk drive spinning up its media).

If this state was entered with a Transitioning From Idle argument, then:

- a) the SCSI device server is capable of processing and completing the same commands, except a START STOP UNIT command with the IMMED bit set to zero (see SBC-3), that the SCSI device server is able to process and complete while in the SA\_PC\_2:Idle state; and

- b) the peak power consumed in this state shall be no more than the typical peak power consumed in the SA\_PC\_1: Active state.

If this state was entered with a Transitioning From Standby argument, then:

- a) the SCSI device server is capable of processing and completing the same commands, except a START STOP UNIT command with the IMMED bit set to zero, that the SCSI device server is able to process and complete while in the SA\_PC\_3:Standby state;
- b) if the CCF STANDBY field in the Power Conditions mode page (see SPC-4) is set to 10b (i.e., enabled), then the SCSI device server shall terminate any command, except a START STOP UNIT command, that requires the logical unit be in the SA\_PC\_1:Active state or SA\_PC\_2:Idle state to continue processing, with CHECK CONDITION status, with the sense key set to NOT READY and:
  - A) if power control is enabled (see 6.14) and the Power Use Granted confirmation has not been received, then the additional sense code set to LOGICAL UNIT NOT READY, ADDITIONAL POWER USE NOT YET GRANTED;
  - B) if power control is not enabled (see 6.14) and the Power Use Granted confirmation has not been received, then the additional sense code set to LOGICAL UNIT NOT READY, NOTIFY (ENABLE SPINUP) REQUIRED; or
  - C) if the Power Use Granted confirmation has been received, then the additional sense code set to LOGICAL UNIT IN THE PROCESS OF BECOMING READY;
- c) if a Power Use Granted confirmation has not been received, then the peak power consumption shall be no more than the typical peak power consumed in the SA\_PC\_1: Active state; and
- d) if a Power Use Granted confirmation has been received, then the peak power consumption is not limited by this standard.

If this state was entered with a Transitioning From Stopped argument, then:

- a) the SCSI device server is capable of processing and completing the same commands, except a START STOP UNIT command with the IMMED bit set to zero, that the SCSI device server is able to process and complete while in the SA\_PC\_8:Stopped state;
- b) if the CCF STOPPED field in the Power Conditions mode page (see SPC-4) is set to 10b (i.e., enabled), then the SCSI device server shall terminate any TEST UNIT READY command or media access command, with CHECK CONDITION status, with the sense key set to NOT READY and:
  - A) if power control is enabled (see 6.14) and the Power Use Granted confirmation has not been received, then the additional sense code set to LOGICAL UNIT NOT READY, ADDITIONAL POWER USE NOT YET GRANTED;
  - B) if power control is not enabled (see 6.14) and the Power Use Granted confirmation has not been received, then the additional sense code set to LOGICAL UNIT NOT READY, NOTIFY (ENABLE SPINUP) REQUIRED; or
  - C) if the Power Use Granted confirmation has been received, then the additional sense code set to LOGICAL UNIT IN THE PROCESS OF BECOMING READY;
- c) if a Power Use Granted confirmation has not been received, then the peak power consumption shall be no more than the typical peak power consumed in the SA\_PC\_1: Active state; and
- d) if a Power Use Granted confirmation has been received, then the peak power consumption is not limited.

If an idle condition timer or a standby condition timer is enabled and expires, then that timer shall be ignored.

When the SCSI device server meets the requirements for the logical unit being in the:

- a) idle\_a power condition, if this state was entered with a Transitioning To Idle\_a argument;
- b) idle\_b power condition, if this state was entered with a Transitioning To Idle\_b argument; or
- c) idle\_c power condition, if this state was entered with a Transitioning To Idle\_c argument,

this state shall:

- 1) if this state sent a Request Additional Power request to the SL\_P\_C state machine (see 6.14.5), then send a Power Use Complete request to the SL\_P\_C state machine (see 6.14.5); and
- 2) send a Phy Power Condition Status (Enable Low Phy Power Conditions) message to the management application layer.

**9.2.10.2.9.2 Transition SA\_PC\_7:Idle\_Wait to SA\_PC\_2:Idle**

This transition shall occur:

- a) after sending a Phy Power Condition Status (Enable Low Phy Power Conditions) message to the management application layer.

**9.2.10.2.10 SA\_PC\_8:Stopped state****9.2.10.2.10.1 State description**

This state is only implemented in logical units that are direct-access block devices.

See the SSU\_PC8:Stopped state in SBC-3 for details about this state.

**9.2.10.2.10.2 Transition SA\_PC\_8:Stopped to SA\_PC\_4:Active\_Wait**

See the SSU\_PC8:Stopped to SSU\_PC4:Active\_Wait transition in SBC-3 for details about this transition.

The transition shall include:

- a) a From Stopped argument.

**9.2.10.2.10.3 Transition SA\_PC\_8:Stopped to SA\_PC\_7:Idle\_Wait**

See the SSU\_PC8:Stopped to SSU\_PC7:Idle\_Wait transition in SBC-3 for details about this transition.

**9.2.10.2.10.4 Transition SA\_PC\_8:Stopped to SA\_PC\_9:Standby\_Wait**

See the SSU\_PC8:Stopped to SSU\_PC9:Standby\_Wait transition in SBC-3 for details about this transition.

**9.2.10.2.11 SA\_PC\_9:Standby\_Wait state****9.2.10.2.11.1 SA\_PC\_9:Standby\_Wait state description**

This state is only implemented in logical units that are direct-access block devices.

See the SSU\_PC9:Standby\_Wait state in SBC-3 for details about this state.

**9.2.10.2.11.2 Transition SA\_PC\_9:Standby\_Wait to SA\_PC\_3:Standby**

See the SSU\_PC9:Standby\_Wait to SSU\_PC3:Standby transition in SBC-3 for details about this transition.

**9.2.10.2.12 SA\_PC\_10:Wait\_Stopped state****9.2.10.2.12.1 SA\_PC\_10:Wait\_Stopped state description**

This state is only implemented in logical units that are direct-access block devices.

See the SSU\_PC10:Standby\_Wait state in SBC-3 for details about this state.

**9.2.10.2.12.2 Transition SA\_PC\_10:Wait\_Stopped to SA\_PC\_8:Stopped**

See the SSU\_PC10:Wait\_Stopped to SSU\_PC8:Stopped transition in SBC-3 for details about this transition.



## 9.2.11 SCSI vital product data (VPD)

### 9.2.11.1 SCSI vital product data (VPD) overview

Table 282 lists VPD pages for which this standard defines special requirements.

**Table 282 – VPD pages with special requirements for SAS SSP**

Page code	VPD page name	Reference	Support requirements
83h	Device Identification VPD page	9.2.11.2 and SPC-4	Mandatory
90h	Protocol Specific Logical Unit Information VPD page	9.2.11.3 and SPC-4	Mandatory
91h	Protocol Specific Port Information VPD page	9.2.11.4 and SPC-4	Mandatory

### 9.2.11.2 Device Identification VPD page

In the Device Identification VPD page (83h) returned in response to an INQUIRY command (see SPC-4), each logical unit in a SAS target device shall include the designation descriptors for the target port identifier (see 4.2.9) and the relative target port identifier (see SAM-5 and SPC-4) listed in table 283.

**Table 283 – Device Identification VPD page designation descriptors for the SAS target port**

Field in designation descriptor	Designation descriptor	
	Target port identifier	Relative target port identifier
DESIGNATOR TYPE	3h (i.e., NAA)	4h (i.e., relative target port identifier)
ASSOCIATION	01b (i.e., SCSI target port)	01b (i.e., SCSI target port)
CODE SET	1h (i.e., binary)	1h (i.e., binary)
DESIGNATOR LENGTH	8	4
PIV (protocol identifier valid)	1	1
PROTOCOL IDENTIFIER	6h (i.e., SAS)	6h (i.e., SAS)
DESIGNATOR	SAS address <sup>a</sup> (see 4.2.4)	Relative port identifier <sup>b</sup> as described in SAM-5 and SPC-4
<sup>a</sup> The DESIGNATOR field contains the SAS address of the SSP target port through which the INQUIRY command was received. <sup>b</sup> The DESIGNATOR field contains the relative port identifier of the SSP target port through which the INQUIRY command was received.		

In the Device Identification VPD page (83h) returned in response to an INQUIRY command (see SPC-4), each logical unit in a SAS target device shall include a designation descriptor for the SAS target device name (see 4.2.6) using NAA format and may include a designation descriptor for the SAS target device name using the SCSI name string format as listed in table 284.

**Table 284 – Device Identification VPD page designation descriptors for the SAS target device**

Field in designation descriptor	Designation descriptor	
	NAA format (required)	SCSI name string format (optional)
DESIGNATOR TYPE	3h (i.e., NAA)	8h (i.e., SCSI name string)
ASSOCIATION	10b (i.e., SCSI target device)	10b (i.e., SCSI target device)
CODE SET	1h (i.e., binary)	3h (i.e., UTF-8)
DESIGNATOR LENGTH	8	24
PIV (protocol identifier valid)	1	0
PROTOCOL IDENTIFIER	6h (i.e., SAS)	0h <sup>a</sup>
DESIGNATOR	Device name of the SAS target device in SAS address format (see 4.2.4)	Device name of the SAS target device in SCSI name string format (e.g., “naa.” followed by 16 hexadecimal digits followed by 4 ASCII null characters)
<sup>a</sup> The PROTOCOL IDENTIFIER field is reserved if the PIV bit is set to zero.		

Logical units may include designation descriptors in addition to those required by this standard (e.g., SCSI target devices with SCSI target ports using other SCSI transport protocols may return additional SCSI target device names for those other SCSI transport protocols).

### 9.2.11.3 Protocol Specific Logical Unit Information VPD page

The Protocol Specific Logical Unit Information VPD page (see SPC-4) contains parameters for the logical unit that are protocol specific based on the I\_T nexus being used to access the logical unit.

The Protocol Specific Logical Unit Information VPD page shall only return information relating to SAS target ports.

Table 285 defines the Protocol Specific Logical Unit Information VPD page for logical units with SSP ports.

**Table 285 – Protocol Specific Logical Unit Information VPD page for SAS SSP**

Byte\Bit	7	6	5	4	3	2	1	0
0	PERIPHERAL QUALIFIER			PERIPHERAL DEVICE TYPE				
1	PAGE CODE (90h)							
2	(MSB)							
	PAGE LENGTH (n - 3)							
3	(LSB)							
Logical unit information descriptor list								
4	Logical unit information descriptor (first) (see table 286)							
...								
...	...							
	Logical unit information descriptor (last) (see table 286)							
...								
n								

The PERIPHERAL QUALIFIER field and the PERIPHERAL DEVICE TYPE field are defined in SPC-4.

The PAGE CODE field is defined in SPC-4 and shall be set as shown in table 285 for the Protocol Specific Logical Unit Information VPD page for SAS SSP.

The PAGE LENGTH field is defined in SPC-4.

The logical unit information descriptor list is defined in SPC-4 and shall contain a logical unit information descriptor for each SAS target port known to the SCSI device server.

Table 286 defines the logical unit information descriptor for logical units with SSP target ports.

**Table 286 – Logical unit information descriptor for SAS SSP**

Byte\Bit	7	6	5	4	3	2	1	0
0	(MSB) _____							
1	RELATIVE PORT IDENTIFIER							_____ (LSB)
2	Reserved				PROTOCOL IDENTIFIER (6h)			
3	Reserved _____							
...								
5								
6	(MSB) _____							
7	DESCRIPTOR LENGTH (0004h)							_____ (LSB)
Per logical unit SCSI transport specific data								
8	Reserved							TLR CONTROL SUPPORTED
9	Reserved _____							
...								
11								

The RELATIVE PORT IDENTIFIER field is defined in SPC-4.

The PROTOCOL IDENTIFIER field is defined in SPC-4 and shall be set as shown in table 286 for the logical unit information descriptor for SAS SSP indicating that this is a SAS SSP specific descriptor.

The DESCRIPTOR LENGTH field is defined in SPC-4 and shall be set as shown in table 286 for the logical unit information descriptor for SAS SSP.

A TLR CONTROL SUPPORTED bit set to one indicates that the combination of the SSP target port and logical unit support the TLR CONTROL field in the SSP frame header (see 8.2.1). A TLR CONTROL SUPPORTED bit set to zero indicates that the combination of the SSP target port and logical unit do not support the TLR CONTROL field in the SSP frame header.

#### 9.2.11.4 Protocol Specific Port Information VPD page

The Protocol Specific Port Information VPD page (see SPC-4) contains parameters for the SAS target ports that are protocol specific and are the same for all logical units in the SAS target device.

Table 287 defines the Protocol Specific Port Information VPD page for the SSP target ports.

**Table 287 – Protocol Specific Port Information VPD page for SAS SSP**

Byte\Bit	7	6	5	4	3	2	1	0
0	PERIPHERAL QUALIFIER			PERIPHERAL DEVICE TYPE				
1	PAGE CODE (91h)							
2	(MSB)							
3	PAGE LENGTH (n - 3)							
	(LSB)							
Port information descriptor list								
4	Port information descriptor (first) (see table 288)							
...								
...	...							
	Port information descriptor (last) (see table 288)							
...								
n								

The PERIPHERAL QUALIFIER field and the PERIPHERAL DEVICE TYPE field are defined in SPC-4.

The PAGE CODE field is defined in SPC-4 and shall be set as shown in table 287 for the Protocol Specific Port Information VPD page for SAS SSP.

The PAGE LENGTH field is defined in SPC-4.

The port information descriptor list is defined in SPC-4 and shall contain a port information descriptor for each SAS target port known to the SCSI device server.

Table 288 defines the port information descriptor for the SSP target ports.

**Table 288 – Port information descriptor for SAS SSP**

Byte\Bit	7	6	5	4	3	2	1	0
0	(MSB) _____							
1	RELATIVE PORT IDENTIFIER _____ (LSB)							
2	Reserved				PROTOCOL IDENTIFIER (6h)			
3	Reserved							PWR_D_S
4	Reserved _____							
5	Reserved _____							
6	(MSB) _____							
7	DESCRIPTOR LENGTH (n-7) _____ (LSB)							
SAS phy information descriptor list								
8	SAS phy information descriptor (first) (see table 289) _____							
...								
11								
...	...							
n - 3	SAS phy information descriptor (last) (see table 289) _____							
...								
n								

The RELATIVE PORT IDENTIFIER field is defined in SPC-4.

The PROTOCOL IDENTIFIER field is defined in SPC-4 and shall be set as shown in table 288 for the port information descriptor for SAS SSP indicating that this is a port information descriptor for SAS SSP.

A power disable supported (PWR\_D\_S) bit set to one indicates that the POWER DISABLE signal (see SAS-4) is supported. A PWR\_D\_S bit set to zero indicates that the POWER DISABLE signal is not supported.

The DESCRIPTOR LENGTH field is defined in SPC-4.

The SAS phy information descriptor list contains a SAS phy information descriptor for each SAS phy in the SAS target port.

Table 289 defines the SAS phy information descriptor.

**Table 289 – SAS phy information descriptor for SAS SSP**

Byte\Bit	7	6	5	4	3	2	1	0
0	Reserved							
1	PHY IDENTIFIER							
2	Reserved							SSP PERSISTENT CAPABLE
3	Reserved							

The PHY IDENTIFIER field is defined in the SMP DISCOVER response (see 9.4.3.10).

An SSP PERSISTENT CAPABLE bit set to one indicates that the phy supports a persistent connection (see 4.1.13).

An SSP PERSISTENT CAPABLE bit set to zero indicates that the phy does not support persistent connections.

### 9.3 ATA application layer

No SAS-specific ATA features are defined by this standard.

### 9.4 Management application layer

#### 9.4.1 READY LED signal behavior

A SAS target device uses the READY LED signal to activate an externally visible LED that indicates the state of readiness and activity of the SAS target device. The READY LED signal electrical characteristics are described in SAS-4.

The system is not required to generate any visual output when the READY LED signal is asserted. Additional vendor specific flashing patterns may be used to signal vendor specific conditions.

SAS target devices without SSP target ports may transmit the READY LED signal using vendor specific methods and patterns.

SAS target devices with SSP target ports shall follow the behavior indicated by the READY LED MEANING bit in the Protocol Specific Port mode page (see 9.2.7.4) as described in table 290.

**Table 290 – READY LED signal behavior**

Power condition <sup>a</sup> (see 9.2.10) or activity	READY LED MEANING <b>bit set to zero</b> <sup>b</sup>	READY LED MEANING <b>bit set to one</b>
active power condition or an idle power condition	The SAS target device shall: a) while not processing a command, assert the READY LED signal continuously; or b) while processing a command, toggle the READY LED signal between the asserted and negated states in a vendor-specified manner. (i.e., the LED is usually on, but flashes on and off while commands are processed)	The SAS target device shall: a) while not processing a command, negate the READY LED signal continuously; or b) while processing a command, toggle the READY LED signal between the asserted and negated states in a vendor-specified manner. (i.e., the LED is usually off, but flashes on and off while commands are processed)
a standby power condition or stopped power condition	The SAS target device shall: a) while not processing a command, negate the READY LED signal continuously; or b) while processing a command, toggle the READY LED signal between the asserted and negated states in a vendor-specified manner. (i.e., the LED is usually off, but flashes on and off while commands are processed)  After a vendor specific amount of time in this state, SAS target devices with rotating media may be removed with minimum risk of mechanical or electrical damage.	
spinup/spindown	If the SAS target device has rotating media and is in the process of performing a spinup or spindown, then the SAS target device shall toggle the READY LED signal between the asserted and negated states with a 1 s ± 0.1 s cycle using a 50 % ± 10 % duty cycle (e.g., the LED is on for 0.5 s and off for 0.5 s).	
formatting media	If the SAS target device is in the process of formatting media, then the SAS target device shall toggle the READY LED signal between the asserted and negated states in a vendor-specified manner (e.g., with each cylinder change on a disk drive).	
<sup>a</sup> If the SAS target device has more than one logical unit and any logical unit is active or idle, then the power condition of that logical unit should be used to control the READY LED signal. <sup>b</sup> If the target device has rotating media, then a READY LED MEANING bit set to zero results in a READY LED signal behavior that provides an indication of the target device's readiness for removal. A SAS target device with rotating media that is not in a state for safe removal shall either toggle the READY LED signal at a significant rate during spinup, during spindown, and while formatting media, or assert the READY LED signal continuously. When removal is safe from a mechanical standpoint, the READY LED signal shall be negated.		



## 9.4.2 Management protocol services

The management application client and management device server use the following four-step process to perform SMP functions:

- 1) the management application client invokes Send SMP Function;
- 2) the SMP target port invokes SMP Function Received;
- 3) the management device server invokes Send SMP Function Response; and
- 4) the SMP initiator port invokes Received SMP Function Complete.

## 9.4.3 SMP functions

### 9.4.3.1 SMP functions overview

Table 291 defines the SMP functions.

**Table 291 – SMP functions (FUNCTION field) (part 1 of 3)**

Function code	SMP function	Description	Reference
SMP input functions (00h to 7Fh)			
General SMP input functions (00h to 0Fh)			
00h	REPORT GENERAL	Return general information about the SMP target device	9.4.3.4
01h	REPORT MANUFACTURER INFORMATION	Return vendor and product identification	9.4.3.5
02h	Obsolete		
03h	REPORT SELF-CONFIGURATION STATUS	Return status of the discover process in a self-configuring expander device	9.4.3.6
04h	REPORT ZONE PERMISSION TABLE	Return zone permission table values	9.4.3.7
05h	REPORT ZONE MANAGER PASSWORD	Return the zone manager password	9.4.3.8
06h	REPORT BROADCAST	Return information about Broadcast counters	9.4.3.9
07h	Restricted for SFF		
08h to 0Fh	Reserved		
Phy-based SMP input functions (10h to 1Fh)			
10h	DISCOVER	Return information about the specified phy	9.4.3.10
11h	REPORT PHY ERROR LOG	Return error logging information about the specified phy	9.4.3.11
12h	REPORT PHY SATA	Return information about a phy currently attached to a SATA phy	9.4.3.12

Table 291 – SMP functions (FUNCTION field) (part 2 of 3)

Function code	SMP function	Description	Reference
13h	REPORT ROUTE INFORMATION	Return phy-based expander route table information	9.4.3.13
14h	REPORT PHY EVENT	Return phy events for the specified phy	9.4.3.14
15h to 1Fh	Reserved		
Descriptor list-based SMP input functions (20h to 2Fh)			
20h	DISCOVER LIST	Return information about the specified phys	9.4.3.15
21h	REPORT PHY EVENT LIST	Return phy events	9.4.3.16
22h	REPORT EXPANDER ROUTE TABLE LIST	Return contents of the expander-based expander route table	9.4.3.17
23h to 2Fh	Reserved		
Other SMP input functions (30h to 7Fh)			
30h to 3Fh	Reserved		
40h to 7Fh	Vendor specific		
SMP output functions (80h to FFh)			
General SMP output functions (80h to 8Fh)			
80h	CONFIGURE GENERAL	Configure the SMP target device	9.4.3.18
81h	ENABLE DISABLE ZONING	Enable or disable zoning	9.4.3.19
82h	Obsolete		
83h	Restricted for SFF		
84h	Reserved		
85h	ZONED BROADCAST	Transmit the specified Broadcast on the expander ports in the specified zone groups	9.4.3.20
86h	ZONE LOCK	Lock a zoning expander device	9.4.3.21
87h	ZONE ACTIVATE	Set the zoning expander current values equal to the zoning expander shadow values	9.4.3.22
88h	ZONE UNLOCK	Unlock a zoning expander device	9.4.3.23
89h	CONFIGURE ZONE MANAGER PASSWORD	Configure the zone manager password	9.4.3.24
8Ah	CONFIGURE ZONE PHY INFORMATION	Configure zone phy information	9.4.3.25

**Table 291 – SMP functions (FUNCTION field) (part 3 of 3)**

Function code	SMP function	Description	Reference
8Bh	CONFIGURE ZONE PERMISSION TABLE	Configure the zone permission table	9.4.3.26
8Ch to 8Fh	Reserved		
Phy-based SMP output functions (90h to 9Fh)			
90h	CONFIGURE ROUTE INFORMATION	Change phy-based expander route table information	9.4.3.27
91h	PHY CONTROL	Request actions by the specified phy	9.4.3.28
92h	PHY TEST FUNCTION	Request a test function by the specified phy	9.4.3.29
93h	CONFIGURE PHY EVENT	Configure phy events for the specified phy	9.4.3.30
94h to 9Fh	Reserved		
Other SMP output functions (A0h to FFh)			
A0h to BFh	Reserved		
C0h to FFh	Vendor specific		

### 9.4.3.2 SMP function request frame format

#### 9.4.3.2.1 SMP function request frame format overview

An SMP request frame is sent by a management application client via an SMP initiator port to request an SMP function be performed by a management device server. Table 292 defines the SMP request frame format.

**Table 292 – SMP request frame format**

Byte <sup>a</sup> \Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (40h)							
1	FUNCTION							
2	ALLOCATED RESPONSE LENGTH							
3	REQUEST LENGTH ((00h) or ((n - 7) / 4))							
4	Additional request bytes							
...								
n - 4								
n - 3	(MSB)							
...	CRC							
n	(LSB)							
<sup>a</sup> Shaded byte numbers (e.g., bytes 0 to 3 and (n - 3) to n) show the bytes that are included in the request frame if the REQUEST LENGTH field is set to 00h. Functions defined in SAS-1.1 may be defined as containing more than eight bytes if the REQUEST LENGTH field is set to 00h.								

#### 9.4.3.2.2 SMP FRAME TYPE field

The SMP FRAME TYPE field is defined by the SMP transport layer (see 8.4.1) and parsed by the SMP transport layer state machines (see 8.4.5). The SMP FRAME TYPE field shall be set as shown in table 292 for the SMP request frame format.

#### 9.4.3.2.3 FUNCTION field

The FUNCTION field specifies which SMP function is being requested and is defined in table 291 (see 9.4.3.1). If the management device server does not support the value in the FUNCTION field, then the management device server shall return a function result of UNKNOWN SMP FUNCTION as described in table 294 (see 9.4.3.3.4).

#### 9.4.3.2.4 ALLOCATED RESPONSE LENGTH field

The ALLOCATED RESPONSE LENGTH field specifies the maximum number of dwords that the management application client has allocated in the data-in buffer for the additional response bytes in the response frame (see 9.4.3.3).

For compatibility with SAS-1.1, an ALLOCATED RESPONSE LENGTH field set to 00h specifies that a specific number of dwords are to be transferred as defined in the SMP function description. If the SMP function description does not specify a specific number of dwords, then the number of dwords to be transferred is zero. This condition shall not be considered as an error.

If the LONG RESPONSE bit is set to one in the REPORT GENERAL response (see 9.4.3.4), then the management application client may set the ALLOCATED RESPONSE LENGTH field to a non-zero value in all SMP request frames. If the LONG RESPONSE bit is set to zero in the REPORT GENERAL response, then the management application client shall set the ALLOCATED RESPONSE LENGTH field to 00h in all SMP request frames.

If the ALLOCATED RESPONSE LENGTH field is set to a non-zero value, then the management device server shall truncate the additional response bytes to the number of dwords specified by the ALLOCATED RESPONSE LENGTH field.

If the ALLOCATED RESPONSE LENGTH field is set to 00h, then the management device server shall truncate the additional response bytes to the number of dwords specified by the SMP function description.

The allocated response length is used to limit the maximum amount of variable length data returned to the management application client. Fields in the additional response bytes (e.g., fields containing counts of the number of dwords in some or all of the data) shall not be altered to reflect the truncation, if any, that results from an insufficient allocated response length.

#### 9.4.3.2.5 REQUEST LENGTH field

A REQUEST LENGTH field set to 00h specifies that either:

- a) no dwords follow the REQUEST LENGTH field before the CRC field; or
- b) a non-zero number of dwords follow the REQUEST LENGTH field before the CRC field. This is for compatibility with SAS-1.1.

The function description defines the interpretation of a REQUEST LENGTH field set to 00h.

A REQUEST LENGTH field set to a non-zero value (i.e., the non-zero value defined in table 292 (see 9.4.3.2.1)) specifies the number of dwords that follow the REQUEST LENGTH field before the CRC field (i.e., the length of the entire request frame minus two).

If the LONG RESPONSE bit is set to one in the REPORT GENERAL response (see 9.4.3.4), then the management application client may set the REQUEST LENGTH field to a non-zero value in the SMP request frame for any SMP function. If the LONG RESPONSE bit is set to zero in the REPORT GENERAL response, then the management application client shall set the REQUEST LENGTH field to 00h in the SMP request frame for every SMP function.

If the request frame size including the CRC field is less than 8 bytes or the REQUEST LENGTH field does not match the request frame size, then the management device server shall return a function result of INVALID REQUEST FRAME LENGTH.

The management device server shall consider any fields not included in the request frame to be set to zero.

#### 9.4.3.2.6 Additional request bytes

The additional request bytes definition and length are based on the SMP function.

The number of additional request bytes are an integer multiple of four, so the CRC field is aligned on a four byte boundary.

The maximum number of additional request bytes is 1 020, making the maximum size of the frame 1 028 bytes (i.e., 4 bytes of header + 1 020 bytes of data + 4 bytes of CRC).

NOTE 68 - If a management application client compliant with SAS-1.1 sends a vendor specific SMP request frame containing 1 024 additional request bytes, then the SMP\_TP state machine discards that SMP request frame as it exceeds the maximum allowed request size of 1 023 bytes (see 6.22.6.4.2.2). SMP request frames defined in SAS 1.1 do not have more than 36 additional request bytes.

If the management device server receives more additional request bytes than it expects (e.g., the management device server complies with a previous version of this standard (i.e., SAS-1.1) defining 24 additional request bytes, but receives a request frame containing 36 additional request bytes), then the management device server shall return a function result of INVALID REQUEST FRAME LENGTH.

For additional request bytes containing a DESCRIPTOR LENGTH field and a descriptor list, if the management device server receives more bytes in each descriptor than it expects (e.g., the management device server complies with a previous version of this standard (i.e., SAS-1.1) defining that a descriptor is 12 bytes, but receives a request frame containing a descriptor list with 16 byte descriptors), then the management device server shall return a function result of INVALID REQUEST FRAME LENGTH.

#### 9.4.3.2.7 CRC field

The CRC field is defined by the SMP transport layer (see 8.4.1) and parsed by the SMP link layer state machines (see 6.22.6).

#### 9.4.3.3 SMP function response frame format

##### 9.4.3.3.1 SMP function response frame format overview

An SMP response frame is sent by a management device server via an SMP target port in response to an SMP request frame. Table 293 defines the SMP response frame format.

**Table 293 – SMP response frame format**

Byte <sup>a</sup> \Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (41h)							
1	FUNCTION							
2	FUNCTION RESULT							
3	RESPONSE LENGTH (00h or ((n - 7) / 4))							
4	Additional response bytes							
...								
n - 4								
n - 3	(MSB)	CRC						
...								
n	(LSB)							

<sup>a</sup> Shaded byte numbers (e.g., bytes 0 to 3 and (n - 3) to n) show the bytes that are included in the response frame if the ALLOCATED RESPONSE LENGTH field is set to 00h in the request frame. Functions defined in SAS-1.1 may be defined as returning more than eight bytes if the ALLOCATED RESPONSE LENGTH field is set to 00h.

**9.4.3.3.2 SMP FRAME TYPE field**

The SMP FRAME TYPE field is defined by the SMP transport layer (see 8.4.1) and parsed by the MT state machines (see 8.4.5). The SMP FRAME TYPE field shall be set as shown in table 293 for the SMP response frame format.

**9.4.3.3.3 FUNCTION field**

The FUNCTION field indicates the SMP function to which this frame is a response and is defined in table 291 (see 9.4.3.1).

**9.4.3.3.4 FUNCTION RESULT field**

The FUNCTION RESULT field is defined in table 294.

**Table 294 – FUNCTION RESULT field (part 1 of 4)**

Code	Name	SMP functions	Description
00h	SMP FUNCTION ACCEPTED	All	The management device server supports the SMP function and processed the SMP function.
01h	UNKNOWN SMP FUNCTION	Unknown	The management device server does not support the requested SMP function.
02h	SMP FUNCTION FAILED	All	The requested SMP function failed.
03h	INVALID REQUEST FRAME LENGTH	All	The SMP request frame length was invalid (see 9.4.3.2).
04h	INVALID EXPANDER CHANGE COUNT	CONFIGURE GENERAL, ENABLE DISABLE ZONING, ZONE LOCK, ZONE ACTIVATE, CONFIGURE ZONE MANAGER PASSWORD, CONFIGURE ZONE PHY INFORMATION, CONFIGURE ZONE PERMISSION TABLE, CONFIGURE ROUTE INFORMATION, PHY CONTROL, PHY TEST FUNCTION, CONFIGURE PHY EVENT	The management device server supports the SMP function, but the EXPECTED EXPANDER CHANGE COUNT field does not match the current expander change count.

**Table 294 – FUNCTION RESULT field** (part 2 of 4)

Code	Name	SMP functions	Description
05h	BUSY	ZONE UNLOCK, ENABLE DISABLE ZONING, CONFIGURE ZONE MANAGER PASSWORD, CONFIGURE ZONE PHY INFORMATION, CONFIGURE ZONE PERMISSION TABLE	For ZONE UNLOCK, the locked zoning expander device is processing the activate step.  For the other functions, the management device server is currently saving zoning values.
06h	INCOMPLETE DESCRIPTOR LIST	ZONED BROADCAST, CONFIGURE ZONE PHY INFORMATION, CONFIGURE ZONE PERMISSION TABLE, CONFIGURE PHY EVENT	The request frame length results in the truncation of a multi-byte field or descriptor list (e.g., in the ZONED BROADCAST request (see table 362), the request frame is not large enough to contain the number of Broadcast source zone groups specified by the NUMBER OF BROADCAST SOURCE ZONE GROUPS field).
10h	PHY DOES NOT EXIST	DISCOVER, REPORT PHY ERROR LOG, REPORT PHY SATA, REPORT ROUTE INFORMATION, REPORT PHY EVENT, DISCOVER LIST, CONFIGURE ZONE PHY INFORMATION, CONFIGURE ROUTE INFORMATION, PHY CONTROL, PHY TEST FUNCTION, CONFIGURE PHY EVENT	The phy specified by the PHY IDENTIFIER field or the STARTING PHY IDENTIFIER field in the SMP request frame does not exist (e.g., the value is not less than the value indicated in the NUMBER OF PHYs field in the SMP REPORT GENERAL response).
11h	INDEX DOES NOT EXIST	REPORT ROUTE INFORMATION, CONFIGURE ROUTE INFORMATION	The phy specified by the PHY IDENTIFIER field in the SMP request frame does not have the table routing attribute (see 4.5.7.1) or the expander route index specified by the EXPANDER ROUTE INDEX field does not exist (i.e., the value is not in the range of 0000h to the value of the EXPANDER ROUTE INDEXES field in the SMP REPORT GENERAL response). The ADDITIONAL RESPONSE BYTES field may be present but shall be ignored.
12h	PHY DOES NOT SUPPORT SATA	REPORT PHY SATA, PHY CONTROL	See 9.4.3.12 and 9.4.3.28
13h	UNKNOWN PHY OPERATION	PHY CONTROL	See 9.4.3.28



**Table 294 – FUNCTION RESULT field** (part 3 of 4)

Code	Name	SMP functions	Description
14h	UNKNOWN PHY TEST FUNCTION	PHY TEST FUNCTION	See 9.4.3.29
15h	PHY TEST FUNCTION IN PROGRESS	PHY TEST FUNCTION	See 9.4.3.29
16h	PHY VACANT	DISCOVER, REPORT PHY ERROR LOG, REPORT PHY SATA, REPORT ROUTE INFORMATION, REPORT PHY EVENT, CONFIGURE ZONE PHY INFORMATION, CONFIGURE ROUTE INFORMATION, PHY CONTROL, PHY TEST FUNCTION, CONFIGURE PHY EVENT	The management device server processing the SMP request frame does not have access to a specified phy (e.g., because of zoning or vendor specific reasons), although the value is less than the value indicated in the NUMBER OF PHYS field in the SMP REPORT GENERAL response.
17h	UNKNOWN PHY EVENT SOURCE	CONFIGURE PHY EVENT	See 9.4.3.30.3
18h	UNKNOWN DESCRIPTOR TYPE	DISCOVER LIST	The descriptor type specified by the DESCRIPTOR TYPE field is not supported.
19h	UNKNOWN PHY FILTER	DISCOVER LIST	The phy filter specified by the PHY FILTER field is not supported.
1Ah	AFFILIATION VIOLATION	PHY CONTROL	The specified phy operation is not allowed due to the current state of affiliations.
20h	SMP ZONE VIOLATION	CONFIGURE GENERAL, ZONED BROADCAST, PHY CONTROL, PHY TEST FUNCTION, CONFIGURE PHY EVENT	Zoning is enabled and the SMP initiator port does not have access to a necessary zone group according to the zone permission table (see 4.8.3.2).
21h	NO MANAGEMENT ACCESS RIGHTS	REPORT ZONE MANAGER PASSWORD, ZONE LOCK, CONFIGURE ZONE MANAGER PASSWORD	For ZONE LOCK see 9.4.3.21. For REPORT ZONE MANAGER PASSWORD, see 9.4.3.8. For CONFIGURE ZONE MANAGER PASSWORD, see 9.4.3.24.
22h	UNKNOWN ENABLE DISABLE ZONING VALUE	ENABLE DISABLE ZONING	See 9.4.3.19

**Table 294 – FUNCTION RESULT field** (part 4 of 4)

Code	Name	SMP functions	Description
23h	ZONE LOCK VIOLATION	ENABLE DISABLE ZONING, ZONE LOCK, ZONE ACTIVATE, ZONE UNLOCK, CONFIGURE ZONE PHY INFORMATION, CONFIGURE ZONE PERMISSION TABLE	Zoning is enabled and: a) an unlocked zoning expander device receives an SMP zone configuration function request, a ZONE ACTIVATE request, or a ZONE UNLOCK request; or b) a locked zoning expander device receives an SMP zone configuration function request, a ZONE ACTIVATE request, or a ZONE UNLOCK request from an SMP initiator port that is not the active zone manager.
24h	NOT ACTIVATED	ZONE UNLOCK	The following conditions are true: a) the ACTIVATE REQUIRED bit is set to one in the request; and b) the locked zoning expander device has not processed the activate step.
25h	ZONE GROUP OUT OF RANGE	CONFIGURE ZONE PHY INFORMATION, CONFIGURE ZONE PERMISSION TABLE	The ZONE GROUP field or NUMBER OF ZONE GROUPS field contains a value that is not supported.
26h	NO PHYSICAL PRESENCE	CONFIGURE ZONE MANAGER PASSWORD	The following conditions are true: a) the NEW ZONE MANAGER PASSWORD field is set to DISABLED (see table 35 in 4.8.1); and b) physical presence is not asserted.
27h	SAVING NOT SUPPORTED	REPORT ZONE PERMISSION TABLE, REPORT ZONE MANAGER PASSWORD, ENABLE DISABLE ZONING, CONFIGURE ZONE MANAGER PASSWORD, CONFIGURE ZONE PHY INFORMATION, CONFIGURE ZONE PERMISSION TABLE	For REPORT ZONE PERMISSION TABLE, see 9.4.3.7.  For REPORT ZONE MANAGER PASSWORD, see 9.4.3.8.  For ENABLE DISABLE ZONING, CONFIGURE ZONE MANAGER PASSWORD, CONFIGURE ZONE PHY INFORMATION, and CONFIGURE ZONE PERMISSION TABLE, the following conditions are true the: a) SAVE field is set to 01b or 11b; and b) management device server does not support saved values for the specified information.
28h	SOURCE ZONE GROUP DOES NOT EXIST	REPORT ZONE PERMISSION TABLE	See 9.4.3.7
29h	DISABLED PASSWORD NOT SUPPORTED	CONFIGURE ZONE MANAGER PASSWORD	See 9.4.3.24
2Ah	INVALID FIELD IN SMP REQUEST	All	The management device server does not support a field in the requested SMP function.
All others	Reserved		

Table 295 defines the priority of the SMP function results defined in table 294.

**Table 295 – Function result priority (part 1 of 5)**

<b>SMP function</b>	<b>SMP function result priority is as follows:</b>
REPORT GENERAL (see 9.4.3.4)	1) INVALID REQUEST FRAME LENGTH; 2) SMP FUNCTION FAILED; 3) INVALID FIELD IN SMP REQUEST; and 4) SMP FUNCTION ACCEPTED.
REPORT MANUFACTURER INFORMATION (see 9.4.3.5)	1) INVALID REQUEST FRAME LENGTH; 2) SMP FUNCTION FAILED; 3) INVALID FIELD IN SMP REQUEST; and 4) SMP FUNCTION ACCEPTED.
REPORT SELF-CONFIGURATION STATUS (see 9.4.3.6)	1) INVALID REQUEST FRAME LENGTH; 2) SMP FUNCTION FAILED; 3) INVALID FIELD IN SMP REQUEST; and 4) SMP FUNCTION ACCEPTED.
REPORT ZONE PERMISSION TABLE (see 9.4.3.7)	1) INVALID REQUEST FRAME LENGTH; 2) SOURCE ZONE GROUP DOES NOT EXIST; 3) SAVING NOT SUPPORTED; 4) SMP FUNCTION FAILED; 5) INVALID FIELD IN SMP REQUEST; and 6) SMP FUNCTION ACCEPTED.
REPORT ZONE MANAGER PASSWORD (see 9.4.3.8)	1) INVALID REQUEST FRAME LENGTH; 2) NO MANAGEMENT ACCESS RIGHTS; 3) SAVING NOT SUPPORTED; 4) SMP FUNCTION FAILED; 5) INVALID FIELD IN SMP REQUEST; and 6) SMP FUNCTION ACCEPTED.
REPORT BROADCAST (see 9.4.3.9)	1) INVALID REQUEST FRAME LENGTH; 2) SMP FUNCTION FAILED; 3) INVALID FIELD IN SMP REQUEST; and 4) SMP FUNCTION ACCEPTED.
DISCOVER (see 9.4.3.10)	1) INVALID REQUEST FRAME LENGTH; 2) PHY DOES NOT EXIST; 3) PHY VACANT; 4) SMP FUNCTION FAILED; 5) INVALID FIELD IN SMP REQUEST; and 6) SMP FUNCTION ACCEPTED.
REPORT PHY ERROR LOG (see 9.4.3.11)	1) INVALID REQUEST FRAME LENGTH; 2) PHY DOES NOT EXIST; 3) PHY VACANT; 4) SMP FUNCTION FAILED; 5) INVALID FIELD IN SMP REQUEST; and 6) SMP FUNCTION ACCEPTED.

Table 295 – Function result priority (part 2 of 5)

SMP function	SMP function result priority is as follows:
REPORT PHY SATA (see 9.4.3.12)	1) INVALID REQUEST FRAME LENGTH; 2) PHY DOES NOT EXIST; 3) PHY VACANT; 4) PHY DOES NOT SUPPORT SATA; 5) SMP FUNCTION FAILED; 6) INVALID FIELD IN SMP REQUEST; and 7) SMP FUNCTION ACCEPTED.
REPORT ROUTE INFORMATION (see 9.4.3.13)	1) INVALID REQUEST FRAME LENGTH; 2) PHY DOES NOT EXIST; 3) PHY VACANT; 4) INDEX DOES NOT EXIST; 5) SMP FUNCTION FAILED; 6) INVALID FIELD IN SMP REQUEST; and 7) SMP FUNCTION ACCEPTED.
REPORT PHY EVENT (see 9.4.3.14)	1) INVALID REQUEST FRAME LENGTH; 2) PHY DOES NOT EXIST; 3) PHY VACANT; 4) SMP FUNCTION FAILED; 5) INVALID FIELD IN SMP REQUEST; and 6) SMP FUNCTION ACCEPTED.
DISCOVER LIST (see 9.4.3.15)	1) INVALID REQUEST FRAME LENGTH; 2) PHY DOES NOT EXIST; 3) UNKNOWN DESCRIPTOR TYPE; 4) UNKNOWN PHY FILTER; 5) SMP FUNCTION FAILED; 6) INVALID FIELD IN SMP REQUEST; and 7) SMP FUNCTION ACCEPTED.
REPORT PHY EVENT LIST (see 9.4.3.16)	1) INVALID REQUEST FRAME LENGTH; 2) SMP FUNCTION FAILED; 3) INVALID FIELD IN SMP REQUEST; and 4) SMP FUNCTION ACCEPTED.
REPORT EXPANDER ROUTE TABLE LIST (see 9.4.3.17)	1) INVALID REQUEST FRAME LENGTH; 2) SMP FUNCTION FAILED; 3) INVALID FIELD IN SMP REQUEST; and 4) SMP FUNCTION ACCEPTED.
CONFIGURE GENERAL (see 9.4.3.18)	1) INVALID REQUEST FRAME LENGTH; 2) SMP ZONE VIOLATION; 3) INVALID EXPANDER CHANGE COUNT; 4) SMP FUNCTION FAILED; 5) INVALID FIELD IN SMP REQUEST; and 6) SMP FUNCTION ACCEPTED.

Table 295 – Function result priority (part 3 of 5)

SMP function	SMP function result priority is as follows:
ENABLE DISABLE ZONING (see 9.4.3.19)	<ol style="list-style-type: none"> <li>1) INVALID REQUEST FRAME LENGTH;</li> <li>2) ZONE LOCK VIOLATION;</li> <li>3) UNKNOWN ENABLE DISABLE ZONING VALUE;</li> <li>4) INVALID EXPANDER CHANGE COUNT;</li> <li>5) SAVING NOT SUPPORTED;</li> <li>6) SMP FUNCTION FAILED;</li> <li>7) INVALID FIELD IN SMP REQUEST; and</li> <li>8) SMP FUNCTION ACCEPTED.</li> </ol>
ZONED BROADCAST (see 9.4.3.20)	<ol style="list-style-type: none"> <li>1) INVALID REQUEST FRAME LENGTH;</li> <li>2) INCOMPLETE DESCRIPTOR LIST;</li> <li>3) SMP ZONE VIOLATION;</li> <li>4) SMP FUNCTION FAILED;</li> <li>5) INVALID FIELD IN SMP REQUEST; and</li> <li>6) SMP FUNCTION ACCEPTED.</li> </ol>
ZONE LOCK (see 9.4.3.21)	<ol style="list-style-type: none"> <li>1) INVALID REQUEST FRAME LENGTH;</li> <li>2) NO MANAGEMENT ACCESS RIGHTS;</li> <li>3) INVALID EXPANDER CHANGE COUNT;</li> <li>4) SMP FUNCTION FAILED;</li> <li>5) INVALID FIELD IN SMP REQUEST; and</li> <li>6) SMP FUNCTION ACCEPTED.</li> </ol>
ZONE ACTIVATE (see 9.4.3.22)	<ol style="list-style-type: none"> <li>1) INVALID REQUEST FRAME LENGTH;</li> <li>2) ZONE LOCK VIOLATION;</li> <li>3) INVALID EXPANDER CHANGE COUNT;</li> <li>4) SMP FUNCTION FAILED;</li> <li>5) INVALID FIELD IN SMP REQUEST; and</li> <li>6) SMP FUNCTION ACCEPTED.</li> </ol>
ZONE UNLOCK (see 9.4.3.23)	<ol style="list-style-type: none"> <li>1) INVALID REQUEST FRAME LENGTH;</li> <li>2) ZONE LOCK VIOLATION;</li> <li>3) NOT ACTIVATED;</li> <li>4) BUSY;</li> <li>5) SMP FUNCTION FAILED;</li> <li>6) INVALID FIELD IN SMP REQUEST; and</li> <li>7) SMP FUNCTION ACCEPTED.</li> </ol>
CONFIGURE ZONE MANAGER PASSWORD (see 9.4.3.24)	<ol style="list-style-type: none"> <li>1) INVALID REQUEST FRAME LENGTH;</li> <li>2) INVALID EXPANDER CHANGE COUNT;</li> <li>3) NO MANAGEMENT ACCESS RIGHTS;</li> <li>4) NO PHYSICAL PRESENCE;</li> <li>5) SAVING NOT SUPPORTED;</li> <li>6) DISABLED PASSWORD NOT SUPPORTED;</li> <li>7) SMP FUNCTION FAILED;</li> <li>8) INVALID FIELD IN SMP REQUEST; and</li> <li>9) SMP FUNCTION ACCEPTED.</li> </ol>

Table 295 – Function result priority (part 4 of 5)

SMP function	SMP function result priority is as follows:
CONFIGURE ZONE PHY INFORMATION (see 9.4.3.25)	1) INVALID REQUEST FRAME LENGTH; 2) INCOMPLETE DESCRIPTOR LIST; 3) PHY DOES NOT EXIST; 4) PHY VACANT; 5) ZONE LOCK VIOLATION; 6) INVALID EXPANDER CHANGE COUNT; 7) SAVING NOT SUPPORTED; 8) ZONE GROUP OUT OF RANGE; 9) SMP FUNCTION FAILED; 10) INVALID FIELD IN SMP REQUEST; and 11) SMP FUNCTION ACCEPTED.
CONFIGURE ZONE PERMISSION TABLE (see 9.4.3.26)	1) INVALID REQUEST FRAME LENGTH; 2) INCOMPLETE DESCRIPTOR LIST; 3) ZONE LOCK VIOLATION; 4) INVALID EXPANDER CHANGE COUNT; 5) SAVING NOT SUPPORTED; 6) ZONE GROUP OUT OF RANGE; 7) SMP FUNCTION FAILED; 8) INVALID FIELD IN SMP REQUEST; and 9) SMP FUNCTION ACCEPTED.
CONFIGURE ROUTE INFORMATION (see 9.4.3.27)	1) INVALID REQUEST FRAME LENGTH; 2) PHY DOES NOT EXIST; 3) PHY VACANT; 4) INVALID EXPANDER CHANGE COUNT; 5) INDEX DOES NOT EXIST; 6) SMP FUNCTION FAILED; 7) INVALID FIELD IN SMP REQUEST; and 8) SMP FUNCTION ACCEPTED.
PHY CONTROL (see 9.4.3.28)	1) INVALID REQUEST FRAME LENGTH; 2) PHY DOES NOT EXIST; 3) PHY VACANT; 4) SMP ZONE VIOLATION; 5) INVALID EXPANDER CHANGE COUNT; 6) UNKNOWN PHY OPERATION; 7) PHY DOES NOT SUPPORT SATA; 8) AFFILIATION VIOLATION; 9) SMP FUNCTION FAILED; 10) INVALID FIELD IN SMP REQUEST; and 11) SMP FUNCTION ACCEPTED.

Table 295 – Function result priority (part 5 of 5)

SMP function	SMP function result priority is as follows:
PHY TEST FUNCTION (see 9.4.3.29)	1) INVALID REQUEST FRAME LENGTH; 2) PHY DOES NOT EXIST; 3) PHY VACANT; 4) SMP ZONE VIOLATION; 5) INVALID EXPANDER CHANGE COUNT; 6) UNKNOWN PHY TEST FUNCTION; 7) PHY TEST FUNCTION IN PROGRESS; 8) SMP FUNCTION FAILED; 9) INVALID FIELD IN SMP REQUEST; and 10) SMP FUNCTION ACCEPTED.
CONFIGURE PHY EVENT (see 9.4.3.30)	1) INVALID REQUEST FRAME LENGTH; 2) INCOMPLETE DESCRIPTOR LIST; 3) PHY DOES NOT EXIST; 4) PHY VACANT; 5) SMP ZONE VIOLATION; 6) INVALID EXPANDER CHANGE COUNT; 7) UNKNOWN PHY EVENT SOURCE; 8) SMP FUNCTION FAILED; 9) INVALID FIELD IN SMP REQUEST; and 10) SMP FUNCTION ACCEPTED.

#### 9.4.3.3.5 RESPONSE LENGTH field

A RESPONSE LENGTH field set to 00h indicates that either:

- a) no dwords follow the RESPONSE LENGTH field before the CRC field; or
- b) a non-zero number of dwords follow the RESPONSE LENGTH field before the CRC field. This is for compatibility with SAS-1.1.

The function description defines the interpretation of a RESPONSE LENGTH field set to 00h.

A RESPONSE LENGTH field set to a non-zero value (i.e., the non-zero value defined in table 293 (see 9.4.3.3.1)) indicates the number of dwords that follow the RESPONSE LENGTH field before the CRC field (i.e., the length of the entire response frame minus two).

#### 9.4.3.3.6 Additional response bytes

If the FUNCTION RESULT field is set to 00h, then the additional response bytes definition depends on the SMP function requested. If the FUNCTION RESULT field is set to a value other than 00h, then the additional response bytes may be present but shall be ignored.

The number of additional response bytes are an integer multiple of four, so the CRC field is aligned on a four byte boundary.

The maximum number of additional response bytes is 1 020, making the maximum size of the frame 1 028 bytes (i.e., 4 bytes of header + 1 020 bytes of data + 4 bytes of CRC).

NOTE 69 - If a management device server compliant with SAS-1.1 sends a vendor specific SMP response frame containing 1 024 additional response bytes then, the SMP\_IP state machine discards that SMP response frame as it exceeds the maximum allowed request size of 1 023 bytes (see 6.22.6.3.4). SMP response frames defined in SAS-1.1 do not have more than 56 additional response bytes.

The management application client should ignore any additional response bytes beyond those that it expects (e.g., if the management application client complies with a version of this standard defining 24 additional response bytes, but receives a response frame containing 36 additional response bytes, then it should ignore the last 12 additional response bytes).

For additional response bytes containing a DESCRIPTOR LENGTH field and a descriptor list, the management application client should ignore any bytes in each descriptor beyond those that it expects (e.g., if the management application client complies with a version of this standard defining that a descriptor has 24 bytes, but receives a response frame containing a descriptor list with 36 byte descriptors, then it should ignore the last 12 bytes of each descriptor).

#### 9.4.3.3.7 CRC field

The CRC field is defined by the SMP transport layer (see 8.4.1) and parsed by the SMP link layer state machines (see 6.22.6).

#### 9.4.3.4 REPORT GENERAL function

The REPORT GENERAL function returns general information about the SAS device (e.g., a SAS device contained in an expander device). This SMP function shall be implemented by all management device servers.

Table 296 defines the REPORT GENERAL request format.

**Table 296 – REPORT GENERAL request**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (40h)							
1	FUNCTION (00h)							
2	ALLOCATED RESPONSE LENGTH							
3	REQUEST LENGTH (00h)							
4	(MSB)							
...	CRC							
7	(LSB)							

The SMP FRAME TYPE field is defined in 9.4.3.2.2 and shall be set as shown in table 296 for the REPORT GENERAL request.

The FUNCTION field is defined in 9.4.3.2.3 and shall be set as shown in table 296 for the REPORT GENERAL request.

The ALLOCATED RESPONSE LENGTH field is defined in 9.4.3.2.4.

If the ALLOCATED RESPONSE LENGTH field is set to 00h, then the management device server shall:

- set the RESPONSE LENGTH field to 00h in the response frame; and
- return the first 28 bytes defined in table 297 plus the CRC field as the response frame.

If the ALLOCATED RESPONSE LENGTH field is not set to 00h, then the management device server shall:

- set the RESPONSE LENGTH field in the response frame to the non-zero value defined in table 297; and
- return the response frame as defined in 9.4.3.2.4.



The REQUEST LENGTH field is defined in 9.4.3.2.5 and shall be set as shown in table 296 for the REPORT GENERAL request.

The CRC field is defined in 9.4.3.2.7.

Table 297 defines the response format.

**Table 297 – REPORT GENERAL response (part 1 of 3)**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (41h)							
1	FUNCTION (00h)							
2	FUNCTION RESULT							
3	RESPONSE LENGTH (00h or 11h)							
4	(MSB) EXPANDER CHANGE COUNT (LSB)							
5								
6	(MSB) EXPANDER ROUTE INDEXES (LSB)							
7								
8	LONG RESPONSE	Reserved						
9	NUMBER OF PHYS							
10	TABLE TO TABLE SUPPORTED	ZONE CONFIGURING	SELF CONFIGURING	STP CONTINUE AWT	OPEN REJECT RETRY SUPPORTED	CONFIGURES OTHERS	CONFIGURING	EXTERNALLY CONFIGURABLE ROUTE TABLE
11	Reserved						EXTENDED FAIRNESS	INITIATES SSP CLOSE
12	ENCLOSURE LOGICAL IDENTIFIER							
...								
19								
20	Reserved							
...								
27								
28	(MSB) SSP CONNECT TIME LIMIT (LSB)							
29								
30	(MSB) STP BUS INACTIVITY LIMIT (LSB)							
31								

Table 297 – REPORT GENERAL response (part 2 of 3)

Byte\Bit	7	6	5	4	3	2	1	0
32	(MSB)							
	STP CONNECT TIME LIMIT							
33	(LSB)							
34	(MSB)							
	STP SMP I _T NEXUS LOSS TIME							
35	(LSB)							
36	NUMBER OF ZONE GROUPS	Reserved	ZONE LOCKED	PHYSICAL PRESENCE SUPPORTED	PHYSICAL PRESENCE ASSERTED	ZONING SUPPORTED	ZONING ENABLED	
37	Reserved		SAVING	SAVING ZONE MANAGER PASSWORD SUPPORTED	SAVING ZONE PHY INFORMATION SUPPORTED	SAVING ZONE PERMISSION TABLE SUPPORTED	SAVING ZONING ENABLED SUPPORTED	
38	(MSB)							
	MAXIMUM NUMBER OF ROUTED SAS ADDRESSES							
39	(LSB)							
40	ACTIVE ZONE MANAGER SAS ADDRESS							
...								
47								
48	(MSB)							
	ZONE LOCK INACTIVITY TIME LIMIT							
49	(LSB)							
50	Reserved							
51								
52	POWER DONE TIMEOUT							
53	FIRST ENCLOSURE CONNECTOR ELEMENT INDEX							
54	NUMBER OF ENCLOSURE CONNECTOR ELEMENT INDEXES							
55	INITIAL TIME TO DELAY EXPANDER FORWARD OPEN INDICATION							
56	REDUCED FUNCTIONALITY	Reserved						
57	TIME TO REDUCED FUNCTIONALITY							
58	INITIAL TIME TO REDUCED FUNCTIONALITY							
59	MAXIMUM REDUCED FUNCTIONALITY TIME							
60	(MSB)							
	LAST SELF-CONFIGURATION STATUS DESCRIPTOR INDEX							
61	(LSB)							

**Table 297 – REPORT GENERAL response (part 3 of 3)**

Byte\Bit	7	6	5	4	3	2	1	0
62	(MSB)							
63	MAXIMUM NUMBER OF STORED SELF-CONFIGURATION STATUS DESCRIPTORS							(LSB)
64	(MSB)							
65	LAST PHY EVENT LIST DESCRIPTOR INDEX							(LSB)
66	(MSB)							
67	MAXIMUM NUMBER OF STORED PHY EVENT LIST DESCRIPTORS							(LSB)
68	(MSB)							
69	STP REJECT TO OPEN LIMIT							(LSB)
70	Reserved							
71								
72	(MSB)							
...	CRC							
75								(LSB)

The SMP FRAME TYPE field is defined in 9.4.3.3.2 and shall be set as shown in table 297 for the REPORT GENERAL response.

The FUNCTION field is defined in 9.4.3.3.3 and shall be set as shown in table 297 for the REPORT GENERAL response.

The FUNCTION RESULT field is defined in 9.4.3.3.4.

The RESPONSE LENGTH field is defined in 9.4.3.3.5 and shall be set to one of the values defined in table 297 based on the ALLOCATED RESPONSE LENGTH field in the request frame.

The EXPANDER CHANGE COUNT field indicates the number of Broadcast (Change)s originated by an expander device (see 6.15). Management device servers in expander devices shall support this field. Management device servers in other SAS device types (e.g., end devices) shall set this field to 0000h. This field shall be set to at least 0001h at power on. If the expander device has originated Broadcast (Change) for any reason described in 6.15 since transmitting any SMP response frame containing an EXPANDER CHANGE COUNT field, then it:

- a) shall increment this field at least once from the value in the previous REPORT GENERAL response; and
- b) shall not increment this field when forwarding a Broadcast (Change).

This field shall wrap to at least 0001h after reaching the maximum value (i.e., FFFFh).

NOTE 70 - If a management application client uses the EXPANDER CHANGE COUNT field, then reading that field often ensures that the field does not increment a multiple of 65 535 times between reading the field in an expander device compliant with this standard or a multiple of 65 536 times between reading the field in an expander device compliant with SAS-1.1.

NOTE 71 - Management device servers in expander devices compliant with SAS-1.1 wrap the EXPANDER CHANGE COUNT field to 0000h.

NOTE 72 - The originated Broadcast (Change) count is also reported in the REPORT BROADCAST response (see 9.4.3.9).

The EXPANDER ROUTE INDEXES field indicates the maximum number of expander route indexes per phy for the expander device (see 4.5.7.4). Management device servers in externally configurable expander devices containing phy-based expander route tables shall support this field. Management device servers in other SAS device types (e.g., end devices, externally configurable expander devices with expander-based expander route tables, and self-configuring expander devices) shall set the EXPANDER ROUTE INDEXES field to 0000h. Not all phys in an externally configurable expander device are required to support the maximum number indicated by this field.

A LONG RESPONSE bit set to one indicates that the management device server supports returning non-zero values in the RESPONSE LENGTH field of the response frame for any SMP function if the ALLOCATED RESPONSE LENGTH field in the request frame for that SMP function is set to a non-zero value. The LONG RESPONSE bit shall be set to one.

NOTE 73 - SMP target devices compliant with SAS-1.1 set the LONG RESPONSE bit to zero in the REPORT GENERAL response and set the RESPONSE LENGTH field to 00h in all SMP response frames.

The NUMBER OF PHYS field indicates the number of phys in the SAS device type, including any virtual phys and any vacant phys.

A TABLE TO TABLE SUPPORTED bit set to one indicates that the expander device is a self-configuring expander device that supports its table routing phys being attached to table routing phys in other expander devices (i.e., table-to-table attachment). The TABLE TO TABLE SUPPORTED bit shall only be set to one if the EXTERNALLY CONFIGURABLE ROUTE TABLE bit is set to zero. A TABLE TO TABLE SUPPORTED bit set to zero indicates that the expander device is not a self-configuring expander device that supports its table routing phys being attached to table routing phys in other expander devices.

A ZONE CONFIGURING bit set to one indicates that the zoning expander device is locked and the zoning expander shadow values differ from the zoning expander current values. A ZONE CONFIGURING bit set to zero indicates that is not true. Management device servers in zoning expander devices shall support this bit. Management device servers in non-zoning expander devices and in other SAS device types shall set this bit to zero.

A SELF CONFIGURING bit set to one indicates that the management device server is in a self-configuring expander device, the self-configuring expander device's management application client is currently performing the discover process (see 4.6), and that management application client has identified at least one change to its expander routing table. Management device servers in self-configuring expander devices shall support this bit. Management device servers in externally configurable expander devices and in other SAS device types shall set this bit to zero.

An STP CONTINUE AWT bit set to one specifies that the STP port shall not stop the Arbitration Wait Time timer and shall not set the Arbitration Wait Time timer to zero when the STP port receives an OPEN\_REJECT (RETRY). An STP CONTINUE AWT bit set to zero specifies that the STP port shall stop the Arbitration Wait Time timer and shall set the Arbitration Wait Time timer to zero when the STP port receives an OPEN\_REJECT (RETRY).

An OPEN REJECT RETRY SUPPORTED bit set to one indicates that the expander device returns OPEN\_REJECT (RETRY) for any connection requests that detects a condition that results in OPEN\_REJECT (NO DESTINATION) while the SELF CONFIGURING bit is set to one (see 4.6.4) or the ZONE CONFIGURING bit is set to one (see 4.8.6.3). An OPEN REJECT RETRY SUPPORTED bit set to zero indicates that the expander device complies with SAS-1.1 (i.e., it returns OPEN\_REJECT (NO DESTINATION) while the CONFIGURING bit is set to one). Self-configuring expander devices compliant with this standard shall set the OPEN REJECT RETRY SUPPORTED bit to one.

A CONFIGURES OTHERS bit set to one indicates that the expander device is a self-configuring expander device that performs the configuration subprocess defined in 4.7. A CONFIGURES OTHERS bit set to zero indicates that the expander device may or may not perform the configuration subprocess. Self-configuring expander devices compliant with this standard shall set the CONFIGURES OTHERS bit to one.

The CONFIGURING bit indicates the logical OR of the ZONE CONFIGURING bit and the SELF CONFIGURING bit. Changes in this bit from one to zero result in a Broadcast (Change) being originated (see 6.15). Management device servers that support the ZONE CONFIGURING bit or the SELF CONFIGURING bit shall support this bit.

An EXTERNALLY CONFIGURABLE ROUTE TABLE bit set to one indicates that the management device server is in an externally configurable expander device that has a phy-based expander route table that is required to be configured with the SMP CONFIGURE ROUTE INFORMATION function (see 4.5.7.4). An EXTERNALLY CONFIGURABLE ROUTE TABLE bit set to zero indicates that the management device server is not in an externally configurable expander device (e.g., the management device server is in an end device, in a self-configuring expander device, or in an expander device with no phys with table routing attributes).

The EXTENDED FAIRNESS bit set to one indicates that the expander device supports CLOSE primitive parameters (see 6.2.6.5.2) and the Delay Expander Forward Open Indication timer. An EXTENDED FAIRNESS bit set to zero indicates that the expander device ignores CLOSE primitive parameters.

The INITIATES SSP CLOSE bit set to one indicates that the expander device is capable of initiating the closing of SSP connections (see 6.16.9). An INITIATES SSP CLOSE bit set to zero indicates that the expander device is not capable of initiating closing of SSP connections.

The ENCLOSURE LOGICAL IDENTIFIER field identifies the enclosure, if any, in which the SMP target device is located, and is defined in SES-3. The ENCLOSURE LOGICAL IDENTIFIER field shall be set to the same value reported by the enclosure services process, if any, for the enclosure. An ENCLOSURE LOGICAL IDENTIFIER field set to 00000000 00000000h indicates no enclosure information is available.

The SSP CONNECT TIME LIMIT field indicates the maximum connect time limit for expander device SSP connections. The maximum time limit is specified by the CONFIGURE GENERAL function (see 9.4.3.18).

The STP BUS INACTIVITY LIMIT field indicates the bus inactivity time limit for STP connections. The STP bus inactivity limit is specified by the CONFIGURE GENERAL function (see 9.4.3.18).

The STP CONNECT TIME LIMIT field indicates the maximum connect time limit for STP connections. The STP maximum connect time limit is specified by the CONFIGURE GENERAL function (see 9.4.3.18).

The STP SMP I\_T NEXUS LOSS TIME field indicates the minimum time that an STP target port and an SMP initiator port retry certain connection requests. The STP SMP I\_T nexus loss time timer is specified by the CONFIGURE GENERAL function (see 9.4.3.18).

The NUMBER OF ZONE GROUPS field indicates the number of zone groups (e.g., the number of entries in the zone group permission table) supported by the expander device and is defined in table 298.

**Table 298 – NUMBER OF ZONE GROUPS field**

Code	Description
00b	128 zone groups
01b	256 zone groups
All others	Reserved

A ZONE LOCKED bit set to one indicates that the zoning expander device is locked (see 4.8.6.2). A ZONE LOCKED bit set to zero indicates that the zoning expander device is not locked.

A PHYSICAL PRESENCE SUPPORTED bit set to one indicates that the expander device supports physical presence as a mechanism for allowing locking from phys in zone groups without access to zone group 2. A PHYSICAL PRESENCE SUPPORTED bit set to zero indicates that the expander device does not support physical presence as a mechanism for allowing locking.

A PHYSICAL PRESENCE ASSERTED bit set to one indicates that the expander device is currently detecting physical presence. A PHYSICAL PRESENCE ASSERTED bit set to zero indicates that the expander device is not currently detecting physical presence. The PHYSICAL PRESENCE ASSERTED bit shall be set to zero if the PHYSICAL PRESENCE SUPPORTED bit is set to zero.

A ZONING SUPPORTED bit set to one indicates that zoning is supported by the expander device (i.e., it is a zoning expander device). A ZONING SUPPORTED bit set to zero indicates that zoning is not supported by the expander device.

A ZONING ENABLED bit set to one indicates that zoning is enabled in the expander device. A ZONING ENABLED bit set to zero indicates that zoning is disabled in the expander device. The ZONING ENABLED bit shall be set to zero if the ZONING SUPPORTED bit is set to zero.

A SAVING bit set to one indicates that the management device server is currently saving zoning values to non-volatile storage and may return a function result of BUSY for SMP zone management functions that access saved zoning values. A SAVING bit set to zero indicates that the management device server is not currently saving zoning values to non-volatile storage.

A SAVING ZONE MANAGER PASSWORD SUPPORTED bit set to one indicates that saving the zone manager password is supported. A SAVING ZONE MANAGER PASSWORD SUPPORTED bit set to zero indicates that saving the zone manager password is not supported.

A SAVING ZONE PHY INFORMATION SUPPORTED bit set to one indicates that saving the zone phy information is supported. A SAVING ZONE PHY INFORMATION SUPPORTED bit set to zero indicates that saving the zone phy information is not supported.

A SAVING ZONE PERMISSION TABLE SUPPORTED bit set to one indicates that saving the zone permission table is supported. A SAVING ZONE PERMISSION TABLE SUPPORTED bit set to zero indicates that saving the zone permission table is not supported.

A SAVING ZONING ENABLED SUPPORTED bit set to one indicates that saving the ZONING ENABLED bit is supported. A SAVING ZONING ENABLED SUPPORTED bit set to zero indicates that saving the ZONING ENABLED bit is not supported.

The MAXIMUM NUMBER OF ROUTED SAS ADDRESSES field indicates the number of routed SAS addresses in an expander-based expander route table (see 4.5.7.4 and 4.8.3.4). Management device servers in expander devices containing expander-based expander route tables shall support this field. Management device servers in other SAS device types (e.g., end devices and expander devices with phy-based expander route tables) shall set this field to 0000h.

The ACTIVE ZONE MANAGER SAS ADDRESS field indicates the SAS address (see 4.2.4) of the zone manager that last locked the zoning expander device. If the zoning expander device is currently being configured by a vendor specific sideband method, then the ACTIVE ZONE MANAGER SAS ADDRESS field shall be set to 00000000 00000000h. This field shall be set to 00000000 00000000h at power on.

The ZONE LOCK INACTIVITY TIME LIMIT field indicates the minimum time between any SMP ZONE LOCK requests, SMP zone configuration function requests, or SMP ZONE ACTIVATE requests from the active zone manager that the locked expander device allows and is set in the SMP ZONE LOCK request (see 9.4.3.21).

The POWER DONE TIMEOUT field indicates the maximum time the management application layer allows a power consumer device (see 6.14.2) to consume additional power. The power done timeout is specified by the CONFIGURE GENERAL function (see 9.4.3.18). A POWER DONE TIMEOUT field set to 00h or FFh indicates that the maximum time is vendor specific.

The FIRST ENCLOSURE CONNECTOR ELEMENT INDEX field indicates the lowest CONNECTOR ELEMENT INDEX field of all the expander phys in all the expander devices in the enclosure that indicate an internal connector to an end device (see the SAS Connector element in SES-3) in their SMP DISCOVER responses.

The NUMBER OF ENCLOSURE CONNECTOR ELEMENT INDEXES field indicates the number of expander phys in all the expander devices in the enclosure that indicate an internal connector to an end device (see the SAS Connector element in SES-3) in their SMP DISCOVER responses.

NOTE 74 - The NUMBER OF ENCLOSURE CONNECTOR ELEMENT INDEXES field assumes that all internal connectors to end devices are assigned to a contiguous range of CONNECTOR ELEMENT INDEX field values.

The INITIAL TIME TO DELAY EXPANDER FORWARD OPEN INDICATION field indicates the time, in 100 ns increments, that an expander phy uses, in conjunction with the contents of the HOP COUNT field (see 6.2.6.5.3) to determine the time to wait before requesting the ECM assign path resources to a connection. The expander device should set the default value for the INITIAL TIME TO DELAY EXPANDER FORWARD OPEN INDICATION field to at least 500 ns (i.e., 05h). The length of time the expander phy waits is determined from the following calculation:

$$\text{delay in assigning resources} = 100 \text{ ns} \times (\text{initial delay}) \times (\text{hop count})$$

where:

delay in assigning resources	is the number of nanoseconds the phy delays before allowing the ECM assign path resources to a connection;
initial delay	is the contents of the INITIAL TIME TO DELAY EXPANDER FORWARD OPEN INDICATION field; and
hop count	is the contents of the HOP COUNT field.

A REDUCED FUNCTIONALITY bit set to one indicates that:

- a) the expander device is scheduled to reduce its functionality (see 4.5.8) in the time indicated in the TIME TO REDUCED FUNCTIONALITY field; or
- b) the expander device is currently operating with reduced functionality (see 4.5.8).

A REDUCED FUNCTIONALITY bit set to zero indicates that the expander device is not scheduled to reduce functionality and that the contents of the TIME TO REDUCED FUNCTIONALITY field shall be ignored.

If the REDUCED FUNCTIONALITY bit is set to one, then the TIME TO REDUCED FUNCTIONALITY field indicates the time, in 100 ms increments, remaining until the expander device is scheduled to reduce functionality. The expander device starts the reduced functionality delay timer after originating a Broadcast (Expander) (see 4.5.8).

The INITIAL TIME TO REDUCED FUNCTIONALITY field indicates the minimum time, in 100 ms increments, that an expander device waits from originating a Broadcast (Expander) to reducing functionality. The expander device should set the default value for the INITIAL TIME TO REDUCED FUNCTIONALITY field to at least 2 000 ms (i.e., 14h).

The MAXIMUM REDUCED FUNCTIONALITY TIME field indicates the maximum time, in one second increments, that the expander device responds with OPEN\_REJECT (RETRY) to connection requests that map to an expander phy or an SMP target port that is not accessible during expander device reduced functionality. This timer starts after the reduced functionality delay timer expires.

The LAST SELF-CONFIGURATION STATUS DESCRIPTOR INDEX field is defined in the REPORT SELF-CONFIGURATION STATUS response (see 9.4.3.6).

The MAXIMUM NUMBER OF STORED SELF-CONFIGURATION STATUS DESCRIPTORS field indicates the maximum number of self-configuration status descriptors (see 9.4.3.6.4) that the management device server supports.

The LAST PHY EVENT LIST DESCRIPTOR INDEX field is defined in the REPORT PHY EVENT LIST response (see 9.4.3.16).

The MAXIMUM NUMBER OF STORED PHY EVENT LIST DESCRIPTORS field indicates the maximum number of phy event list descriptors (see 9.4.3.14.4) that the management device server supports.

The STP REJECT TO OPEN LIMIT field indicates the minimum time, in 10  $\mu$ s increments, that an STP port waits to establish a connection request with an initiator port on an I\_T nexus after receiving an OPEN\_REJECT (RETRY), OPEN\_REJECT (RESERVED CONTINUE 0), or OPEN\_REJECT (RESERVED CONTINUE 1). An STP REJECT TO OPEN LIMIT field set to 0000h indicates that the time limit is vendor specific.

The CRC field is defined in 9.4.3.3.7.

#### 9.4.3.5 REPORT MANUFACTURER INFORMATION function

The REPORT MANUFACTURER INFORMATION function returns vendor and product identification. This SMP function may be implemented by any management device server.

Table 299 defines the request format.

**Table 299 – REPORT MANUFACTURER INFORMATION request**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (40h)							
1	FUNCTION (01h)							
2	ALLOCATED RESPONSE LENGTH							
3	REQUEST LENGTH (00h)							
4	(MSB)							
...	CRC							
7	(LSB)							

The SMP FRAME TYPE field is defined in 9.4.3.2.2 and shall be set as shown in table 299 for the REPORT MANUFACTURER INFORMATION request.

The FUNCTION field is defined in 9.4.3.2.3 and shall be set as shown in table 299 for the REPORT MANUFACTURER INFORMATION request.

The ALLOCATED RESPONSE LENGTH field is defined in 9.4.3.2.4.

If the ALLOCATED RESPONSE LENGTH field is set to 00h, then the management device server shall:

- a) set the RESPONSE LENGTH field to 00h in the response frame; and
- b) return the first 60 bytes defined in table 300 plus the CRC field as the response frame.

If the ALLOCATED RESPONSE LENGTH field is not set to 00h, then the management device server shall:

- a) set the RESPONSE LENGTH field in the response frame to the non-zero value defined in table 300; and
- b) return the response frame as defined in 9.4.3.2.4.

The REQUEST LENGTH field is defined in 9.4.3.2.5 and shall be set as shown in table 299 for the REPORT MANUFACTURER INFORMATION request.

The CRC field is defined in 9.4.3.2.7.



Table 300 defines the response format.

**Table 300 – REPORT MANUFACTURER INFORMATION response**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (41h)							
1	FUNCTION (01h)							
2	FUNCTION RESULT							
3	RESPONSE LENGTH (00h or 0Eh)							
4	(MSB)	EXPANDER CHANGE COUNT						(LSB)
5								
6	Reserved							
7								
8	Reserved							SAS-1.1 FORMAT
9	Reserved							
...								
11								
12	(MSB)	VENDOR IDENTIFICATION						(LSB)
...								
19								
20	(MSB)	PRODUCT IDENTIFICATION						(LSB)
...								
35								
36	(MSB)	PRODUCT REVISION LEVEL						(LSB)
...								
39								
40	(MSB)	COMPONENT VENDOR IDENTIFICATION						(LSB)
...								
47								
48	(MSB)	COMPONENT ID						(LSB)
49								
50	COMPONENT REVISION LEVEL							
51	Reserved							
52	Vendor specific							
...								
59								
60	(MSB)	CRC						(LSB)
...								
63								

The SMP FRAME TYPE field is defined in 9.4.3.3.2 and shall be set as shown in table 300 for the REPORT MANUFACTURER INFORMATION response.

The FUNCTION field is defined in 9.4.3.3.3 and shall be set as shown in table 300 for the REPORT MANUFACTURER INFORMATION response.

The FUNCTION RESULT field is defined in 9.4.3.3.4.

The RESPONSE LENGTH field is defined in 9.4.3.3.5 and shall be set to one of the values defined in table 300 based on the ALLOCATED RESPONSE LENGTH field in the request frame.

The EXPANDER CHANGE COUNT field is defined in the SMP REPORT GENERAL response (see 9.4.3.4).

A SAS-1.1 FORMAT bit set to one indicates that bytes 40 to 59 are as defined in this standard. A SAS-1.1 FORMAT bit set to zero indicates that bytes 40 to 59 are vendor specific as defined in the original version of this standard.

ASCII data fields (e.g., the VENDOR IDENTIFICATION field, the PRODUCT IDENTIFICATION field, the PRODUCT REVISION LEVEL field, and the COMPONENT VENDOR IDENTIFICATION field) shall contain only graphic codes (i.e., code values 20h to 7Eh). Left-aligned fields shall place any unused bytes at the end of the field (i.e., at the highest offset) and the unused bytes shall be filled with space characters (i.e., 20h).

The VENDOR IDENTIFICATION field contains eight bytes of ASCII data identifying the vendor of the subsystem (e.g., the board or enclosure) containing the component. The data shall be left-aligned within the field. The vendor identification string shall be one assigned by INCITS for use in the standard INQUIRY data VENDOR IDENTIFICATION field. A list of assigned vendor identification strings is in SPC-4 and on the T10 web site (see <http://www.t10.org>).

The PRODUCT IDENTIFICATION field contains 16 bytes of ASCII data identifying the type of the subsystem (e.g., the board or enclosure model number) containing the component, as defined by the vendor of the subsystem. The data shall be left-aligned within the field. The PRODUCT IDENTIFICATION field should be changed whenever the subsystem design changes in a way noticeable to a user (e.g., a different stock-keeping unit (SKU)).

The PRODUCT REVISION LEVEL field contains four bytes of ASCII data identifying the revision level of the subsystem (e.g., the board or enclosure) containing the component, as defined by the vendor of the subsystem. The data shall be left-aligned within the field. The PRODUCT REVISION LEVEL field should be changed whenever the subsystem design changes (e.g., any component change, even including resistor values).

All components on a subsystem should have the same values for their VENDOR IDENTIFICATION fields, PRODUCT IDENTIFICATION fields, and PRODUCT REVISION LEVEL fields.

NOTE 75 - A use of the VENDOR IDENTIFICATION field and PRODUCT IDENTIFICATION field is for the management application client to identify the subsystem (e.g., for a user interface). Another use of the VENDOR IDENTIFICATION field, PRODUCT IDENTIFICATION field, and PRODUCT REVISION LEVEL field is for a management application client to perform workarounds for problems in a specific revision of a subsystem.

The COMPONENT VENDOR IDENTIFICATION field contains eight bytes of ASCII data identifying the vendor of the component (e.g., the expander device) containing the management device server. The data shall be left-aligned within the field. The component vendor identification string shall be one assigned by INCITS for use in the standard INQUIRY data VENDOR IDENTIFICATION field. A list of assigned vendor identification strings is in SPC-4 and on the T10 web site (see <http://www.t10.org>).

The COMPONENT ID field contains a 16-bit identifier identifying the type of the component (e.g., the expander device model number) containing the management device server, as defined by the vendor of the component. The COMPONENT ID field should be changed whenever the component's programming interface (e.g., the management device server definition) changes.

The COMPONENT REVISION LEVEL field contains an 8-bit identifier identifying the revision level of the component (e.g., the expander device) containing the management device server, as defined by the vendor of the component. The COMPONENT REVISION LEVEL field should be changed whenever the component changes but its programming interface does not change.

NOTE 76 - A use of the COMPONENT VENDOR IDENTIFICATION field and the COMPONENT ID field is for the management application client to interpret vendor specific information (e.g., vendor specific SMP functions) correctly for that component. Another use of the COMPONENT VENDOR IDENTIFICATION field, the COMPONENT ID field, and the COMPONENT REVISION LEVEL field is for the management application client to perform workarounds for problems in a specific revision of a component.

The vendor specific bytes are defined by the vendor of the subsystem (e.g., the board or enclosure) containing the component.

The CRC field is defined in 9.4.3.3.7.

#### 9.4.3.6 REPORT SELF-CONFIGURATION STATUS function

##### 9.4.3.6.1 REPORT SELF-CONFIGURATION STATUS function overview

The REPORT SELF-CONFIGURATION STATUS function returns self-configuration expander device status. This SMP function shall be implemented by the management device server in self-configuring expander devices and shall not be implemented by any other management device servers.

##### 9.4.3.6.2 REPORT SELF-CONFIGURATION STATUS request

Table 301 defines the request format.

**Table 301 – REPORT SELF-CONFIGURATION STATUS request**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (40h)							
1	FUNCTION (03h)							
2	ALLOCATED RESPONSE LENGTH							
3	REQUEST LENGTH (01h)							
4	Reserved							
5								
6	(MSB)	STARTING SELF-CONFIGURATION STATUS DESCRIPTOR INDEX						
7								(LSB)
8	(MSB)	CRC						
...								
11								(LSB)

The SMP FRAME TYPE field is defined in 9.4.3.2.2 and shall be set as shown in table 301 for the REPORT SELF-CONFIGURATION STATUS request.

The FUNCTION field is defined in 9.4.3.2.3 and shall be set as shown in table 301 for the REPORT SELF-CONFIGURATION STATUS request.

The ALLOCATED RESPONSE LENGTH field is defined in 9.4.3.2.4.

The REQUEST LENGTH field is defined in 9.4.3.2.5 and shall be set as shown in table 301 for the REPORT SELF-CONFIGURATION STATUS request.

The STARTING SELF-CONFIGURATION STATUS DESCRIPTOR INDEX field specifies the first self-configuration status descriptor that the management device server shall return in the SMP response frame. If the specified index does not contain a valid self-configuration status descriptor, then the STARTING SELF-CONFIGURATION STATUS DESCRIPTOR INDEX field in the response may differ from the specified index. A STARTING SELF-CONFIGURATION STATUS DESCRIPTOR INDEX field set to 0000h specifies that the management device server shall return no self-configuration status descriptors.

The CRC field is defined in 9.4.3.2.7.

## 9.4.3.6.3 REPORT SELF-CONFIGURATION STATUS response

Table 302 defines the response format.

Table 302 – REPORT SELF-CONFIGURATION STATUS response

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (41h)							
1	FUNCTION (03h)							
2	FUNCTION RESULT							
3	RESPONSE LENGTH ((n - 7) / 4)							
4	(MSB)	EXPANDER CHANGE COUNT						
5								(LSB)
6	(MSB)	STARTING SELF-CONFIGURATION STATUS DESCRIPTOR INDEX						
7								(LSB)
8	(MSB)	TOTAL NUMBER OF SELF-CONFIGURATION STATUS DESCRIPTORS						
9								(LSB)
10	(MSB)	LAST SELF-CONFIGURATION STATUS DESCRIPTOR INDEX						
11								(LSB)
12	SELF-CONFIGURATION STATUS DESCRIPTOR LENGTH							
13	Reserved							
...								
18								
19	NUMBER OF SELF-CONFIGURATION STATUS DESCRIPTORS							
Self-configuration status descriptor list								
20	Self-configuration status descriptor (first) (see table 303 in 9.4.3.6.4)							
...								
...	...							
	Self-configuration status descriptor (last) (see table 303 in 9.4.3.6.4)							
...								
n - 4								
n - 3	(MSB)	CRC						
...								
n								(LSB)

The SMP FRAME TYPE field is defined in 9.4.3.3.2 and shall be set as shown in table 302 for the REPORT SELF-CONFIGURATION STATUS response.

The FUNCTION field is defined in 9.4.3.3.3 and shall be set as shown in table 302 for the REPORT SELF-CONFIGURATION STATUS response.

The FUNCTION RESULT field is defined in 9.4.3.3.4.

The RESPONSE LENGTH field is defined in 9.4.3.3.5 and shall be set as shown in table 302 for the REPORT SELF-CONFIGURATION STATUS response. A RESPONSE LENGTH field set to 00h does not have a special meaning based on the ALLOCATED RESPONSE LENGTH field in the request frame.

The EXPANDER CHANGE COUNT field is defined in the SMP REPORT GENERAL response (see 9.4.3.4). If the management application client detects a change in the value of this field while retrieving multiple response frames, then it should retrieve the response frames again.

The STARTING SELF-CONFIGURATION STATUS DESCRIPTOR INDEX field indicates the index of the first self-configuration status descriptor being returned. If the STARTING SELF-CONFIGURATION STATUS DESCRIPTOR INDEX field in the SMP request frame is set to 0000h, then the management device server shall:

- a) set the STARTING SELF-CONFIGURATION STATUS DESCRIPTOR INDEX field to 0000h;
- b) set the TOTAL NUMBER OF SELF-CONFIGURATION STATUS DESCRIPTORS field to 0000h; and
- c) return no descriptors.

If the STARTING SELF-CONFIGURATION STATUS DESCRIPTOR INDEX field in the SMP request frame does not specify a valid descriptor, then the management device server shall:

- a) set the STARTING SELF-CONFIGURATION STATUS DESCRIPTOR INDEX field to the next index, in ascending order wrapping from FFFFh to 0001h, that contains a valid descriptor.

If the STARTING SELF-CONFIGURATION STATUS DESCRIPTOR INDEX field is not set to 0000h and specifies a valid descriptor, then this field shall be set to the same value as the STARTING SELF-CONFIGURATION STATUS DESCRIPTOR INDEX field in the SMP request frame.

The TOTAL NUMBER OF SELF-CONFIGURATION STATUS DESCRIPTORS field indicates the number of self-configuration status descriptors are available at this time from the management device server.

The LAST SELF-CONFIGURATION STATUS DESCRIPTOR INDEX field indicates the index of the last recorded self-configuration status descriptor.

The SELF-CONFIGURATION STATUS DESCRIPTOR LENGTH field indicates the length, in dwords, of the self-configuration status descriptor (see table 303 in 9.4.3.6.4).

The NUMBER OF SELF-CONFIGURATION STATUS DESCRIPTORS field indicates the number of self-configuration status descriptors in the self-configuration status descriptor list.

The self-configuration status descriptor list contains self-configuration status descriptors (see table 303). The management device server shall return either all the self-configuration status descriptors that fit in one SMP response frame or all the self-configuration status descriptors until the index indicated in the LAST SELF-CONFIGURATION STATUS DESCRIPTOR INDEX field is reached. The self-configuration status descriptor list shall start with the self-configuration status descriptor specified by the STARTING SELF-CONFIGURATION STATUS DESCRIPTOR INDEX field and shall continue with self-configuration status descriptors sorted in ascending order, wrapping from FFFFh to 0001h, based on the self-configuration status descriptor index. The self-configuration status descriptor list shall not contain any truncated self-configuration status descriptors. If the STARTING SELF-CONFIGURATION STATUS DESCRIPTOR INDEX field is equal to the LAST SELF-CONFIGURATION STATUS DESCRIPTOR INDEX field, then the self-configuration status descriptor at that index shall be returned.

The CRC field is defined in 9.4.3.3.7.

#### 9.4.3.6.4 Self-configuration status descriptor

Each self-configuration status descriptor follows the format defined in table 303.

**Table 303 – Self-configuration status descriptor**

Byte\Bit	7	6	5	4	3	2	1	0
0	STATUS TYPE							
1	Reserved							FINAL
2	Reserved							
3	PHY IDENTIFIER							
4	Reserved							
...								
7								
8	SAS ADDRESS							
...								
15								

The STATUS TYPE field indicates the type of status being reported and is defined in table 304.

**Table 304 – STATUS TYPE field (part 1 of 3)**

Code	Description
Status not related to specific layers (00h to 0Fh)	
00h	Reserved
01h	Error not related to a specific layer
02h	The expander device currently has a connection or is currently attempting to establish a connection with the SMP target port with the indicated SAS address.
03h	Expander route table is full. The expander device was not able to add the indicated SAS address to the expander route table.
04h	Expander device is out of resources (e.g., it discovered too many SAS addresses while performing the discover process through a subtractive port). This does not affect the expander route table.
05h to 1Fh	Reserved
Status reported by the phy layer (20h to 3Fh)	
20h	Error reported by the phy layer
21h	All phys in the expander port containing the indicated phy lost dword synchronization
22h to 3Fh	Reserved
Status reported by the link layer (40h to 5Fh)	

Table 304 – STATUS TYPE field (part 2 of 3)

Code	Description
40h	Error reported by the link layer
41h	Connection request failed as a result of an Open Timeout timer expiring
42h	Connection request failed as a result of receiving an abandon-class OPEN_REJECT (e.g., BAD DESTINATION, PROTOCOL NOT SUPPORTED, ZONE VIOLATION, STP RESOURCES BUSY, WRONG DESTINATION)
43h	Connection request failed as a result of receiving a vendor specific number of retry-class OPEN_REJECTs (e.g., RETRY, PATHWAY BLOCKED)
44h	Connection request failed as a result of an I_T nexus loss occurring (e.g., OPEN_REJECT (NO DESTINATION) received for longer than the time specified by the STP SMP I_T NEXUS LOSS TIME field in the CONFIGURE GENERAL function)
45h	Connection request failed as a result of receiving a BREAK
46h	Connection established as a result of an SMP response frame having a CRC error
47h to 5Fh	Reserved
Status reported by the port layer (60h to 7Fh)	
60h	Error reported by the port layer
61h	During an SMP connection, there was no SMP response frame within the maximum SMP connection time
62h to 7Fh	Reserved
Status reported by the SMP transport layer (80h to 9Fh)	
80h	Error reported by the SMP transport layer
81h to 9Fh	Reserved
Status reported by the management application layer (A0h to BFh)	
A0h	Error reported by the management application layer
A1h	SMP response frame is too short
A2h	SMP response frame contains fields with unsupported values
A3h	SMP response frame contains results inconsistent with other SMP response frames (e.g., the DISCOVER response ATTACHED SAS ADDRESS field does not contain the SAS address the expander device expected)
A4h	<p>The SAS ADDRESS field contains the SAS address of a self-configuring expander device that returned a REPORT GENERAL response with the CONFIGURING bit set to one, the SELF CONFIGURING bit set to zero, and the ZONE CONFIGURING bit set to zero (e.g., compliant with a previous version of this standard). Accesses to SAS addresses two or more levels beyond this expander device may not succeed until the indicated expander device completes configuration.</p> <p>This may or may not be an error.</p>
A5h	<p>The SAS ADDRESS field contains the SAS address of a self-configuring expander device that returned a REPORT GENERAL response with the SELF CONFIGURING bit set to one. Accesses to SAS addresses two or more levels beyond this expander device may not succeed until the indicated expander device completes configuration.</p> <p>This may or may not be an error.</p>



**Table 304 – STATUS TYPE field (part 3 of 3)**

Code	Description
A6h	The SAS ADDRESS field contains the SAS address of a self-configuring expander device that returned a REPORT GENERAL response with the ZONE CONFIGURING bit set to one. Accesses to SAS addresses two or more levels beyond this expander device may not succeed until the indicated expander device completes configuration.  This may or may not be an error.
A7h to BFh	Reserved
Other status (C0h to FFh)	
C0h to DFh	Reserved
E0h to FFh	Vendor specific

A FINAL bit set to one indicates that the expander device is no longer attempting to establish connections to the SMP target port with the indicated SAS address as part of the discover process because of the error indicated by the descriptor. A FINAL bit set to zero indicates that the expander device is still attempting to access the SMP target port with the indicated SAS address as part of the discover process.

The PHY IDENTIFIER field indicates the phy identifier (see 4.2.10) of the phy that was used to request a connection with the SMP target port with the indicated SAS address.

The SAS ADDRESS field indicates the SAS address (see 4.2.4) of the SMP target port to which the expander device established a connection or attempted to establish a connection.

#### **9.4.3.7 REPORT ZONE PERMISSION TABLE function**

##### **9.4.3.7.1 REPORT ZONE PERMISSION TABLE function overview**

The REPORT ZONE PERMISSION function returns a set of zone permission table entries. This function shall be supported by all zoning expander devices.

**9.4.3.7.2 REPORT ZONE PERMISSION TABLE request**

Table 305 defines the request format.

**Table 305 – REPORT ZONE PERMISSION TABLE request**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (40h)							
1	FUNCTION (04h)							
2	ALLOCATED RESPONSE LENGTH							
3	REQUEST LENGTH (01h)							
4	Reserved						REPORT TYPE	
5	Reserved							
6	STARTING SOURCE ZONE GROUP							
7	MAXIMUM NUMBER OF ZONE PERMISSION DESCRIPTORS							
8	<div>(MSB)<div>CRC</div>(LSB)</div>							
...								
11								

The SMP FRAME TYPE field is defined in 9.4.3.2.2 and shall be set as shown in table 305 for the REPORT ZONE PERMISSION TABLE request.

The FUNCTION field is defined in 9.4.3.2.3 and shall be set as shown in table 305 for the REPORT ZONE PERMISSION TABLE request.

The ALLOCATED RESPONSE LENGTH field is defined in 9.4.3.2.4.

The REQUEST LENGTH field is defined in 9.4.3.2.5 and shall be set as shown in table 305 for the REPORT ZONE PERMISSION TABLE request.

The REPORT TYPE field specifies the zone permission table values that the management device server shall return and is defined in table 306.

**Table 306 – REPORT TYPE field**

Code	Description
00b	Current zone permission table
01b	Shadow zone permission table
10b	Saved zone permission table. If the expander device does not support saving, then it shall return a function result of SAVING NOT SUPPORTED in the response frame (see table 294 in 9.4.3.3).
11b	Default zone permission table

The STARTING SOURCE ZONE GROUP field specifies the first source zone group (i.e., s) returned. If the value in this field exceeds the end of the zone permission table, then the management device server shall return a function result of SOURCE ZONE GROUP DOES NOT EXIST in the response frame (see table 294 in 9.4.3.3).

The MAXIMUM NUMBER OF ZONE PERMISSION DESCRIPTORS field specifies the maximum number of complete zone permission descriptors that the management device server shall return.

The CRC field is defined in 9.4.3.2.7.

**9.4.3.7.3 REPORT ZONE PERMISSION TABLE response**

Table 307 defines the response format.

**Table 307 – REPORT ZONE PERMISSION TABLE response**

ByteBit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (41h)							
1	FUNCTION (04h)							
2	FUNCTION RESULT							
3	RESPONSE LENGTH ((n - 7) / 4)							
4	(MSB) EXPANDER CHANGE COUNT (LSB)							
5								
6	ZONE LOCKED	Reserved					REPORT TYPE	
7	NUMBER OF ZONE GROUPS		Reserved					
8	Reserved							
...								
12								
13	ZONE PERMISSION DESCRIPTOR LENGTH							
14	STARTING SOURCE ZONE GROUP							
15	NUMBER OF ZONE PERMISSION DESCRIPTORS							
Zone permission descriptor list								
16	Zone permission descriptor (first) (see table 309 or table 310 in 9.4.3.7.4)							
...								
31 or 47								
...	...							
(n - 20) or (n - 36)	Zone permission descriptor (last) (see table 309 or table 310 in 9.4.3.7.4)							
...								
n - 4								
n - 3	(MSB)							
...	CRC							
n	(LSB)							

The SMP FRAME TYPE field is defined in 9.4.3.3.2 and shall be set as shown in table 307 for the REPORT ZONE PERMISSION TABLE response.

The FUNCTION field is defined in 9.4.3.3.3 and shall be set as shown in table 307 for the REPORT ZONE PERMISSION TABLE response.

The FUNCTION RESULT field is defined in 9.4.3.3.4.

The RESPONSE LENGTH field is defined in 9.4.3.3.5 and shall be set to one of the values defined in table 307 based on the ALLOCATED RESPONSE LENGTH field in the request frame.

The EXPANDER CHANGE COUNT field is defined in the SMP REPORT GENERAL response (see 9.4.3.4). If the SMP initiator port detects a change in the value of this field while retrieving multiple response frames, then it should retrieve the response frames again because the status information returned is incomplete and inconsistent.

The ZONE LOCKED bit is defined in the SMP REPORT GENERAL response.

The REPORT TYPE field indicates the value of the REPORT TYPE field in the request frame.

The NUMBER OF ZONE GROUPS field indicates the number of zone groups supported by the expander device and is defined in the REPORT GENERAL response (see table 298 in 9.4.3.4).

The ZONE PERMISSION DESCRIPTOR LENGTH field indicates the length, in dwords, of the zone permission descriptor (see 9.4.3.7.4).

The STARTING SOURCE ZONE GROUP field indicates the first source zone group (i.e., s) being returned and shall be set to the same value as the STARTING SOURCE ZONE GROUP field in the SMP request frame.

The NUMBER OF ZONE PERMISSION DESCRIPTORS field indicates the number of zone permission descriptors in the zone permission descriptor list.

The zone permission descriptor list contains a zone permission descriptor as defined in 9.4.3.7.4 for each source zone group in ascending order starting with the source zone group specified in the STARTING SOURCE ZONE GROUP field in the request.

The CRC field is defined in 9.4.3.3.7.

#### 9.4.3.7.4 Zone permission descriptor

The zone permission descriptor format is based on the NUMBER OF ZONE GROUPS field as defined in table 308.

**Table 308 – Zone permission descriptors**

NUMBER OF ZONE GROUPS field	Zone permission descriptor format
00b	Table 309
01b	Table 310
All others	Reserved

Table 309 defines the zone permission descriptor containing 128 zone groups.

**Table 309 – Zone permission descriptor for a source zone group (i.e., s) with 128 zone groups**

Byte\Bit	7	6	5	4	3	2	1	0
0	ZP[s, 127]	ZP[s, 126]	ZP[s, 125]	ZP[s, 124]	ZP[s, 123]	ZP[s, 122]	ZP[s, 121]	ZP[s, 120]
...	...							
15	ZP[s, 7] (0b)	ZP[s, 6] (0b)	ZP[s, 5] (0b)	ZP[s, 4] (0b)	ZP[s, 3]	ZP[s, 2]	ZP[s, 1] (1b)	ZP[s, 0] (0b)

Table 310 defines the zone permission descriptor containing 256 zone groups.

**Table 310 – Zone permission descriptor for a source zone group (i.e., s) with 256 zone groups**

Byte\Bit	7	6	5	4	3	2	1	0
0	ZP[s, 255]	ZP[s, 254]	ZP[s, 253]	ZP[s, 252]	ZP[s, 251]	ZP[s, 250]	ZP[s, 249]	ZP[s, 248]
...	...							
31	ZP[s, 7] (0b)	ZP[s, 6] (0b)	ZP[s, 5] (0b)	ZP[s, 4] (0b)	ZP[s, 3]	ZP[s, 2]	ZP[s, 1] (1b)	ZP[s, 0] (0b)

The zone permission descriptor contains all of the zone permission table entries for the source zone group (i.e., s).

Table 311 defines how the zone permission descriptor bits shall be set by the management device server.

**Table 311 – Zone permission descriptor bit requirements**

Source zone group (i.e., s)	Management device server requirements <sup>a</sup>
0	ZP[s, 0] shall be set to zero. ZP[s, 1] shall be set to one. ZP[s, 2 to (z-1)] shall be set to zero.
1	ZP[s, 0 to (z-1)] shall be set to one.
4, 5, 6, or 7	ZP[s, 0] shall be set to zero. ZP[s, 1] shall be set to one. ZP[s, 4 to (z-1)] shall be set to zero.
2, 3, or 8 to (z-1) <sup>a</sup>	ZP[s, 0] shall be set to zero. ZP[s, 1] shall be set to one. ZP[s, 2 to 3] shall be set to zero or one as specified by the CONFIGURE ZONE PERMISSION TABLE function (see 9.4.3.26). ZP[s, 4 to 7] shall be set to zero. ZP[s, 8 to (z-1)] shall be set to zero or one as specified by the CONFIGURE ZONE PERMISSION TABLE function.
<sup>a</sup> The number of zone groups (i.e., z) is reported in NUMBER OF ZONE GROUPS field.	

#### 9.4.3.8 REPORT ZONE MANAGER PASSWORD function

The REPORT ZONE MANAGER PASSWORD function returns the zone manager password (see 4.8.1). This SMP function may be implemented by a management device server in a zoning expander device and shall be implemented if the management device server supports the CONFIGURE ZONE MANAGER PASSWORD function (see 9.4.3.24). Other management device servers shall not support this SMP function. This function shall only be processed if the request is received from:

- a) an SMP initiator port that has access to zone group 2 (see 4.8.3.2); or
- b) any SMP initiator port while physical presence is asserted.

If physical presence is not asserted and the SMP initiator port does not have access to zone group 2, then the management device server shall return a function result of NO MANAGEMENT ACCESS RIGHTS in the response frame (see table 294 in 9.4.3.3).

Table 312 defines the request format.

**Table 312 – REPORT ZONE MANAGER PASSWORD request**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (40h)							
1	FUNCTION (05h)							
2	ALLOCATED RESPONSE LENGTH							
3	REQUEST LENGTH (01h)							
4	Reserved						REPORT TYPE	
5	Reserved							
...								
7								
8	(MSB)							
...	CRC							
11	(LSB)							

The SMP FRAME TYPE field is defined in 9.4.3.2.2 and shall be set as shown in table 312 for the REPORT ZONE MANAGER PASSWORD request.

The FUNCTION field is defined in 9.4.3.2.3 and shall be set as shown in table 312 for the REPORT ZONE MANAGER PASSWORD request.

The ALLOCATED RESPONSE LENGTH field is defined in 9.4.3.2.4.

The REQUEST LENGTH field is defined in 9.4.3.2.5 and shall be set as shown in table 312 for the REPORT ZONE MANAGER PASSWORD request.

The REPORT TYPE field specifies the zone manager password value that the management device server shall return and is defined in table 313.

**Table 313 – REPORT TYPE field**

Code	Description
00b	Current zone manager password
01b	Reserved <sup>a</sup>
10b	Saved zone manager password. If the expander device does not support saving, then it shall return a function result of SAVING NOT SUPPORTED in the response frame (see table 294 in 9.4.3.3).
11b	Default zone manager password
<sup>a</sup> The CONFIGURE ZONE PASSWORD function updates the current zone manager password, not a shadow zone manager password.	

The CRC field is defined in 9.4.3.2.7.

Table 314 defines the response format.

**Table 314 – REPORT ZONE MANAGER PASSWORD response**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (41h)							
1	FUNCTION (05h)							
2	FUNCTION RESULT							
3	RESPONSE LENGTH (09h)							
4	(MSB)	EXPANDER CHANGE COUNT						
5								(LSB)
6	Reserved						REPORT TYPE	
7	Reserved							
8	ZONE MANAGER PASSWORD							
...								
39								
40	(MSB)	CRC						
...								
43								(LSB)

The SMP FRAME TYPE field is defined in 9.4.3.3.2 and shall be set as shown in table 314 for the REPORT ZONE MANAGER PASSWORD response.



The FUNCTION field is defined in 9.4.3.3.3 and shall be set as shown in table 314 for the REPORT ZONE MANAGER PASSWORD response.

The FUNCTION RESULT field is defined in 9.4.3.3.4.

The RESPONSE LENGTH field is defined in 9.4.3.3.5 and shall be set as shown in table 314 for the REPORT ZONE MANAGER PASSWORD response. A RESPONSE LENGTH field set to 00h does not have a special meaning based on the ALLOCATED RESPONSE LENGTH field in the request frame.

The EXPANDER CHANGE COUNT field is defined in the SMP REPORT GENERAL response (see 9.4.3.4).

The REPORT TYPE field indicates the value of the REPORT TYPE field in the request frame.

The ZONE MANAGER PASSWORD field indicates the zone manager password of the type indicated by the REPORT TYPE field.

The CRC field is defined in 9.4.3.3.7.

### 9.4.3.9 REPORT BROADCAST function

#### 9.4.3.9.1 REPORT BROADCAST function overview

The REPORT BROADCAST function returns information about Broadcasts (see 4.1.15) that were either:

- a) originated from this expander device or SAS device; or
- b) received on a phy directly attached to an end device.

This SMP function may be implemented by any management device server.

#### 9.4.3.9.2 REPORT BROADCAST request

Table 315 defines the request format.

**Table 315 – REPORT BROADCAST request**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (40h)							
1	FUNCTION (06h)							
2	ALLOCATED RESPONSE LENGTH							
3	REQUEST LENGTH (01h)							
4	Reserved				BROADCAST TYPE			
5	Reserved							
...								
7								
8	(MSB)							
...	CRC							
11	(LSB)							

The SMP FRAME TYPE field is defined in 9.4.3.2.2 and shall be set as shown in table 315 for the REPORT BROADCAST request.

The FUNCTION field is defined in 9.4.3.2.3 and shall be set as shown in table 315 for the REPORT BROADCAST request.

The ALLOCATED RESPONSE LENGTH field is defined in 9.4.3.2.4.

The REQUEST LENGTH field is defined in 9.4.3.2.5 and shall be set as shown in table 315 for the REPORT BROADCAST request.

The BROADCAST TYPE field, defined in the ZONED BROADCAST request (see table 363 in 9.4.3.20), specifies the type of Broadcast for which counts shall be returned in the response frame.

The CRC field is defined in 9.4.3.2.7.

**9.4.3.9.3 REPORT BROADCAST response**

Table 316 defines the response format.

**Table 316 – REPORT BROADCAST response**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (41h)							
1	FUNCTION (06h)							
2	FUNCTION RESULT							
3	RESPONSE LENGTH ((n - 7) / 4)							
4	(MSB)	EXPANDER CHANGE COUNT						
5								(LSB)
6	Reserved				BROADCAST TYPE			
7	Reserved							
...								
9								
10	BROADCAST DESCRIPTOR LENGTH							
11	NUMBER OF BROADCAST DESCRIPTORS							
Broadcast descriptor list								
12	Broadcast descriptor (first) (see table 317 in 9.4.3.9.4)							
...								
19								
...	...							
n - 11	Broadcast descriptor (last) (see table 317 in 9.4.3.9.4)							
...								
n - 4								
n - 3	(MSB)	CRC						
...								(LSB)
n								

The SMP FRAME TYPE field is defined in 9.4.3.3.2 and shall be set as shown in table 316 for the REPORT BROADCAST response.

The FUNCTION field is defined in 9.4.3.3.3 and shall be set as shown in table 316 for the REPORT BROADCAST response.

The FUNCTION RESULT field is defined in 9.4.3.3.4.

The RESPONSE LENGTH field is defined in 9.4.3.3.5 and shall be set as shown in table 316 for the REPORT BROADCAST response. A RESPONSE LENGTH field set to 00h does not have a special meaning based on the ALLOCATED RESPONSE LENGTH field in the request frame.

The EXPANDER CHANGE COUNT field is defined in the SMP REPORT GENERAL response (see 9.4.3.4).

The BROADCAST TYPE field indicates the value of the BROADCAST TYPE field in the request frame.

The BROADCAST DESCRIPTOR LENGTH field indicates the length, in dwords, of the Broadcast descriptor (see 9.4.3.9.4).

The NUMBER OF BROADCAST DESCRIPTORS field indicates the number of Broadcast descriptors in the Broadcast descriptor list.

NOTE 77 - If Broadcast descriptors are 8 bytes, then the number of Broadcast descriptors is limited to 126 by the SMP response frame size (see 9.4.3.3.6).

The Broadcast descriptor list contains Broadcast descriptors as defined in 9.4.3.9.4. Broadcast descriptors shall be returned for all Broadcasts of the type specified in the BROADCAST TYPE field for which the count is non-zero. Broadcast descriptors shall be returned with the descriptor, if any, pertaining to no particular phy (i.e., PHY IDENTIFIER field set to FFh) first, followed by descriptors, if any, in ascending order sorted by the PHY IDENTIFIER field in each descriptor.

The CRC field is defined in 9.4.3.3.7.

#### 9.4.3.9.4 Broadcast descriptor

Table 317 defines the Broadcast descriptor.

**Table 317 – Broadcast descriptor**

Byte\Bit	7	6	5	4	3	2	1	0
0	Reserved				BROADCAST TYPE			
1	PHY IDENTIFIER							
2	Reserved				BROADCAST REASON			
3	Reserved							
4	BROADCAST COUNT							
5								
6	Reserved							
7								

The BROADCAST TYPE field, defined in the ZONED BROADCAST request (see table 363 in 9.4.3.20), indicates the type of Broadcast described by this Broadcast descriptor.

The PHY IDENTIFIER field indicates the phy identifier (see 4.2.10) of the phy that caused the Broadcast described by this Broadcast descriptor to be originated or the phy on which the Broadcast was received. A PHY IDENTIFIER field set to FFh indicates that no specific phy caused the Broadcast described by this Broadcast descriptor.

The BROADCAST COUNT field indicates the number of Broadcasts that were either:

- a) originated by the SAS device or expander device; or
- b) received by a phy attached to an end device.

If the SAS device or expander device has originated the Broadcast or received the Broadcast since transmitting a REPORT BROADCAST response, then it shall increment this field at least once from the value in the previous REPORT BROADCAST response. It shall not increment this field when forwarding a Broadcast. This field shall wrap to at least 0001h after the maximum value (i.e., FFFFh) has been reached.

NOTE 78 - If a management application client uses the BROADCAST COUNT field, then reading and saving all the BROADCAST COUNT field values after performing the discover process (see 4.6), this allows the management application client to read them after each receipt of each Broadcast to ensure that none of the counts increments a multiple of 65 535 times between reading them.

For Broadcasts that are received, the BROADCAST REASON field shall be set to Fh. For Broadcasts that are originated, the BROADCAST REASON field indicates the reason that the Broadcast described by this Broadcast descriptor was originated and is defined in table 318.

**Table 318 – BROADCAST REASON field for originated Broadcasts**

BROADCAST TYPE field	BROADCAST REASON field	Description
0h (i.e., Broadcast (Change))	0h	Unspecified <sup>a b</sup>
4h (i.e., Broadcast (Expander))	0h	Unspecified
	1h	A phy event peak value detector has reached its threshold value.
	2h	A phy event peak value detector has been cleared by the SMP CONFIGURE PHY EVENT function (see 9.4.3.30).
	3h	The expander device is going to have reduced functionality (e.g., disable SMP access, reduced performance, disable phy to phy communication) for a period of time (see 4.5.8).
8h (i.e., Broadcast (Zone Activate))	0h	Unspecified
All others		Reserved
<sup>a</sup> In an expander device, the Broadcast (Change) count is also reported in the REPORT GENERAL response (see 9.4.3.4) and in other SMP response frames containing an EXPANDER CHANGE COUNT field. <sup>b</sup> Broadcast (Change)s originated by this expander device or SAS device shall be counted, with the PHY IDENTIFIER field set to FFh.		

### 9.4.3.10 DISCOVER function

The DISCOVER function returns information about the specified phy. This SMP function provides information from the IDENTIFY address frame received by the phy during the last identification sequence and additional phy-specific information. This SMP function shall be implemented by all management device servers.

NOTE 79 - The DISCOVER LIST function (see 9.4.3.15) returns information about one or more phys.

Table 319 defines the request format.

**Table 319 – DISCOVER request**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (40h)							
1	FUNCTION (10h)							
2	ALLOCATED RESPONSE LENGTH							
3	REQUEST LENGTH (00h or 02h)							
4	Reserved							
...								
7								
8	Reserved							IGNORE ZONE GROUP
9	PHY IDENTIFIER							
10	Reserved							
11								
12	(MSB)	CRC						
...								
15		(LSB)						

The SMP FRAME TYPE field is defined in 9.4.3.2.2 and shall be set as shown in table 319 for the DISCOVER request.

The FUNCTION field is defined in 9.4.3.2.3 and shall be set as shown in table 319 for the DISCOVER request.

The ALLOCATED RESPONSE LENGTH field is defined in 9.4.3.2.4.

If the ALLOCATED RESPONSE LENGTH field is set to 00h, then the management device server shall:

- set the RESPONSE LENGTH field to 00h in the response frame; and
- return the first 52 bytes defined in table 320 plus the CRC field as the response frame.

If the ALLOCATED RESPONSE LENGTH field is not set to 00h, then the management device server shall:

- a) set the RESPONSE LENGTH field in the response frame to the non-zero value defined in table 320; and
- b) return the response frame as defined in 9.4.3.2.4.

The REQUEST LENGTH field is defined in 9.4.3.2.5 and shall be set to one of the values defined in table 319 based on the LONG RESPONSE bit in the REPORT GENERAL response (see 9.4.3.4). A REQUEST LENGTH field set to 00h specifies that there are two dwords before the CRC field.

An IGNORE ZONE GROUP bit set to one specifies that the management device server shall return information about the specified phy (i.e., the phy specified by the PHY IDENTIFIER field) regardless of the zone permission table.

An IGNORE ZONE GROUP bit set to zero specifies that the management device server shall if the SMP initiator port:

- a) has access to the specified phy based on the zone permission table, then return the requested information; or
- b) does not have access to the specified phy, then return a function result of PHY VACANT in the response frame (see table 294 in 9.4.3.3).

If the management device server is not in a zoning expander device with zoning enabled, then it shall ignore the IGNORE ZONE GROUP bit.

The PHY IDENTIFIER field specifies the phy (see 4.2.10) for which the information is being requested.

The CRC field is defined in 9.4.3.2.7.

Table 320 defines the response format.

**Table 320 – DISCOVER response (part 1 of 4)**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (41h)							
1	FUNCTION (10h)							
2	FUNCTION RESULT							
3	RESPONSE LENGTH (00h or 1Dh)							
4	(MSB)	EXPANDER CHANGE COUNT						(LSB)
5								
6								
...								
8								
9								
10								
11								
12	Reserved	ATTACHED SAS DEVICE TYPE			ATTACHED REASON			
13	Reserved				NEGOTIATED LOGICAL LINK RATE			
14	Reserved				ATTACHED SSP INITIATOR PORT	ATTACHED STP INITIATOR PORT	ATTACHED SMP INITIATOR PORT	ATTACHED SATA HOST
15	ATTACHED SATA PORT SELECTOR	Reserved		STP BUFFER TOO SMALL	ATTACHED SSP TARGET PORT	ATTACHED STP TARGET PORT	ATTACHED SMP TARGET PORT	ATTACHED SATA DEVICE

Table 320 – DISCOVER response (part 2 of 4)

Byte\Bit	7	6	5	4	3	2	1	0
16	SAS ADDRESS							
...								
23								
24	ATTACHED SAS ADDRESS							
...								
31								
32	ATTACHED PHY IDENTIFIER							
33	ATTACHED PERSISTENT CAPABLE	ATTACHED POWER CAPABLE		ATTACHED SLUMBER CAPABLE	ATTACHED PARTIAL CAPABLE	ATTACHED INSIDE ZPSDS PERSISTENT	ATTACHED REQUESTED INSIDE ZPSDS	ATTACHED BREAK_REPLY CAPABLE
34	Reserved for IDENTIFY address frame-related fields					ATTACHED APTA CAPABLE	ATTACHED SMP PRIORITY CAPABLE	ATTACHED PWR_DIS CAPABLE
35	Reserved for IDENTIFY address frame-related fields							
...								
39								
40	PROGRAMMED MINIMUM PHYSICAL LINK RATE				HARDWARE MINIMUM PHYSICAL LINK RATE			
41	PROGRAMMED MAXIMUM PHYSICAL LINK RATE				HARDWARE MAXIMUM PHYSICAL LINK RATE			
42	PHY CHANGE COUNT							
43	VIRTUAL PHY	Reserved			PARTIAL PATHWAY TIMEOUT VALUE			
44	Reserved				ROUTING ATTRIBUTE			
45	Reserved	CONNECTOR TYPE						
46	CONNECTOR ELEMENT INDEX							
47	CONNECTOR PHYSICAL LINK							
48	PHY POWER CONDITION		SAS POWER CAPABLE		SAS SLUMBER CAPABLE	SAS PARTIAL CAPABLE	SATA SLUMBER CAPABLE	SATA PARTIAL CAPABLE
49	PWR_DIS SIGNAL		PWR_DIS CONTROL CAPABLE		SAS SLUMBER ENABLED	SAS PARTIAL ENABLED	SATA SLUMBER ENABLED	SATA PARTIAL ENABLED
50	Vendor specific							
51								
52	ATTACHED DEVICE NAME							
...								
59								
60	Reserved	REQUESTED INSIDE ZPSDS CHANGED BY EXPANDER	INSIDE ZPSDS PERSISTENT	REQUESTED INSIDE ZPSDS	Reserved	ZONE GROUP PERSISTENT	INSIDE ZPSDS	ZONING ENABLED
61	Reserved for zoning-related fields							
62								
63	ZONE GROUP							
64	SELF-CONFIGURATION STATUS							
65	SELF-CONFIGURATION LEVELS COMPLETED							



Table 320 – DISCOVER response (part 3 of 4)

Byte\Bit	7	6	5	4	3	2	1	0
66	Reserved for self-configuration related fields							
67								
68	SELF-CONFIGURATION SAS ADDRESS							
...								
75	PROGRAMMED PHY CAPABILITIES							
76								
...	CURRENT PHY CAPABILITIES							
79								
80	ATTACHED PHY CAPABILITIES							
...								
83	Reserved							
84								
...	Reserved							
87								
88	Reserved							
...								
93	Reserved							
94								
94	REASON				NEGOTIATED PHYSICAL LINK RATE			
95	Reserved					OPTICAL MODE ENABLED	NEGOTIATED SSC	HARDWARE MUXING SUPPORTED
96	Reserved	DEFAULT INSIDE ZPSDS PERSISTENT	DEFAULT REQUESTED INSIDE ZPSDS	Reserved	DEFAULT ZONE GROUP PERSISTENT	Reserved	DEFAULT ZONING ENABLED	
97	Reserved							
98	Reserved							
99	DEFAULT ZONE GROUP							
100	Reserved	SAVED INSIDE ZPSDS PERSISTENT	SAVED REQUESTED INSIDE ZPSDS	Reserved	SAVED ZONE GROUP PERSISTENT	Reserved	SAVED ZONING ENABLED	
101	Reserved							
102	Reserved							
103	SAVED ZONE GROUP							
104	Reserved	SHADOW INSIDE ZPSDS PERSISTENT	SHADOW REQUESTED INSIDE ZPSDS	Reserved	SHADOW ZONE GROUP PERSISTENT	Reserved	SHADOW ZONING ENABLED	
105	Reserved							

**Table 320 – DISCOVER response** (part 4 of 4)

Byte\Bit	7	6	5	4	3	2	1	0
106	Reserved							
107	SHADOW ZONE GROUP							
108	DEVICE SLOT NUMBER							
109	DEVICE SLOT GROUP NUMBER							
110	(MSB)	DEVICE SLOT GROUP OUTPUT CONNECTOR						
115								
116	(MSB)	STP BUFFER SIZE						
117								
118	BUFFERED PHY BURST SIZE							
119	Reserved							
120	(MSB)	CRC						
123								

The SMP FRAME TYPE field is defined in 9.4.3.3.2 and shall be set as shown in table 320 for the DISCOVER response.

The FUNCTION field is defined in 9.4.3.3.3 and shall be set as shown in table 320 for the DISCOVER response.

The FUNCTION RESULT field is defined in 9.4.3.3.4.

The RESPONSE LENGTH field is defined in 9.4.3.3.5 and shall be set to one of the values defined in table 320 based on the ALLOCATED RESPONSE LENGTH field in the request frame.

The EXPANDER CHANGE COUNT field is defined in the SMP REPORT GENERAL response (see 9.4.3.4).

The PHY IDENTIFIER field indicates the phy identifier (see 4.2.10) of the phy for which information is being returned.

The ATTACHED SAS DEVICE TYPE field indicates the SAS device type attached to this phy and is defined in table 321.

**Table 321 – ATTACHED SAS DEVICE TYPE field**

Code	Description
000b	No device attached
001b	SAS device or SATA device
010b	Expander device
011b	Obsolete
All others	Reserved

If the phy is a physical phy, then the ATTACHED SAS DEVICE TYPE field shall only be set to a value other than 000b:

- a) if a SAS device or expander device is attached, then after the identification sequence is complete;
- b) if a SATA phy is attached and the STP SATA bridge does not retrieve IDENTIFY DEVICE data (see ACS-4), then after the STP SATA bridge receives the initial Register - Device to Host FIS; or
- c) if a SATA phy is attached and the STP SATA bridge retrieves IDENTIFY DEVICE data, then after the STP SATA bridge receives IDENTIFY DEVICE data or it encounters a failure retrieving that data.

If the NEGOTIATED PHYSICAL LINK RATE field (see table 322) is not set to a physical link rate, then the management device server may set the ATTACHED SAS DEVICE TYPE field to 000b.

If the phy is a physical phy and a SAS phy or expander phy is attached, then the ATTACHED REASON field indicates the value of the REASON field received in the IDENTIFY address frame (see 6.10.2) during the identification sequence. If the phy is a physical phy and a SATA phy is attached, then the ATTACHED REASON field shall be set to 0h after the initial Register - Device to Host FIS has been received. If the phy is a virtual phy, then the ATTACHED REASON field shall be set to 0h.

The NEGOTIATED LOGICAL LINK RATE field is defined in table 322 and indicates the logical link rate being used by the phy. For physical phys, this is negotiated during the link reset sequence. For virtual phys, this field should be set to the maximum physical link rate supported by the expander device. This field may be different from the negotiated physical link rate when multiplexing is enabled (see table 323).

**Table 322 – NEGOTIATED LOGICAL LINK RATE field and NEGOTIATED PHYSICAL LINK RATE field**

SP state machine ResetStatus state machine variable	Code	Description
UNKNOWN	0h	Phy is enabled, unknown physical link rate. <sup>a</sup>
DISABLED	1h	Phy is disabled.
PHY_RESET_PROBLEM	2h	Phy is enabled and a phy reset problem occurred (see 5.11.4.2.4).
SPINUP_HOLD	3h	Phy is enabled, did not detect a SAS phy or an expander phy (i.e., the attached phy did not respond with COMSAS within the COMSAS timeout) and entered the SATA spinup hold state. The SMP PHY CONTROL function (see 9.4.3.28) phy operations of LINK RESET and HARD RESET may be used to release the phy.
PORT_SELECTOR	4h	Phy is enabled and detected a SATA port selector. The physical link rate has not been negotiated since the last time the phy's SP state machine entered the SP0:OOB_COMINIT state. The SATA spinup hold state has not been entered since the last time the phy's SP state machine entered the SP0:OOB_COMINIT state. The value in this field may change to 3h, 8h, 9h, or Ah if attached to the active phy of the SATA port selector. Presence of a SATA port selector is indicated by the ATTACHED SATA PORT SELECTOR bit (see table 324).
RESET_IN_PROGRESS	5h	Phy is enabled and the expander phy is performing an SMP PHY CONTROL function (see 9.4.3.28) phy operation of LINK RESET or HARD RESET.  This value is returned if the specified phy contained a value of 8h to Fh in this field when an SMP PHY CONTROL function phy operation of LINK RESET or HARD RESET phy operation is processed.
UNSUPPORTED_PHY_ATTACHED	6h	Phy is enabled and a phy is attached without any commonly supported settings.
Reserved	7h	Reserved
G1	8h	Phy is enabled with a 1.5 Gbit/s physical link rate or logical link rate.
G2	9h	Phy is enabled with a 3 Gbit/s physical link rate or logical link rate.
G3	Ah	Phy is enabled with a 6 Gbit/s physical link rate or logical link rate.
G4	Bh	Phy is enabled with a 12 Gbit/s physical link rate or logical link rate.
G5	Ch	Phy is enabled with a 22.5 Gbit/s physical link rate or logical link rate.
Reserved	Dh to Fh	Phy is enabled and reserved for future logical link rate or physical link rates.
<sup>a</sup> This code may be used by a management application client in its local data structures to indicate an unknown negotiated logical link rate or physical link rate (e.g., before the discover process has queried the phy).		

**Table 323 – NEGOTIATED PHYSICAL LINK RATE field and NEGOTIATED LOGICAL LINK RATE field combinations based on multiplexing**

NEGOTIATED PHYSICAL LINK RATE field	Multiplexing	NEGOTIATED LOGICAL LINK RATE field
9h (i.e., G2)	Disabled	9h (i.e., 3 Gbit/s)
	Enabled	8h (i.e., 1.5 Gbit/s)
Ah (i.e., G3)	Disabled	Ah (i.e., 6 Gbit/s)
	Enabled	9h (i.e., 3 Gbit/s)
Bh (i.e., G4)	Disabled	Bh (i.e., 12 Gbit/s)
Ch (i.e., G5)	Disabled	Ch (i.e., 22.5 Gbit/s)

NOTE 80 - In SAS-1.1 which did not define multiplexing, the NEGOTIATED LOGICAL LINK RATE field was called the NEGOTIATED PHYSICAL LINK RATE field and the NEGOTIATED PHYSICAL LINK RATE field in byte 94 did not exist.

Table 324 defines the ATTACHED SATA PORT SELECTOR bit and the ATTACHED SATA DEVICE bit.

**Table 324 – ATTACHED SATA PORT SELECTOR and ATTACHED SATA DEVICE bits**

ATTACHED SATA PORT SELECTOR bit value <sup>a b d</sup>	ATTACHED SATA DEVICE bit value <sup>c d</sup>	Description
0	0	Either the phy is: a) a virtual phy; or b) a physical phy, and neither a SATA port selector nor a SATA device is attached and ready on the selected phy.
0	1	The phy is a physical phy and the attached phy is neither a SAS phy nor an expander phy (i.e., the attached phy did not respond with COMSAS within the COMSAS timeout). No SATA port selector is present (i.e., the SP state machine did not detect COMWAKE in response to the initial COMINIT, detected COMINIT, and then timed out waiting for COMSAS).
1	0	The phy is a physical phy, the attached phy is a SATA port selector host phy, and either the attached phy is: a) the inactive host phy; or b) the active host phy and a SATA device is either not present or not ready behind the SATA port selector.
1	1	The phy is a physical phy, the attached phy is a SATA port selector's active host phy and neither a SAS phy nor an expander phy is present behind the SATA port selector (i.e., the SP state machine detected COMWAKE while waiting for COMINIT, detected COMINIT, and then timed out waiting for COMSAS).
<sup>a</sup> The ATTACHED SATA PORT SELECTOR bit shall be ignored if the NEGOTIATED LOGICAL LINK RATE field is set to UNKNOWN (i.e., 0h), DISABLED (i.e., 1h), or RESET_IN_PROGRESS (i.e., 5h). <sup>b</sup> Whenever the ATTACHED SATA PORT SELECTOR bit changes, the phy shall originate a Broadcast (Change) (see 6.15). <sup>c</sup> For the purposes of the ATTACHED SATA DEVICE bit, a SATA port selector is not considered a SATA device. <sup>d</sup> The ATTACHED SATA PORT SELECTOR bit and the ATTACHED SATA DEVICE bit are updated as specified in the SP state machine (see 5.14).		

An ATTACHED SATA HOST bit set to one indicates a SATA host port is attached. An ATTACHED SATA HOST bit set to zero indicates a SATA host port is not attached.

NOTE 81 - Support for SATA hosts is outside the scope of this standard.

If a SAS phy reset sequence occurs (see 5.11.4) (i.e., one or more of the ATTACHED SSP INITIATOR PORT bit, the ATTACHED STP INITIATOR PORT bit, the ATTACHED SMP INITIATOR PORT bit, the ATTACHED SSP TARGET PORT bit, the ATTACHED STP TARGET PORT bit, and/or the ATTACHED SMP TARGET PORT bit is set to one), then the ATTACHED SATA PORT SELECTOR bit, the ATTACHED SATA DEVICE bit, and the ATTACHED SATA HOST bit shall each be set to zero.

An ATTACHED SSP INITIATOR PORT bit set to one indicates that the attached phy supports an SSP initiator port. An ATTACHED SSP INITIATOR PORT bit set to zero indicates that the attached phy does not support an SSP initiator port. If the phy is a physical phy, then the ATTACHED SSP INITIATOR PORT bit indicates the value of the SSP INITIATOR PORT bit received in the IDENTIFY address frame (see 6.10.2) during the identification sequence.

An ATTACHED STP INITIATOR PORT bit set to one indicates that the attached phy supports an STP initiator port. An ATTACHED STP INITIATOR PORT bit set to zero indicates that the attached phy does not support an STP initiator port. If the phy is a physical phy, then the ATTACHED STP INITIATOR PORT bit indicates the value of the STP INITIATOR PORT bit received in the IDENTIFY address frame (see 6.10.2) during the identification sequence.

An ATTACHED SMP INITIATOR PORT bit set to one indicates that the attached phy supports an SMP initiator port. An ATTACHED SMP INITIATOR PORT bit set to zero indicates that the attached phy does not support an SMP initiator port. If the phy is a physical phy, then the ATTACHED SMP INITIATOR PORT bit indicates the value of the SMP INITIATOR PORT bit received in the IDENTIFY address frame (see 6.10.2) during the identification sequence.

The STP BUFFER TOO SMALL bit set to one indicates that the phy does not contain sufficient buffers to support STP connections for the attached cable assembly (see 6.21.4). The STP BUFFER TOO SMALL bit set to zero indicates that the phy may contain sufficient buffers to support STP connections for the attached cable assembly.

An ATTACHED SSP TARGET PORT bit set to one indicates that the attached phy supports an SSP target port. An ATTACHED SSP TARGET PORT bit set to zero indicates that the attached phy does not support an SSP target port. If the phy is a physical phy, then the ATTACHED SSP TARGET PORT bit indicates the value of the SSP TARGET PORT bit received in the IDENTIFY address frame (see 6.10.2) during the identification sequence.

An ATTACHED STP TARGET PORT bit set to one indicates that the attached phy supports an STP target port. An ATTACHED STP TARGET PORT bit set to zero indicates that the attached phy does not support an STP target port. If the phy is a physical phy, then the ATTACHED STP TARGET PORT bit indicates the value of the STP TARGET PORT bit received in the IDENTIFY address frame (see 6.10.2) during the identification sequence.

An ATTACHED SMP TARGET PORT bit set to one indicates that the attached phy supports an SMP target port. An ATTACHED SMP TARGET PORT bit set to zero indicates that the attached phy does not support an SMP target port. If the phy is a physical phy, then the ATTACHED SMP TARGET PORT bit indicates the value of the SMP TARGET PORT bit received in the IDENTIFY address frame (see 6.10.2) during the identification sequence.

If the phy is a physical phy, then the ATTACHED SSP INITIATOR PORT bit, the ATTACHED STP INITIATOR PORT bit, the ATTACHED SMP INITIATOR PORT bit, the ATTACHED SSP TARGET PORT bit, the ATTACHED STP TARGET PORT bit, and the ATTACHED SMP TARGET PORT bit shall be updated at the end of the identification sequence.

If a SATA phy reset sequence occurs (see 5.11.3) (i.e., the ATTACHED SATA PORT SELECTOR bit is set to one, the ATTACHED SATA DEVICE bit is set to one, or the ATTACHED SATA HOST bit is set to one), then the ATTACHED SSP INITIATOR PORT bit, the ATTACHED STP INITIATOR PORT bit, the ATTACHED SMP INITIATOR PORT bit, the ATTACHED SSP TARGET PORT bit, the ATTACHED STP TARGET PORT bit, and the ATTACHED SMP TARGET PORT bit shall each be set to zero.

If the phy is an expander phy, then the SAS ADDRESS field contains the SAS address of the expander device (see 4.2.6). If the phy is a SAS phy, then the SAS ADDRESS field contains the SAS address of the SAS port (see 4.2.9). If the phy is a physical phy, then the SAS ADDRESS field contains the value of the SAS ADDRESS field transmitted in the IDENTIFY address frame (see 6.10.2) during the identification sequence.

The ATTACHED SAS ADDRESS field is defined as follows:

- a) if the attached port is an expander port, then the ATTACHED SAS ADDRESS field contains the SAS address of the attached expander device (see 4.2.6);
- b) if the attached port is a SAS port, then the ATTACHED SAS ADDRESS field contains SAS address of the attached SAS port (see 4.2.9); or
- c) if the attached port is a SATA device port, then the ATTACHED SAS ADDRESS field contains the SAS address of the STP SATA bridge (see 4.5.2).

For a physical phy, the ATTACHED SAS ADDRESS field contains the value of the SAS ADDRESS field received in the IDENTIFY address frame (see 6.10.2) during the identification sequence and shall be updated after:

- a) the identification sequence completes, if a SAS phy or expander phy is attached; or
- b) the COMSAS Detect Timeout timer expires (see 5.14.3.9), if a SATA phy is attached.

An STP initiator port should not make a connection request to the attached SAS address until the ATTACHED SAS DEVICE TYPE field is set to a value other than 000b (see table 321).

The ATTACHED PHY IDENTIFIER field is defined as follows:

- a) if the attached phy is a SAS phy, then the ATTACHED PHY IDENTIFIER field contains the phy identifier of the attached SAS phy in the attached SAS device;
- b) if the attached phy is an expander phy, then the ATTACHED PHY IDENTIFIER field contains the phy identifier (see 4.2.10) of the attached expander phy in the attached expander device;
- c) if the attached phy is a SATA device phy, then the ATTACHED PHY IDENTIFIER field contains 00h;
- d) if the attached phy is a SATA port selector phy and the expander device is able to determine the port of the SATA port selector to which it is attached, then the ATTACHED PHY IDENTIFIER field contains 00h or 01h; or
- e) if the attached phy is a SATA port selector phy and the expander device is not able to determine the port of the SATA port selector to which it is attached, then the ATTACHED PHY IDENTIFIER field contains 00h.

If the phy is a physical phy and the attached phy is a SAS phy or an expander phy, then the ATTACHED PHY IDENTIFIER field contains the value of the PHY IDENTIFIER field received in the IDENTIFY address frame (see 6.10.2) during the identification sequence.

For a physical phy, the ATTACHED PHY IDENTIFIER field shall be updated after:

- a) the identification sequence completes, if a SAS phy or expander phy is attached; or
- b) the COMSAS Detect Timeout timer expires (see 5.14.3.9), if a SATA phy is attached.

An ATTACHED PERSISTENT CAPABLE bit indicates the value of the PERSISTENT CAPABLE bit received in the IDENTIFY address frame (see 6.10.2) from the attached phy during the identification sequence.

The ATTACHED POWER CAPABLE field indicates the value of the POWER CAPABLE field received in the IDENTIFY address frame (see 6.10.2) from the attached phy during the identification sequence.

The ATTACHED SLUMBER CAPABLE bit indicates the value of the SLUMBER CAPABLE bit received in the IDENTIFY address frame (see 6.10.2) from the attached phy during the identification sequence.

The ATTACHED PARTIAL CAPABLE bit indicates the value of the PARTIAL CAPABLE bit received in the IDENTIFY address frame (see 6.10.2) from the attached phy during the identification sequence.

If the phy is a physical phy, then the ATTACHED INSIDE ZPSDS PERSISTENT bit indicates the value of the INSIDE ZPSDS PERSISTENT bit received in the IDENTIFY address frame (see 6.10.2) from the attached phy during the identification sequence. If the phy is a virtual phy, then the ATTACHED INSIDE ZPSDS PERSISTENT bit shall be set to zero.

If the phy is a physical phy, then the ATTACHED REQUESTED INSIDE ZPSDS bit indicates the value of the REQUESTED INSIDE ZPSDS bit received in the IDENTIFY address frame (see 6.10.2) from the attached phy during the identification sequence. If the phy is a virtual phy, then the ATTACHED REQUESTED INSIDE ZPSDS bit shall be set to zero.

If the phy is a physical phy, then the ATTACHED BREAK\_REPLY CAPABLE bit indicates the value of the BREAK\_REPLY CAPABLE bit received in the IDENTIFY address frame (see 6.10.2) during the identification sequence. If a phy reset sequence occurs (see 5.11), then the ATTACHED BREAK\_REPLY CAPABLE bit shall be set to zero. If the phy is a virtual phy, then the ATTACHED BREAK\_REPLY CAPABLE bit shall be set to zero.

The ATTACHED APTA CAPABLE bit indicates the value of the APTA CAPABLE bit received in the IDENTIFY address frame (see 6.10.2) from the attached phy during the identification sequence.

The ATTACHED SMP PRIORITY CAPABLE bit indicates the value of the SMP PRIORITY CAPABLE bit received in the IDENTIFY address frame (see 6.10.2) from the attached phy during the identification sequence.

The ATTACHED PWR\_DIS CAPABLE bit indicates the value of the PWR\_DIS CAPABLE bit received in the IDENTIFY address frame (see 6.10.2) from the attached phy during the identification sequence.

The PROGRAMMED MINIMUM PHYSICAL LINK RATE field indicates the minimum physical link rate set by the PHY CONTROL function (see 9.4.3.28). The values are defined in table 325. The default value shall be the value of the HARDWARE MINIMUM PHYSICAL LINK RATE field.



The **HARDWARE MINIMUM PHYSICAL LINK RATE** field indicates the minimum physical link rate supported by the phy. The values are defined in table 326.

The **PROGRAMMED MAXIMUM PHYSICAL LINK RATE** field indicates the maximum physical link rate set by the PHY CONTROL function (see 9.4.3.28). The values are defined in table 325. The default value shall be the value of the **HARDWARE MAXIMUM PHYSICAL LINK RATE** field.

**Table 325 – PROGRAMMED MINIMUM PHYSICAL LINK RATE field and PROGRAMMED MAXIMUM PHYSICAL LINK RATE field**

Code	Description
0h	Not programmable
1h to 7h	Reserved
8h	1.5 Gbit/s
9h	3 Gbit/s
Ah	6 Gbit/s
Bh	12 Gbit/s
Ch	22.5 Gbit/s
Dh to Fh	Reserved

The **HARDWARE MAXIMUM PHYSICAL LINK RATE** field indicates the maximum physical link rate supported by the phy. The values are defined in table 326. If the phy is a virtual phy, then this field should be set to the maximum physical link rate supported by the expander device.

**Table 326 – The HARDWARE MINIMUM PHYSICAL LINK RATE field and the HARDWARE MAXIMUM PHYSICAL LINK RATE field**

Code	Description
0h to 7h	Reserved
8h	1.5 Gbit/s
9h	3 Gbit/s
Ah	6 Gbit/s
Bh	12 Gbit/s
Ch	22.5 Gbit/s
Dh to Fh	Reserved

The PHY CHANGE COUNT field indicates the number of Broadcast (Change)s originated by an expander phy. Expander devices shall support this field. Other SAS device types shall not support this field. This field shall be set to 00h at power on. The expander device shall increment this field at least once when:

- a) the expander device originates a Broadcast (Change) for an expander phy-related reason described in 6.15 from the specified expander phy; or
- b) the zone phy information changes for the specified expander phy (e.g., when a locked expander device is unlocked (see 4.8.6.5)).

The expander device shall not increment this field when forwarding a Broadcast (Change).

After incrementing the PHY CHANGE COUNT field, the expander device is not required to increment the PHY CHANGE COUNT field again unless a DISCOVER response or a DISCOVER LIST response for the phy is transmitted. The PHY CHANGE COUNT field shall wrap to 00h after the maximum value (i.e., FFh) has been reached.

NOTE 82 - If a management application client uses the PHY CHANGE COUNT field, then reading it often ensures that it does not increment a multiple of 256 times between reading the field.

A VIRTUAL PHY bit set to one indicates that the phy is a virtual phy and is part of an internal port and the attached device is contained within the expander device. A VIRTUAL PHY bit set to zero indicates that the phy is a physical phy and the attached device is not contained within the expander device.

The PARTIAL PATHWAY TIMEOUT VALUE field indicates the partial pathway timeout value in microseconds (see 6.16.5.4) set by the PHY CONTROL function (see 9.4.3.28). The recommended default value for PARTIAL PATHWAY TIMEOUT VALUE is 7  $\mu$ s.

The ROUTING ATTRIBUTE field indicates the routing attribute supported by the phy (see 4.5.7.1) and is defined in table 327.

**Table 327 – ROUTING ATTRIBUTE field**

Code	Name	Description
0h	Direct routing attribute	Direct routing method for attached end devices. Attached expander devices are not supported on this phy.
1h	Subtractive routing attribute	Either: a) subtractive routing method for attached expander devices; or b) direct routing method for attached end devices.
2h	Table routing attribute	Either: a) table routing method for attached expander devices; or b) direct routing method for attached end devices.
All others	Reserved	

The ROUTING ATTRIBUTE field shall not change based on the attached SAS device type.

The CONNECTOR TYPE field indicates the type of connector used to access the phy, as reported by the enclosure services process for the enclosure (see the SAS Connector element in SES-3). A CONNECTOR TYPE field set to 00h indicates no connector information is available and that the CONNECTOR ELEMENT INDEX field and the CONNECTOR PHYSICAL LINK fields shall be ignored.

The CONNECTOR ELEMENT INDEX field indicates the element index of the SAS Connector element representing the connector used to access the phy, as reported by the enclosure services process for the enclosure (see the SAS Connector element in SES-3).

The CONNECTOR PHYSICAL LINK field indicates the physical link in the connector used to access the phy, as reported by the enclosure services process for the enclosure (see the SAS Connector element in SES-3).

The PHY POWER CONDITION field is defined in table 328 and indicates the power condition of the phy.

**Table 328 – PHY POWER CONDITION field**

Code	Description
00b	Active phy power condition
01b	Partial phy power condition
10b	Slumber phy power condition
11b	Reserved

The SAS POWER CAPABLE field is defined in table 329.

**Table 329 – SAS POWER CAPABLE field**

Code	Description
00b	The SAS device containing the phy: a) does not respond to PWR_GRANT with PWR_ACK, PWR_REQ with PWR_ACK, or PWR_DONE with PWR_ACK; and b) does not issue PWR_REQ or PWR_DONE.
01b	Reserved
10b	The SAS device containing the phy is capable of managing the additional consumption of power (see 6.14.1) by responding to: a) PWR_REQ with PWR_ACK; b) PWR_REQ with PWR_GRANT; and c) PWR_DONE with PWR_ACK.
11b	Reserved

A SAS SLUMBER CAPABLE bit set to one indicates that the phy supports the slumber phy power condition (see 4.10.1.4). A SAS SLUMBER CAPABLE bit set to zero indicates that the phy does not support the slumber phy power condition.

A SAS PARTIAL CAPABLE bit set to one indicates that the phy supports the partial phy power condition (see 4.10.1.3). A SAS PARTIAL CAPABLE bit set to zero indicates that the phy does not support the partial phy power condition.

A SATA SLUMBER CAPABLE bit set to one indicates that the phy supports the SATA slumber interface power management sequence (see 4.10.2). A SATA SLUMBER CAPABLE bit set to zero indicates that the phy does not support the SATA slumber interface power management sequence.

A SATA PARTIAL CAPABLE bit set to one indicates that the phy supports the SATA partial interface power management sequence (see 4.10.2). A SATA PARTIAL CAPABLE bit set to zero indicates that the phy does not support the SATA partial interface power management sequence.

Table 330 defines the PWR\_DIS SIGNAL field.

**Table 330 – PWR\_DIS SIGNAL field**

Code	Description
00b	Not capable of reporting the POWER DISABLE signal (see 4.13 and SAS-4) associated with the phy.
01b	Reserved
10b	The POWER DISABLE signal associated with the phy is negated.
11b	The POWER DISABLE signal associated with the phy is asserted.

Table 331 defines the PWR\_DIS CONTROL CAPABLE field.

**Table 331 – PWR\_DIS CONTROL CAPABLE field**

Code	Description
00b	Not capable of controlling the POWER DISABLE signal (see 4.13.2 and SAS-4) associated with the phy.
01b	Capable of controlling the POWER DISABLE signal (see 4.13.2 and SAS-4) associated with the phy using the PWR_DIS CONTROL field in the SMP PHY CONTROL function.
10b	Capable of controlling the POWER DISABLE signal (see 4.13.2 and SAS-4) associated with the phy and controlled by a method outside the scope of this standard.
11b	Reserved

A SAS SLUMBER ENABLED bit set to one indicates that the slumber phy power condition (see 4.10.1.4) is enabled on the phy (see table 391). A SAS SLUMBER ENABLED bit set to zero indicates that the slumber phy power condition is disabled on the phy.

A SAS PARTIAL ENABLED bit set to one indicates that the partial phy power condition (see 4.10.1.3) is enabled on the phy (see table 392). A SAS PARTIAL ENABLED bit set to zero indicates that the partial phy power condition is disabled on the phy.

A SATA SLUMBER ENABLED bit set to one indicates that the SATA slumber interface power management sequence (see 4.10.2) is enabled on the phy (see table 393). A SATA SLUMBER ENABLED bit set to zero indicates that the SATA slumber interface power management sequence is disabled on the phy.

A SATA PARTIAL ENABLED bit set to one indicates that the SATA partial interface power management sequence (see 4.10.2) is enabled on the phy (see table 394). A SATA PARTIAL ENABLED bit set to zero indicates that the SATA partial interface power management sequence is disabled on the phy.

The ATTACHED DEVICE NAME field is defined as follows:

- if the attached phy is an expander phy, then the ATTACHED DEVICE NAME field contains the device name of the attached expander device (see 4.2.6);
- if the attached phy is a SAS phy, then the ATTACHED DEVICE NAME field contains the device name of the attached SAS device (see 4.2.6); or
- if the attached phy is a SATA device phy, then the ATTACHED DEVICE NAME field contains the world wide name of the SATA device (see 4.2.7) or 00000000 00000000h.

For physical phys, table 332 defines how the ATTACHED DEVICE NAME field is updated.

**Table 332 – ATTACHED DEVICE NAME field**

Condition	Update time	Value
A SAS phy or expander phy is attached	Completion of the identification sequence	The management device server shall set this field to the DEVICE NAME field in the incoming IDENTIFY address frame (i.e., the attached expander device name or attached SAS device name (see 4.2.6)).
A SATA phy is attached	Expiration of the COMSAS Detect Timeout timer (see 5.7.3)	The management device server shall set this field to 00000000 00000000h.
	Reception of IDENTIFY DEVICE data (see ACS-4) from the SATA device <sup>a</sup>	Either: a) if IDENTIFY DEVICE data word 255 (i.e., the Integrity word) is correct and words 108 to 111 (i.e., the World Wide Name field) are not set to zero (see ACS-4), then the management device server shall set this field to the world wide name indicated by words 108 to 111 according to table 21 in 4.2.7; b) if IDENTIFY DEVICE data word 255 (i.e., the Integrity word) is correct and words 108 to 111 (i.e., the World Wide Name) are set to zero, then the management device server shall set this field to 00000000 00000000h; or c) if IDENTIFY DEVICE data word 255 (i.e., the Integrity word) is not correct, then the management device server shall set this field to 00000000 00000000h.
	Processing a PHY CONTROL function SET ATTACHED DEVICE NAME phy operation	The management device server shall set this field to the value specified in the ATTACHED DEVICE NAME field in the PHY CONTROL request (see 9.4.3.28).
<sup>a</sup> This row only applies if the expander device originates the IDENTIFY DEVICE command.		

A REQUESTED INSIDE ZPSDS CHANGED BY EXPANDER bit set to one indicates that the zoning expander device set the REQUESTED INSIDE ZPSDS bit to zero in the zone phy information at the completion of the last link reset sequence. A REQUESTED INSIDE ZPSDS CHANGED BY EXPANDER bit set to zero indicates that the zoning expander device did not set the REQUESTED INSIDE ZPSDS bit to zero in the zone phy information at the completion of the last link reset sequence. The REQUESTED INSIDE ZPSDS CHANGED BY EXPANDER bit shall be set to zero if the management device server is not in a zoning expander device.

NOTE 83 - A use of the REQUESTED INSIDE ZPSDS CHANGED BY EXPANDER bit is for the zone manager to determine why the REQUESTED INSIDE ZPSDS bit has changed in the DISCOVER response from the value to which it last set the bit.

The INSIDE ZPSDS PERSISTENT bit indicates the value of the INSIDE ZPSDS PERSISTENT bit in the zone phy information (see 4.8.3.1). The INSIDE ZPSDS PERSISTENT bit shall be set to zero if the management device server is not in a zoning expander device.

The REQUESTED INSIDE ZPSDS bit indicates the value of the REQUESTED INSIDE ZPSDS bit in the zone phy information (see 4.8.3.1). The REQUESTED INSIDE ZPSDS bit shall be set to zero if the management device server is not in a zoning expander device.

The ZONE GROUP PERSISTENT bit indicates the value of the ZONE GROUP PERSISTENT bit in the zone phy information (see 4.8.3.1). The ZONE GROUP PERSISTENT bit shall be set to zero if the management device server is not in a zoning expander device.

The INSIDE ZPSDS bit indicates the value of the INSIDE ZPSDS bit in the zone phy information (see 4.8.3.1). The INSIDE ZPSDS bit shall be set to zero if the management device server is not in a zoning expander device.

The ZONING ENABLED bit is defined in the SMP REPORT GENERAL response (see 9.4.3.4).

The ZONE GROUP field indicates the value of the ZONE GROUP field in the zone phy information (see 4.8.3.1). The ZONE GROUP field shall be set to 00h if the management device server is not in a zoning expander device.

The SELF-CONFIGURATION STATUS field indicates the status of a self-configuring expander device pertaining to the specified phy and is defined in table 333.

**Table 333 – SELF-CONFIGURATION STATUS field**

Code	Description
00h	No status available.
01h to FFh	As defined for the STATUS TYPE field in the self-configuration status descriptor in the REPORT SELF-CONFIGURATION STATUS response (see table 302 in 9.4.3.6).

The SELF-CONFIGURATION LEVELS COMPLETED field indicates the number of levels of expander devices beyond the expander port containing the specified phy for which the self-configuring expander device's management application client has completed the discover process and is defined in table 334.

**Table 334 – SELF-CONFIGURATION LEVELS COMPLETED field**

Code	Description
00h	The management application client: <ul style="list-style-type: none"> <li>a) has not begun the discover process through the expander port containing the specified phy;</li> <li>b) has not completed the discover process through the expander port containing the specified phy; or</li> <li>c) an expander device is not attached to the expander port containing the specified phy.</li> </ul>
01h	The management application client has completed discovery of the expander device attached to the expander port containing the specified phy (i.e., level 1).
02h	The management application client has completed discovery of the expander devices attached to the expander device attached to the expander port containing the specified phy (i.e., level 2).
...	...
FFh	The management application client has completed discovery of the expander devices attached at level 255.

NOTE 84 - The SELF-CONFIGURATION LEVELS COMPLETED field does not reflect the level of externally configurable expander devices that the configuration subprocess updates to enable the discover process to proceed to higher levels.

The SELF-CONFIGURATION SAS ADDRESS field indicates the SAS address (see 4.2.4) of the SMP target port to which the self-configuring expander device established a connection or attempted to establish a connection using the specified phy and resulted in the status indicated by the SELF-CONFIGURATION STATUS field.

The PROGRAMMED PHY CAPABILITIES field indicates the SNW-3 phy capabilities bits that are going to be transmitted in the next link reset sequence containing SNW-3 as defined in table 70.

The CURRENT PHY CAPABILITIES field indicates the outgoing SNW-3 phy capabilities bits transmitted in the last link reset sequence as defined in table 70. If the last link reset sequence did not include SNW-3 or was a SATA link reset sequence, then the CURRENT PHY CAPABILITIES field shall be set to 00000000h.

The ATTACHED PHY CAPABILITIES field indicates the incoming SNW-3 phy capabilities bits received in the last SNW-3 as defined in table 70. If the last link reset sequence did not include SNW-3 or was a SATA link reset sequence, then the ATTACHED PHY CAPABILITIES field shall be set to 00000000h.

The REASON field indicates the reason for the last reset of the phy. If the phy is a physical phy, then the REASON field indicates the value of the REASON field transmitted in the IDENTIFY address frame (see 6.10.2) during the identification sequence. If the phy is a physical phy and a SATA phy is attached, then the REASON field indicates the reason for the link reset sequence (see 6.10.2).

The NEGOTIATED PHYSICAL LINK RATE field is defined in table 322. If the phy is a physical phy, then this field indicates the physical link rate negotiated during the link reset sequence. If the phy is a virtual phy, then this field should be set to the maximum physical link rate supported by the expander device. The negotiated physical link rate may be less than the programmed minimum physical link rate or greater than the programmed maximum physical link rate if the programmed physical link rates have been changed since the last link reset sequence.

An OPTICAL MODE ENABLED bit set to one indicates that optical mode is enabled on the phy. An OPTICAL MODE ENABLED bit set to zero indicates that D.C. mode is enabled on the phy.

A NEGOTIATED SSC bit set to one indicates that SSC is enabled (see SAS-4). A NEGOTIATED SSC bit set to zero indicates that SSC is disabled. The NEGOTIATED SSC bit is only valid if the NEGOTIATED PHYSICAL LINK RATE field is greater than or equal to 8h.

A HARDWARE MUXING SUPPORTED bit set to one indicates that the phy supports multiplexing (see 5.20). A HARDWARE MUXING SUPPORTED bit set to zero indicates that the phy does not support multiplexing. This value is not adjusted based on the negotiated physical link rate.

The DEFAULT INSIDE ZPSDS PERSISTENT bit contains the default value of the INSIDE ZPSDS PERSISTENT bit in the zone phy information (see 4.8.3.1).

The DEFAULT REQUESTED INSIDE ZPSDS bit contains the default value of the REQUESTED INSIDE ZPSDS bit in the zone phy information (see 4.8.3.1).

The DEFAULT ZONE GROUP PERSISTENT bit contains the default value of the ZONE GROUP PERSISTENT bit in the zone phy information (see 4.8.3.1).

The DEFAULT ZONING ENABLED bit contains the default value of the ZONING ENABLED bit (see 4.8.3.1).

The DEFAULT ZONE GROUP field contains the default value of the ZONE GROUP field in the zone phy information (see 4.8.3.1).

The SAVED INSIDE ZPSDS PERSISTENT bit contains the saved value of the INSIDE ZPSDS PERSISTENT bit in the zone phy information (see 4.8.3.1).

The SAVED REQUESTED INSIDE ZPSDS bit contains the saved value of the REQUESTED INSIDE ZPSDS bit in the zone phy information (see 4.8.3.1).

The SAVED ZONE GROUP PERSISTENT bit contains the saved value of the ZONE GROUP PERSISTENT bit in the zone phy information (see 4.8.3.1).

The SAVED ZONING ENABLED bit contains the saved value of the ZONING ENABLED bit (see 4.8.3.1).

The SAVED ZONE GROUP field contains the saved value of the ZONE GROUP field in the zone phy information (see 4.8.3.1).

The SHADOW INSIDE ZPSDS PERSISTENT bit contains the shadow value of the INSIDE ZPSDS PERSISTENT bit in the zone phy information (see 4.8.3.1).

The SHADOW REQUESTED INSIDE ZPSDS bit contains the shadow value of the REQUESTED INSIDE ZPSDS bit in the zone phy information (see 4.8.3.1).

The SHADOW ZONE GROUP PERSISTENT bit contains the shadow value of the ZONE GROUP PERSISTENT bit in the zone phy information (see 4.8.3.1).

The SHADOW ZONING ENABLED bit contains the shadow value of the ZONING ENABLED bit (see 4.8.3.1).

The SHADOW ZONE GROUP field contains the shadow value of the ZONE GROUP field in the zone phy information (see 4.8.3.1).

The DEVICE SLOT NUMBER field indicates the number of the enclosure device slot to which the phy provides access, as reported by the enclosure services process for the enclosure (see the Additional Element Status descriptor for Device Slot and Array Device Slot elements in SES-3). A DEVICE SLOT NUMBER field set to FFh indicates that no device slot number is available.

The DEVICE SLOT GROUP NUMBER field indicates the number of the group of device slots containing the device slot indicated by the DEVICE SLOT NUMBER field. A DEVICE SLOT GROUP NUMBER field set to FFh indicates that no device slot group number is available.

Contents of the DEVICE SLOT GROUP NUMBER field may be the same as the Group ID reported via the SGPIO input stream from the enclosure (see SFF-8485).

The DEVICE SLOT GROUP OUTPUT CONNECTOR field contains a left-aligned ASCII string describing the connector of the enclosure containing the management device server attached to the device slot group indicated by the DEVICE SLOT GROUP NUMBER field. A DEVICE SLOT GROUP OUTPUT CONNECTOR field set to 2020 202020h (i.e., six space characters) indicates that no device slot group output connector information is available.

The STP BUFFER SIZE field indicates the largest buffer size in data dwords that is supported by the phy. An STP BUFFER SIZE field set to 00h indicates unknown buffer size.

The BUFFERED PHY BURST SIZE field multiplied by 1 024 bytes indicates the optimum transfer size for the phy, if that phy contains buffers. An initiator device may use this information to optimize write data transfers.

A BUFFERED PHY BURST SIZE field set to 00h indicates that the optimum transfer size is not specified or the phy does not contain buffers. A BUFFERED PHY BURST SIZE field set to FFh indicates an optimum transfer size that is greater than or equal to 261 120 bytes.

The CRC field is defined in 9.4.3.3.7.

#### **9.4.3.11 REPORT PHY ERROR LOG function**

The REPORT PHY ERROR LOG function returns error logging information about the specified phy. This SMP function may be implemented by any management device server.



Table 335 defines the request format.

**Table 335 – REPORT PHY ERROR LOG request**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (40h)							
1	FUNCTION (11h)							
2	ALLOCATED RESPONSE LENGTH							
3	REQUEST LENGTH (00h or 02h)							
4	Reserved							
...								
8								
9	PHY IDENTIFIER							
10	Reserved							
11								
12	(MSB)	CRC						(LSB)
...								
15								

The SMP FRAME TYPE field is defined in 9.4.3.2.2 and shall be set as shown in table 335 for the REPORT PHY ERROR LOG request.

The FUNCTION field is defined in 9.4.3.2.3 and shall be set as shown in table 335 for the REPORT PHY ERROR LOG request.

The ALLOCATED RESPONSE LENGTH field is defined in 9.4.3.2.4.

If the ALLOCATED RESPONSE LENGTH field is set to 00h, then the management device server shall:

- set the RESPONSE LENGTH field to 00h in the response frame; and
- return the first 28 bytes defined in table 336 plus the CRC field as the response frame.

If the ALLOCATED RESPONSE LENGTH field is not set to 00h, then the management device server shall:

- set the RESPONSE LENGTH field in the response frame to the non-zero value defined in table 336; and
- return the response frame as defined in 9.4.3.2.4.

The REQUEST LENGTH field is defined in 9.4.3.2.5 and shall be set to one of the values defined in table 335 based on the LONG RESPONSE bit in the REPORT GENERAL response (see 9.4.3.4). A REQUEST LENGTH field set to 00h specifies that there are two dwords before the CRC field.

The PHY IDENTIFIER field specifies the phy (see 4.2.10) for which information shall be reported.

The CRC field is defined in 9.4.3.2.7.

Table 336 defines the response format.

**Table 336 – REPORT PHY ERROR LOG response**

Byte\Bit	7	6	5	4	3	2	1	0					
0	SMP FRAME TYPE (41h)												
1	FUNCTION (11h)												
2	FUNCTION RESULT												
3	RESPONSE LENGTH (00h or 06h)												
4	(MSB)	EXPANDER CHANGE COUNT											
5								(LSB)					
6	Reserved												
...													
8													
9	PHY IDENTIFIER												
10	Reserved												
11													
12	(MSB)	INVALID DWORD COUNT											
...													
15								(LSB)					
16	(MSB)	RUNNING DISPARITY ERROR COUNT											
...													
19								(LSB)					
20	(MSB)	LOSS OF DWORD SYNCHRONIZATION COUNT											
...													
23								(LSB)					
24	(MSB)	PHY RESET PROBLEM COUNT											
...													
27								(LSB)					
28	(MSB)	CRC											
...													
31								(LSB)					

The SMP FRAME TYPE field is defined in 9.4.3.3.2 and shall be set as shown in table 336 for the REPORT PHY ERROR LOG response.

The FUNCTION field is defined in 9.4.3.3.3 and shall be set as shown in table 336 for the REPORT PHY ERROR LOG response.

The FUNCTION RESULT field is defined in 9.4.3.3.4.

The RESPONSE LENGTH field is defined in 9.4.3.3.5 and shall be set to one of the values defined in table 336 based on the ALLOCATED RESPONSE LENGTH field in the request frame.

The EXPANDER CHANGE COUNT field is defined in the SMP REPORT GENERAL response (see 9.4.3.4).

The PHY IDENTIFIER field indicates the phy (see 4.2.10) for which information is being reported and is the same as the PHY IDENTIFIER field in the request frame.

The INVALID DWORD COUNT field indicates the number of invalid dwords that have been received outside of phy reset sequences (i.e., between when the SP state machine (see 5.14) sends a Phy Layer Ready (SAS) confirmation or Phy Layer Ready (SATA) confirmation and when it sends a Phy Layer Not Ready confirmation to the link layer). The count shall stop at the maximum value. The INVALID DWORD COUNT field is set to a vendor specific value after power on.

The RUNNING DISPARITY ERROR COUNT field indicates the number of dwords containing running disparity errors (see 5.3.5) that have been received outside of phy reset sequences. The count shall stop at the maximum value. The RUNNING DISPARITY ERROR COUNT field is set to a vendor specific value after power on.

The LOSS OF DWORD SYNCHRONIZATION COUNT field indicates the number of times the phy has restarted the link reset sequence because it lost dword synchronization (see 5.15) (i.e., the SP state machine transitioned from SP15:SAS\_PHY\_Ready or SP22:SATA\_PHY\_Ready to SP0:OOB\_COMINIT (see 5.14)). The count shall stop at the maximum value. The LOSS OF DWORD SYNCHRONIZATION COUNT field is set to a vendor specific value after power on.

The PHY RESET PROBLEM COUNT field indicates the number of times a phy reset problem (see 5.11.4.2.4) occurred. The count shall stop at the maximum value. The PHY RESET PROBLEM COUNT field is set to a vendor specific value after power on.

The CRC field is defined in 9.4.3.3.7.

#### **9.4.3.12 REPORT PHY SATA function**

The REPORT PHY SATA function returns information about the SATA state for a specified phy. This SMP function shall be implemented by management device servers behind SMP target ports that share SAS addresses with STP target ports and by management device servers in expander devices with STP SATA bridges. This SMP function shall not be implemented by any other type of management device server.

Table 337 defines the request format.

**Table 337 – REPORT PHY SATA request**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (40h)							
1	FUNCTION (12h)							
2	ALLOCATED RESPONSE LENGTH							
3	REQUEST LENGTH (00h or 02h)							
4	Reserved							
...								
8								
9	PHY IDENTIFIER							
10	AFFILIATION CONTEXT							
11	Reserved							
12	CRC							
...								
15								

The SMP FRAME TYPE field is defined in 9.4.3.2.2 and shall be set as shown in table 337 for the REPORT PHY SATA request.

The FUNCTION field is defined in 9.4.3.2.3 and shall be set as shown in table 337 for the REPORT PHY SATA request.

The ALLOCATED RESPONSE LENGTH field is defined in 9.4.3.2.4.

If the ALLOCATED RESPONSE LENGTH field is set to 00h, then the management device server shall:

- set the RESPONSE LENGTH field to 00h in the response frame; and
- return the first 56 bytes defined in table 338 plus the CRC field as the response frame.

If the ALLOCATED RESPONSE LENGTH field is not set to 00h, then the management device server shall:

- set the RESPONSE LENGTH field in the response frame to the non-zero value defined in table 338; and
- return the response frame as defined in 9.4.3.2.4.

The REQUEST LENGTH field is defined in 9.4.3.2.5 and shall be set to one of the values defined in table 337 based on the LONG RESPONSE bit in the REPORT GENERAL response (see 9.4.3.4). A REQUEST LENGTH field set to 00h specifies that there are two dwords before the CRC field.

The PHY IDENTIFIER field specifies the phy (see 4.2.10) for which information shall be reported.

The AFFILIATION CONTEXT field specifies the relative identifier of the affiliation context for which information shall be reported (see 6.21.6).

The CRC field is defined in 9.4.3.2.7.

Table 338 defines the response format.

**Table 338 – REPORT PHY SATA response**

Byte\Bit	7	6	5	4	3	2	1	0							
0	SMP FRAME TYPE (41h)														
1	FUNCTION (12h)														
2	FUNCTION RESULT														
3	RESPONSE LENGTH (00h or 10h)														
4	(MSB)	EXPANDER CHANGE COUNT						(LSB)							
5															
6	Reserved														
...															
8															
9	PHY IDENTIFIER														
10	Reserved														
11	Reserved					STP I_T NEXUS LOSS OCCURRED	AFFILIATIONS SUPPORTED	AFFILIATION VALID							
12	Reserved														
...															
15															
16	STP SAS ADDRESS														
...															
23															
24	REGISTER DEVICE TO HOST FIS														
...															
43															
44	Reserved														
...															
47															
48	AFFILIATED STP INITIATOR SAS ADDRESS														
...															
55															
56	STP I_T NEXUS LOSS SAS ADDRESS														
...															
63															
64	Reserved														
65	AFFILIATION CONTEXT														
66	CURRENT AFFILIATION CONTEXTS														
67	MAXIMUM AFFILIATION CONTEXTS														
68	(MSB)	CRC						(LSB)							
...															
71															

The SMP FRAME TYPE field is defined in 9.4.3.3.2 and shall be set as shown in table 338 for the REPORT PHY SATA response.

The FUNCTION field is defined in 9.4.3.3.3 and shall be set as shown in table 338 for the REPORT PHY SATA response.

The FUNCTION RESULT field is defined in 9.4.3.3.4.

The RESPONSE LENGTH field is defined in 9.4.3.3.5 and shall be set to one of the values defined in table 338 based on the ALLOCATED RESPONSE LENGTH field in the request frame.

The EXPANDER CHANGE COUNT field is defined in the SMP REPORT GENERAL response (see 9.4.3.4).

The PHY IDENTIFIER field indicates the phy (see 4.2.10) for which information is being reported and is the same as the PHY IDENTIFIER field in the request frame.

An STP I\_T NEXUS LOSS OCCURRED bit set to one indicates that the STP target port encountered an I\_T nexus loss (see 4.4.3) in the specified affiliation context for the STP initiator port whose SAS address is indicated in the STP I\_T NEXUS LOSS SAS ADDRESS field. An STP I\_T NEXUS LOSS OCCURRED bit set to zero indicates that:

- a) an I\_T nexus loss has not occurred in the specified affiliation context;
- b) an I\_T nexus loss has occurred in the specified affiliation context and been cleared by the SMP PHY CONTROL function CLEAR STP I\_T NEXUS LOSS phy operation (see table 389 in 9.4.3.28); or
- c) the STP target port has established a connection with the indicated STP initiator port in the specified affiliation context.

An AFFILIATIONS SUPPORTED bit set to one indicates that the specified affiliation context is supported by the STP target port containing the specified phy. An AFFILIATIONS SUPPORTED bit set to zero indicates that the specified affiliation context is not supported by the STP target port containing the specified phy.

An AFFILIATION VALID bit set to one indicates that the STP target port is currently maintaining an affiliation in the specified affiliation context and the AFFILIATED STP INITIATOR SAS ADDRESS field is valid. An AFFILIATION VALID bit set to zero indicates that the STP target port is not currently maintaining an affiliation in the specified affiliation context and the AFFILIATED STP INITIATOR SAS ADDRESS field is not valid.

The STP SAS ADDRESS field indicates the SAS address (see 4.2.4) of the STP target port that contains the specified phy.

The REGISTER DEVICE TO HOST FIS field indicates the contents of the initial Register - Device to Host FIS. For an STP SATA bridge, this is delivered by the attached SATA device after a link reset sequence (see SATA). For a native STP target port in an end device, this is directly provided.

The FIS contents shall be stored with little-endian byte ordering (e.g., the first byte of the field (i.e., byte 24) contains the FIS Type).

For an STP SATA bridge, the first byte of the field (i.e., the FIS Type) shall be set to 00h on power on and whenever the phy has restarted the link reset sequence after losing dword synchronization (see 5.15) (i.e., the SP state machine transitioned from SP22:SATA\_PHY\_Ready to SP0:OOB\_COMINIT (see 5.14)) to indicate that the REGISTER DEVICE TO HOST FIS field does not contain the Register - Device to Host FIS contents of the currently attached SATA device. The first byte of the field shall be set to 34h when the attached SATA device has delivered the initial Register - Device to Host FIS. The remaining contents of the REGISTER DEVICE TO HOST FIS field shall remain constant until a link reset sequence causes the attached SATA device to deliver another initial Register - Device to Host FIS.

If the AFFILIATION VALID bit is set to one, then the AFFILIATED STP INITIATOR SAS ADDRESS field indicates the SAS address (see 4.2.4) of the STP initiator port that has an affiliation in the specified affiliation context with the STP target port that contains the specified phy. If the AFFILIATION VALID bit is set to zero, then the AFFILIATED STP INITIATOR SAS ADDRESS field may contain the SAS address of the STP initiator port that previously had an affiliation in the specified affiliation context with the STP target port that contains the specified phy.

The STP I\_T NEXUS LOSS SAS ADDRESS field indicates the SAS address (see 4.2.4) of the last STP initiator port for which the STP target port experienced an I\_T nexus loss (see 4.4.3) in the specified affiliation context.

The AFFILIATION CONTEXT field indicates the relative identifier of the affiliation context for which affiliation-related information (i.e., the AFFILIATIONS SUPPORTED bit, the AFFILIATION VALID bit, the AFFILIATED STP INITIATOR SAS ADDRESS field, the STP I\_T NEXUS LOSS OCCURRED bit, and the STP I\_T NEXUS LOSS SAS ADDRESS field) is being reported (see 6.21.6) and is the same as the AFFILIATION CONTEXT field in the request frame.

The CURRENT AFFILIATION CONTEXTS field indicates the current number of affiliations established by the STP target port.

The MAXIMUM AFFILIATION CONTEXTS field indicates the maximum number of affiliation contexts supported by the STP target port.

The CRC field is defined in 9.4.3.3.7.

#### **9.4.3.13 REPORT ROUTE INFORMATION function**

The REPORT ROUTE INFORMATION function returns an expander route entry from a phy-based expander route table within an expander device (see 4.5.7.4). This SMP function shall be supported by management device servers in expander devices if the EXPANDER ROUTE INDEXES field is set to a non-zero value in the SMP REPORT GENERAL response (see 9.4.3.4). This SMP function may be used as a diagnostic tool to resolve topology issues.

Table 339 defines the request format.

**Table 339 – REPORT ROUTE INFORMATION request**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (40h)							
1	FUNCTION (13h)							
2	ALLOCATED RESPONSE LENGTH							
3	REQUEST LENGTH (00h or 02h)							
4	Reserved							
5								
6	(MSB)	EXPANDER ROUTE INDEX						(LSB)
7								
8	Reserved							
9	PHY IDENTIFIER							
10	Reserved							
11								
12	(MSB)	CRC						(LSB)
...								
15								

The SMP FRAME TYPE field is defined in 9.4.3.2.2 and shall be set as shown in table 339 for the REPORT ROUTE INFORMATION request.

The FUNCTION field is defined in 9.4.3.2.3 and shall be set as shown in table 339 for the REPORT ROUTE INFORMATION request.

The ALLOCATED RESPONSE LENGTH field is defined in 9.4.3.2.4.

If the ALLOCATED RESPONSE LENGTH field is set to 00h, then the management device server shall:

- set the RESPONSE LENGTH field to 00h in the response frame; and
- return the first 40 bytes defined in table 340 plus the CRC field as the response frame.

If the ALLOCATED RESPONSE LENGTH field is not set to 00h, then the management device server shall:

- set the RESPONSE LENGTH field in the response frame to the non-zero value defined in table 340; and
- return the response frame as defined in 9.4.3.2.4.

The REQUEST LENGTH field is defined in 9.4.3.2.5 and shall be set to one of the values defined in table 339 based on the LONG RESPONSE bit in the REPORT GENERAL response (see 9.4.3.4). A REQUEST LENGTH field set to 00h specifies that there are two dwords before the CRC field.



The EXPANDER ROUTE INDEX field specifies the expander route index for the expander route entry being requested (see 4.5.7.4).

The PHY IDENTIFIER field specifies the phy identifier (see 4.2.10) of the phy for which the expander route entry is being requested.

The CRC field is defined in 9.4.3.2.7.

Table 340 defines the response format.

**Table 340 – REPORT ROUTE INFORMATION response**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (41h)							
1	FUNCTION (13h)							
2	FUNCTION RESULT							
3	RESPONSE LENGTH (00h or 09h)							
4	(MSB)	EXPANDER CHANGE COUNT						
5								(LSB)
6	(MSB)	EXPANDER ROUTE INDEX						
7								(LSB)
8	Reserved							
9	PHY IDENTIFIER							
10	Reserved							
11								
12	EXPANDER ROUTE ENTRY DISABLED	Reserved						
13	Reserved							
...								
15								
16	ROUTED SAS ADDRESS							
...								
23								
24	Reserved							
...								
39								
40	(MSB)	CRC						
...								
43								(LSB)

The SMP FRAME TYPE field is defined in 9.4.3.3.2 and shall be set as shown in table 340 for the REPORT ROUTE INFORMATION response.

The FUNCTION field is defined in 9.4.3.3.3 and shall be set as shown in table 340 for the REPORT ROUTE INFORMATION response.

The FUNCTION RESULT field is defined in 9.4.3.3.4.

The RESPONSE LENGTH field is defined in 9.4.3.3.5 and shall be set to one of the values defined in table 340 based on the ALLOCATED RESPONSE LENGTH field in the request frame.

The EXPANDER CHANGE COUNT field is defined in the SMP REPORT GENERAL response (see 9.4.3.4).

The EXPANDER ROUTE INDEX field indicates the expander route index for the expander route entry being returned (see 4.5.7.4).

The PHY IDENTIFIER field indicates the phy identifier (see 4.2.10) for the expander route entry being returned and is the same as the PHY IDENTIFIER field in the request frame.

The EXPANDER ROUTE ENTRY DISABLED bit indicates whether the ECM shall use the expander route entry to route connection requests (see 4.5.7.4). If the EXPANDER ROUTE ENTRY DISABLED bit is set to zero, then the ECM shall use the expander route entry to route connection requests. If the EXPANDER ROUTE ENTRY DISABLED bit is set to one, then the ECM shall not use the expander route entry to route connection requests.

The ROUTED SAS ADDRESS field indicates the SAS address (see 4.2.4) in the expander route entry (see 4.5.7.4).

The CRC field is defined in 9.4.3.3.7.

#### **9.4.3.14 REPORT PHY EVENT function**

##### **9.4.3.14.1 REPORT PHY EVENT function overview**

The REPORT PHY EVENT function returns phy events (see 4.12) concerning the specified phy. This SMP function may be implemented by any management device server.

NOTE 85 - The REPORT PHY EVENT LIST function (see 9.4.3.16) returns information about one or more phys.

**9.4.3.14.2 REPORT PHY EVENT request**

Table 341 defines the request format.

**Table 341 – REPORT PHY EVENT request**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (40h)							
1	FUNCTION (14h)							
2	ALLOCATED RESPONSE LENGTH							
3	REQUEST LENGTH (02h)							
4	Reserved							
5	Reserved							
...								
8								
9	PHY IDENTIFIER							
10	Reserved							
11								
12	(MSB)	CRC						(LSB)
...								
15								

The SMP FRAME TYPE field is defined in 9.4.3.2.2 and shall be set as shown in table 341 for the REPORT PHY EVENT request.

The FUNCTION field is defined in 9.4.3.2.3 and shall be set as shown in table 341 for the REPORT PHY EVENT request.

The ALLOCATED RESPONSE LENGTH field is defined in 9.4.3.2.4.

The REQUEST LENGTH field is defined in 9.4.3.2.5 and shall be set as shown in table 341 for the REPORT PHY EVENT request.

The PHY IDENTIFIER field specifies the phy (see 4.2.9) for which information shall be reported.

The CRC field is defined in 9.4.3.2.7.

**9.4.3.14.3 REPORT PHY EVENT response**

Table 342 defines the response format.

**Table 342 – REPORT PHY EVENT response**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (41h)							
1	FUNCTION (14h)							
2	FUNCTION RESULT							
3	RESPONSE LENGTH ((n - 7) / 4)							
4	(MSB)	EXPANDER CHANGE COUNT						(LSB)
5								
6								
...								
8	Reserved							
9	PHY IDENTIFIER							
10	Reserved							
...								
13								
14	PHY EVENT DESCRIPTOR LENGTH							
15	NUMBER OF PHY EVENT DESCRIPTORS							
Phy event descriptor list								
16	Phy event descriptor (first) (see table 343 in 9.4.3.14.4)							
...								
27								
...	...							
n - 15	Phy event descriptor (last) (see table 343 in 9.4.3.14.4)							
...								
n - 4								
n - 3	(MSB)	CRC						(LSB)
...								
n								

The SMP FRAME TYPE field is defined in 9.4.3.3.2 and shall be set as shown in table 342 for the REPORT PHY EVENT response.

The FUNCTION field is defined in 9.4.3.3.3 and shall be set as shown in table 342 for the REPORT PHY EVENT response.

The FUNCTION RESULT field is defined in 9.4.3.3.4.

The RESPONSE LENGTH field is defined in 9.4.3.3.5 and shall be set as shown in table 342 for the REPORT PHY EVENT response. A RESPONSE LENGTH field set to 00h does not have a special meaning based on the ALLOCATED RESPONSE LENGTH field in the request frame.

The EXPANDER CHANGE COUNT field is defined in the SMP REPORT GENERAL response (see 9.4.3.4).

The PHY IDENTIFIER field indicates the phy identifier (see 4.2.10) of the phy for which information is being reported and is the same as the PHY IDENTIFIER field in the request frame.

The PHY EVENT DESCRIPTOR LENGTH field indicates the length, in dwords, of the phy event descriptor (see 9.4.3.14.4).

The NUMBER OF PHY EVENT DESCRIPTORS field indicates the number of phy event descriptors in the phy event descriptor list.

The phy event descriptor list contains phy event descriptors as defined in 9.4.3.14.4.

The CRC field is defined in 9.4.3.3.7.

#### 9.4.3.14.4 Phy event descriptor

Table 343 defines the phy event descriptor.

**Table 343 – Phy event descriptor**

Byte\Bit	7	6	5	4	3	2	1	0
0	Reserved							
...								
2								
3	PHY EVENT SOURCE							
4	PHY EVENT							
...								
7								
8	PEAK VALUE DETECTOR THRESHOLD							
...								
11								

The PHY EVENT SOURCE field, defined in table 46 in 4.12, indicates the type of phy event being reported in the PHY EVENT field.

The PHY EVENT field indicates the value (i.e., the count or peak value detected) of the phy event indicated by the PHY EVENT SOURCE field.

If the phy event source is a peak value detector, then the PEAK VALUE DETECTOR THRESHOLD field indicates the value of the peak value detector that causes the expander device to originate a Broadcast (Expander) (see 6.2.6.4). If the phy event source is not a peak value detector, then the PEAK VALUE DETECTOR THRESHOLD field is reserved.

### **9.4.3.15 DISCOVER LIST function**

#### **9.4.3.15.1 DISCOVER LIST function overview**

The DISCOVER LIST function returns information about the device (i.e., some fields from the REPORT GENERAL response (see 9.4.3.4)) and one or more phys (i.e., some fields from the DISCOVER response (see 9.4.3.10)). This SMP function shall be implemented by all management device servers. This function provides the necessary information in a single SMP response for a self-configuring expander device to perform the discover process and configure its own expander routing table.

**9.4.3.15.2 DISCOVER LIST request**

Table 344 defines the request format.

**Table 344 – DISCOVER LIST request**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (40h)							
1	FUNCTION (20h)							
2	ALLOCATED RESPONSE LENGTH							
3	REQUEST LENGTH (06h)							
4	Reserved							
...								
7								
8	STARTING PHY IDENTIFIER							
9	MAXIMUM NUMBER OF DISCOVER LIST DESCRIPTORS							
10	IGNORE ZONE GROUP	Reserved			PHY FILTER			
11	Reserved				DESCRIPTOR TYPE			
12	Reserved							
...								
15								
16	Vendor specific							
...								
27								
28	(MSB)							
...	CRC							
31	(LSB)							

The SMP FRAME TYPE field is defined in 9.4.3.2.2 and shall be set as shown in table 344 for the DISCOVER LIST request.

The FUNCTION field is defined in 9.4.3.2.3 and shall be set as shown in table 344 for the DISCOVER LIST request.

The ALLOCATED RESPONSE LENGTH field is defined in 9.4.3.2.4.



The REQUEST LENGTH field is defined in 9.4.3.2.5 and shall be set as shown in table 344 for the DISCOVER LIST request.

The STARTING PHY IDENTIFIER field specifies the phy identifier of the first phy for which the information is being requested.

The MAXIMUM NUMBER OF DISCOVER LIST DESCRIPTORS field specifies the maximum number of complete DISCOVER LIST descriptors that the management device server shall return.

The IGNORE ZONE GROUP bit is defined in the SMP DISCOVER request (see 9.4.3.10).

The PHY FILTER field is defined in table 345 and specifies a filter limiting the phys that the management device server shall return in the DISCOVER LIST descriptor list in the DISCOVER response.

**Table 345 – PHY FILTER field**

Code	Description
0h	All phys. If the management device server is a zoning expander device with zoning enabled and the IGNORE ZONE GROUP bit is set to zero, then for any phy that is not accessible the FUNCTION RESULT field is set to PHY VACANT (see table 294).
1h	Phys with: a) the ATTACHED SAS DEVICE TYPE field (see 9.4.3.10) set to 010b or 011b (i.e., phys attached to expander devices); and b) the FUNCTION RESULT field not set to PHY VACANT.
2h	Phys with: a) the ATTACHED SAS DEVICE TYPE field (see 9.4.3.10) set to a value other than 000b (i.e., phys attached to end devices or expander devices); and b) the FUNCTION RESULT field not set to PHY VACANT.
3h	Phys with: a) the ATTACHED DEVICE TYPE field (see 9.4.3.10) set to 001b (i.e., phys attached to end devices); and b) the FUNCTION RESULT field not set to PHY VACANT.
All others	Reserved

The DESCRIPTOR TYPE field is defined in table 346 and specifies the DISCOVER LIST descriptor format and length.

**Table 346 – DESCRIPTOR TYPE field**

Code	DISCOVER LIST descriptor format	Descriptor length
0h	DISCOVER response defined in table 320 (see 9.4.3.10), starting with byte 0 and not including the CRC field.	The length of the DISCOVER response, not including the CRC field <sup>a</sup>
1h	SHORT FORMAT descriptor defined in table 348 (see 9.4.3.15.4)	24 bytes <sup>b</sup>
All others	Reserved	
<sup>a</sup> A maximum response frame size of 1 028 bytes supports eight 120-byte DISCOVER LIST descriptors containing DISCOVER responses. <sup>b</sup> A maximum response frame size of 1 028 bytes supports 40 24-byte DISCOVER LIST descriptors containing SHORT FORMAT descriptors.		

The CRC field is defined in 9.4.3.2.7.

## 9.4.3.15.3 DISCOVER LIST response

Table 347 defines the response format.

Table 347 – DISCOVER LIST response

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (41h)							
1	FUNCTION (20h)							
2	FUNCTION RESULT							
3	RESPONSE LENGTH ((n - 7) / 4)							
4	(MSB) EXPANDER CHANGE COUNT (LSB)							
5								
6	Reserved							
7								
8	STARTING PHY IDENTIFIER							
9	NUMBER OF DISCOVER LIST DESCRIPTORS							
10	Reserved				PHY FILTER			
11	Reserved				DESCRIPTOR TYPE			
12	DISCOVER LIST DESCRIPTOR LENGTH							
13	Reserved							
...								
15								
16	ZONING SUPPORTED	ZONING ENABLED	Reserved		SELF CONFIGURING	ZONE CONFIGURING	CONFIGURING	EXTERNALLY CONFIGURABLE ROUTE TABLE
17	Reserved							
18	(MSB) LAST SELF-CONFIGURATION STATUS DESCRIPTOR INDEX (LSB)							
19								
20	(MSB) LAST PHY EVENT LIST DESCRIPTOR INDEX (LSB)							
21								
22	Reserved							
...								
31								
32	Vendor specific							
...								
47								
DISCOVER LIST descriptor list								
48	DISCOVER LIST descriptor (first) (see table 346 in 9.4.3.15.1, and table 320 in 9.4.3.10 or table 348 in 9.4.3.15.4)							
...								
...								
...								
...	DISCOVER LIST descriptor (last) (see table 346 in 9.4.3.15.1, and table 320 in 9.4.3.10 or table 348 in 9.4.3.15.4)							
n - 4								
n - 3	(MSB)							
...	CRC (LSB)							
n								

The SMP FRAME TYPE field is defined in 9.4.3.3.2 and shall be set as shown in table 347 for the DISCOVER LIST response.

The FUNCTION field is defined in 9.4.3.3.3 and shall be set as shown in table 347 for the DISCOVER LIST response.

The FUNCTION RESULT field is defined in 9.4.3.3.4.

The RESPONSE LENGTH field is defined in 9.4.3.3.5 and shall be set as shown in table 347 for the DISCOVER LIST response. A RESPONSE LENGTH field set to 00h does not have a special meaning based on the ALLOCATED RESPONSE LENGTH field in the request frame.

The EXPANDER CHANGE COUNT field is defined in the SMP REPORT GENERAL response (see 9.4.3.4).

The STARTING PHY IDENTIFIER field indicates the phy identifier (see 4.2.10) of the first phy in the DISCOVER LIST descriptor list. As a result of the filter specified by the PHY FILTER field in the request frame, the STARTING PHY IDENTIFIER field may be different than the STARTING PHY IDENTIFIER field in the request frame (see 9.4.3.15.2).

The NUMBER OF DISCOVER LIST DESCRIPTORS field indicates the number of DISCOVER LIST descriptors returned in the DISCOVER LIST descriptor list.

The PHY FILTER field indicates the phy filter (see table 345 in 9.4.3.15.2) being used and is the same as the PHY FILTER field in the request frame.

The DESCRIPTOR TYPE field indicates the descriptor type (see table 346) being used and is the same as the DESCRIPTOR TYPE field in the request frame.

The DISCOVER LIST DESCRIPTOR LENGTH field indicates the length, in dwords, of the DISCOVER LIST descriptor (see table 346 in 9.4.3.15.2).

The ZONING SUPPORTED bit is defined in the SMP REPORT GENERAL response (see 9.4.3.4).

The ZONING ENABLED bit is defined in the SMP REPORT GENERAL response (see 9.4.3.4).

The SELF CONFIGURING bit is defined in the REPORT GENERAL response (see 9.4.3.4).

The ZONE CONFIGURING bit is defined in the REPORT GENERAL response (see 9.4.3.4).

The CONFIGURING bit is defined in the REPORT GENERAL response (see 9.4.3.4).

The EXTERNALLY CONFIGURABLE ROUTE TABLE bit is defined in the SMP REPORT GENERAL response (see 9.4.3.4).

The LAST SELF-CONFIGURATION STATUS DESCRIPTOR INDEX field is defined in the REPORT SELF-CONFIGURATION STATUS response (see 9.4.3.6).

The LAST PHY EVENT LIST DESCRIPTOR INDEX field is defined in the REPORT PHY EVENT LIST response (see 9.4.3.16).

The DISCOVER LIST descriptor list contains DISCOVER LIST descriptors for each phy:

- a) starting with the phy whose phy identifier is specified in the STARTING PHY IDENTIFIER field in the request (see 9.4.3.15.2);
- b) satisfying the filter specified in the PHY FILTER field in the request (see table 345 in 9.4.3.15.2);
- c) sorted in ascending order by phy identifier; and
- d) that is able to be included in the response frame without being truncated.

Each DISCOVER LIST descriptor shall use the format specified in the DESCRIPTOR TYPE field in the request (see table 346 in 9.4.3.15.2).

The management device server shall not include DISCOVER LIST descriptors for phys with phy identifiers greater than or equal to the NUMBER OF PHYs field reported in the SMP REPORT GENERAL response (see 9.4.3.4). The management device server shall not include partial DISCOVER LIST descriptors.

The CRC field is defined in 9.4.3.3.7.

## 9.4.3.15.4 DISCOVER LIST response SHORT FORMAT descriptor

Table 348 defines the SHORT FORMAT descriptor.

Table 348 – SHORT FORMAT descriptor

Byte\Bit	7	6	5	4	3	2	1	0
0	PHY IDENTIFIER							
1	FUNCTION RESULT							
2	Restricted for DISCOVER response byte 12	ATTACHED SAS DEVICE TYPE			ATTACHED REASON			
3	Restricted for DISCOVER response byte 13				NEGOTIATED LOGICAL LINK RATE			
4	Restricted for DISCOVER response byte 14				ATTACHED SSP INITIATOR PORT	ATTACHED STP INITIATOR PORT	ATTACHED SMP INITIATOR PORT	ATTACHED SATA HOST
5	ATTACHED SATA PORT SELECTOR	Restricted for DISCOVER response byte 15		STP BUFFER TOO SMALL	ATTACHED SSP TARGET PORT	ATTACHED STP TARGET PORT	ATTACHED SMP TARGET PORT	ATTACHED SATA DEVICE
6	VIRTUAL PHY	Reserved			ROUTING ATTRIBUTE			
7	REASON				NEGOTIATED PHYSICAL LINK RATE			
8	ZONE GROUP							
9	Restricted for DISCOVER response byte 60		INSIDE ZPSDS PERSISTENT	REQUESTED INSIDE ZPSDS	Reserved	ZONE GROUP PERSISTENT	INSIDE ZPSDS	Reserved
10	ATTACHED PHY IDENTIFIER							
11	PHY CHANGE COUNT							
12	ATTACHED SAS ADDRESS							
...								
19								
20	BUFFERED PHY BURST SIZE							
21	Reserved							
...								
23								

The PHY IDENTIFIER field indicates the phy identifier (see 4.2.10) of the phy for which information is being returned.

The FUNCTION RESULT field indicates the value that is returned in the FUNCTION RESULT field in the SMP DISCOVER response for the specified phy (e.g., SMP FUNCTION ACCEPTED, PHY VACANT, or PHY DOES NOT EXIST). If the FUNCTION RESULT field is set to PHY VACANT or PHY DOES NOT EXIST, then the rest of the fields in the SHORT FORMAT descriptor shall be ignored.

The fields in the SHORT FORMAT descriptor not defined in this subclause are defined in the SMP DISCOVER response (see 9.4.3.10).

### 9.4.3.16 REPORT PHY EVENT LIST function

#### 9.4.3.16.1 REPORT PHY EVENT LIST function overview

The REPORT PHY EVENT LIST function returns phy events (see 4.12). This SMP function may be implemented by any management device server.

#### 9.4.3.16.2 REPORT PHY EVENT LIST request

Table 349 defines the request format.

**Table 349 – REPORT PHY EVENT LIST request**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (40h)							
1	FUNCTION (21h)							
2	ALLOCATED RESPONSE LENGTH							
3	REQUEST LENGTH (01h)							
4	Reserved							
5								
6	(MSB)	STARTING PHY EVENT LIST DESCRIPTOR INDEX						
7								(LSB)
8	(MSB)	CRC						
...								
11								(LSB)

The SMP FRAME TYPE field is defined in 9.4.3.2.2 and shall be set as shown in table 349 for the REPORT PHY EVENT LIST request.

The FUNCTION field is defined in 9.4.3.2.3 and shall be set as shown in table 349 for the REPORT PHY EVENT LIST request.

The ALLOCATED RESPONSE LENGTH field is defined in 9.4.3.2.4.

The REQUEST LENGTH field is defined in 9.4.3.2.5 and shall be set as shown in table 349 for the REPORT PHY EVENT LIST request.

The STARTING PHY EVENT LIST DESCRIPTOR INDEX field specifies the first phy event list descriptor that the management device server shall return in the SMP response frame. A STARTING PHY EVENT LIST DESCRIPTOR INDEX field set to 0000h specifies that the management device server shall return no phy event list descriptors. The requested starting index and the indicated starting index in the response may differ.

The CRC field is defined in 9.4.3.2.7.

## 9.4.3.16.3 REPORT PHY EVENT LIST response

Table 350 defines the response format.

Table 350 – REPORT PHY EVENT LIST response

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (41h)							
1	FUNCTION (21h)							
2	FUNCTION RESULT							
3	RESPONSE LENGTH ((n - 7) / 4)							
4	(MSB)	EXPANDER CHANGE COUNT						
5								(LSB)
6	(MSB)	FIRST PHY EVENT LIST DESCRIPTOR INDEX						
7								(LSB)
8	(MSB)	LAST PHY EVENT LIST DESCRIPTOR INDEX						
9								(LSB)
10	PHY EVENT LIST DESCRIPTOR LENGTH							
11	Reserved							
...								
14								
15	NUMBER OF PHY EVENT LIST DESCRIPTORS							
Phy event list descriptor list								
16	Phy event list descriptor (first) (see table 351 in 9.4.3.16.4)							
...								
...	...							
	Phy event list descriptor (last) (see table 351 in 9.4.3.16.4)							
...								
n - 4								
n - 3	(MSB)	CRC						
...								
n								(LSB)

The SMP FRAME TYPE field is defined in 9.4.3.3.2 and shall be set as shown in table 350 for the REPORT PHY EVENT LIST response.

The FUNCTION field is defined in 9.4.3.3.3 and shall be set as shown in table 350 for the REPORT PHY EVENT LIST response.



The FUNCTION RESULT field is defined in 9.4.3.3.4.

The RESPONSE LENGTH field is defined in 9.4.3.3.5 and shall be set as shown in table 350 for the REPORT PHY EVENT LIST response. A RESPONSE LENGTH field set to 00h does not have a special meaning based on the ALLOCATED RESPONSE LENGTH field in the request frame.

The EXPANDER CHANGE COUNT field is defined in the SMP REPORT GENERAL response (see 9.4.3.4).

The FIRST PHY EVENT LIST DESCRIPTOR INDEX field indicates the index of the first phy event list descriptor being returned. If the STARTING PHY EVENT LIST DESCRIPTOR INDEX field in the SMP request frame is set to 0000h, then the management device server shall:

- a) set the FIRST PHY EVENT LIST DESCRIPTOR INDEX field to 0000h;
- b) set the NUMBER OF PHY EVENT LIST DESCRIPTORS field to 00h; and
- c) return no descriptors.

If the STARTING PHY EVENT LIST DESCRIPTOR INDEX field specified in the SMP request frame does not contain a valid descriptor, then the device management server shall set the FIRST PHY EVENT LIST DESCRIPTOR INDEX field to the next index, in ascending order wrapping from FFFFh to 0001h, that contains a valid descriptor, otherwise this field shall be set to the same value as the STARTING PHY EVENT LIST DESCRIPTOR INDEX field in the SMP request frame.

The LAST PHY EVENT LIST DESCRIPTOR INDEX field indicates the last index of the last recorded phy event list descriptor.

The PHY EVENT LIST DESCRIPTOR LENGTH field indicates the length, in dwords, of the phy event list descriptor (see table 351 in 9.4.3.16.4).

The NUMBER OF PHY EVENT LIST DESCRIPTORS field indicates the number of phy event list descriptors in the phy event list descriptor list.

The phy event list descriptor list contains phy event list descriptors as defined in 9.4.3.16.4. The management device server shall return either all the phy event list descriptors that fit in one SMP response frame or all the phy event list descriptors until the index indicated in the LAST PHY EVENT LIST DESCRIPTOR INDEX field is reached. The phy event list descriptor list shall start with the phy event list descriptor indicated by the FIRST PHY EVENT LIST DESCRIPTOR INDEX field and continue with phy event list descriptors sorted in ascending order, wrapping from FFFFh to 0001h, based on the phy event list descriptor index. The phy event list descriptor list shall not contain any truncated phy event list descriptors. If the FIRST PHY EVENT LIST DESCRIPTOR INDEX field is equal to the LAST PHY EVENT LIST DESCRIPTOR INDEX field, then the phy event list descriptor at that index shall be returned.

The CRC field is defined in 9.4.3.3.7.

#### 9.4.3.16.4 Phy event list descriptor

Table 351 defines the phy event list descriptor.

**Table 351 – Phy event list descriptor**

Byte\Bit	7	6	5	4	3	2	1	0
0	Reserved							
1								
2	PHY IDENTIFIER							
3	PHY EVENT SOURCE							
4	PHY EVENT							
...								
7								
8	PEAK VALUE DETECTOR THRESHOLD							
...								
11								

The PHY IDENTIFIER field indicates the phy identifier (see 4.2.10) of the phy for which information is being returned.

The PHY EVENT SOURCE field, defined in table 46 in 4.12, indicates the type of phy event being reported in the PHY EVENT field.

The PHY EVENT field indicates the value (i.e., the count or peak value detected) of the phy event indicated by the PHY EVENT SOURCE field.

If the phy event source is a peak value detector, then the PEAK VALUE DETECTOR THRESHOLD field indicates the value of the peak value detector that causes the expander device to originate a Broadcast (Expander) (see 6.2.6.4). If the phy event source is not a peak value detector, then the PEAK VALUE DETECTOR THRESHOLD field is reserved.

#### 9.4.3.17 REPORT EXPANDER ROUTE TABLE LIST function

##### 9.4.3.17.1 REPORT EXPANDER ROUTE TABLE LIST function overview

The REPORT EXPANDER ROUTE TABLE LIST function returns the contents of an expander-based expander route table (see 4.5.7.4 and 4.8.3.4). The list may be in any order. Self-configuring expander devices shall support this function.

**9.4.3.17.2 REPORT EXPANDER ROUTE TABLE LIST request**

Table 352 defines the request format.

**Table 352 – REPORT EXPANDER ROUTE TABLE LIST request**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (40h)							
1	FUNCTION (22h)							
2	ALLOCATED RESPONSE LENGTH							
3	REQUEST LENGTH (06h)							
4	Reserved							
...								
7								
8	(MSB)	MAXIMUM NUMBER OF EXPANDER ROUTE TABLE DESCRIPTORS						(LSB)
9								
10	(MSB)	STARTING ROUTED SAS ADDRESS INDEX						(LSB)
11								
12	Reserved							
...								
18								
19	STARTING PHY IDENTIFIER							
20	Reserved							
...								
27								
28	(MSB)	CRC						(LSB)
...								
31								

The SMP FRAME TYPE field is defined in 9.4.3.2.2 and shall be set as shown in table 352 for the REPORT EXPANDER ROUTE TABLE LIST request.

The FUNCTION field is defined in 9.4.3.2.3 and shall be set as shown in table 352 for the REPORT EXPANDER ROUTE TABLE LIST request.

The ALLOCATED RESPONSE LENGTH field is defined in 9.4.3.2.4.

The REQUEST LENGTH field is defined in 9.4.3.2.5 and shall be set as shown in table 352 for the REPORT EXPANDER ROUTE TABLE LIST request.

The MAXIMUM NUMBER OF EXPANDER ROUTE TABLE DESCRIPTORS field specifies the maximum number of expander route table descriptors that the management device server shall return.

The STARTING ROUTED SAS ADDRESS INDEX field specifies the index of the first routed SAS address that the management device server shall return in the expander route table descriptor list.

The STARTING PHY IDENTIFIER field specifies the first phy identifier of the phy identifier bit map returned in each expander route table descriptor (see table 354 in 9.4.3.17.3). This field should be set to a multiple of 48 (e.g., 0, 48, or 96) and shall be less than the value indicated in the NUMBER OF PHYS field in the REPORT GENERAL response (see 9.4.3.4).

The CRC field is defined in 9.4.3.2.7.

## 9.4.3.17.3 REPORT EXPANDER ROUTE TABLE LIST response

Table 353 defines the response format.

Table 353 – REPORT EXPANDER ROUTE TABLE LIST response

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (41h)							
1	FUNCTION (22h)							
2	FUNCTION RESULT							
3	RESPONSE LENGTH ((n - 7) / 4)							
4	(MSB)	EXPANDER CHANGE COUNT						(LSB)
5								
6	(MSB)	EXPANDER ROUTE TABLE CHANGE COUNT						(LSB)
7								
8	Reserved				SELF CONFIGURING	ZONE CONFIGURING	CONFIGURING	ZONING ENABLED
9	Reserved							
10	EXPANDER ROUTE TABLE DESCRIPTOR LENGTH							
11	NUMBER OF EXPANDER ROUTE TABLE DESCRIPTORS							
12	(MSB)	FIRST ROUTED SAS ADDRESS INDEX						(LSB)
13								
14	(MSB)	LAST ROUTED SAS ADDRESS INDEX						(LSB)
15								
16	Reserved							
...								
18								
19	STARTING PHY IDENTIFIER							
20	Reserved							
...								
31								
Expander route table descriptor list								
32	Expander route table descriptor (first) (see table 354 in 9.4.3.17.4)							
...								
47								
...	...							
n - 20	Expander route table descriptor (last) (see table 354 in 9.4.3.17.4)							
...								
n - 4								
n - 3	(MSB)	CRC						(LSB)
...								
n								

The SMP FRAME TYPE field is defined in 9.4.3.3.2 and shall be set as shown in table 353 for the REPORT EXPANDER ROUTE TABLE LIST response.

The FUNCTION field is defined in 9.4.3.3.3 and shall be set as shown in table 353 for the REPORT EXPANDER ROUTE TABLE LIST response.

The FUNCTION RESULT field is defined in 9.4.3.3.4.

The RESPONSE LENGTH field is defined in 9.4.3.3.5 and shall be set as shown in table 353 for the REPORT EXPANDER ROUTE TABLE LIST response. A RESPONSE LENGTH field set to 00h does not have a special meaning based on the ALLOCATED RESPONSE LENGTH field in the request frame.

The EXPANDER CHANGE COUNT field is defined in the SMP REPORT GENERAL response (see 9.4.3.4).

The EXPANDER ROUTE TABLE CHANGE COUNT field indicates the number of times the expander route table has been modified by the self-configuring expander device. Self-configuring expander devices shall support this field. This field shall be set to at least 0001h at power on. If the self-configuring expander device modified the expander route table since responding to a previous REPORT EXPANDER ROUTE TABLE LIST request, then it shall increment this field at least once from the value in the previous REPORT EXPANDER ROUTE TABLE LIST response. This field shall wrap to at least 0001h after the maximum value (i.e., FFFFh) has been reached.

NOTE 86 - if a management application client uses the EXPANDER ROUTE TABLE CHANGE COUNT field, then reading it often ensures that it does not increment a multiple of 65 535 times between reading the field.

The SELF CONFIGURING bit is defined in the REPORT GENERAL response (see 9.4.3.4).

The ZONE CONFIGURING bit is defined in the REPORT GENERAL response (see 9.4.3.4).

The CONFIGURING bit is defined in the REPORT GENERAL response (see 9.4.3.4).

The ZONING ENABLED bit is defined in the SMP REPORT GENERAL response (see 9.4.3.4). A ZONING ENABLED bit set to one indicates that the ZONE GROUP field in each expander route table descriptor (see 9.4.3.17.4) is valid. A ZONING ENABLED bit set to zero indicates that the ZONE GROUP field in each expander route table descriptor is not valid.

The EXPANDER ROUTE TABLE DESCRIPTOR LENGTH field indicates the length, in dwords, of each expander route table descriptor (see 9.4.3.17.4).

The NUMBER OF EXPANDER ROUTE TABLE DESCRIPTORS field indicates the number of expander route table descriptors in the expander route table descriptor list.

The FIRST ROUTED SAS ADDRESS INDEX field indicates the index of the first expander route table descriptor reported in the expander route table descriptor list.

The LAST ROUTED SAS ADDRESS INDEX field indicates the index of the last expander route table descriptor reported in the expander route table descriptor list. The management application client may set the STARTING ROUTED SAS ADDRESS INDEX field in its next REPORT EXPANDER ROUTE TABLE LIST request to the value of this field plus one.

The STARTING PHY IDENTIFIER field indicates the value of the STARTING PHY IDENTIFIER field in the request frame, rounded down to a multiple of 48.

The expander route table descriptor list contains expander route table descriptors as defined in 9.4.3.17.4. The management device server shall return either all the expander route table descriptors that fit in one SMP response frame or all the expander route table descriptors until the index indicated in the LAST ROUTED SAS ADDRESS INDEX field is reached. The expander route table descriptor list shall start with the expander route table descriptor indicated by the FIRST ROUTED SAS ADDRESS INDEX field and continue with expander route table descriptors sorted in a vendor specific order based on the routed SAS address index. The expander route table descriptor list shall not contain any truncated expander route table descriptors. If the FIRST ROUTED SAS ADDRESS INDEX field is equal to the LAST ROUTED SAS ADDRESS INDEX field, then the expander route table descriptor at that index shall be returned.

The CRC field is defined in 9.4.3.3.7.

**9.4.3.17.4 Expander route table descriptor**

Table 354 defines the expander route table descriptor.

**Table 354 – Expander route table descriptor**

Byte\Bit	7	6	5	4	3	2	1	0
0	ROUTED SAS ADDRESS							
...								
7								
8	(starting phy identifier + 47)	PHY BIT MAP						(starting phy identifier + 40)
...								
13	(starting phy identifier + 7)							(starting phy identifier)
14	Reserved							
15	ZONE GROUP							

The ROUTED SAS ADDRESS field indicates the routed SAS address.

The PHY BIT MAP field indicates the phys to which connection requests to the SAS address indicated by the ROUTED SAS ADDRESS field may be forwarded. This field is a bit map where each bit position indicates a corresponding phy (e.g., bit zero of byte 13 indicates the phy indicated by the starting phy identifier). A bit set to one indicates that connection requests to the SAS address indicated by the ROUTED SAS ADDRESS field may be forwarded to the corresponding phy. A bit set to zero indicates that connection requests to the SAS address indicated by the ROUTED SAS ADDRESS field are not forwarded to that corresponding phy. Bits representing phys beyond the value of the NUMBER OF PHYS field reported in the REPORT GENERAL response (see 9.4.3.4) shall be set to zero.

The ZONE GROUP field is defined in 4.8.3.1. The ZONE GROUP field is only valid if the ZONING ENABLED bit is set to one (see 9.4.3.17.3).

**9.4.3.18 CONFIGURE GENERAL function**

The CONFIGURE GENERAL function requests actions by the SMP target device containing the management device server. This SMP function may be implemented by any management device server. In zoning expander devices, if zoning is enabled, then this function shall only be processed from SMP initiator ports that have access to zone group 2 (see 4.8.3.2).

Table 355 defines the request format.

**Table 355 – CONFIGURE GENERAL request**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (40h)							
1	FUNCTION (80h)							
2	ALLOCATED RESPONSE LENGTH							
3	REQUEST LENGTH (04h)							
4	(MSB)	EXPECTED EXPANDER CHANGE COUNT						(LSB)
5								
6	(MSB)	SSP CONNECT TIME LIMIT						(LSB)
7								
8	UPDATE TIME TO DELAY	UPDATE SSP TIME LIMIT	UPDATE POWER DONE TIMEOUT	UPDATE STP REJECT TO OPEN LIMIT	UPDATE INITIAL TIME TO REDUCED FUNCTIONALITY	UPDATE STP SMP I_T NEXUS LOSS TIME	UPDATE STP CONNECT TIME LIMIT	UPDATE STP BUS INACTIVITY LIMIT
9	INITIAL TIME TO DELAY EXPANDER FORWARD OPEN INDICATION							
10	(MSB)	STP BUS INACTIVITY LIMIT						(LSB)
11								
12	(MSB)	STP CONNECT TIME LIMIT						(LSB)
13								
14	(MSB)	STP SMP I_T NEXUS LOSS TIME						(LSB)
15								
16	INITIAL TIME TO REDUCED FUNCTIONALITY							
17	POWER DONE TIMEOUT							
18	(MSB)	STP REJECT TO OPEN LIMIT						(LSB)
19								
20	(MSB)	CRC						(LSB)
...								
23								

The SMP FRAME TYPE field is defined in 9.4.3.2.2 and shall be set as shown in table 355 for the CONFIGURE GENERAL request.

The FUNCTION field is defined in 9.4.3.2.3 and shall be set as shown in table 355 for the CONFIGURE GENERAL request.



The ALLOCATED RESPONSE LENGTH field is defined in 9.4.3.2.4.

The REQUEST LENGTH field is defined in 9.4.3.2.5 and shall be set as shown in table 355 for the CONFIGURE GENERAL request.

If the management device server is not in an expander device or the EXPECTED EXPANDER CHANGE COUNT field is set to 0000h, then the EXPECTED EXPANDER CHANGE COUNT field shall be ignored. If the management device server is in an expander device and the EXPECTED EXPANDER CHANGE COUNT field is not set to 0000h, then:

- a) if the EXPECTED EXPANDER CHANGE COUNT field contains the current expander change count (i.e., the value of the EXPANDER CHANGE COUNT field that is returned by an SMP REPORT GENERAL response at this time), then the management device server shall process the function; and
- b) if the EXPECTED EXPANDER CHANGE COUNT field does not contain the current expander change count, then the management device server shall return a function result of INVALID EXPANDER CHANGE COUNT in the response frame (see table 294 in 9.4.3.3).

The SSP CONNECT TIME LIMIT field specifies the maximum duration of an SSP connection (see 4.1.12) in 100  $\mu$ s increments (e.g., a value of 0001h in this field means that the time is less than or equal to 100  $\mu$ s and a value of 0002h in this field means that the time is less than or equal to 200  $\mu$ s). If this time is exceeded, then the expander logical phy requests the end device close the connection (see 6.20.8). A value of 0000h in this field specifies that there is no maximum connection time limit. This value is reported in the SSP CONNECT TIME LIMIT field in the SMP REPORT GENERAL response (see 9.4.3.4). For expander logical phys the maximum connection time limit is enforced by the expander link layer (see 6.19.9).

An UPDATE TIME TO DELAY bit set to one specifies that the INITIAL TIME TO DELAY EXPANDER FORWARD OPEN INDICATION field shall be honored. An UPDATE TIME TO DELAY bit set to zero specifies that the INITIAL TIME TO DELAY EXPANDER FORWARD OPEN INDICATION field shall be ignored.

An UPDATE SSP TIME LIMIT bit set to one specifies that the SSP CONNECT TIME LIMIT field shall be honored. An UPDATE SSP TIME LIMIT bit set to zero specifies that the SSP CONNECT TIME LIMIT field shall be ignored.

An UPDATE POWER DONE TIMEOUT bit set to one specifies that the POWER DONE TIMEOUT field shall be honored. An UPDATE POWER DONE TIMEOUT bit set to zero specifies that the POWER DONE TIMEOUT field shall be ignored.

An UPDATE STP REJECT TO OPEN LIMIT bit set to one specifies that the STP REJECT TO OPEN LIMIT field shall be honored. An UPDATE STP REJECT TO OPEN LIMIT bit set to zero specifies that the STP REJECT TO OPEN LIMIT field shall be ignored.

An UPDATE INITIAL TIME TO REDUCED FUNCTIONALITY bit set to one specifies that the INITIAL TIME TO REDUCED FUNCTIONALITY field shall be honored. An UPDATE INITIAL TIME TO REDUCED FUNCTIONALITY bit set to zero specifies that the INITIAL TIME TO REDUCED FUNCTIONALITY field shall be ignored.

An UPDATE STP SMP I\_T NEXUS LOSS TIME bit set to one specifies that the STP SMP I\_T NEXUS LOSS TIME field shall be honored. An UPDATE STP SMP I\_T NEXUS LOSS TIME bit set to zero specifies that the STP SMP I\_T NEXUS LOSS TIME field shall be ignored.

An UPDATE STP CONNECT TIME LIMIT bit set to one specifies that the STP CONNECT TIME LIMIT field shall be honored. An UPDATE STP CONNECT TIME LIMIT bit set to zero specifies that the STP CONNECT TIME LIMIT field shall be ignored.

An UPDATE STP BUS INACTIVITY LIMIT bit set to one specifies that the STP BUS INACTIVITY LIMIT field shall be honored. An UPDATE STP BUS INACTIVITY LIMIT bit set to zero specifies that the STP BUS INACTIVITY LIMIT field shall be ignored.

The INITIAL TIME TO DELAY EXPANDER FORWARD OPEN INDICATION field specifies the maximum time, in 100 ns increments, that an expander phy shall use, in conjunction with the contents of the HOP COUNT field (see 6.2.6.5.3) to wait before requesting the ECM assign path resources to a connection. A value of 00h in this field specifies that the ECM assign path resources to a connection as resources become available. This value is reported in the INITIAL TIME TO DELAY EXPANDER FORWARD OPEN INDICATION field (see 9.4.3.4). The length of time the expander phy shall wait is determined from the following calculation:

$$\text{delay in assigning resources} = 100 \text{ ns} \times (\text{initial delay}) \times (\text{hop count})$$

where:

delay in assigning resources	is the number of nanoseconds the phy delays before allowing the ECM assign path resources to a connection;
initial delay	is the contents of the INITIAL TIME TO DELAY EXPANDER FORWARD OPEN INDICATION field; and
hop count	is the contents of the HOP COUNT field.

The STP BUS INACTIVITY LIMIT field specifies the maximum time, in 100  $\mu$ s increments, that an STP target port is permitted to maintain a connection (see 4.1.12) while transmitting and receiving SATA\_SYNC. When this time is exceeded, the STP target port shall close the connection. A value of 0000h in this field specifies that there is no bus inactivity time limit. This value is reported in the STP BUS INACTIVITY LIMIT field in the SMP REPORT GENERAL response (see 9.4.3.4). The bus inactivity time limit is enforced by the port layer (see 7.2.3).

The STP CONNECT TIME LIMIT field specifies the maximum duration of a connection (see 4.1.12) in 100  $\mu$ s increments (e.g., a value of 0001h in this field means that the time is less than or equal to 100  $\mu$ s and a value of 0002h in this field means that the time is less than or equal to 200  $\mu$ s). When this time is exceeded, the STP target port shall close the connection at the next opportunity. If the STP target port is transferring a frame when the maximum connection time limit is exceeded, then the STP target port shall complete transfer of the frame before closing the connection. A value of 0000h in this field specifies that there is no maximum connection time limit. This value is reported in the STP CONNECT TIME LIMIT field in the SMP REPORT GENERAL response (see 9.4.3.4). The maximum connection time limit is enforced by the port layer (see 7.2.3).

The STP SMP I\_T NEXUS LOSS TIME field specifies the minimum time that an STP target port or SMP initiator port shall retry connection requests that are rejected with responses indicating the destination port may no longer be present (see 7.2.2) before recognizing an I\_T nexus loss (see 4.4.3).

An STP initiator port or an SMP initiator port should retry connection requests for at least the time indicated by the STP SMP I\_T NEXUS LOSS TIME field in the SMP REPORT GENERAL response for the STP target port to which it is trying to establish a connection.

Table 356 defines the values of the STP SMP I\_T NEXUS LOSS TIME field. This value is enforced by the port layer (see 7.2.2).

**Table 356 – STP SMP I\_T NEXUS LOSS TIME field**

Code <sup>a</sup>	Description
0000h	Vendor specific amount of time.
0001h to FFFEh	Time in one millisecond increments.
FFFFh	The port shall never recognize an I_T nexus loss (i.e., it shall retry the connection requests forever).
<sup>a</sup> The default value of the STP SMP I_T NEXUS LOSS TIME field should be non-zero. It is recommended that this value be 07D0h (i.e., 2 000 ms).	

The INITIAL TIME TO REDUCED FUNCTIONALITY field specifies the minimum time, in 100 ms increments, that an expander device shall wait from originating a Broadcast (Expander) to reducing functionality (see 4.5.8). This value is reported in the INITIAL TIME TO REDUCED FUNCTIONALITY field in the SMP REPORT GENERAL response (see 9.4.3.4).

The POWER DONE TIMEOUT field specifies the maximum time, in one second increments, that a management application layer allows a power consumer device (see 6.14.2) to consume additional power. This value is reported in the POWER DONE TIMEOUT field in the SMP REPORT GENERAL response (see 9.4.3.4). A POWER DONE TIMEOUT field set to 00h specifies that the time limit shall not be changed from the current value. A POWER DONE TIMEOUT field set to FFh specifies that the time limit is vendor specific. The power done timeout limit (see 6.14.4) is enforced by the management application layer.

The STP REJECT TO OPEN LIMIT field specifies the minimum time, in 10  $\mu$ s increments, that an STP port shall wait to establish a connection request with an initiator port on an I\_T nexus after receiving an OPEN\_REJECT (RETRY), OPEN\_REJECT (RESERVED CONTINUE 0), or OPEN\_REJECT (RESERVED CONTINUE 1). This value may be rounded as defined in SPC-4. An STP REJECT TO OPEN LIMIT field set to 0000h specifies that the minimum time is vendor specific. This minimum time is enforced by the port layer (see 7.2.3). This value is reported in the STP REJECT TO OPEN LIMIT field in the SMP REPORT GENERAL response (see 9.4.3.4).

The CRC field is defined in 9.4.3.2.7.

Table 357 defines the response format.

**Table 357 – CONFIGURE GENERAL response**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (41h)							
1	FUNCTION (80h)							
2	FUNCTION RESULT							
3	RESPONSE LENGTH (00h)							
4	(MSB)							
...	CRC							
7	(LSB)							

The SMP FRAME TYPE field is defined in 9.4.3.3.2 and shall be set as shown in table 357 for the CONFIGURE GENERAL response.

The FUNCTION field is defined in 9.4.3.3.3 and shall be set as shown in table 357 for the CONFIGURE GENERAL response.

The FUNCTION RESULT field is defined in 9.4.3.3.4.

The RESPONSE LENGTH field is defined in 9.4.3.3.5 and shall be set as shown in table 357 for the CONFIGURE GENERAL response. A RESPONSE LENGTH field set to 00h does not have a special meaning based on the ALLOCATED RESPONSE LENGTH field in the request frame.

The CRC field is defined in 9.4.3.3.7.

#### 9.4.3.19 ENABLE DISABLE ZONING function

The ENABLE DISABLE ZONING function enables or disables zoning. This SMP function shall be supported by SMP target ports in zoning expander devices (see 4.8). Other SMP target ports shall not support this SMP function. This function is an SMP zone configuration function (see 4.8.6.3).

SMP zone configuration functions change the zoning expander shadow values, which do not become zoning expander current values until the activate step (see 4.8.6.4).

Table 358 defines the request format.

**Table 358 – ENABLE DISABLE ZONING request**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (40h)							
1	FUNCTION (81h)							
2	ALLOCATED RESPONSE LENGTH							
3	REQUEST LENGTH (02h)							
4	(MSB) EXPECTED EXPANDER CHANGE COUNT (LSB)							
5								
6	Reserved						SAVE	
7	Reserved							
8	Reserved						ENABLE DISABLE ZONING	
9	Reserved							
...								
11								
12	(MSB) CRC (LSB)							
...								
15								

The SMP FRAME TYPE field is defined in 9.4.3.2.2 and shall be set as shown in table 358 for the ENABLE DISABLE ZONING request.

The FUNCTION field is defined in 9.4.3.2.3 and shall be set as shown in table 358 for the ENABLE DISABLE ZONING request.

The ALLOCATED RESPONSE LENGTH field is defined in 9.4.3.2.4.

The REQUEST LENGTH field is defined in 9.4.3.2.5 and shall be set as shown in table 358 for the ENABLE DISABLE ZONING request.

The EXPECTED EXPANDER CHANGE COUNT field is defined in the CONFIGURE GENERAL request (see 9.4.3.18).

The SAVE field specifies whether the management device server shall apply the specified changes to the shadow value and/or the saved value of the zoning enabled setting and is defined in table 359.

**Table 359 – SAVE field**

Code	Values updated	Return function result of SAVING NOT SUPPORTED if saving is not supported
00b	Shadow	no
01b	Saved <sup>a</sup>	yes
10b	Saved <sup>a</sup> , if saving is supported, and shadow.	no
11b	Saved <sup>a</sup> and shadow.	yes
<sup>a</sup> Saving only begins during the activate step (see 4.8.6.4). The management device server shall return the function result without waiting for the save to complete and set the SAVING bit to one in the REPORT GENERAL response until the save is complete.		

The ENABLE DISABLE ZONING field is defined in table 360.

**Table 360 – ENABLE DISABLE ZONING field**

Code	Description
00b	No change
01b	Enable zoning
10b	Disable zoning
11b	Reserved

If the ENABLE DISABLE ZONING field is set to 11b (i.e., reserved), then the management device server shall return a function result of UNKNOWN ENABLE DISABLE ZONING VALUE in the response frame (see table 294 in 9.4.3.3).

The CRC field is defined in 9.4.3.2.7.

Table 361 defines the response format.

**Table 361 – ENABLE DISABLE ZONING response**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (41h)							
1	FUNCTION (81h)							
2	FUNCTION RESULT							
3	RESPONSE LENGTH (00h)							
4	CRC							
...								
7								
	(LSB)							

The SMP FRAME TYPE field is defined in 9.4.3.3.2 and shall be set as shown in table 361 for the ENABLE DISABLE ZONING response.

The FUNCTION field is defined in 9.4.3.3.3 and shall be set as shown in table 361 for the ENABLE DISABLE ZONING response.

The FUNCTION RESULT field is defined in 9.4.3.3.4.

The RESPONSE LENGTH field is defined in 9.4.3.3.5 and shall be set as shown in table 361 for the ENABLE DISABLE ZONING response. A RESPONSE LENGTH field set to 00h does not have a special meaning based on the ALLOCATED RESPONSE LENGTH field in the request frame.

The CRC field is defined in 9.4.3.3.7.

#### **9.4.3.20 ZONED BROADCAST function**

The ZONED BROADCAST function requests that the specified Broadcast (see 4.1.15) be forwarded as specified in 4.8.5. This SMP function shall be supported by management device servers in zoning expander devices (see 4.8). Other management device servers shall not support this SMP function. This SMP function shall only be processed from SMP initiator ports that have access to zone group 3 (see 4.8.3.2).

Table 362 defines the request format.

**Table 362 – ZONED BROADCAST request**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (40h)							
1	FUNCTION (85h)							
2	ALLOCATED RESPONSE LENGTH							
3	REQUEST LENGTH ((n - 7) / 4)							
4	Restricted (for an EXPECTED EXPANDER CHANGE COUNT field)							
5								
6	Reserved				BROADCAST TYPE			
7	NUMBER OF BROADCAST SOURCE ZONE GROUPS							
Broadcast source zone group list								
8	BROADCAST SOURCE ZONE GROUP (first)							
...	...							
	BROADCAST SOURCE ZONE GROUP (last)							
	PAD (if needed)							
...								
n - 4								
n - 3	(MSB)							
...	CRC							
n	(LSB)							

The SMP FRAME TYPE field is defined in 9.4.3.2.2 and shall be set as shown in table 362 for the ZONED BROADCAST request.

The FUNCTION field is defined in 9.4.3.2.3 and shall be set as shown in table 362 for the ZONED BROADCAST request.

The ALLOCATED RESPONSE LENGTH field is defined in 9.4.3.2.4.

The REQUEST LENGTH field is defined in 9.4.3.2.5 and shall be set as shown in table 362 for the ZONED BROADCAST request.

The BROADCAST TYPE field specifies the type of Broadcast that shall be forwarded and is defined in Table 363.

**Table 363 – BROADCAST TYPE field**

Code	Description
0000b	Broadcast (Change)
0001b	Broadcast (Reserved Change 0)
0010b	Broadcast (Reserved Change 1)
0011b	Broadcast (SES)
0100b	Broadcast (Expander)
0101b	Broadcast (Asynchronous Event)
0110b	Broadcast (Reserved 3)
0111b	Broadcast (Reserved 4)
1000b	Broadcast (Zone Activate)
All others	Reserved

The NUMBER OF BROADCAST SOURCE ZONE GROUPS field specifies the number of zone groups to which the specified Broadcast is to be forwarded.

The Broadcast source zone group list contains BROADCAST SOURCE ZONE GROUP fields. The Broadcast source zone group list shall contain no more than one entry for each source zone group.

Each BROADCAST SOURCE ZONE GROUP field specifies a source zone group for the Broadcast. The expander device forwards the Broadcast to each destination zone group accessible to that source zone group as specified in 4.8.5.

The PAD field contains zero, one, two, or three bytes set to 00h such that the total length of the SMP request frame is a multiple of four bytes.

The CRC field is defined in 9.4.3.2.7.



Table 364 defines the response format.

**Table 364 – ZONED BROADCAST response**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (41h)							
1	FUNCTION (85h)							
2	FUNCTION RESULT							
3	RESPONSE LENGTH (00h)							
4	CRC							
...								
7								

The SMP FRAME TYPE field is defined in 9.4.3.3.2 and shall be set as shown in table 364 for the ZONED BROADCAST response.

The FUNCTION field is defined in 9.4.3.3.3 and shall be set as shown in table 364 for the ZONED BROADCAST response.

The FUNCTION RESULT field is defined in 9.4.3.3.4.

The RESPONSE LENGTH field is defined in 9.4.3.3.5 and shall be set as shown in table 364 for the ZONED BROADCAST response. A RESPONSE LENGTH field set to 00h does not have a special meaning based on the ALLOCATED RESPONSE LENGTH field in the request frame.

The CRC field is defined in 9.4.3.3.7.

#### 9.4.3.21 ZONE LOCK function

The ZONE LOCK function locks a zoning expander device to provide exclusive access to SMP zone configuration functions (see 4.8.6.3) for one zone manager. All zoning expander devices shall support this function.

If:

- the ZONING ENABLED bit is set to one, the ZONE LOCKED bit is set to zero in the REPORT GENERAL response (see 9.4.3.4), and the SMP initiator port has access to zone group 2 (see 4.8.3.2);
- the ZONE LOCKED bit is set to zero in the REPORT GENERAL response and the PHYSICAL PRESENCE ASSERTED bit is set to one in the REPORT GENERAL response;
- the ZONE LOCKED bit is set to zero in the REPORT GENERAL response and the request contains the correct zone manager password (see 4.8.1); or
- the ZONE LOCKED bit is set to one in the REPORT GENERAL response and the request originated from the active zone manager,

then the management device server shall:

- set the ACTIVE ZONE MANAGER SAS ADDRESS field to the SAS address of the SMP initiator port in the ZONE LOCK response and the REPORT GENERAL response; and
- set the ZONE LOCKED bit to one in the REPORT GENERAL response,

otherwise the management device server shall return a function result of NO MANAGEMENT ACCESS RIGHTS.

When the management device server changes the ZONE LOCKED bit from zero to one, the locked zoning expander device sets the zoning expander shadow values equal to the zoning expander current values.

Table 365 defines the request format.

**Table 365 – ZONE LOCK request**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (40h)							
1	FUNCTION (86h)							
2	ALLOCATED RESPONSE LENGTH							
3	REQUEST LENGTH (09h)							
4	(MSB)	EXPECTED EXPANDER CHANGE COUNT						
5								(LSB)
6	(MSB)	ZONE LOCK INACTIVITY TIME LIMIT						
7								(LSB)
8	(MSB)	ZONE MANAGER PASSWORD						
...								
39								(LSB)
40	(MSB)	CRC						
...								
43								(LSB)

The SMP FRAME TYPE field is defined in 9.4.3.2.2 and shall be set as shown in table 365 for the ZONE LOCK request.

The FUNCTION field is defined in 9.4.3.2.3 and shall be set as shown in table 365 for the ZONE LOCK request.

The ALLOCATED RESPONSE LENGTH field is defined in 9.4.3.2.4.

The REQUEST LENGTH field is defined in 9.4.3.2.5 and shall be set as shown in table 365 for the ZONE LOCK request.

The EXPECTED EXPANDER CHANGE COUNT field is defined in the CONFIGURE GENERAL request (see 9.4.3.18).

The ZONE LOCK INACTIVITY TIME LIMIT field specifies the minimum time that the locked expander device shall allow between any SMP zone configuration function requests or SMP ZONE LOCK requests from the active zone manager (i.e., the maximum time that a zone manager may allow to pass without accessing the locked expander device) and is reported in the SMP REPORT GENERAL response (see 9.4.3.21). This field specifies the number of 100 ms increments that a locked zoning expander device shall remain locked without processing any SMP zone configuration function or SMP ZONE LOCK function (e.g., a value of 0001h in this field means that the time is less than or equal to 100 ms and a value of 0002h in this field means that the time is less than or equal to 200 ms). A value of 0000h in this field specifies that there is no zone lock inactivity time limit (i.e., the zone lock inactivity timer is disabled).

The ZONE MANAGER PASSWORD field specifies a password used to allow permission to lock without physical presence being asserted.

The CRC field is defined in 9.4.3.2.7.

Table 366 defines the response format.

**Table 366 – ZONE LOCK response**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (41h)							
1	FUNCTION (86h)							
2	FUNCTION RESULT							
3	RESPONSE LENGTH (03h)							
4	Reserved							
...								
7								
8	ACTIVE ZONE MANAGER SAS ADDRESS							
...								
15								
16	(MSB)	CRC						(LSB)
...								
19								

The SMP FRAME TYPE field is defined in 9.4.3.3.2 and shall be set as shown in table 366 for the ZONE LOCK response.

The FUNCTION field is defined in 9.4.3.3.3 and shall be set as shown in table 366 for the ZONE LOCK response.

The FUNCTION RESULT field is defined in 9.4.3.3.4.

The RESPONSE LENGTH field is defined in 9.4.3.3.5 and shall be set as shown in table 366 for the ZONE LOCK response. A RESPONSE LENGTH field set to 00h does not have a special meaning based on the ALLOCATED RESPONSE LENGTH field in the request frame.

The ACTIVE ZONE MANAGER SAS ADDRESS field is defined in the REPORT GENERAL response (see 9.4.3.4).

The CRC field is defined in 9.4.3.3.7.

#### 9.4.3.22 ZONE ACTIVATE function

The ZONE ACTIVATE function causes the zoning expander device to set the zoning expander current values equal to the zoning expander shadow values (see 4.8.6.4). All zoning expander devices shall support this function. This function is an SMP zone configuration function (see 4.8.6.3).

Table 367 defines the request format.

**Table 367 – ZONE ACTIVATE request**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (40h)							
1	FUNCTION (87h)							
2	ALLOCATED RESPONSE LENGTH							
3	REQUEST LENGTH (01h)							
4	(MSB)	EXPECTED EXPANDER CHANGE COUNT						
5								(LSB)
6	Reserved							
7								
8	(MSB)	CRC						
...								
11								(LSB)

The SMP FRAME TYPE field is defined in 9.4.3.2.2 and shall be set as shown in table 367 for the ZONE ACTIVATE request.

The FUNCTION field is defined in 9.4.3.2.3 and shall be set as shown in table 367 for the ZONE ACTIVATE request.

The ALLOCATED RESPONSE LENGTH field is defined in 9.4.3.2.4.

The REQUEST LENGTH field is defined in 9.4.3.2.5 and shall be set as shown in table 367 for the ZONE ACTIVATE request.

The EXPECTED EXPANDER CHANGE COUNT field is defined in the CONFIGURE GENERAL request (see 9.4.3.18).

The CRC field is defined in 9.4.3.2.7.

Table 368 defines the response format.

**Table 368 – ZONE ACTIVATE response**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (41h)							
1	FUNCTION (87h)							
2	FUNCTION RESULT							
3	RESPONSE LENGTH (00h)							
4	CRC							
...								
7								

The SMP FRAME TYPE field is defined in 9.4.3.3.2 and shall be set as shown in table 368 for the ZONE ACTIVATE response.

The FUNCTION field is defined in 9.4.3.3.3 and shall be set as shown in table 368 for the ZONE ACTIVATE request.

The FUNCTION RESULT field is defined in 9.4.3.3.4.

The RESPONSE LENGTH field is defined in 9.4.3.3.5 and shall be set as shown in table 368 for the ZONE ACTIVATE request. A RESPONSE LENGTH field set to 00h does not have a special meaning based on the ALLOCATED RESPONSE LENGTH field in the request frame.

The CRC field is defined in 9.4.3.3.7.

#### 9.4.3.23 ZONE UNLOCK function

The ZONE UNLOCK function unlocks a zoning expander device (see 4.8.6.5). All zoning expander devices shall support this function. This function is an SMP zone configuration function (see 4.8.6.3).

If a locked zoning expander device processes a ZONE UNLOCK request from the active zone manager, then the management device server shall set the ZONE LOCKED bit to zero in the REPORT GENERAL response (see 9.4.3.4).

Table 369 defines the request format.

**Table 369 – ZONE UNLOCK request**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (40h)							
1	FUNCTION (88h)							
2	ALLOCATED RESPONSE LENGTH							
3	REQUEST LENGTH (01h)							
4	Restricted (for an EXPECTED EXPANDER CHANGE COUNT field)							
5								
6	Reserved							ACTIVATE REQUIRED
7	Reserved							
8	(MSB)							
...	CRC							
11	(LSB)							

The SMP FRAME TYPE field is defined in 9.4.3.2.2 and shall be set as shown in table 369 for the ZONE UNLOCK request.

The FUNCTION field is defined in 9.4.3.2.3 and shall be set as shown in table 369 for the ZONE UNLOCK request.

The ALLOCATED RESPONSE LENGTH field is defined in 9.4.3.2.4.

The REQUEST LENGTH field is defined in 9.4.3.2.5 and shall be set as shown in table 369 for the ZONE UNLOCK request.

An ACTIVATE REQUIRED bit set to one specifies that the management device server shall unlock the zoning expander device only if the activate step has been completed. An ACTIVATE REQUIRED bit set to zero specifies that the management device server shall unlock the zoning expander device regardless of whether the activate step has been completed.

The CRC field is defined in 9.4.3.2.7.

Table 370 defines the response format.

**Table 370 – ZONE UNLOCK response**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (41h)							
1	FUNCTION (88h)							
2	FUNCTION RESULT							
3	RESPONSE LENGTH (00h)							
4	CRC							
...								
7								

The SMP FRAME TYPE field is defined in 9.4.3.3.2 and shall be set as shown in table 370 for the ZONE UNLOCK response.

The FUNCTION field is defined in 9.4.3.3.3 and shall be set as shown in table 370 for the ZONE UNLOCK response.

The FUNCTION RESULT field is defined in 9.4.3.3.4.

The RESPONSE LENGTH field is defined in 9.4.3.3.5 and shall be set as shown in table 370 for the ZONE UNLOCK response. A RESPONSE LENGTH field set to 00h does not have a special meaning based on the ALLOCATED RESPONSE LENGTH field in the request frame.

The CRC field is defined in 9.4.3.3.7.

#### **9.4.3.24 CONFIGURE ZONE MANAGER PASSWORD function**

The CONFIGURE ZONE MANAGER PASSWORD function configures the zone manager password (see 4.8.1). This SMP function may be supported by a management device server in a zoning expander device. Other management device servers shall not support this SMP function. This SMP function shall only be processed if the request is received from any:

- a) SMP initiator port and specifies the correct zone manager password; or
- b) SMP initiator port while physical presence is asserted.

Table 371 defines the request format.

**Table 371 – CONFIGURE ZONE MANAGER PASSWORD request**

Byte\Bit	7	6	5	4	3	2	1	0	
0	SMP FRAME TYPE (40h)								
1	FUNCTION (89h)								
2	ALLOCATED RESPONSE LENGTH								
3	REQUEST LENGTH (11h)								
4	(MSB)	EXPECTED EXPANDER CHANGE COUNT						(LSB)	
5									
6	Reserved						SAVE		
7	Reserved								
8	(MSB)	ZONE MANAGER PASSWORD						(LSB)	
...									
39									
40	(MSB)	NEW ZONE MANAGER PASSWORD						(LSB)	
...									
71									
72	(MSB)	CRC						(LSB)	
...									
75									

The SMP FRAME TYPE field is defined in 9.4.3.2.2 and shall be set as shown in table 371 for the CONFIGURE ZONE MANAGER PASSWORD request.

The FUNCTION field is defined in 9.4.3.2.3 and shall be set as shown in table 371 for the CONFIGURE ZONE MANAGER PASSWORD request.

The ALLOCATED RESPONSE LENGTH field is defined in 9.4.3.2.4.

The REQUEST LENGTH field is defined in 9.4.3.2.5 and shall be set as shown in table 371 for the CONFIGURE ZONE MANAGER PASSWORD request.

The EXPECTED EXPANDER CHANGE COUNT field is defined in the CONFIGURE GENERAL request (see 9.4.3.18).



The SAVE field specifies whether the management device server shall apply the specified changes to the current value and/or the saved value of the zone manager password and is defined in table 372.

Table 372 – SAVE field

Code	Values updated	Return function result of SAVING NOT SUPPORTED if saving is not supported
00b	Current <sup>a</sup>	no
01b	Saved <sup>b</sup>	yes
10b	Saved <sup>b</sup> , if saving is supported, and current	no
11b	Saved <sup>b</sup> and current	yes
<sup>a</sup> The CONFIGURE ZONE PASSWORD function updates the current zone manager password, not a shadow zone manager password. <sup>b</sup> The management device server shall return the function result without waiting for the save to complete and set the SAVING bit to one in the REPORT GENERAL response until the save is complete.		

If physical presence is not asserted and the ZONE MANAGER PASSWORD field does not match the current zone manager password maintained by the management device server, then the management device server shall return a function result of NO MANAGEMENT ACCESS RIGHTS in the response frame (see table 294 in 9.4.3.3).

The NEW ZONE MANAGER PASSWORD field specifies a new value for the zone manager password maintained by the management device server. A NEW ZONE MANAGER PASSWORD field set to ZERO (see table 35 in 4.8.1) specifies that the zone manager password is disabled and all zone managers have access. A NEW ZONE MANAGER PASSWORD field set to DISABLED (see table 35 in 4.8.1) specifies that the zone manager password is disabled and access shall only be allowed if physical presence is asserted. If the expander device does not support a zone manager password of DISABLED, then the management device server shall return a function result of DISABLED PASSWORD NOT SUPPORTED in the response frame (see table 294 in 9.4.3.3).

The CRC field is defined in 9.4.3.2.7.

Table 373 defines the response format.

Table 373 – CONFIGURE ZONE MANAGER PASSWORD response

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (41h)							
1	FUNCTION (89h)							
2	FUNCTION RESULT							
3	RESPONSE LENGTH (00h)							
4	(MSB)							
...	CRC							
7	(LSB)							

The SMP FRAME TYPE field is defined in 9.4.3.3.2 and shall be set as shown in table 373 for the CONFIGURE ZONE MANAGER PASSWORD response.

The FUNCTION field is defined in 9.4.3.3.3 and shall be set as shown in table 373 for the CONFIGURE ZONE MANAGER PASSWORD response.

The FUNCTION RESULT field is defined in 9.4.3.3.4.

The RESPONSE LENGTH field is defined in 9.4.3.3.5 and shall be set as shown in table 373 for the CONFIGURE ZONE MANAGER PASSWORD response. A RESPONSE LENGTH field set to 00h does not have a special meaning based on the ALLOCATED RESPONSE LENGTH field in the request frame.

The CRC field is defined in 9.4.3.3.7.

#### **9.4.3.25 CONFIGURE ZONE PHY INFORMATION function**

##### **9.4.3.25.1 CONFIGURE ZONE PHY INFORMATION function overview**

The CONFIGURE ZONE PHY INFORMATION function configures zone phy information for one or more phys in a locked zoning expander device. This function shall be supported by all zoning expander devices. This function is an SMP zone configuration function (see 4.8.6.3).

SMP zone configuration functions change the zoning expander shadow values, which do not become zoning expander current values until the activate step (see 4.8.6.4).

**9.4.3.25.2 CONFIGURE ZONE PHY INFORMATION request**

Table 374 defines the request format.

**Table 374 – CONFIGURE ZONE PHY INFORMATION request**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (40h)							
1	FUNCTION (8Ah)							
2	ALLOCATED RESPONSE LENGTH							
3	REQUEST LENGTH ((n - 7) / 4)							
4	(MSB) _____ EXPECTED EXPANDER CHANGE COUNT _____ (LSB)							
5								
6	ZONE PHY CONFIGURATION DESCRIPTOR LENGTH						SAVE	
7	NUMBER OF ZONE PHY CONFIGURATION DESCRIPTORS							
Zone phy configuration descriptor list								
8	Zone phy configuration descriptor (first) (see table 376 in 9.4.3.25.3) _____							
...								
11								
...	...							
n - 7	Zone phy configuration descriptor (last) (see table 376 in 9.4.3.25.3) _____							
...								
n - 4								
n - 3	(MSB) _____							
...	CRC _____							
n	(LSB)							

The SMP FRAME TYPE field is defined in 9.4.3.2.2 and shall be set as shown in table 374 for the CONFIGURE ZONE PHY INFORMATION request.

The FUNCTION field is defined in 9.4.3.2.3 and shall be set as shown in table 374 for the CONFIGURE ZONE PHY INFORMATION request.

The ALLOCATED RESPONSE LENGTH field is defined in 9.4.3.2.4.

The REQUEST LENGTH field is defined in 9.4.3.2.5 and shall be set as shown in table 374 for the CONFIGURE ZONE PHY INFORMATION request.

The EXPECTED EXPANDER CHANGE COUNT field is defined in the SMP CONFIGURE GENERAL request (see 9.4.3.18).

The ZONE PHY CONFIGURATION DESCRIPTOR LENGTH field indicates the length, in dwords, of the zone phy configuration descriptor (see 9.4.3.25.3).

The SAVE field specifies whether the management device server shall apply the specified changes to the shadow value and/or the saved value of the zone phy information and is defined in table 375.

**Table 375 – SAVE field**

Code	Values updated	Return function result of SAVING NOT SUPPORTED if saving is not supported
00b	Shadow	no
01b	Saved <sup>a</sup>	yes
10b	Saved <sup>a</sup> , if saving is supported, and shadow	no
11b	Saved <sup>a</sup> and shadow	yes
<sup>a</sup> Saving only begins during the activate step (see 4.8.6.4). The management device server shall return the function result without waiting for the save to complete and set the SAVING bit to one in the REPORT GENERAL response until the save is complete.		

The NUMBER OF ZONE PHY CONFIGURATION DESCRIPTORS field specifies the number of zone phy configuration descriptors in the zone phy configuration descriptor list.

The zone phy configuration descriptor list contain a zone phy configuration descriptors as defined in 9.4.3.25.3 for each expander phy in the expander device. The zone phy configuration descriptor list shall contain no more than one zone phy configuration descriptor with the same value in the PHY IDENTIFIER field.

NOTE 87 - Because the maximum number of response bytes is 1 023 bytes (see 8.4.3), the length of the header is 8 bytes, and the length of the zone phy configuration descriptor is 4 bytes, the zone phy configuration descriptor list has a maximum of 254 entries.

The CRC field is defined in 9.4.3.2.7.

### 9.4.3.25.3 Zone phy configuration descriptor

Table 376 defines the zone phy configuration descriptor.

**Table 376 – Zone phy configuration descriptor**

Byte\Bit	7	6	5	4	3	2	1	0
0	PHY IDENTIFIER							
1	Reserved		INSIDE ZPSDS PERSISTENT	REQUESTED INSIDE ZPSDS	Reserved	ZONE GROUP PERSISTENT	Reserved	
2	Reserved							
3	ZONE GROUP							

The PHY IDENTIFIER field specifies the phy to which the zone phy configuration descriptor information shall be applied.

The INSIDE ZPSDS PERSISTENT bit specifies the value of the INSIDE ZPSDS PERSISTENT bit in the zone phy information (see 4.8.3.1).

The REQUESTED INSIDE ZPSDS bit specifies the value of the REQUESTED INSIDE ZPSDS bit in the zone phy information (see 4.8.3.1).

The ZONE GROUP PERSISTENT bit specifies the value of the ZONE GROUP PERSISTENT bit in the zone phy information (see 4.8.3.1).

The ZONE GROUP field specifies the value of the ZONE GROUP field in the zone phy information (see 4.8.3.1).

### 9.4.3.25.4 CONFIGURE ZONE PHY INFORMATION response

Table 377 defines the response format.

**Table 377 – CONFIGURE ZONE PHY INFORMATION response**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (41h)							
1	FUNCTION (8Ah)							
2	FUNCTION RESULT							
3	RESPONSE LENGTH (00h)							
4	(MSB)							
...	CRC							
7	(LSB)							

The SMP FRAME TYPE field is defined in 9.4.3.3.2 and shall be set as shown in table 377 for the CONFIGURE ZONE PHY INFORMATION response.

The FUNCTION field is defined in 9.4.3.3.3 and shall be set as shown in table 377 for the CONFIGURE ZONE PHY INFORMATION response.

The FUNCTION RESULT field is defined in 9.4.3.3.4.

The RESPONSE LENGTH field is defined in 9.4.3.3.5 and shall be set as shown in table 377 for the CONFIGURE ZONE PHY INFORMATION response. A RESPONSE LENGTH field set to 00h does not have a special meaning based on the ALLOCATED RESPONSE LENGTH field in the request frame.

The CRC field is defined in 9.4.3.3.7.

#### **9.4.3.26 CONFIGURE ZONE PERMISSION TABLE function**

##### **9.4.3.26.1 CONFIGURE ZONE PERMISSION TABLE function overview**

The CONFIGURE ZONE PERMISSION TABLE function configures the zone permission table. This function shall be supported by all zoning expander devices. This function is an SMP zone configuration function (see 4.8.6.3).

SMP zone configuration functions change the zoning expander shadow values, which do not become zoning expander current values until the zoning expander device processes the activate step (see 4.8.6.4).

Annex I describes examples of using multiple zone configuration descriptors.

## 9.4.3.26.2 CONFIGURE ZONE PERMISSION TABLE request

Table 378 defines the request format.

Table 378 – CONFIGURE ZONE PERMISSION TABLE request

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (40h)							
1	FUNCTION (8Bh)							
2	ALLOCATED RESPONSE LENGTH							
3	REQUEST LENGTH ((n - 7) / 4)							
4	(MSB) _____ EXPECTED EXPANDER CHANGE COUNT _____ (LSB)							
5								
6	STARTING SOURCE ZONE GROUP							
7	NUMBER OF ZONE PERMISSION CONFIGURATION DESCRIPTORS							
8	NUMBER OF ZONE GROUPS	Reserved					SAVE	
9	ZONE PERMISSION CONFIGURATION DESCRIPTOR LENGTH							
10	Reserved							
...								
15								
Zone permission configuration descriptor list								
16	Zone permission configuration descriptor (first) (see table 382 or table 383 in 9.4.3.26.3)							
...								
31 or 47								
...	...							
(n - 20) or (n - 36)	Zone permission configuration descriptor (last) (see table 382 or table 383 in 9.4.3.26.3)							
...								
n - 4								
n - 3	(MSB) _____							
...	CRC							
n	(LSB)							

The SMP FRAME TYPE field is defined in 9.4.3.2.2 and shall be set as shown in table 378 for the CONFIGURE ZONE PERMISSION TABLE request.

The FUNCTION field is defined in 9.4.3.2.3 and shall be set as shown in table 378 for the CONFIGURE ZONE PERMISSION TABLE request.

The ALLOCATED RESPONSE LENGTH field is defined in 9.4.3.2.4.

The REQUEST LENGTH field is defined in 9.4.3.2.5 and shall be set as shown in table 378 for the CONFIGURE ZONE PERMISSION TABLE request.

The EXPECTED EXPANDER CHANGE COUNT field is defined in the SMP CONFIGURE GENERAL request (see 9.4.3.18).

The STARTING SOURCE ZONE GROUP field specifies the first source zone group (i.e., s) to be written with the first zone permission configuration descriptor.

The NUMBER OF ZONE PERMISSION CONFIGURATION DESCRIPTORS field specifies the number of zone permission configuration descriptors in the zone permission configuration descriptor list.

The NUMBER OF ZONE GROUPS field specifies the number of elements in each zone permission configuration descriptor and is defined in table 379.

**Table 379 – NUMBER OF ZONE GROUPS field**

Code	Description
00b	128 zone groups
01b	256 zone groups
All others	Reserved

The SAVE field specifies whether the management device server shall apply the specified changes to the shadow value and/or the saved value of the zone permission table and is defined in table 380.

**Table 380 – SAVE field**

Code	Values updated	Return function result of SAVING NOT SUPPORTED if saving is not supported
00b	Shadow	no
01b	Saved <sup>a</sup>	yes
10b	Saved <sup>a</sup> , if saving is supported, and shadow	no
11b	Saved <sup>a</sup> and shadow	yes
<sup>a</sup> Saving only begins during the activate step (see 4.8.6.4). The management device server shall return the function result without waiting for the save to complete and set the SAVING bit to one in the REPORT GENERAL response until the save is complete.		

The ZONE PERMISSION CONFIGURATION DESCRIPTOR LENGTH field indicates the length, in dwords, of the zone permission configuration descriptor (see 9.4.3.26.3).



The zone permission configuration descriptor list contains a zone permission configuration descriptor as defined in 9.4.3.26.3 for each source zone group in ascending order starting with the source zone group specified in the STARTING SOURCE ZONE GROUP field. The management device server shall process the zone permission configuration descriptors in order (i.e., a subsequent zone permission configuration descriptor overrides a previous zone permission configuration descriptor).

The CRC field is defined in 9.4.3.2.7.

#### 9.4.3.26.3 Zone permission configuration descriptor

The zone permission configuration descriptor format is based on the NUMBER OF ZONE GROUPS field as defined in table 381.

**Table 381 – Zone permission configuration descriptors**

NUMBER OF ZONE GROUPS field	Zone permission configuration descriptor format
00b	Table 382
01b	Table 383
All others	Reserved

Table 382 defines the zone permission configuration descriptor for a source zone group (i.e., s) containing 128 zone groups.

**Table 382 – Zone permission configuration descriptor for source zone group for 128 zone groups**

Byte\Bit	7	6	5	4	3	2	1	0
0	ZP[s, 127]	ZP[s, 126]	ZP[s, 125]	ZP[s, 124]	ZP[s, 123]	ZP[s, 122]	ZP[s, 121]	ZP[s, 120]
...	...							
15	ZP[s, 7] (ignored)	ZP[s, 6] (ignored)	ZP[s, 5] (ignored)	ZP[s, 4] (ignored)	ZP[s, 3]	ZP[s, 2]	ZP[s, 1] (ignored)	ZP[s, 0] (ignored)

Table 383 defines the zone permission configuration descriptor for a source zone group (i.e., s) containing 256 zone groups.

**Table 383 – Zone permission configuration descriptor for source zone group for 256 zone groups**

Byte\Bit	7	6	5	4	3	2	1	0
0	ZP[s, 255]	ZP[s, 254]	ZP[s, 253]	ZP[s, 252]	ZP[s, 251]	ZP[s, 250]	ZP[s, 249]	ZP[s, 248]
...	...							
31	ZP[s, 7] (ignored)	ZP[s, 6] (ignored)	ZP[s, 5] (ignored)	ZP[s, 4] (ignored)	ZP[s, 3]	ZP[s, 2]	ZP[s, 1] (ignored)	ZP[s, 0] (ignored)

The zone permission configuration descriptor contains all of the zone permission table entries for the source zone group (i.e.,  $s$ ). To preserve symmetry about the  $ZP[s, s]$  table axis, the management device server shall apply the same value to both the source and destination zone groups for the zone permission entries.

Table 384 defines how the zone permission descriptor bits shall be set by the management application client and processed by the management device server.

**Table 384 – Zone permission configuration descriptor bit requirements**

Source zone group (i.e., $s$ )	Management application client requirements <sup>a</sup>	Management device server requirements <sup>a</sup>
0	$ZP[s, 0]$ shall be set to zero. $ZP[s, 1]$ shall be set to one. $ZP[s, 2 \text{ to } (z-1)]$ shall be set to zero.	$ZP[s, 0 \text{ to } (z-1)]$ shall be ignored.
1	$ZP[s, 0 \text{ to } (z-1)]$ shall be set to one.	$ZP[s, 0 \text{ to } (z-1)]$ shall be ignored.
4, 5, 6, or 7	$ZP[s, 0]$ shall be set to zero. $ZP[s, 1]$ shall be set to one. $ZP[s, 4 \text{ to } (z-1)]$ shall be set to zero.	$ZP[s, 0 \text{ to } (z-1)]$ shall be ignored.
2, 3, or 8 to $(z-1)$ <sup>a</sup>	$ZP[s, 0]$ shall be set to zero. $ZP[s, 1]$ shall be set to one. $ZP[s, 2 \text{ to } 3]$ may be set to zero or one. $ZP[s, 4 \text{ to } 7]$ shall be set to zero. $ZP[s, 8 \text{ to } (z-1)]$ may be set to zero or one.	$ZP[s, 0 \text{ to } 1]$ shall be ignored. $ZP[s, 2 \text{ to } 3]$ shall be processed. $ZP[s, 4 \text{ to } 7]$ shall be ignored. $ZP[s, 8 \text{ to } (z-1)]$ shall be processed. For each source zone group $t$ other than $s$ , $ZP[t, s]$ shall be set to $ZP[s, t]$ .
<sup>a</sup> The number of zone groups (i.e., $z$ ) is specified in NUMBER OF ZONE GROUPS field.		

#### 9.4.3.26.4 CONFIGURE ZONE PERMISSION TABLE response

Table 385 defines the response format.

**Table 385 – CONFIGURE ZONE PERMISSION TABLE response**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (41h)							
1	FUNCTION (8Bh)							
2	FUNCTION RESULT							
3	RESPONSE LENGTH (00h)							
4	(MSB)							
...	CRC							
7	(LSB)							

The SMP FRAME TYPE field is defined in 9.4.3.3.2 and shall be set as shown in table 385 for the CONFIGURE ZONE PERMISSION TABLE response.

The FUNCTION field is defined in 9.4.3.3.3 and shall be set as shown in table 385 for the CONFIGURE ZONE PERMISSION TABLE response.

The FUNCTION RESULT field is defined in 9.4.3.3.4.

The RESPONSE LENGTH field is defined in 9.4.3.3.5 and shall be set as shown in table 385 for the CONFIGURE ZONE PERMISSION TABLE response. A RESPONSE LENGTH field set to 00h does not have a special meaning based on the ALLOCATED RESPONSE LENGTH field in the request frame.

The CRC field is defined in 9.4.3.3.7.

#### **9.4.3.27 CONFIGURE ROUTE INFORMATION function**

The CONFIGURE ROUTE INFORMATION function sets an expander route entry within the expander route table of a configurable expander device. This SMP function shall be supported by management device servers in expander devices if the CONFIGURABLE ROUTE TABLE field is set to one in the SMP REPORT GENERAL response (see 9.4.3.4). Other management device servers shall not support this SMP function.

Table 386 defines the request format.

**Table 386 – CONFIGURE ROUTE INFORMATION request**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (40h)							
1	FUNCTION (90h)							
2	ALLOCATED RESPONSE LENGTH							
3	REQUEST LENGTH (00h or 09h)							
4	(MSB)	EXPECTED EXPANDER CHANGE COUNT						
5								(LSB)
6	(MSB)	EXPANDER ROUTE INDEX						
7								(LSB)
8	Reserved							
9	PHY IDENTIFIER							
10	Reserved							
11								
12	DISABLE EXPANDER ROUTE ENTRY	Reserved						
13								
...								
15								
16	ROUTED SAS ADDRESS							
...								
23								
24								
...	Reserved							
39								
40	(MSB)	CRC						
...								
43								(LSB)

The SMP FRAME TYPE field is defined in 9.4.3.2.2 and shall be set as shown in table 386 for the CONFIGURE ROUTE INFORMATION request.

The FUNCTION field is defined in 9.4.3.2.3 and shall be set as shown in table 386 for the CONFIGURE ROUTE INFORMATION request.

The ALLOCATED RESPONSE LENGTH field is defined in 9.4.3.2.4.

If the ALLOCATED RESPONSE LENGTH field is set to 00h, then the management device server shall:

- a) set the RESPONSE LENGTH field to 00h in the response frame; and
- b) return the first 4 bytes defined in table 387 plus the CRC field as the response frame.

If the ALLOCATED RESPONSE LENGTH field is not set to 00h, then the management device server shall:

- a) set the RESPONSE LENGTH field in the response frame to the value defined in table 387 (i.e., 00h); and
- b) return the response frame as defined in 9.4.3.2.4.

NOTE 88 - Future versions of this standard may change the value defined in table 387.

The REQUEST LENGTH field is defined in 9.4.3.2.5 and shall be set to one of the values defined in table 386 based on the LONG RESPONSE bit in the REPORT GENERAL response (see 9.4.3.4). A REQUEST LENGTH field set to 00h specifies that there are 9 dwords before the CRC field.

The EXPECTED EXPANDER CHANGE COUNT field is defined in the SMP CONFIGURE GENERAL request (see 9.4.3.18).

The EXPANDER ROUTE INDEX field specifies the expander route index for the expander route entry being configured (see 4.5.7.4).

The PHY IDENTIFIER field specifies the phy identifier (see 4.2.10) of the phy for which the expander route entry is being configured (see 4.5.7.4).

The DISABLE EXPANDER ROUTE ENTRY bit specifies whether the ECM shall use the expander route entry to route connection requests (see 4.5.7.4). If the DISABLE EXPANDER ROUTE ENTRY bit is set to zero, then the ECM shall use the expander route entry to route connection requests. If the DISABLE EXPANDER ROUTE ENTRY bit is set to one, then the ECM shall not use the expander route entry to route connection requests.

The ROUTED SAS ADDRESS field specifies the SAS address for the expander route entry being configured (see 4.5.7.4).

The CRC field is defined in 9.4.3.2.7.

Table 387 defines the response format.

**Table 387 – CONFIGURE ROUTE INFORMATION response**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (41h)							
1	FUNCTION (90h)							
2	FUNCTION RESULT							
3	RESPONSE LENGTH (00h)							
4	(MSB)							
...	CRC							
7	(LSB)							

The SMP FRAME TYPE field is defined in 9.4.3.3.2 and shall be set as shown in table 387 for the CONFIGURE ROUTE INFORMATION response.

The FUNCTION field is defined in 9.4.3.3.3 and shall be set as shown in table 387 for the CONFIGURE ROUTE INFORMATION response.

The FUNCTION RESULT field is defined in 9.4.3.3.4.

The RESPONSE LENGTH field is defined in 9.4.3.3.5 and shall be set as shown in table 387 for the CONFIGURE ROUTE INFORMATION response.

The CRC field is defined in 9.4.3.3.7.

#### **9.4.3.28 PHY CONTROL function**

The PHY CONTROL function requests actions by the specified phy. This SMP function may be implemented by any management device server. In zoning expander devices if zoning is enabled, then this function shall only be processed from SMP initiator ports that have access to zone group 2 or the zone group of the specified phy (see 4.8.3.2).

Table 388 defines the request format.

**Table 388 – PHY CONTROL request**

Byte\Bit	7	6	5	4	3	2	1	0							
0	SMP FRAME TYPE (40h)														
1	FUNCTION (91h)														
2	ALLOCATED RESPONSE LENGTH														
3	REQUEST LENGTH (00h or 09h)														
4	(MSB)	EXPECTED EXPANDER CHANGE COUNT						(LSB)							
5															
6	Reserved														
...															
8															
9	PHY IDENTIFIER														
10	PHY OPERATION														
11	Reserved							UPDATE PARTIAL PATHWAY TIMEOUT VALUE							
12	Reserved														
...															
23															
24	ATTACHED DEVICE NAME														
...															
31															
32	PROGRAMMED MINIMUM PHYSICAL LINK RATE				Reserved										
33	PROGRAMMED MAXIMUM PHYSICAL LINK RATE				Reserved										
34	ENABLE SAS SLUMBER		ENABLE SAS PARTIAL		ENABLE SATA SLUMBER		ENABLE SATA PARTIAL								
35	PWR_DIS CONTROL		Reserved												
36	Reserved				PARTIAL PATHWAY TIMEOUT VALUE										
37	Reserved														
...															
39															
40	(MSB)	CRC						(LSB)							
...															
43															

The SMP FRAME TYPE field is defined in 9.4.3.2.2 and shall be set as shown in table 388 for the PHY CONTROL request.

The FUNCTION field is defined in 9.4.3.2.3 and shall be set as shown in table 388 for the PHY CONTROL request.

The ALLOCATED RESPONSE LENGTH field is defined in 9.4.3.2.4.

If the ALLOCATED RESPONSE LENGTH field is set to 00h, then the management device server shall:

- a) set the RESPONSE LENGTH field to 00h in the response frame; and
- b) return the first 4 bytes defined in table 396 plus the CRC field as the response frame.

If the ALLOCATED RESPONSE LENGTH field is not set to 00h, then the management device server shall:

- a) set the RESPONSE LENGTH field in the response frame to the value defined in table 396 (i.e., 00h); and
- b) return the response frame as defined in 9.4.3.2.4.

NOTE 89 - Future versions of this standard may change the value defined in table 396.

The REQUEST LENGTH field is defined in 9.4.3.2.5 and shall be set to one of the values defined in table 388 based on the LONG RESPONSE bit in the REPORT GENERAL response (see 9.4.3.4). A REQUEST LENGTH field set to 00h specifies that there are 9 dwords before the CRC field.

The EXPECTED EXPANDER CHANGE COUNT field is defined in the SMP CONFIGURE GENERAL request (see 9.4.3.18).

The PHY IDENTIFIER field specifies the phy (see 4.2.10) to which the SMP PHY CONTROL request applies.



Table 389 defines the PHY OPERATION field.

**Table 389 – PHY OPERATION field (part 1 of 2)**

Code	Operation	Description
00h	NOP	No operation.
01h	LINK RESET	<p>If:</p> <ul style="list-style-type: none"> <li>a) a SAS phy is attached;</li> <li>b) a SATA phy is attached and there is no affiliation; or</li> <li>c) a SATA phy is attached and an affiliation exists for the STP initiator port with the same SAS address as the SMP initiator port that opened this SMP connection,</li> </ul> <p>then:</p> <ul style="list-style-type: none"> <li>a) if the specified phy is a physical phy, then perform a link reset sequence (see 4.4) on the specified phy and enable the specified phy even if the specified phy is in a connection; and</li> <li>b) if the specified phy is a virtual phy, then perform an internal reset and enable the specified phy even if the specified phy is in a connection.</li> </ul> <p>If a SATA phy is attached and an affiliation does not exist for the STP initiator port with the same SAS address as the SMP initiator port that opened this SMP connection, then the management device server shall return a function result of AFFILIATION VIOLATION in the response frame (see table 294 in 9.4.3.3). <sup>a</sup></p> <p>See 6.15 for Broadcast (Change) requirements related to this phy operation in an expander device.</p> <p>Any affiliation (see 6.21.6) shall continue to be present. The phy shall bypass the SATA spinup hold state, if implemented (see 5.14.3.9).</p> <p>The management device server shall:</p> <ul style="list-style-type: none"> <li>1) send a Management Reset request to the SP state machine;</li> <li>2) return the PHY CONTROL response; and</li> <li>3) wait for the LINK RESET phy operation to complete.</li> </ul>
02h	HARD RESET	<p>If the specified phy is a physical phy, then perform a link reset sequence (see 4.4) on the specified phy and enable the specified phy even if the specified phy is in a connection. If the attached phy is a SAS phy or an expander phy, then the link reset sequence shall include a hard reset sequence (see 4.4.2). If the attached phy is a SATA phy, then the phy shall bypass the SATA spinup hold state. See 6.15 for Broadcast (Change) requirements related to this phy operation in an expander device.</p> <p>If the specified phy is a virtual phy, then perform an internal reset and enable the specified phy even if the specified phy is in a connection.</p> <p>Any affiliation (see 6.21.6) shall be cleared.</p> <p>The management device server shall return the PHY CONTROL response without waiting for the HARD RESET phy operation to complete.</p>
<p><sup>a</sup> Phys compliant with SAS-1.1 did not reject this phy operation due to affiliations.</p> <p><sup>b</sup> Phys compliant with SAS-1.1 returned SMP FUNCTION REJECTED.</p>		

**Table 389 – PHY OPERATION field (part 2 of 2)**

Code	Operation	Description
03h	DISABLE	Disable the specified phy (i.e., stop transmitting valid dwords and receiving dwords on the specified phy). The LINK RESET and HARD RESET operations may be used to enable the phy. See 6.15 for Broadcast (Change) requirements related to this phy operation in an expander device.
04h	Reserved	
05h	CLEAR ERROR LOG	Clear the error log counters reported in the REPORT PHY ERROR LOG function (see 9.4.3.11) for the specified phy.
06h	CLEAR AFFILIATION	Clear an affiliation (see 6.21.6) from the STP initiator port with the same SAS address as the SMP initiator port that opened this SMP connection. If there is no such affiliation, then the management device server shall return a function result of AFFILIATION VIOLATION <sup>b</sup> in the response frame (see table 294 in 9.4.3.3).
07h	TRANSMIT SATA PORT SELECTION SIGNAL	<p>This function shall only be supported by phys in an expander device.</p> <p>If the expander phy incorporates an STP SATA bridge and supports SATA port selectors, then the phy shall transmit the SATA port selection signal (see 5.7.4) which causes the SATA port selector to select the attached phy as the active host phy and make its other host phy inactive. See 6.15 for Broadcast (Change) requirements related to this phy operation in an expander device.</p> <p>Any affiliation (see 6.21.6) shall be cleared.</p> <p>If the expander phy does not support SATA port selectors, then the management device server shall return a function result of PHY DOES NOT SUPPORT SATA.</p> <p>If the expander phy supports SATA port selectors but is attached to a SAS phy or an expander phy, then the management device server shall return a function result of SMP FUNCTION FAILED in the response frame (see table 294 in 9.4.3.3).</p>
08h	CLEAR STP I_T NEXUS LOSS	The STP I_T NEXUS LOSS OCCURRED bit shall be set to zero in the REPORT PHY SATA function (see 9.4.3.12).
09h	SET ATTACHED DEVICE NAME	If the expander phy is attached to a SATA phy, then set the ATTACHED DEVICE NAME field reported in the DISCOVER response (see 9.4.3.10) to the value of the ATTACHED DEVICE NAME field in the PHY CONTROL request.
All others	Reserved	
<sup>a</sup> Phys compliant with SAS-1.1 did not reject this phy operation due to affiliations. <sup>b</sup> Phys compliant with SAS-1.1 returned SMP FUNCTION REJECTED.		

If the operation specified by the PHY OPERATION field is unknown, then the management device sever shall return a function result of UNKNOWN PHY OPERATION in the response frame (see table 294 in 9.4.3.3) and not process any other fields in the request.

If the PHY IDENTIFIER field specifies the phy that is being used for the SMP connection and a phy operation of LINK RESET, HARD RESET, or DISABLE is requested, then the management device server shall not perform the requested operation and shall return a function result of SMP FUNCTION FAILED in the response frame (see table 294 in 9.4.3.3).

An UPDATE PARTIAL PATHWAY TIMEOUT VALUE bit set to one specifies that the PARTIAL PATHWAY TIMEOUT VALUE field shall be honored. An UPDATE PARTIAL PATHWAY TIMEOUT VALUE bit set to zero specifies that the PARTIAL PATHWAY TIMEOUT VALUE field shall be ignored.

The ATTACHED DEVICE NAME field is used by the SET ATTACHED DEVICE NAME phy operation and is reserved for all other phy operations. If a management application client detects the ATTACHED DEVICE NAME field set to 00000000 00000000h in the DISCOVER response when a SATA device is attached, then it shall set the ATTACHED DEVICE NAME field based on the IDENTIFY DEVICE data (see ACS-4) retrieved by an ATA application client in the same SAS initiator device as follows:

- a) if IDENTIFY DEVICE data word 255 (i.e., the Integrity word) is correct and words 108 to 111 (i.e., the World Wide Name field) are not set to zero, then set this field to the world wide name indicated by words 108 to 111 according to table 21 in 4.2.7;
- b) if IDENTIFY DEVICE data word 255 (i.e., the Integrity word) is correct and words 108 to 111 (i.e., the World Wide Name) are set to zero, then set this field to 00000000 00000000h; or
- c) if IDENTIFY DEVICE data word 255 (i.e., the Integrity word) is not correct, then set this field to 00000000 00000000h.

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 390 defines the values for this field. This value is reported in the DISCOVER response (see 9.4.3.10). If this field is changed along with a phy operation of LINK RESET or HARD RESET, then 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 390 defines the values for this field. This value is reported in the DISCOVER response (see 9.4.3.10). If this field is changed along with a phy operation of LINK RESET or HARD RESET, then that phy operation shall utilize the new value for this field.

**Table 390 – PROGRAMMED MINIMUM PHYSICAL LINK RATE field and PROGRAMMED MAXIMUM PHYSICAL LINK RATE field**

Code	Description
0h	Do not change current value
1h to 7h	Reserved
8h	1.5 Gbit/s
9h	3 Gbit/s
Ah	6 Gbit/s
Bh	12 Gbit/s
Ch	22.5 Gbit/s
Dh to Fh	Reserved

If:

- a) the PROGRAMMED MINIMUM PHYSICAL LINK RATE field or the PROGRAMMED MAXIMUM PHYSICAL LINK RATE field is set to an unsupported or reserved value; or

- b) 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),

then the management device server shall not change either of their values and may return a function result of SMP FUNCTION FAILED in the response frame (see table 294 in 9.4.3.3). If the management device server returns a function result of SMP FUNCTION FAILED, then it shall not perform the requested phy operation.

Table 391 defines the ENABLE SAS SLUMBER field.

**Table 391 – ENABLE SAS SLUMBER field**

Code	Description
00b	No change
01b	If supported, then the management device server shall manage slumber phy power conditions (see 4.10.1.6).
10b	If supported, then the management device server shall disable slumber phy power conditions (see 4.10.1.6).
11b	Reserved

If the ENABLE SAS SLUMBER field is set to an unsupported or reserved value, then the management device server shall not issue a Manage Power Conditions request to any XL state machine and shall return a function result of SMP FUNCTION FAILED in the response frame (see table 294).

Table 392 defines the ENABLE SAS PARTIAL field.

**Table 392 – ENABLE SAS PARTIAL field**

Code	Description
00b	No change
01b	If supported, then the management device server shall manage partial phy power conditions (see 4.10.1.6).
10b	If supported, then the management device server shall disable partial phy power conditions (see 4.10.1.6).
11b	Reserved

If the ENABLE SAS PARTIAL field is set to an unsupported or reserved value, then the management device server shall not issue a Manage Power Conditions request to any XL state machine and shall return a function result of SMP FUNCTION FAILED in the response frame (see table 294).

Table 393 defines the ENABLE SATA SLUMBER field.

**Table 393 – ENABLE SATA SLUMBER field**

Code	Description
00b	No change
01b	If supported, then the management device server shall manage SATA slumber interface power management sequences (see 4.10.2).
10b	If supported, then the management device server shall disable SATA slumber interface power management sequences (see 4.10.2).
11b	Reserved

If the ENABLE SATA SLUMBER field is set to an unsupported or reserved value, then the management device server shall not issue a Manage Power Conditions request to any XL state machine and shall return a function result of SMP FUNCTION FAILED in the response frame (see table 294).

Table 394 defines the ENABLE SATA PARTIAL field.

**Table 394 – ENABLE SATA PARTIAL field**

Code	Description
00b	No change
01b	If supported, then the management device server shall manage SATA partial interface power management sequences (see 4.10.2).
10b	If supported, then the management device server shall disable SATA partial interface power management sequences (see 4.10.2).
11b	Reserved

If the ENABLE SATA PARTIAL field is set to an unsupported or reserved value, then the management device server shall not issue a Manage Power Conditions request to any XL state machine and shall return a function result of SMP FUNCTION FAILED in the response frame (see table 294).

Table 395 defines the PWR\_DIS CONTROL field.

**Table 395 – PWR\_DIS CONTROL field**

Code	Description
00b	No change
01b	Reserved
10b	If supported, then the management device server shall negate the POWER DISABLE signal (see 4.13.3 and SAS-4) associated with the phy.
11b	If supported, then the management device server shall assert the POWER DISABLE signal (see 4.13.3 and SAS-4) associated with the phy.

The PARTIAL PATHWAY TIMEOUT VALUE field specifies the amount of time, in one microsecond intervals, the expander phy shall wait after receiving an Arbitrating (Blocked On Partial) confirmation from the ECM before requesting that the ECM resolve pathway blockage (see 6.16.5.5). A PARTIAL PATHWAY TIMEOUT VALUE field value of zero (i.e., 0  $\mu$ s) specifies that partial pathway resolution shall be requested by the expander phy after receiving an Arbitrating (Blocked On Partial) confirmation from the ECM. This value is reported in the DISCOVER response (see 9.4.3.10). The PARTIAL PATHWAY TIMEOUT VALUE field is only honored if the UPDATE PARTIAL PATHWAY TIMEOUT VALUE bit is set to one.

The CRC field is defined in 9.4.3.2.7.

Table 396 defines the response format.

**Table 396 – PHY CONTROL response**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (41h)							
1	FUNCTION (91h)							
2	FUNCTION RESULT							
3	RESPONSE LENGTH (00h)							
4	(MSB)							
...	CRC							
7	(LSB)							

The SMP FRAME TYPE field is defined in 9.4.3.3.2 and shall be set as shown in table 396 for the PHY CONTROL response.

The FUNCTION field is defined in 9.4.3.3.3 and shall be set as shown in table 396 for the PHY CONTROL response.

The FUNCTION RESULT field is defined in 9.4.3.3.4.

The RESPONSE LENGTH field is defined in 9.4.3.3.5 and shall be set as shown in table 396 for the PHY CONTROL response.

The CRC field is defined in 9.4.3.3.7.

#### **9.4.3.29 PHY TEST FUNCTION function**

The PHY TEST FUNCTION function requests actions by the specified phy. This SMP function may be implemented by any management device server. In zoning expander devices if zoning is enabled, then this function shall only be processed from SMP initiator ports that have access to zone group 2 or the zone group of the specified phy (see 4.8.3.2).

Table 397 defines the request format.

**Table 397 – PHY TEST FUNCTION request**

Byte\Bit	7	6	5	4	3	2	1	0								
0	SMP FRAME TYPE (40h)															
1	FUNCTION (92h)															
2	ALLOCATED RESPONSE LENGTH															
3	REQUEST LENGTH (00h or 09h)															
4	(MSB)	EXPECTED EXPANDER CHANGE COUNT						(LSB)								
5																
6	Reserved															
...																
8																
9	PHY IDENTIFIER															
10	PHY TEST FUNCTION															
11	PHY TEST PATTERN															
12	Reserved															
...																
14																
15	Reserved	PHY TEST FUNCTION SATA	PHY TEST FUNCTION SSC	PHY TEST FUNCTION PHYSICAL LINK RATE												
16	Reserved															
...																
18																
19	PHY TEST PATTERN DWORDS CONTROL															
20	PHY TEST PATTERN DWORDS															
...																
27																
28	Reserved															
...																
39																
40	(MSB)	CRC						(LSB)								
...																
43																



The SMP FRAME TYPE field is defined in 9.4.3.2.2 and shall be set as shown in table 397 for the PHY TEST FUNCTION request.

The FUNCTION field is defined in 9.4.3.2.3 and shall be set as shown in table 397 for the PHY TEST FUNCTION request.

The ALLOCATED RESPONSE LENGTH field is defined in 9.4.3.2.4.

If the ALLOCATED RESPONSE LENGTH field is set to 00h, then the management device server shall:

- a) set the RESPONSE LENGTH field to 00h in the response frame; and
- b) return the first 4 bytes defined in table 400 plus the CRC field as the response frame.

If the ALLOCATED RESPONSE LENGTH field is not set to 00h, then the management device server shall:

- a) set the RESPONSE LENGTH field in the response frame to the value defined in table 400 (i.e., 00h); and
- b) return the response frame as defined in 9.4.3.2.4.

NOTE 90 - Future versions of this standard may change the value defined in table 400.

The REQUEST LENGTH field is defined in 9.4.3.2.5 and shall be set to one of the values defined in table 397 based on the LONG RESPONSE bit in the REPORT GENERAL response (see 9.4.3.4). A REQUEST LENGTH field set to 00h specifies that there are 9 dwords before the CRC field.

The EXPECTED EXPANDER CHANGE COUNT field is defined in the SMP CONFIGURE GENERAL request (see 9.4.3.18).

The PHY IDENTIFIER field specifies the phy (see 4.2.10) to which the SMP PHY TEST PATTERN request applies.

If the PHY IDENTIFIER field specifies the phy that is being used for the SMP connection, then the management device server shall not perform the requested operation and shall return a function result of SMP FUNCTION FAILED in the response frame (see table 294 in 9.4.3.3).

The PHY TEST FUNCTION field specifies the phy test function (see 4.11) to be performed and is defined in table 398. If the PHY TEST FUNCTION field specifies a phy test function that is not supported by the phy, then the management device server shall return a function result of UNKNOWN PHY TEST FUNCTION in the response frame (see table 294 in 9.4.3.3).

**Table 398 – PHY TEST FUNCTION field**

Code	Name	Description
00h	STOP	<p>If the selected phy is performing a phy test function, then the selected phy shall stop performing the phy test function and originate a link reset sequence.</p> <p>If the selected phy is not performing a phy test function, then this function has no effect on the selected phy.</p>
01h	TRANSMIT PATTERN	<p>If the selected phy is not performing a phy test function, then the selected phy shall be set to transmit the phy test pattern specified by the PHY TEST PATTERN field at the physical link rate specified by the PHY TEST FUNCTION PHYSICAL LINK RATE field and set to ignore its receiver. If the selected phy receives data while transmitting the pattern (see 4.11.2), then the selected phy shall ignore the received data.</p> <p>If the selected phy is performing a phy test function, then the management device server shall return a function result of PHY TEST FUNCTION IN PROGRESS in the response frame (see table 294).</p>
02h to EFh	Reserved	
F0h to FFh	Vendor specific	

If the PHY TEST FUNCTION field is set to 01h (i.e., TRANSMIT PATTERN), then the PHY TEST PATTERN field specifies the phy test pattern to be performed and is the same as that defined in table 276 for the Protocol Specific diagnostic page (see 9.2.9.2). The phy test pattern shall be sent at the physical link rate specified by the PHY TEST FUNCTION PHYSICAL LINK RATE field.

The PHY TEST FUNCTION SATA bit is as defined in the Protocol Specific diagnostic page (see 9.2.9.2).

The PHY TEST FUNCTION SSC field is as defined in table 277 for the Protocol Specific diagnostic page (see 9.2.9.2).

The PHY TEST FUNCTION PHYSICAL LINK RATE field specifies the physical link rate at which the phy test function, if any, shall be performed. Table 399 defines the values for this field.

**Table 399 – PHY TEST FUNCTION PHYSICAL LINK RATE field**

Code	Description
0h to 7h	Reserved
8h	1.5 Gbit/s
9h	3 Gbit/s
Ah	6 Gbit/s
Bh	12 Gbit/s
Ch	22.5 Gbit/s
Dh to Fh	Reserved

The PHY TEST PATTERN DWORDS CONTROL field and the PHY TEST PATTERN DWORDS field are as defined in table 276 for the Protocol Specific diagnostic page (see 9.2.9.2).

The CRC field is defined in 9.4.3.2.7.

Table 400 defines the response format.

**Table 400 – PHY TEST FUNCTION response**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (41h)							
1	FUNCTION (92h)							
2	FUNCTION RESULT							
3	RESPONSE LENGTH (00h)							
4	CRC							
...								
7								

The SMP FRAME TYPE field is defined in 9.4.3.3.2 and shall be set as shown in table 400 for the PHY TEST FUNCTION response.

The FUNCTION field is defined in 9.4.3.3.3 and shall be set as shown in table 400 for the PHY TEST FUNCTION response.

The FUNCTION RESULT field is defined in 9.4.3.3.4.

The RESPONSE LENGTH field is defined in 9.4.3.3.5 and shall be set as shown in table 400 for the PHY TEST FUNCTION response.

The CRC field is defined in 9.4.3.3.7.

#### **9.4.3.30 CONFIGURE PHY EVENT function**

##### **9.4.3.30.1 CONFIGURE PHY EVENT function overview**

The CONFIGURE PHY EVENT function configures phy events (see 4.12) for the specified phy. This SMP function may be implemented by any management device server. In zoning expander devices, if zoning is enabled, then this function shall only be processed from SMP initiator ports that have access to zone group 2 or the zone group of the specified phy (see 4.8.3.2).

## 9.4.3.30.2 CONFIGURE PHY EVENT request

Table 401 defines the request format.

Table 401 – CONFIGURE PHY EVENT request

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (40h)							
1	FUNCTION (93h)							
2	ALLOCATED RESPONSE LENGTH							
3	REQUEST LENGTH ((n - 7) / 4)							
4	(MSB)	EXPECTED EXPANDER CHANGE COUNT						
5								(LSB)
6	Reserved							CLEAR PEAKS
7	Reserved							
8								
9	PHY IDENTIFIER							
10	PHY EVENT CONFIGURATION DESCRIPTOR LENGTH							
11	NUMBER OF PHY EVENT CONFIGURATION DESCRIPTORS							
Phy event configuration descriptor list								
12	Phy event configuration descriptor (first) (see table 402 in 9.4.3.30.3)							
...								
19								
...	...							
n - 11	Phy event configuration descriptor (last) (see table 402 in 9.4.3.30.3)							
...								
n - 4								
n - 3	(MSB)	CRC						(LSB)
...								
n								

The SMP FRAME TYPE field is defined in 9.4.3.2.2 and shall be set as shown in table 401 for the CONFIGURE PHY EVENT request.

The FUNCTION field is defined in 9.4.3.2.3 and shall be set as shown in table 401 for the CONFIGURE PHY EVENT request.

The ALLOCATED RESPONSE LENGTH field is defined in 9.4.3.2.4.

The REQUEST LENGTH field is defined in 9.4.3.2.5 and shall be set as shown in table 401 for the CONFIGURE PHY EVENT request.

The EXPECTED EXPANDER CHANGE COUNT field is defined in the SMP CONFIGURE GENERAL request (see 9.4.3.18).

A CLEAR PEAKS bit set to one specifies that all phy event peak value detectors shall be set to zero. A CLEAR PEAKS bit set to zero specifies no change to the phy event peak value detectors.

The PHY IDENTIFIER field specifies the phy (see 4.2.9) to which the configure phy event information shall be applied.

The PHY EVENT CONFIGURATION DESCRIPTOR LENGTH field indicates the length, in dwords, of the phy event configuration descriptor (see 9.4.3.30.3).

The NUMBER OF PHY EVENT CONFIGURATION DESCRIPTORS field specifies the number of phy event configuration descriptors in the phy event configuration descriptor list and shall be set to the same value as the NUMBER OF PHY EVENT DESCRIPTORS field in the SMP REPORT PHY EVENT function (see 9.4.3.14).

The phy event configuration descriptor list contains phy event configuration descriptors as defined in 9.4.3.30.3.

The CRC field is defined in 9.4.3.2.7.

#### 9.4.3.30.3 Phy event configuration descriptor

Table 402 defines the phy event configuration descriptor.

**Table 402 – Phy event configuration descriptor**

Byte\Bit	7	6	5	4	3	2	1	0
0	Reserved							
2								
3	PHY EVENT SOURCE							
4	PEAK VALUE DETECTOR THRESHOLD							
...								
7								

The PHY EVENT SOURCE field, defined in table 46 in 4.12, specifies the type of event that shall be recorded by the corresponding phy event monitor.

If the phy event source is a peak value detector, then the PEAK VALUE DETECTOR THRESHOLD field specifies the value of the peak value detector that causes the expander device to originate a Broadcast (Expander) (see 6.2.6.4). If the phy event source is not a peak value detector, then the PEAK VALUE DETECTOR THRESHOLD field is reserved.

If the PHY EVENT SOURCE field contains a value that is not supported, then the management device server shall return a function result of UNKNOWN PHY EVENT SOURCE in the response frame (see table 294 in 9.4.3.3).

#### 9.4.3.30.4 CONFIGURE PHY EVENT response

Table 403 defines the response format.

**Table 403 – CONFIGURE PHY EVENT response**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (41h)							
1	FUNCTION (93h)							
2	FUNCTION RESULT							
3	RESPONSE LENGTH (00h)							
4	CRC							
...								
7								

The SMP FRAME TYPE field is defined in 9.4.3.3.2 and shall be set as shown in table 403 for the CONFIGURE PHY EVENT response.

The FUNCTION field is defined in 9.4.3.3.3 and shall be set as shown in table 403 for the CONFIGURE PHY EVENT response.

The FUNCTION RESULT field is defined in 9.4.3.3.4.

The RESPONSE LENGTH field is defined in 9.4.3.3.5 and shall be set as shown in table 403 for the CONFIGURE PHY EVENT response. A RESPONSE LENGTH field set to 00h does not have a special meaning based on the ALLOCATED RESPONSE LENGTH field in the request frame.

The CRC field is defined in 9.4.3.3.7.

## Annex A

(normative)

### Jitter tolerance patterns when SAS dword mode is enabled

#### A.1 Jitter tolerance pattern (JTPAT)

Table A.1 shows a pattern containing both JTPAT for RD+ and JTPAT for RD-. The 10b pattern resulting from encoding the 8b pattern contains the desired bit sequences for the phase shifts with both starting running disparities. For more details on JTPAT see SAS-4.

**Table A.1 – JTPAT for RD+ and RD-**

Dwords	First character	Second character	Third character	Fourth character	Notes
0 to 40	D30.3(7Eh)	D30.3(7Eh)	D30.3(7Eh)	D30.3(7Eh)	This dword is sent a total of 41 times.
	...	...	...	...	
41	D30.3(7Eh)	D30.3(7Eh)	D30.3(7Eh)	D20.3(74h)	This dword is sent once.
42	D30.3(7Eh)	D11.5(ABh)	D21.5(B5h)	D21.5(B5h)	This dword is sent once.
43 to 54	D21.5(B5h)	D21.5(B5h)	D21.5(B5h)	D21.5(B5h)	This dword is sent a total of 12 times.
	...	...	...	...	
55	D21.5(B5h)	D30.2(5Eh)	D10.2(4Ah)	D30.3(7Eh)	This dword is sent once.
56 to 96	D30.3(7Eh)	D30.3(7Eh)	D30.3(7Eh)	D30.3(7Eh)	This dword is sent a total of 41 times.
	...	...	...	...	
97	D30.3(7Eh)	D30.3(7Eh)	D30.3(7Eh)	D11.3(6Bh)	This dword is sent once.
98	D30.3(7Eh)	D20.2(54h)	D10.2(4Ah)	D10.2(4Ah)	This dword is sent once.
99 to 110	D10.2(4Ah)	D10.2(4Ah)	D10.2(4Ah)	D10.2(4Ah)	This dword is sent a total of 12 times.
	...	...	...	...	
111	D10.2(4Ah)	D30.5(BEh)	D21.5(B5h)	D30.3(7Eh)	This dword is sent once.

#### A.2 Compliant jitter tolerance pattern (CJTPAT)

The compliant jitter tolerance pattern (CJTPAT) is the JTPAT for RD+ and RD- (see table A.1) included as the payload in an SSP DATA frame or an SMP frame. The CJTPAT is:

- 1) SOF;
- 2) six data dwords containing either:
  - A) an SSP DATA frame header; or
  - B) an SMP frame header followed by 23 vendor specific bytes;
- 3) 112 data dwords containing JTPAT for RD+ and RD-;



- 4) one data dword containing a CRC value; and
- 5) EOF.

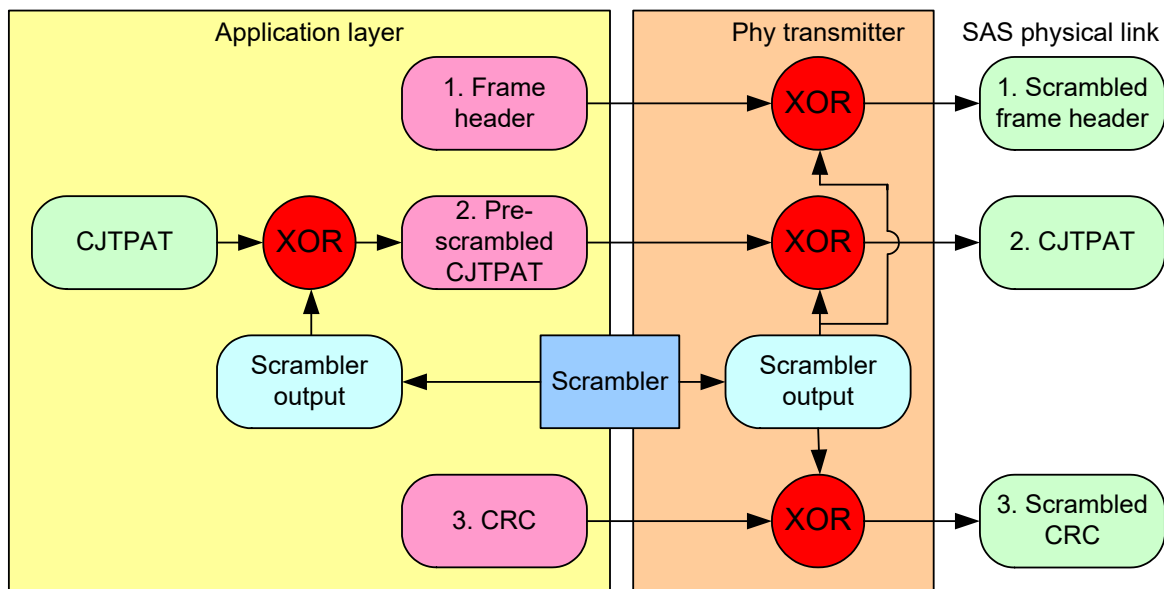
Deletable primitives may be included in the transmission of the CJTPAT, but the number of deletable primitives transmitted should be as small as possible so that the percentage of the transfer that is the JTPAT is as high as possible.

As a result of the SOF, EOF, and CRC being the same in SSP and SMP, CJTPAT complies with:

- a) the SSP frame transmission format defined by the SSP link layer (see figure 164);
- b) the SSP frame format defined by the SSP transport layer (see table 206);
- c) the SMP frame transmission format defined by the SMP link layer (see figure 189); and
- d) the SMP frame format defined by the SMP transport layer (see table 235).

When a phy transmits a frame, it XORs the 8b data provided by the application layer with the output of a scrambler before transmission (see 6.8). The phy reinitializes the scrambler at the beginning of each frame (e.g., at SOF) and does not modify primitives. If the application layer XORs the desired 8b pattern with the expected output of the scrambler prior to submitting it to the transmitter, then the transmitter transmits the desired pattern. The 8b data dwords are scrambled by XORing the pattern with the expected scrambler dword output, taking into account the position of the 8b data dwords within the frame.

Figure A.1 shows how to pre-scramble CJTPAT into the phy's transmitter so CJTPAT results on the physical link.



**Figure A.1 – CJTPAT pre-scrambling**

Table A.2 defines CJTPAT.

The “SSP frame contents” column in table A.2 shows the interpretation of the frame if viewed as an SSP DATA frame.

The “SMP frame contents” column in table A.2 shows the interpretation of the frame if viewed as an SMP frame.

The “Pre-scrambled CJTPAT” column in table A.2 shows the result of XORing CJTPAT with the expected scrambler output before presenting the frame to the phy's transmitter. If the data in this column is supplied to the phy's transmitter where it is scrambled again, then the data in the “CJTPAT” column is transmitted onto the physical link. The frame header is not pre-scrambled, and the CRC is calculated over the frame header and the pre-scrambled CJTPAT.

The “Scrambler output” column in table A.2 shows the scrambler output for each data dword in the frame. The scrambler output is independent of the data pattern.

The “CJTPAT” column in table A.2 shows CJTPAT, transmitted on the physical link.

**Table A.2 – CJTPAT** (part 1 of 6)

Data dword number	SSP frame contents	SMP frame contents	Pre-scrambled CJTPAT	Scrambler output	CJTPAT
Not applicable	SOF		SOF	Not applicable	SOF
0	SSP frame header	SMP frame header and 3 frame-type dependent bytes	unknown	C2D2768Dh	unknown
1		Frame-type dependent bytes	unknown	1F26B368h	unknown
2			unknown	A508436Ch	unknown
3			unknown	3452D354h	unknown
4			unknown	8A559502h	unknown
5			unknown	BB1ABE1Bh	unknown
6	INFORMATION UNIT field (dwords 0 to 7)	Frame-type dependent bytes	8428C943h	FA56B73Dh	7E7E7E7Eh
7			2D887565h	53F60B1Bh	7E7E7E7Eh
8			8EFEE23Fh	F0809C41h	7E7E7E7Eh
9			0A01BD34h	747FC34Ah	7E7E7E7Eh
10			C0F82CEFh	BE865291h	7E7E7E7Eh
11			0411D9C8h	7A6FA7B6h	7E7E7E7Eh
12			4F1D98A8h	3163E6D6h	7E7E7E7Eh
13			8E488072h	F036FE0Ch	7E7E7E7Eh
a The CRC field shall be set to a valid value for the frame.					

Table A.2 – CJTPAT (part 2 of 6)

Data dword number	SSP frame contents	SMP frame contents	Pre-scrambled CJTPAT	Scrambler output	CJTPAT
14	INFORMATION UNIT field (dwords 8 to 15)	Frame-type dependent bytes	608D9457h	1EF3EA29h	7E7E7E7Eh
15			954A58EAh	EB342694h	7E7E7E7Eh
16			2DFB4569h	53853B17h	7E7E7E7Eh
17			9734A233h	E94ADC4Dh	7E7E7E7Eh
18			235E70F6h	5D200E88h	7E7E7E7Eh
19			177F93AEh	6901EDD0h	7E7E7E7Eh
20			84E046A0h	FA9E38DEh	7E7E7E7Eh
21			16A53579h	68DB4B07h	7E7E7E7Eh
22	INFORMATION UNIT field (dwords 16 to 23)	Frame-type dependent bytes	3B743D05h	450A437Bh	7E7E7E7Eh
23			E873A976h	960DD708h	7E7E7E7Eh
24			414B98E6h	3F35E698h	7E7E7E7Eh
25			8008E6DBh	FE7698A5h	7E7E7E7Eh
26			B670896Bh	C80EF715h	7E7E7E7Eh
27			181EEED1h	666090AFh	7E7E7E7Eh
28			848EABB5h	FAF0D5CBh	7E7E7E7Eh
29			55FC7EE1h	2B82009Fh	7E7E7E7Eh
30	INFORMATION UNIT field (dwords 24 to 31)	Frame-type dependent bytes	704F0AEFh	0E317491h	7E7E7E7Eh
31			088A1460h	76F46A1Eh	7E7E7E7Eh
32			8A131736h	F46D6948h	7E7E7E7Eh
33			05B3F4Edh	7BCD8A93h	7E7E7E7Eh
34			6B6DD300h	1513AD7Eh	7E7E7E7Eh
35			600C8090h	1E72FEEEH	7E7E7E7Eh
36			DE6AD445h	A014AA3Bh	7E7E7E7Eh
37			5DD4AA99h	23AAD4E7h	7E7E7E7Eh
a The CRC field shall be set to a valid value for the frame.					

Table A.2 – CJTPAT (part 3 of 6)

Data dword number	SSP frame contents	SMP frame contents	Pre-scrambled CJTPAT	Scrambler output	CJTPAT
38	INFORMATION UNIT field field (dwords 32 to 39)	Frame-type dependent bytes	CEA2E019h	B0DC9E67h	7E7E7E7Eh
39			9EDB0D85h	E0A573FBh	7E7E7E7Eh
40			78B4EA31h	06CA944Fh	7E7E7E7Eh
41			1D9CEC6Ch	63E29212h	7E7E7E7Eh
42			3B061C13h	4578626Dh	7E7E7E7Eh
43			2D5872EDh	53260C93h	7E7E7E7Eh
44			40275C7Ch	3E592202h	7E7E7E7Eh
45			5510B41Dh	2B6ECA63h	7E7E7E7Eh
46	INFORMATION UNIT field field (dwords 40 to 47)	Frame-type dependent bytes	1D146161h	636A1F1Fh	7E7E7E7Eh
47			4BCBD799h	35B5A9EDh	7E7E7E74h
48			34091548h	4AA2A0FDh	7EABB5B5h
49			C41A5423h	71AFE196h	B5B5B5B5h
50			5460CED7h	E1D57B62h	B5B5B5B5h
51			E015E33Fh	55A0568Ah	B5B5B5B5h
52			37643CDDh	82D18968h	B5B5B5B5h
53			96F9014Ah	234CB4FFh	B5B5B5B5h
54	INFORMATION UNIT field field (dwords 48 to 55)	Frame-type dependent bytes	36FDABCAh	83481E7Fh	B5B5B5B5h
55			07AF5DCAh	B21AE87Fh	B5B5B5B5h
56			1C705F78h	A9C5EACDh	B5B5B5B5h
57			D7B41976h	6201ACC3h	B5B5B5B5h
58			43BC8C7Bh	F60939CEh	B5B5B5B5h
59			8CEAC3C8h	395F767Dh	B5B5B5B5h
60			9A10EDF4h	2FA55841h	B5B5B5B5h
61			36330004h	836D4A7Ah	B55E4A7Eh
a The CRC field shall be set to a valid value for the frame.					

Table A.2 – CJTPAT (part 4 of 6)

Data dword number	SSP frame contents	SMP frame contents	Pre-scrambled CJTPAT	Scrambler output	CJTPAT
62	INFORMATION UNIT field field (dwords 56 to 63)	Frame-type dependent bytes	46F32604h	388D587Ah	7E7E7E7Eh
63			09438122h	773DFF5Ch	7E7E7E7Eh
64			425DE2CDh	3C239CB3h	7E7E7E7Eh
65			2833EFDEh	564D91A0h	7E7E7E7Eh
66			3D93759Fh	43ED0BE1h	7E7E7E7Eh
67			E60A57D9h	987429A7h	7E7E7E7Eh
68			9B53A5DCh	E52DDBA2h	7E7E7E7Eh
69			99F3B601h	E78DC87Fh	7E7E7E7Eh
70	INFORMATION UNIT field field (dwords 64 to 71)	Frame-type dependent bytes	74C6B817h	0AB8C669h	7E7E7E7Eh
71			1AAEFDB7h	64D083C9h	7E7E7E7Eh
72			7B438744h	053DF93Ah	7E7E7E7Eh
73			9097A794h	EEE9D9Eah	7E7E7E7Eh
74			3AC345E9h	44BD3B97h	7E7E7E7Eh
75			719C35F2h	0FE24B8Ch	7E7E7E7Eh
76			8CF328EAh	F28D5694h	7E7E7E7Eh
77			1D6EC8A7h	6310B6D9h	7E7E7E7Eh
78	INFORMATION UNIT field field (dwords 72 to 79)	Frame-type dependent bytes	69ECD0B0h	1792AECEh	7E7E7E7Eh
79			742850DFh	0A562EA1h	7E7E7E7Eh
80			CE36A117h	B048DF69h	7E7E7E7Eh
81			68645606h	161A2878h	7E7E7E7Eh
82			2B67B52Fh	5519CB51h	7E7E7E7Eh
83			678BC028h	19F5BE56h	7E7E7E7Eh
84			9181CAC8h	EFFFB4B6h	7E7E7E7Eh
85			CDFC100Ch	B3826E72h	7E7E7E7Eh
a The CRC field shall be set to a valid value for the frame.					

Table A.2 – CJTPAT (part 5 of 6)

Data dword number	SSP frame contents	SMP frame contents	Pre-scrambled CJTPAT	Scrambler output	CJTPAT
86	INFORMATION UNIT field (dwords 80 to 87)	Frame-type dependent bytes	9A0C53A4h	E4722DDAh	7E7E7E7Eh
87			1EC12F57h	60BF5129h	7E7E7E7Eh
88			5AF3EE8Bh	248D90F5h	7E7E7E7Eh
89			3378AC62h	4D06D21Ch	7E7E7E7Eh
90			00E86812h	7E96166Ch	7E7E7E7Eh
91			21D19DCAh	5FAFE3B4h	7E7E7E7Eh
92			2E12C62Bh	506CB855h	7E7E7E7Eh
93			258E4EE6h	5BF03098h	7E7E7E7Eh
94	INFORMATION UNIT field (dwords 88 to 95)	Frame-type dependent bytes	38AAC8CDh	46D4B6B3h	7E7E7E7Eh
95			7B65E06Fh	051B9E11h	7E7E7E7Eh
96			7F22BB28h	015CC556h	7E7E7E7Eh
97			9C6E4B91h	E21035EFh	7E7E7E7Eh
98			281E330Bh	56604D75h	7E7E7E7Eh
99			50081922h	2E76675Ch	7E7E7E7Eh
100			796A088Eh	071476F0h	7E7E7E7Eh
101			D18EF995h	AFF087EBh	7E7E7E7Eh
102	INFORMATION UNIT field (dwords 96 to 103)	Frame-type dependent bytes	651CA57Fh	1B62DB01h	7E7E7E7Eh
103			5D186107h	23661F6Ch	7E7E7E6Bh
104			8623FA6Dh	F877B027h	7E544A4Ah
105			BFA9C3E8h	F5E389A2h	4A4A4A4Ah
106			A48D7C5Bh	EEC73611h	4A4A4A4Ah
107			064EB1D9h	4C04FB93h	4A4A4A4Ah
108			A29D4578h	E8D70F32h	4A4A4A4Ah
109			F5BA761Eh	BFF03C54h	4A4A4A4Ah
<sup>a</sup> The CRC field shall be set to a valid value for the frame.					

Table A.2 – CJTPAT (part 6 of 6)

Data dword number	SSP frame contents	SMP frame contents	Pre-scrambled CJTPAT	Scrambler output	CJTPAT
110	INFORMATION UNIT field (dwords 104 to 111)	Frame-type dependent bytes	A90A764Bh	E3403C01h	4A4A4A4Ah
111			6AB08034h	20FACA7Eh	4A4A4A4Ah
112			D3080FC6h	9942458Ch	4A4A4A4Ah
113			7DA881C3h	37E2CB89h	4A4A4A4Ah
114			1050DDC9h	5A1A9783h	4A4A4A4Ah
115			8402E075h	CE48AA3Fh	4A4A4A4Ah
116			4C83ED2Bh	06C9A761h	4A4A4A4Ah
117			4C7E8BD5h	06C03EABh	4ABEB57Eh
118	CRC field <sup>a</sup>		depends on contents of first 6 data dwords	3D2D7984h	depends on contents of first 6 data dwords
Not applicable	EOF		<primitive>	<primitive>	<primitive>
<sup>a</sup> The CRC field shall be set to a valid value for the frame.					

A phy or test equipment transmitting CJTPAT outside connections may transmit it with fixed content as defined in table A.3.

Table A.3 shows CJTPAT with fixed content:

- a) interpreted as an SSP frame, with the FRAME TYPE field in the frame header set to 01h (i.e., DATA), each other field in the frame header set to zero, the INFORMATION UNIT field containing JTPAT for RD+ and RD-, and the CRC field set to a fixed value; and
- b) interpreted as an SMP frame, with the SMP FRAME TYPE field in the frame header set to 01h (i.e., reserved), the frame-type dependent bytes containing JTPAT for RD+ and RD-, and the CRC field set to a fixed value.

Table A.3 – CJTPAT with fixed content

Data dword number	SSP frame contents	SMP frame contents	Pre-scrambled CJTPAT	Scrambler output	CJTPAT
Not applicable	SOF		SOF	Not applicable	SOF
0	FRAME TYPE field set to 01h (i.e., DATA frame) and 23 subsequent bytes each set to 00h	SMP FRAME TYPE field set to 01h (i.e., reserved) and 23 subsequent frame-type dependent bytes each set to 00h	01000000h	C2D2768Dh	C3D2768Dh
1			00000000h	1F26B368h	1F26B368h
2			00000000h	A508436Ch	A508436Ch
3			00000000h	3452D354h	3452D354h
4			00000000h	8A559502h	8A559502h
5			00000000h	BB1ABE1Bh	BB1ABE1Bh
6	INFORMATION UNIT field	Frame-type dependent bytes	See the INFORMATION UNIT field in table A.2		
...					
117					
118	CRC field		44EF682Eh	3D2D7984h	79C211AAh
Not applicable	EOF		EOF	Not applicable	EOF

### A.3 Considerations for a phy transmitting JTPAT and CJTPAT

A phy may be configured to transmit JTPAT for RD+ and RD- (see clause A.1) by:

- using the SMP PHY TEST FUNCTION function (see 9.4.3.29) or the Protocol Specific diagnostic page (see 9.2.9.2) specifying the phy, with the PHY TEST FUNCTION field set to 01h (i.e., TRANSMIT PATTERN), and the PHY TEST PATTERN field set to 01h (i.e., JTPAT); or
- vendor specific mechanisms.

A phy may be configured to transmit CJTPAT (see clause A.2) by:

- using the SMP PHY TEST FUNCTION function (see 9.4.3.29) or the Protocol Specific diagnostic page (see 9.2.9.2) specifying the phy, with the PHY TEST FUNCTION field set to 01h (i.e., TRANSMIT PATTERN), and the PHY TEST PATTERN field set to 02h (i.e., CJTPAT);
- including CJTPAT as a data pattern while processing SCSI commands (e.g., the WRITE BUFFER command if the phy is in an SSP initiator port or the READ BUFFER command if the phy is in a target port). The frame length shall be selected to ensure that the specified pattern is transmitted on the physical link; or
- vendor specific mechanisms.



## A.4 Considerations for a phy receiving JTPAT and CJTPAT

If a phy receives JTPAT (see SAS-4) inside or outside a connection, then it considers the data dwords to be idle dwords and ignores them.

If a phy receives CJTPAT (see clause A.2) outside a connection, then the SL receiver (see 6.18.2) considers the SOF and EOF to be unexpected dwords and ignores them, and considers the data dwords to be idle dwords and ignores them.

Phy-layer based phy event counters (e.g., invalid dword count, running disparity error count, loss of dword synchronization count, elasticity buffer overflow count, and received ERROR count) count events that occur while receiving idle dwords, so may be used to count events while receiving JTPAT or CTPAT.

If a phy receives CJTPAT inside an SSP connection, then the phy expects it to have a valid frame header (i.e., all fields in the frame header are valid including the FRAME TYPE field and the SOURCE SAS ADDRESS field) and follow SSP frame transmission rules (e.g., SSP frame credit, ACK or NAK exchange).

If a phy receives CJTPAT inside an SMP connection, then the phy expects it to have a valid frame header (e.g., valid frame type) and follow SMP frame transmission rules (e.g., only one frame is transmitted in each direction per connection). Sending CJTPAT inside SMP connections is not recommended.

This standard defines no mechanism for configuring a phy to expect to receive JTPAT or CJTPAT (e.g., to compare the incoming pattern to the expected pattern).

## Annex B

(informative)

### SAS to SAS phy reset sequence examples

Figure B.1 shows a speed negotiation between a phy A that supports only SNW-1 attached to a phy B that only supports SNW-1. Both phys run:

- 1) SNW-1, supported by both phys; and
- 2) SNW-2, supported by neither phy.

Both phys then select 1.5 Gbit/s for Final-SNW, which is used to establish the negotiated physical link rate.

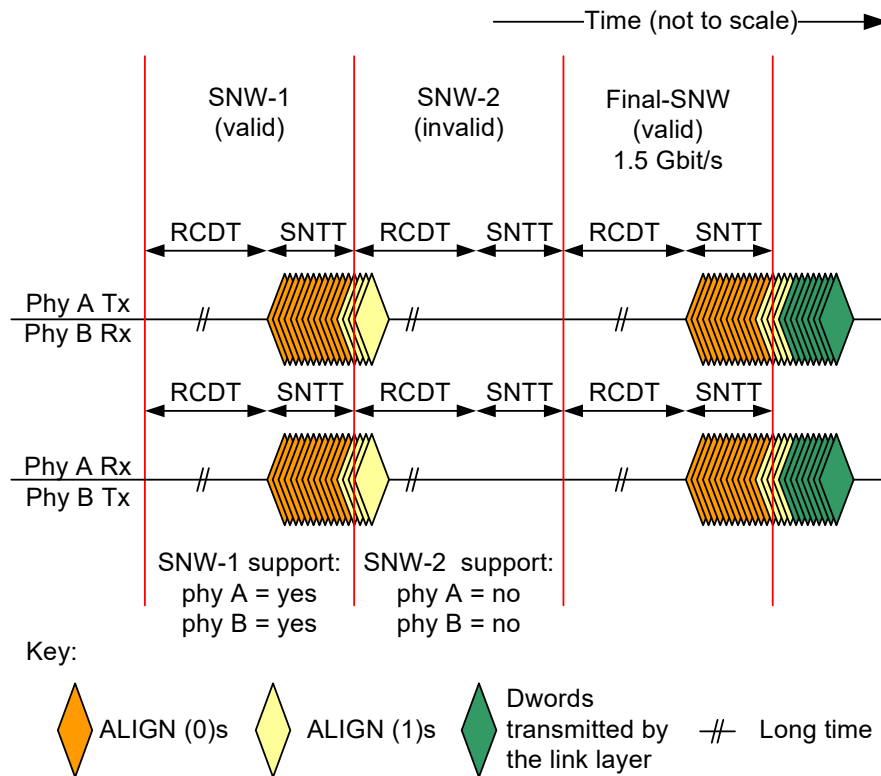


Figure B.1 – SAS speed negotiation sequence (phy A: SNW-1 only, phy B: SNW-1 only)

Figure B.2 shows a speed negotiation between a phy A that supports SNW-1 and SNW-2 attached to a phy B that supports SNW-1 and SNW-2. Both phys run:

- 1) SNW-1, supported by both phys;
- 2) SNW-2, supported by both phys; and
- 3) SNW-3, supported by neither phy.

Both phys then select 3 Gbit/s for Final-SNW, which is used to establish the negotiated physical link rate.

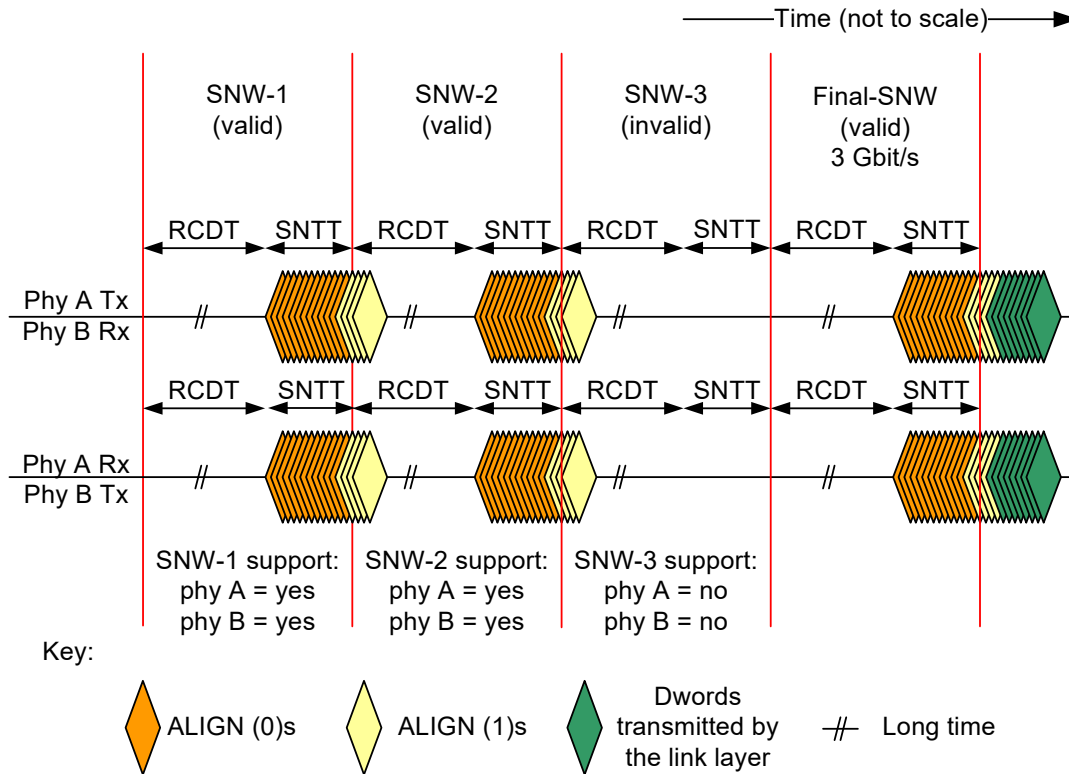
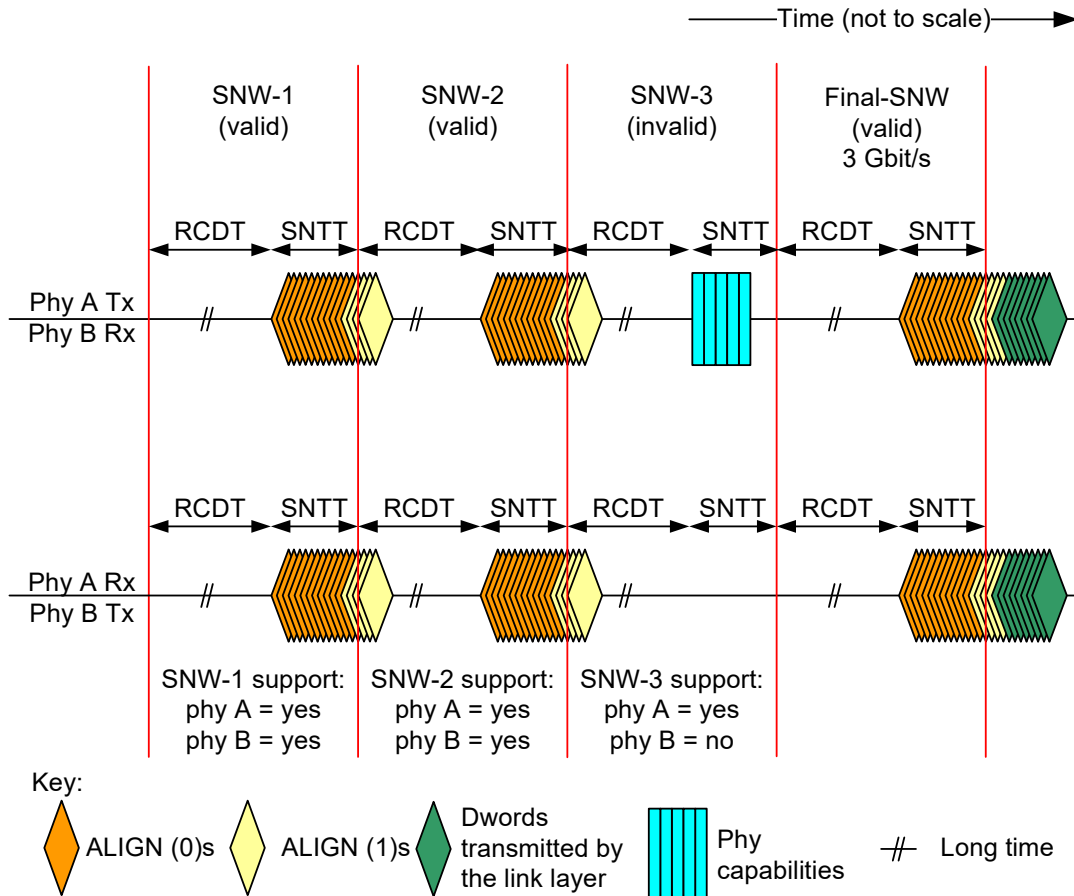


Figure B.2 – SAS speed negotiation sequence (phy A: SNW-1, SNW-2, phy B: SNW-1, SNW-2)

Figure B.3 shows a speed negotiation between a phy A that supports SNW-1 to SNW-3 attached to a phy B that only supports SNW-1 and SNW-2. Both phys run:

- 1) SNW-1, supported by both phys;
- 2) SNW-2, supported by both phys; and
- 3) SNW-3, supported by phy A but not by phy B.

Both phys then select 3 Gbit/s for Final-SNW, which is used to establish the negotiated physical link rate.

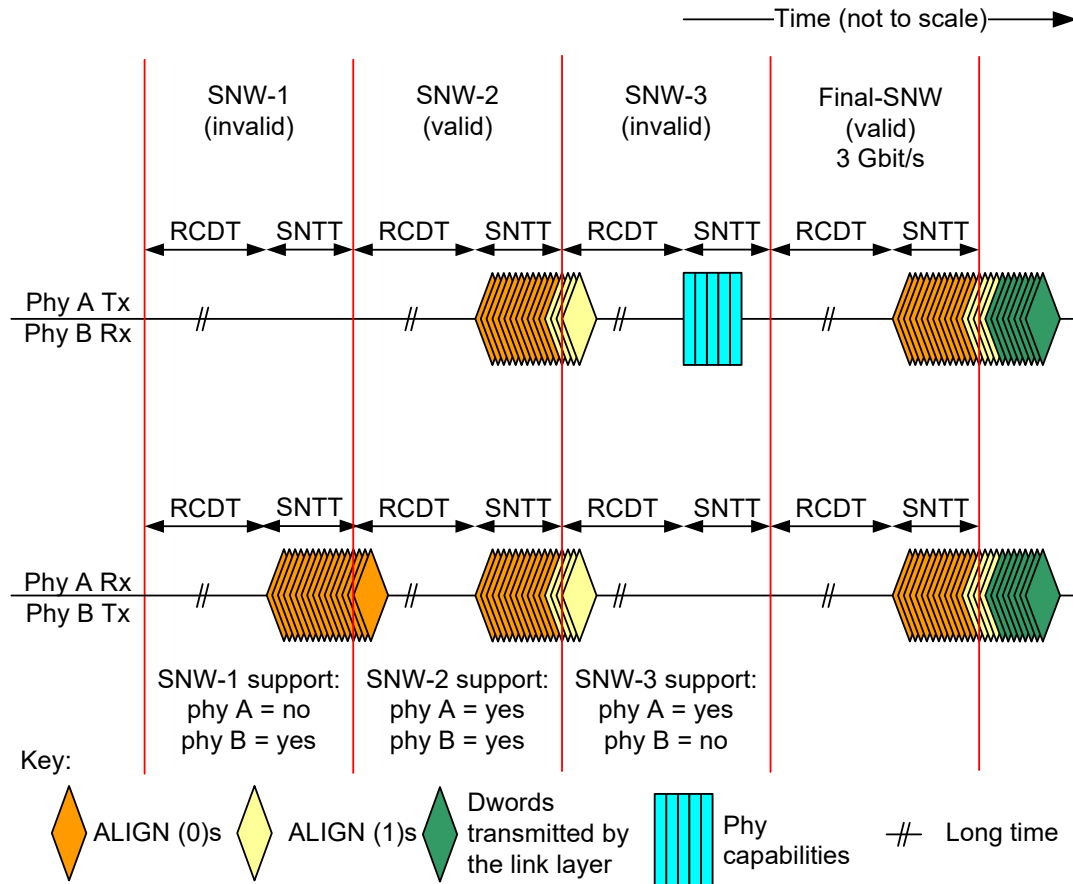


**Figure B.3 – SAS speed negotiation sequence (phy A: SNW-1, SNW-2, and SNW-3, phy B: SNW-1 and SNW-2)**

Figure B.4 shows a speed negotiation between a phy A that supports SNW-2 and SNW-3 attached to a phy B that only supports SNW-1 and SNW-2. Both phys run:

- 1) SNW-1, supported by phy B but not by phy A;
- 2) SNW-2, supported by both phys; and
- 3) SNW-3, supported by phy A but not by phy B.

Both phys then select 3 Gbit/s for Final-SNW, which is used to establish the negotiated physical link rate.



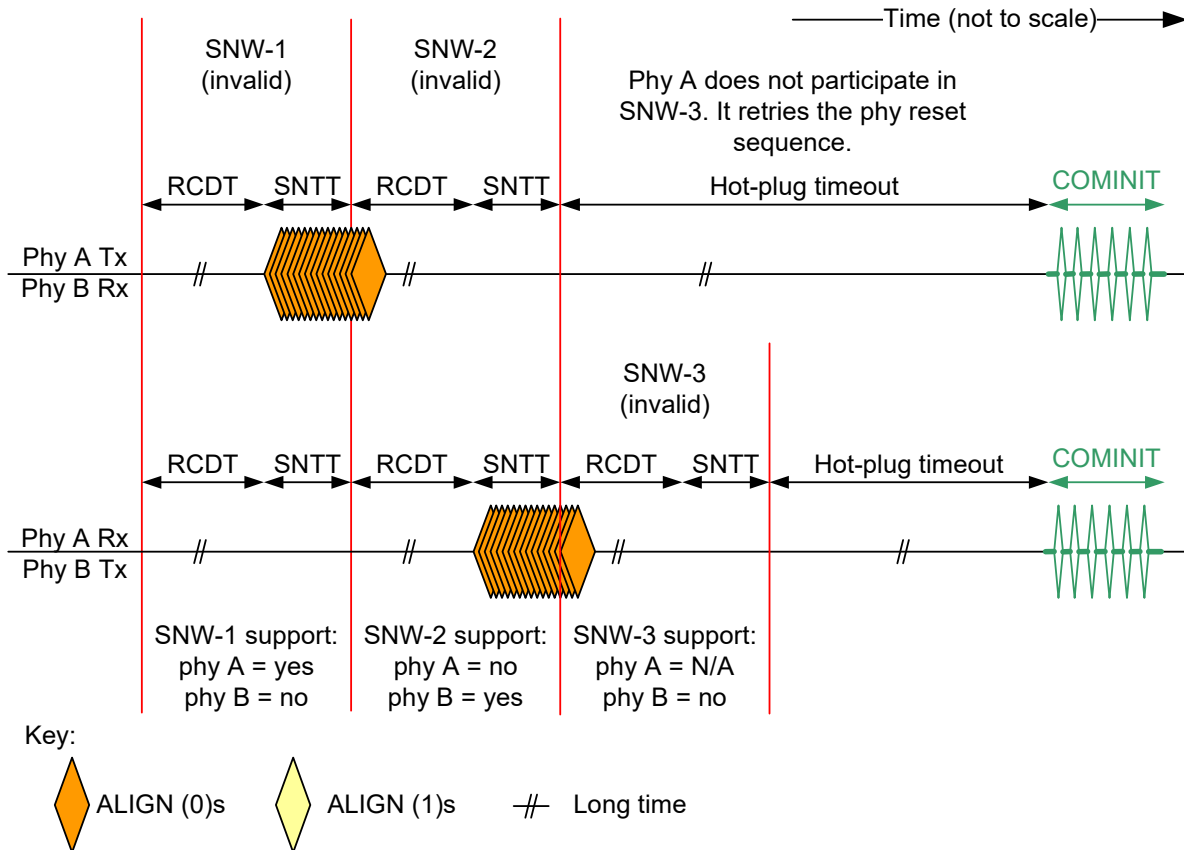
**Figure B.4 – SAS speed negotiation sequence (phy A: SNW-2, SNW-3, phy B: SNW-1, SNW-2)**

Figure B.5 shows a speed negotiation between a phy A that only supports SNW-1 attached to a phy B that only supports SNW-2. Both phys run:

- 1) SNW-1, supported by phy A but not by phy B; and
- 2) SNW-2, supported by phy B but not by phy A.

Phy B continues to run SNW-3, but phy A determines speed negotiation is unsuccessful and may attempt another phy reset sequence after a hot-plug timeout.

Phy B determines speed negotiation is not succeeding after SNW-3 and may retry the phy reset sequence after a hot-plug timeout.



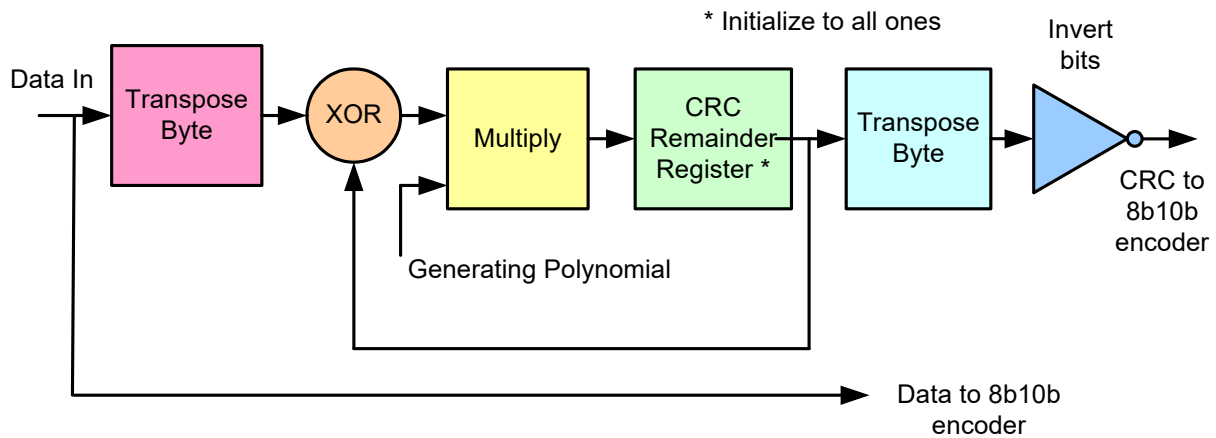
**Figure B.5 – SAS speed negotiation sequence (phy A: SNW-1 only, phy B: SNW-2 only)**

## Annex C (informative)

### CRC

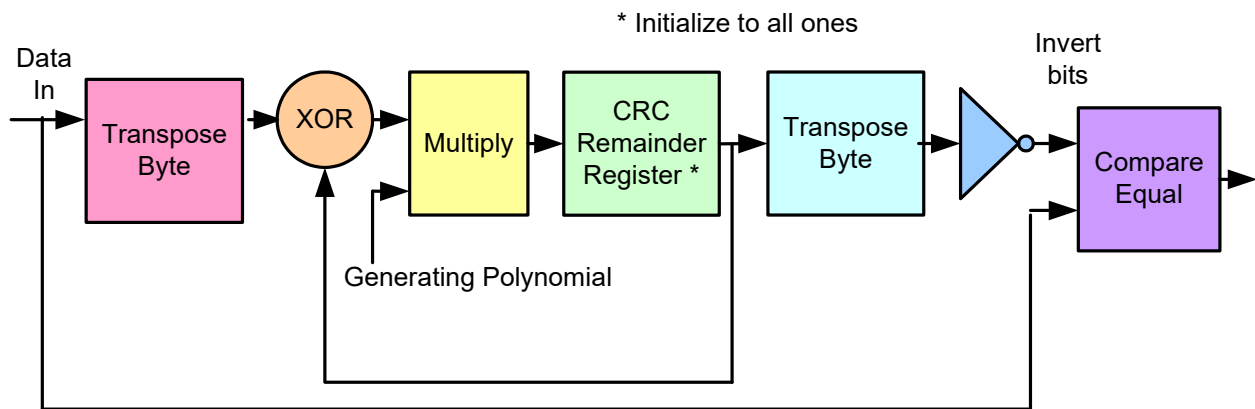
#### C.1 CRC generator and checker implementation examples

Figure C.1 shows an example of a CRC generator.



**Figure C.1 – CRC generator example**

Figure C.2 shows an example of a CRC checker.



**Figure C.2 – CRC checker example**

#### C.2 CRC implementation in C

The following example C program generates the value for the CRC field in frames. The inputs are the data dwords for the frame and the number of data dwords.

```
#include <stdio.h>
```

```

void main (void) {

    static unsigned long data_dwords[] = {
        0x06D0B992L, 0x00B5DF59L, 0x00000000L,
        0x00000000L, 0x1234FFFFL, 0x00000000L,
        0x00000000L, 0x00000000L, 0x00000000L,
        0x08000012L, 0x01000000L, 0x00000000L,
        0x00000000L}; /* example data dwords */

    unsigned long calculate_crc(unsigned long *, unsigned long);
    unsigned long crc;

    crc = calculate_crc(data_dwords, 13);
    printf ("Example CRC is %x\n", crc);
}

/* returns crc value */
unsigned long calculate_crc(unsigned long *frame, unsigned long length) {
    long poly = 0x04C11DB7L;
    unsigned long crc_gen, x;
    union {
        unsigned long lword;
        unsigned char byte[4];
    } b_access;
    static unsigned char xpose[] = {
        0x0, 0x8, 0x4, 0xC, 0x2, 0xA, 0x6, 0xE,
        0x1, 0x9, 0x5, 0xD, 0x3, 0xB, 0x7, 0xF};
    unsigned int i, j, fb;

    crc_gen = ~0; /* seed generator with all ones */
    for (i = 0; i < length; i++) {
        x = *frame++; /* get word */
        b_access.lword = x; /* transpose bits in byte */
        for (j = 0; j < 4; j++) {
            b_access.byte[j] = xpose[b_access.byte[j] >> 4] |
                xpose[b_access.byte[j] & 0xF] << 4;
        }
        x = b_access.lword;

        for (j = 0; j < 32; j++) { /* serial shift register implementation */
            fb = ((x & 0x80000000L) > 0) ^ ((crc_gen & 0x80000000L) > 0);
            x <<= 1;
            crc_gen <<= 1;
            if (fb)
                crc_gen ^= poly;
        }
    }

    b_access.lword = crc_gen; /* transpose bits in CRC */
    for (j = 0; j < 4; j++) {
        b_access.byte[j] = xpose[b_access.byte[j] >> 4] |
            xpose[b_access.byte[j] & 0xF] << 4;
    }
    crc_gen = b_access.lword;

    return ~crc_gen; /* invert output */
}

```



### C.3 CRC implementation with XORs

These equations implement the multiply function shown in figure C.1 and figure C.2. The ^ symbol represents an XOR operation.

```

crc00 = d00^d06^d09^d10^d12^d16^d24^d25^d26^d28^d29^d30^d31;
crc01 = d00^d01^d06^d07^d09^d11^d12^d13^d16^d17^d24^d27^d28;
crc02 = d00^d01^d02^d06^d07^d08^d09^d13^d14^d16^d17^d18^d24^d26^d30^d31;
crc03 = d01^d02^d03^d07^d08^d09^d10^d14^d15^d17^d18^d19^d25^d27^d31;
crc04 =
d00^d02^d03^d04^d06^d08^d11^d12^d15^d18^d19^d20^d24^d25^d29^d30^d31;
crc05 = d00^d01^d03^d04^d05^d06^d07^d10^d13^d19^d20^d21^d24^d28^d29;
crc06 = d01^d02^d04^d05^d06^d07^d08^d11^d14^d20^d21^d22^d25^d29^d30;
crc07 = d00^d02^d03^d05^d07^d08^d10^d15^d16^d21^d22^d23^d24^d25^d28^d29;
crc08 = d00^d01^d03^d04^d08^d10^d11^d12^d17^d22^d23^d28^d31;
crc09 = d01^d02^d04^d05^d09^d11^d12^d13^d18^d23^d24^d29;
crc10 = d00^d02^d03^d05^d09^d13^d14^d16^d19^d26^d28^d29^d31;
crc11 =
d00^d01^d03^d04^d09^d12^d14^d15^d16^d17^d20^d24^d25^d26^d27^d28^d31;
crc12 =
d00^d01^d02^d04^d05^d06^d09^d12^d13^d15^d17^d18^d21^d24^d27^d30^d31;
crc13 = d01^d02^d03^d05^d06^d07^d10^d13^d14^d16^d18^d19^d22^d25^d28^d31;
crc14 = d02^d03^d04^d06^d07^d08^d11^d14^d15^d17^d19^d20^d23^d26^d29;
crc15 = d03^d04^d05^d07^d08^d09^d12^d15^d16^d18^d20^d21^d24^d27^d30;
crc16 = d00^d04^d05^d08^d12^d13^d17^d19^d21^d22^d24^d26^d29^d30;
crc17 = d01^d05^d06^d09^d13^d14^d18^d20^d22^d23^d25^d27^d30^d31;
crc18 = d02^d06^d07^d10^d14^d15^d19^d21^d23^d24^d26^d28^d31;
crc19 = d03^d07^d08^d11^d15^d16^d20^d22^d24^d25^d27^d29;
crc20 = d04^d08^d09^d12^d16^d17^d21^d23^d25^d26^d28^d30;
crc21 = d05^d09^d10^d13^d17^d18^d22^d24^d26^d27^d29^d31;
crc22 = d00^d09^d11^d12^d14^d16^d18^d19^d23^d24^d26^d27^d29^d31;
crc23 = d00^d01^d06^d09^d13^d15^d16^d17^d19^d20^d26^d27^d29^d31;
crc24 = d01^d02^d07^d10^d14^d16^d17^d18^d20^d21^d27^d28^d30;
crc25 = d02^d03^d08^d11^d15^d17^d18^d19^d21^d22^d28^d29^d31;
crc26 = d00^d03^d04^d06^d10^d18^d19^d20^d22^d23^d24^d25^d26^d28^d31;
crc27 = d01^d04^d05^d07^d11^d19^d20^d21^d23^d24^d25^d26^d27^d29;
crc28 = d02^d05^d06^d08^d12^d20^d21^d22^d24^d25^d26^d27^d28^d30;
crc29 = d03^d06^d07^d09^d13^d21^d22^d23^d25^d26^d27^d28^d29^d31;
crc30 = d04^d07^d08^d10^d14^d22^d23^d24^d26^d27^d28^d29^d30;
crc31 = d05^d08^d09^d11^d15^d23^d24^d25^d27^d28^d29^d30^d31;

```

## C.4 CRC examples

Table C.1 shows several CRC examples when SAS dword mode is enabled. Table C.2 shows several CRC examples when SAS packet mode is enabled. Data is shown in dwords, from first to last.

**Table C.1 – CRC examples while SAS dword mode is enabled**

Frame contents	CRC	Frame contents	CRC
<SOF> 00010203h 04050607h 08090A0Bh 0C0D0E0Fh 10111213h 14151617h 18191A1Bh 1C1D1E1Fh <CRC> <EOF>	8A7E2691h	<SOF> 00000000h 00000000h 00000000h 00000000h 00000000h 00000000h 00000000h 00000000h 00000001h <CRC> <EOF>	3B650D6Eh
<SOF> 00000001h 00000000h 00000000h 00000000h 00000000h 00000000h 00000000h 00000000h 00000000h <CRC> <EOF>	898C0D7Ah	<SOF> 06D0B992h 00B5DF59h 00000000h 00000000h 1234FFFFh 00000000h 00000000h 00000000h 00000000h 00000000h 08000012h 01000000h 00000000h 00000000h <CRC> <EOF>	3F4F1C26h

Table C.2 – CRC examples while SAS packet mode is enabled

Frame contents	CRC	Frame contents	CRC
<SOF> 00010203h 04050607h 08090A0Bh 0C0D0E0Fh 10111213h 14151617h 18191A1Bh 1C1D1E1Fh <CRC> <Pad dword> <Pad dword> <Pad dword> <B_EOF (3)>	8A7E2691h	<SOF> 00000000h 00000000h 00000000h 00000000h 00000000h 00000000h 00000000h 00000000h 00000001h <CRC> <Pad dword> <Pad dword> <Pad dword> <B_EOF (3)>	3B650D6Eh
<SOF> 00000001h 00000000h 00000000h 00000000h 00000000h 00000000h 00000000h 00000000h 00000000h <CRC> <Pad dword> <Pad dword> <Pad dword> <B_EOF (3)>	898C0D7Ah	<SOF> 06D0B992h 00B5DF59h 00000000h 00000000h 1234FFFFh 00000000h 00000000h 00000000h 00000000h 00000000h 08000012h 01000000h 00000000h 00000000h <CRC> <Pad dword> <Pad dword> <B_EOF (2)>	3F4F1C26h

## Annex D (informative)

### Forward error correction encoding while in SAS packet mode

#### D.1 Forward error correction encoding overview

Forward error correction utilizes redundant information to detect and correct errors at the receiver. The redundant information is transmitted by way of parity symbols that are generated by a Reed Solomon code encoding function. This annex provides some example implementations that describe the encoding process along with some example results. The specific Reed Solomon code utilized in this standard is defined in 5.5.7.

#### D.2 Forward error correction encoder implementation example

Figure D.1 shows a forward error correction encoder used in SAS packet mode. The SPL packet header and payload are realigned into 26 message symbols. The message symbols are passed into the encoding function to produce four parity symbols. Finally, this example implementation reorders all symbols by interleaving the parity symbols within the original message symbols using the ordering described in 5.5.7.2.

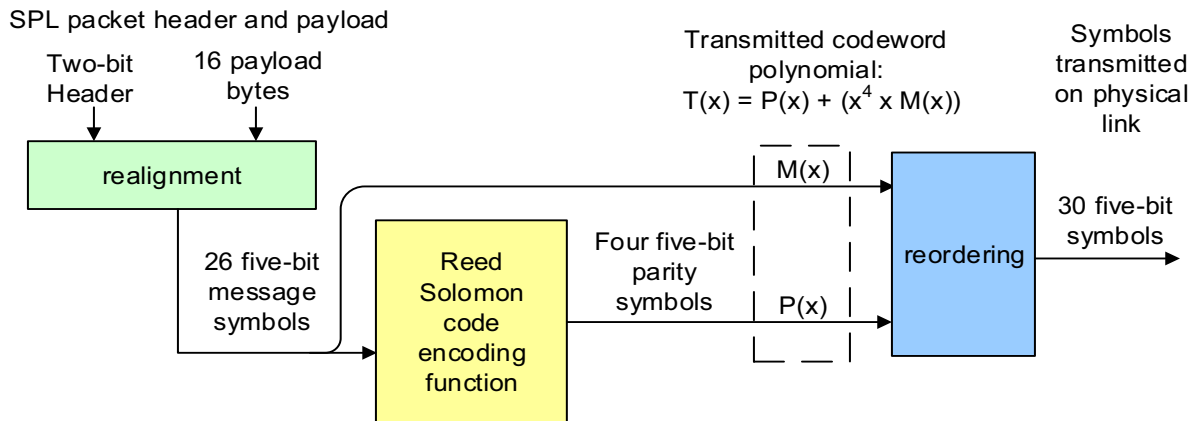
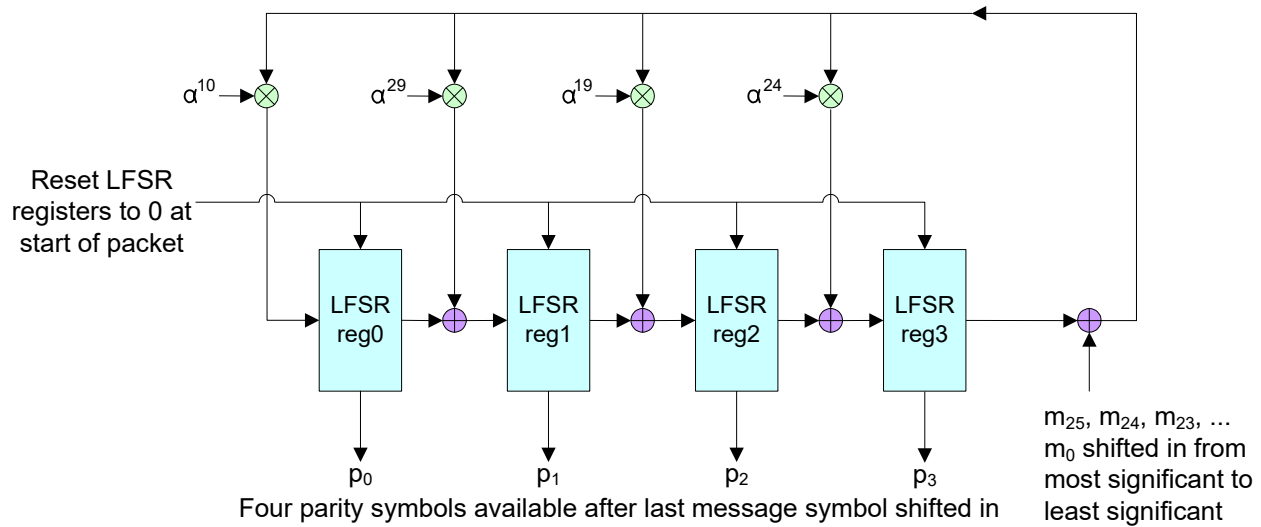


Figure D.1 – SAS packet mode forward error correction encoder

#### D.3 Reed Solomon code encoding function block diagram

An example implementation of a Reed Solomon code encoding function with a linear feedback shift register is shown in figure D.2. In this example, the message symbols are shifted into the linear feedback shift register from most significant symbol to least significant symbol. After the shifting is complete, the registers contain the parity symbols.



Key:

⊗ = Galois Field GF(32) multiplication

⊕ = Galois Field GF(32) addition

**Figure D.2 – Reed Solomon linear feedback shift register implementation**

## D.4 Forward error correction encoder implementation in C

The following example C program:

- 1) realigns a packet header and payload into 26 message symbols;
- 2) calculates four parity symbols based on the message symbols; and
- 3) reorders all symbols by interleaving the parity symbols within the original message symbols.

```
#include <stdio.h>
// gen_parity() determines the parity symbols for a SAS-4 packet and
// interleaves into full set of 30 symbols

// structure definitions
typedef unsigned char symbol;

struct splpacket_byte {
    unsigned char header;
    unsigned char payload[16];
};

struct splpacket_symb {
    symbol symb[30];
};

struct message_symb {
    symbol symb[26];
};

struct parity_symb {
```

```

symbol symb[4];
};

struct encoded_symb {
symbol symb[30];
};

// support functions
struct splpacket_byte packet_load(void);
struct message_symb packet_to_message(struct splpacket_byte);
unsigned char symbol_to_alpha(symbol symb);
struct parity_symb gen_parity(struct message_symb);
struct encoded_symb gen_fec(struct message_symb, struct parity_symb);

// display functions
void packet_display(struct splpacket_byte);
void message_display(struct message_symb);
void parity_display(struct parity_symb);
void encoded_display(struct encoded_symb);

// tables
// Galois Field table where index represents the power of alpha and
// table entry is the decimal representation of the polynomial in alpha
const symbol GFtable[31] = { 1, 2, 4, 8, 16, 5, 10, 20, 13, 26, 17, 7, 14,
    28, 29, 31, 27, 19, 3, 6, 12, 24, 21, 15, 30, 25, 23, 11, 22, 9, 18};
// SPL-4 G(x) where index is power of x and entry is the power of alpha
const symbol GenPoly[5] = {10, 29, 19, 24, 0};

// main process
int main(void) {
struct splpacket_byte packetbytes;
struct message_symb message;
struct parity_symb parity;
struct encoded_symb encoded;
// translate packet bytes to symbols, gen parity symbols, interleave symbols
packetbytes = packet_load();
message = packet_to_message(packetbytes);
parity = gen_parity(message);
encoded = gen_fec(message, parity);
// display various results
packet_display(packetbytes);
message_display(message);
parity_display(parity);
encoded_display(encoded);
}

// support functions
struct splpacket_byte packet_load(void)
{
struct splpacket_byte packet;
    packet.header = 0x01; // primitive packet header
    packet.payload[0] = 0xE2; // PACKET_SYNC Extended Binary
    packet.payload[1] = 0x89;
    packet.payload[2] = 0xE6;
    packet.payload[3] = 0xCA;
    packet.payload[4] = 0x17;
    packet.payload[5] = 0x8E;
}

```

```

    packet.payload[6] = 0x64;
    packet.payload[7] = 0x6B;
    packet.payload[8] = 0x6F;
    packet.payload[9] = 0x2C;
    packet.payload[10] = 0xDD;
    packet.payload[11] = 0x99;
    packet.payload[12] = 0xEA;
    packet.payload[13] = 0x0F;
    packet.payload[14] = 0x04;
    packet.payload[15] = 0x43;
    return packet;
}

struct message_symb packet_to_message(struct splpacket_byte rawpacket)
{
    int i;
    struct message_symb packet;
    // note that this function maintains symbols with LS bit on the right
    // (i.e., not the transmit order)
    packet.symb[0] = ((rawpacket.payload[0] & 0x07) << 2)
        | rawpacket.header;
    packet.symb[1] = ((rawpacket.payload[0] & 0xF8) >> 3);
    packet.symb[2] = (rawpacket.payload[1] & 0x1F);
    packet.symb[3] = ((rawpacket.payload[2] & 0x03) << 3)
        | ((rawpacket.payload[1] & 0xE0) >> 5);
    packet.symb[4] = ((rawpacket.payload[2] & 0x7C) >> 2);
    packet.symb[5] = ((rawpacket.payload[3] & 0x0F) << 1)
        | ((rawpacket.payload[2] & 0x80) >> 7);
    packet.symb[6] = ((rawpacket.payload[4] & 0x01) << 4)
        | ((rawpacket.payload[3] & 0xF0) >> 4);
    packet.symb[7] = ((rawpacket.payload[4] & 0x3E) >> 1);
    packet.symb[8] = ((rawpacket.payload[5] & 0x07) << 2)
        | ((rawpacket.payload[4] & 0xC0) >> 6);
    packet.symb[9] = ((rawpacket.payload[5] & 0xF8) >> 3);
    // alignment repeats similar to symbols 2-9
    packet.symb[10] = (rawpacket.payload[6] & 0x1F);
    packet.symb[11] = ((rawpacket.payload[7] & 0x03) << 3)
        | ((rawpacket.payload[6] & 0xE0) >> 5);
    packet.symb[12] = ((rawpacket.payload[7] & 0x7C) >> 2);
    packet.symb[13] = ((rawpacket.payload[8] & 0x0F) << 1)
        | ((rawpacket.payload[7] & 0x80) >> 7);
    packet.symb[14] = ((rawpacket.payload[9] & 0x01) << 4)
        | ((rawpacket.payload[8] & 0xF0) >> 4);
    packet.symb[15] = ((rawpacket.payload[9] & 0x3E) >> 1);
    packet.symb[16] = ((rawpacket.payload[10] & 0x07) << 2)
        | ((rawpacket.payload[9] & 0xC0) >> 6);
    packet.symb[17] = ((rawpacket.payload[10] & 0xF8) >> 3);
    // alignment repeats similar to symbols 2-9
    packet.symb[18] = (rawpacket.payload[11] & 0x1F);
    packet.symb[19] = ((rawpacket.payload[12] & 0x03) << 3)
        | ((rawpacket.payload[11] & 0xE0) >> 5);
    packet.symb[20] = ((rawpacket.payload[12] & 0x7C) >> 2);
    packet.symb[21] = ((rawpacket.payload[13] & 0x0F) << 1)
        | ((rawpacket.payload[12] & 0x80) >> 7);
    packet.symb[22] = ((rawpacket.payload[14] & 0x01) << 4)
        | ((rawpacket.payload[13] & 0xF0) >> 4);
    packet.symb[23] = ((rawpacket.payload[14] & 0x3E) >> 1);

```

```

packet.symb[24] = ((rawpacket.payload[15] & 0x07) << 2)
                | ((rawpacket.payload[14] & 0xC0) >> 6);
packet.symb[25] = ((rawpacket.payload[15] & 0xF8) >> 3);
return packet;
}

unsigned char symbol_to_alpha(symbol symb)
{
unsigned char i;
    for (i = 0; i < 31; i++) {
        if (symb == GFtable[i]) {
            break;
        }
    }
    return i;
}

// Reed Solomon parity encoder
struct parity_symb gen_parity(struct message_symb message)
{
symbol feedbackPath;
struct parity_symb lfsr;
int i, passProduct;
    lfsr.symb[3] = 0;
    lfsr.symb[2] = 0;
    lfsr.symb[1] = 0;
    lfsr.symb[0] = 0;
    printf("| i|msg|fbp|pass|a(fbp)|lfsr0|lfsr1|lfsr2|lfsr3|\n");
    printf("-----\n");
    for (i = 25; i >= 0; i--) { // iterate from MS to LS message symbol
        feedbackPath = (message.symb[i] ^ lfsr.symb[3]);
        if (feedbackPath == 0) {
            passProduct = 0; // zero out product with G(x) if feedback
                            // is 0 since working in GF
        } else {
            passProduct = 1; // or allow product to pass if feedback
                            // is non zero
        }
        lfsr.symb[3] = passProduct * GFtable[((GenPoly[3] +
            symbol_to_alpha(feedbackPath)) % 31)] ^ lfsr.symb[2];
            // LFSR3 = (message + LFSR3) * G3 + LFSR2
        lfsr.symb[2] = passProduct * GFtable[((GenPoly[2] +
            symbol_to_alpha(feedbackPath)) % 31)] ^ lfsr.symb[1];
            // LFSR2 = (message + LFSR2) * G2 + LFSR1
        lfsr.symb[1] = passProduct * GFtable[((GenPoly[1] +
            symbol_to_alpha(feedbackPath)) % 31)] ^ lfsr.symb[0];
            // LFSR1 = (message + LFSR1) * G1 + LFSR0
        lfsr.symb[0] = passProduct * GFtable[((GenPoly[0] +
            symbol_to_alpha(feedbackPath)) % 31)];
            // LFSR0 = (message + LFSR0) * G0
    }
    printf("|%2d| %02X| %02X| %02X| %2d| %02X| %02X| %02X| %02X|\n",
        i, message.symb[i], feedbackPath, passProduct,
        symbol_to_alpha(feedbackPath),
        lfsr.symb[0], lfsr.symb[1], lfsr.symb[2], lfsr.symb[3]);
    }
    return lfsr;
}

```



```

// generate full FEC code word by interleaving parity with message per SPL-4
struct encoded_symb gen_fec(struct message_symb msg, struct parity_symb par)
{
    struct encoded_symb encoded;
    int i;
    int p = 0;
    encoded.symb[0] = msg.symb[0];
    for (i = 1; i < 30; i++) {
        if ((i % 6) == 0) { // insert parity symbol
            encoded.symb[i] = par.symb[p];
            p++;
        } else { // insert next message symbol
            encoded.symb[i] = msg.symb[i-p];
        }
    }
    return encoded;
}

// display functions
void packet_display(struct splpacket_byte packet)
{
    int i;
    printf("header: %d \n", packet.header);
    printf("payload bytes:\n");
    for (i = 0; i < 16; i++) {
        printf("%2d ", i);
    }
    printf("\n");
    for (i = 0; i < 16; i++) {
        printf("%02X ", packet.payload[i]);
    }
    printf("\n");
}

void message_display(struct message_symb message)
{
    int i;
    printf("message symbols:\n");
    for (i = 0; i < 26; i++) {
        printf("%2d ", i);
    }
    printf("\n");
    for (i = 0; i < 26; i++) {
        printf("%02X ", message.symb[i]);
    }
    printf("\n");
}

void parity_display(struct parity_symb parity)
{
    int i;
    printf("parity symbols:\n");
    for (i = 0; i < 4; i++) {
        printf("%2d ", i);
    }
    printf("\n");
    for (i = 0; i < 4; i++) {

```

```

        printf("%02X ",parity.symb[i]);
    }
    printf("\n");
}

void encoded_display(struct encoded_symb encoded)
{
    int i;
    printf("interleaved codeword symbols (transmit LS to MS bit on wire):\n");
    for (i = 0; i <30; i++) {
        printf("%2d ",i);
    }
    printf("\n");
    for (i = 0; i <30; i++) {
        printf("%02X ",encoded.symb[i]);
    }
    printf("\n");
}

```

The code given in the example C program produces the following output, showing the forward error correction parity symbols produced when encoding a PACKET\_SYNC extended binary primitive.

i	msg	fbp	pass	a(fbp)	lfsr0	lfsr1	lfsr2	lfsr3
25	08	08	01	3	1C	02	15	0B
24	0C	07	01	11	18	06	10	05
23	02	07	01	11	18	02	14	00
22	00	00	00	31	00	18	02	14
21	1F	0B	01	27	0A	19	07	0E
20	1A	14	01	7	13	0F	0E	06
19	14	12	01	30	1A	05	0C	01
18	19	18	01	21	01	1C	1F	11
17	1B	0A	01	6	1B	11	05	0D
16	14	19	01	25	10	14	0D	06
15	16	10	01	4	1D	14	1B	1B
14	06	1D	01	14	1E	13	10	0F
13	1E	11	01	10	0C	13	1A	18
12	1A	02	01	1	07	1E	1F	03
11	1B	18	01	21	01	01	04	02
10	04	06	01	19	09	12	15	0A
9	11	1B	01	16	17	14	02	0F
8	18	17	01	26	05	09	09	04
7	0B	0F	01	23	04	1D	0E	12
6	1C	0E	01	12	15	15	1C	0B
5	15	1E	01	24	08	00	1B	0F
4	19	16	01	28	14	1F	1B	03
3	14	17	01	26	05	0A	02	1D
2	09	14	01	7	13	00	1D	03
1	1C	1F	01	15	19	0F	08	10
0	09	19	01	25	10	16	13	0B

header: 1  
payload bytes:  
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15  
E2 89 E6 CA 17 8E 64 6B 6F 2C DD 99 EA 0F 04 43  
message symbols:  
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25  
09 1C 09 14 19 15 1C 0B 18 11 04 1B 1A 1E 06 16 14 1B 19 14 1A 1F 00 02 0C 08  
parity symbols:

```

0  1  2  3
10 16 13 0B
interleaved codeword symbols (transmit LS to MS bit on wire):
0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
26 27 28 29
09 1C 09 14 19 15 10 1C 0B 18 11 04 16 1B 1A 1E 06 16 13 14 1B 19 14 1A 0B 1F
00 02 0C 08

```

## D.5 Example forward error correction encoder results

Forward error correction encoding results for several different packets are given in table D.1.

**Table D.1 – Example forward error correction coding results (part 1 of 2)**

Packet		Message and Parity results															
PACKET_SYNC extended binary primitive:																	
Input Packet	Header	bytes															
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	01b	E2h	89h	E6h	CAh	17h	8Eh	64h	6Bh	6Fh	2Ch	DDh	99h	EAh	0Fh	04h	43h
Message M(x)	symbols $m_i$																
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
	09h	1Ch	09h	14h	19h	15h	1Ch	0Bh	18h	11h	04h	1Bh	1Ah	1Eh	06h	16h	14h
	17	18	19	20	21	22	23	24	25								
	1Bh	19h	14h	1Ah	1Fh	00h	02h	0Ch	08h								
Parity P(x)	symbols $p_i$																
	0	1	2	3													
	10h	16h	13h	0Bh													
PACKET_SYNC_LOST extended binary primitive:																	
Input Packet	Header	bytes															
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	01b	A6h	FAh	03h	C1h	50h	3Ch	D1h	5Ch	F9h	C2h	91h	6Ch	EDh	8Dh	A5h	3Bh
Message M(x)	symbols $m_i$																
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
	19h	14h	1Ah	1Fh	00h	02h	0Ch	08h	11h	07h	11h	06h	17h	12h	0Fh	01h	07h
	17	18	19	20	21	22	23	24	25								
	12h	0Ch	0Bh	1Bh	1Bh	18h	12h	0Eh	07h								
Parity P(x)	symbols $p_i$																
	0	1	2	3													
	11h	1Fh	03h	01h													
LINK_RATE_MANAGEMENT extended binary primitive:																	
Input Packet	Header	bytes															
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	01b	AEh	7Ch	E1h	48h	B6h	F6h	C6h	D2h	9Dh	A9h	FEh	40h	30h	14h	4Fh	34h

Table D.1 – Example forward error correction coding results (part 2 of 2)

Packet		Message and Parity results															
Message M(x)	symbols $m_i$																
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
	19h	15h	1Ch	0Bh	18h	11h	04h	1Bh	1Ah	1Eh	06h	16h	14h	1Bh	19h	14h	1Ah
	17	18	19	20	21	22	23	24	25								
	1Fh	00h	02h	0Ch	08h	11h	07h	11h	06h								
Parity P(x)	symbols $p_i$																
	0	1	2	3													
	03h	07h	19h	12h													
SPL frame segment:																	
Input Packet	Header	bytes															
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
		10b	00h	11h	22h	33h	44h	55h	66h	77h	88h	99h	AAh	BBh	CCh	DDh	EEh
Message M(x)	symbols $m_i$																
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
	02h	00h	11h	10h	08h	06h	03h	02h	15h	0Ah	06h	1Bh	1Dh	10h	18h	0Ch	0Ah
	17	18	19	20	21	22	23	24	25								
	15h	1Bh	05h	13h	1Bh	0Dh	17h	1Fh	1Fh								
Parity P(x)	symbols $p_i$																
	0	1	2	3													
	0Fh	08h	00h	0Ah													
Scrambled idle segment (e.g., immediately following a PACKET_SYNC):																	
Input Packet	Header	bytes															
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
		00b	6Ch	BDh	94h	98h	53h	C6h	D8h	CEh	50h	6Ah	75h	C1h	04h	4Fh	C3h
Message M(x)	symbols $m_i$																
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
	10h	0Dh	1Dh	05h	05h	11h	19h	09h	19h	18h	18h	16h	13h	01h	05h	15h	15h
	17	18	19	20	21	22	23	24	25								
	0Eh	01h	06h	01h	1Eh	14h	01h	1Fh	00h								
Parity P(x)	symbols $p_i$																
	0	1	2	3													
	05h	1Ch	16h	14h													

## **Annex E**

(informative)

### **SAS address hashing**

#### **E.1 SAS address hashing overview**

See 4.2.2 for a description of hashed SAS addresses and the algorithm used to create them.

#### **E.2 Hash collision probability**

The following are Monte-Carlo simulations evaluating the probability of collision in a system containing 128 addressable SAS ports. Four models were used for the models for the simulations:

- a) random model;
- b) sequential mode;
- c) lots model; and
- d) three lots model.

The random model uses a system with 128 randomly chosen 64-bit integers as SAS addresses.

The sequential model uses a system with 128 sequentially-assigned SAS addresses starting from a random 64-bit base.

The lots model uses:

- a) two sequentially assigned SAS addresses with unique company IDs and random vendor specific identifiers;
- b) 125 randomly drawn SAS addresses from a 10 000-unit production lot. The vendor specific identifiers within the lot were assigned by 10 SAS address-writers, randomly drawn from a pool of 4 096 possible SAS address-writers. Each SAS address-writer assigns vendor specific identifiers sequentially within its own subset of the vendor specific identifiers, starting from a randomly chosen base at the beginning of the production run; and
- c) one randomly chosen SAS address with another unique company ID, representing a replacement unit.

The three lots model uses:

- a) two sequentially assigned SAS addresses with unique company IDs and random vendor specific identifiers;
- b) 125 randomly drawn SAS addresses from three 10 000-unit lots. The vendor specific identifiers within each lot were assigned by 10 SAS address-writers, randomly drawn from a pool of 4 096 possible SAS address-writers for that vendor. Each SAS address-writer assigns vendor specific identifiers sequentially within its own subset of the vendor specific identifiers, starting from a randomly chosen base at the beginning of the production run. Each of the three lots has a different company ID; and
- c) one randomly chosen SAS address with another unique company ID, representing a replacement unit.

Table E.1 lists the results of Monte-Carlo simulation.

Table E.1 – Monte-Carlo simulation results

SAS address model	Trials	Collisions	Average collisions per system
lots	2 000 000 000	45 063	0.000 022 531 5
three lots	2 000 000 000	662 503	0.000 331 251 5
random	10 000 000	4 882	0.000 488 2
sequential	10 000 000	0	0

### E.3 Hash generation

One way to implement the hashing encoder in hardware is to use serial shift registers as shown in figure E.1. For error correction purposes, the number of data bits is limited to 39. For hashing purposes, the circuit shown serves as a divider. Because the period of this generator polynomial is 63, any binary sequence of length exceeding 63 is treated as a 63-bit sequence with  $(\text{bit } 63) \times L + k$  added to  $(\text{bit } k \text{ modulo } 2)$  for  $k = 0$ , to 62 and any integer  $L$ . Therefore, using this generator polynomial to hash a 64-bit address is equivalent to hashing a 63-bit sequence with bit 63 added modulo 2 to bit 0. With this wrapping, a binary sequence of any length is treated as an equivalent binary sequence of 63 bits, which, in turn, is treated as a degree-62 polynomial. After feeding this equivalent degree-62 polynomial into the circuit shown, the shift register contains the remainder from dividing the degree-62 input polynomial by the generator polynomial. This remainder is the hashed result.

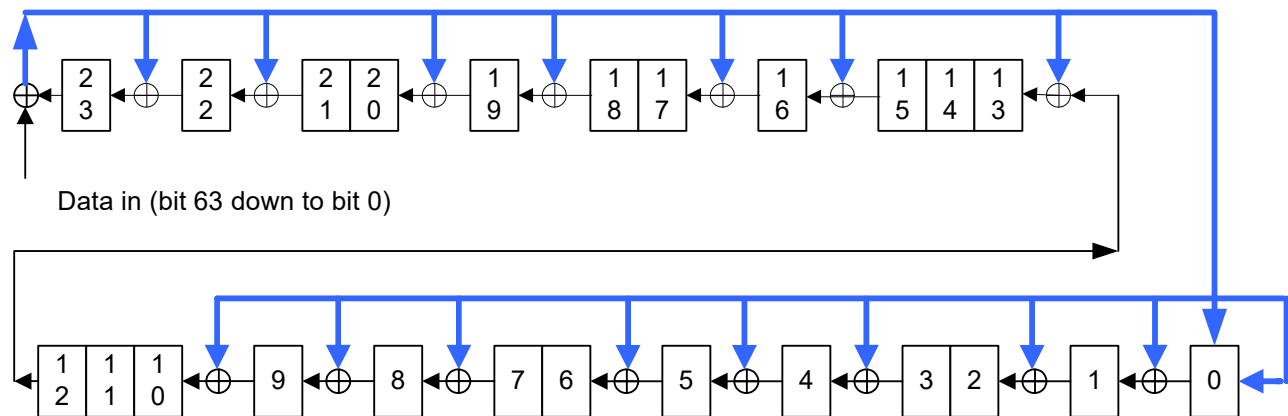


Figure E.1 – BCH(69, 39, 9) code generator

### E.4 Hash implementation in C

The following example C program generates a 24-bit hashed value from a 64-bit value.

```
typedef unsigned int uint32_t;
uint32_t hash(uint32_t upperbits, uint32_t lowerbits)
{
    const unsigned distance_9_poly = 0x01DB2777;
```

```

uint32_t msb = 0x01000000;
uint32_t moving_one, leading_bit;
int i;
unsigned regg;
regg = 0;
moving_one = 0x80000000;
for (i = 31; i >= 0; i--) {
    leading_bit = 0;
    if (moving_one & upperbits) leading_bit = msb;
    regg <= 1;
    regg ^= leading_bit;
    if (regg & msb) regg ^= distance_9_poly;
    moving_one >>= 1;
}
moving_one = 0x80000000;
for (i = 31; i >= 0; i--) { // note lower limit of i = 0;
    leading_bit = 0;
    if (moving_one & lowerbits) leading_bit = msb;
    regg <= 1;
    regg ^= leading_bit;
    if (regg & msb) regg ^= distance_9_poly;
    moving_one >>= 1;
}
return regg & 0x00FFFFFF;
}

```

## E.5 Hash implementation with XORs

These equations generate the 24-bit hashed SAS address for the SSP frame header from a 64-bit SAS address. The ^ symbol represents an XOR.

```

hash00=d00^d01^d03^d05^d07^d09^d10^d11^d12^d15^d16^d17^d18^d19^d20^d21^d22^d
23^d24^d25^d28^d30^d31^d33^d34^d36^d38^d39^d63;
hash01=d00^d02^d03^d04^d05^d06^d07^d08^d09^d13^d15^d26^d28^d29^d30^d32^d33^d
35^d36^d37^d38^d40^d63;
hash02=d00^d04^d06^d08^d11^d12^d14^d15^d17^d18^d19^d20^d21^d22^d23^d24^d25^d
27^d28^d29^d37^d41^d63;
hash03=d01^d05^d07^d09^d12^d13^d15^d16^d18^d19^d20^d21^d22^d23^d24^d25^d26^d
28^d29^d30^d38^d42;
hash04=d00^d01^d02^d03^d05^d06^d07^d08^d09^d11^d12^d13^d14^d15^d18^d26^d27^d
28^d29^d33^d34^d36^d38^d43^d63;
hash05=d00^d02^d04^d05^d06^d08^d11^d13^d14^d17^d18^d20^d21^d22^d23^d24^d25^d
27^d29^d31^d33^d35^d36^d37^d38^d44^d63;
hash06=d00^d06^d10^d11^d14^d16^d17^d20^d26^d31^d32^d33^d37^d45^d63;
hash07=d01^d07^d11^d12^d15^d17^d18^d21^d27^d32^d33^d34^d38^d46;
hash08=d00^d01^d02^d03^d05^d07^d08^d09^d10^d11^d13^d15^d17^d20^d21^d23^d24^d
25^d30^d31^d35^d36^d38^d47^d63;
hash09=d00^d02^d04^d05^d06^d07^d08^d14^d15^d17^d19^d20^d23^d26^d28^d30^d32^d
33^d34^d37^d38^d48^d63;
hash10=d00^d06^d08^d10^d11^d12^d17^d19^d22^d23^d25^d27^d28^d29^d30^d35^d36^d
49^d63;
hash11=d01^d07^d09^d11^d12^d13^d18^d20^d23^d24^d26^d28^d29^d30^d31^d36^d37^d
50;
hash12=d02^d08^d10^d12^d13^d14^d19^d21^d24^d25^d27^d29^d30^d31^d32^d37^d38^d
51;
hash13=d00^d01^d05^d07^d10^d12^d13^d14^d16^d17^d18^d19^d21^d23^d24^d26^d32^d

```

```

34^d36^d52^d63;
hash14=d01^d02^d06^d08^d11^d13^d14^d15^d17^d18^d19^d20^d22^d24^d25^d27^d33^d
35^d37^d53;
hash15=d02^d03^d07^d09^d12^d14^d15^d16^d18^d19^d20^d21^d23^d25^d26^d28^d34^d
36^d38^d54;
hash16=d00^d01^d04^d05^d07^d08^d09^d11^d12^d13^d18^d23^d25^d26^d27^d28^d29^d
30^d31^d33^d34^d35^d36^d37^d38^d55^d63;
hash17=d00^d02^d03^d06^d07^d08^d11^d13^d14^d15^d16^d17^d18^d20^d21^d22^d23^d
25^d26^d27^d29^d32^d33^d35^d37^d56^d63;
hash18=d01^d03^d04^d07^d08^d09^d12^d14^d15^d16^d17^d18^d19^d21^d22^d23^d24^d
26^d27^d28^d30^d33^d34^d36^d38^d57;
hash19=d00^d01^d02^d03^d04^d07^d08^d11^d12^d13^d21^d27^d29^d30^d33^d35^d36^d
37^d38^d58^d63;
hash20=d00^d02^d04^d07^d08^d10^d11^d13^d14^d15^d16^d17^d18^d19^d20^d21^d23^d
24^d25^d33^d37^d59^d63;
hash21=d01^d03^d05^d08^d09^d11^d12^d14^d15^d16^d17^d18^d19^d20^d21^d22^d24^d
25^d26^d34^d38^d60;
hash22=d00^d01^d02^d03^d04^d05^d06^d07^d11^d13^d24^d26^d27^d28^d30^d31^d33^d
34^d35^d36^d38^d61^d63;
hash23=d00^d02^d04^d06^d08^d09^d10^d11^d14^d15^d16^d17^d18^d19^d20^d21^d22^d
23^d24^d27^d29^d30^d32^d33^d35^d37^d38^d62^d63;

```

## E.6 Hash examples

Table E.2 shows examples using simple SAS addresses as input values. Two of the input values hash to the same value.

**Table E.2 – Hash results for simple SAS addresses**

64-bit input value	24-bit hashed value
00000000 00000000h	000000h
00000000 00000001h	DB2777h
FFFFFFFF FFFFFFFFh	DB2777h

Table E.3 shows examples using realistic SAS addresses as input values.

**Table E.3 – Hash results for realistic SAS addresses**

64-bit input value	24-bit hashed value
50010753 4F0CFC88h	D0B992h
50010B92 B3CBF639h	B5DF59h
5002037E 157FEC63h	B064F7h
50004CF6 FBCE3889h	88FF12h
50020374 C4657EC7h	F36570h



**Table E.3 – Hash results for realistic SAS addresses**

64-bit input value	24-bit hashed value
50010D92 A016E450h	9F9571h
50002A58 850ACC66h	64B6B9h
50008C7B EE7910DEh	8D6135h
500508BD C22CAC94h	86ECF1h
500805F3 334B0AD3h	752AB2h
500A0B8A FAA6A820h	5543A7h
500805E6 BCC55C68h	463DEDh

Table E.4 shows examples using a walking ones pattern to generate the input values.

**Table E.4 – Hash results for a walking ones pattern (part 1 of 2)**

64-bit input value	24-bit hashed value	64-bit input value	24-bit hashed value
00000000 00000001h	DB2777h	00000001 00000000h	8232C2h
00000000 00000002h	6D6999h	00000002 00000000h	DF42F3h
00000000 00000004h	DAD332h	00000004 00000000h	65A291h
00000000 00000008h	6E8113h	00000008 00000000h	CB4522h
00000000 00000010h	DD0226h	00000010 00000000h	4DAD33h
00000000 00000020h	61233Bh	00000020 00000000h	9B5A66h
00000000 00000040h	C24676h	00000040 00000000h	ED93BBh
00000000 00000080h	5FAB9Bh	00000080 00000000h	000001h
00000000 00000100h	BF5736h	00000100 00000000h	000002h
00000000 00000200h	A5891Bh	00000200 00000000h	000004h
00000000 00000400h	903541h	00000400 00000000h	000008h
00000000 00000800h	FB4DF5h	00000800 00000000h	000010h
00000000 00001000h	2DBC9Dh	00001000 00000000h	000020h
00000000 00002000h	5B793Ah	00002000 00000000h	000040h
00000000 00004000h	B6F274h	00004000 00000000h	000080h
00000000 00008000h	B6C39Fh	00008000 00000000h	000100h
00000000 00010000h	B6A049h	00010000 00000000h	000200h

**Table E.4 – Hash results for a walking ones pattern (part 2 of 2)**

64-bit input value	24-bit hashed value	64-bit input value	24-bit hashed value
00000000 00020000h	B667E5h	00020000 00000000h	000400h
00000000 00040000h	B7E8BDh	00040000 00000000h	000800h
00000000 00080000h	B4F60Dh	00080000 00000000h	001000h
00000000 00100000h	B2CB6Dh	00100000 00000000h	002000h
00000000 00200000h	BEB1ADh	00200000 00000000h	004000h
00000000 00400000h	A6442Dh	00400000 00000000h	008000h
00000000 00800000h	97AF2Dh	00800000 00000000h	010000h
00000000 01000000h	F4792Dh	01000000 00000000h	020000h
00000000 02000000h	33D52Dh	02000000 00000000h	040000h
00000000 04000000h	67AA5Ah	04000000 00000000h	080000h
00000000 08000000h	CF54B4h	08000000 00000000h	100000h
00000000 10000000h	458E1Fh	10000000 00000000h	200000h
00000000 20000000h	8B1C3Eh	20000000 00000000h	400000h
00000000 40000000h	CD1F0Bh	40000000 00000000h	800000h
00000000 80000000h	411961h	80000000 00000000h	DB2777h

Table E.5 shows examples using a walking zeros pattern to generate the input values.

**Table E.5 – Hash results for a walking zeros pattern (part 1 of 2)**

64-bit input value	24-bit hashed value	64-bit input value	24-bit hashed value
FFFFFFFF FFFFFFFFh	000000h	FFFFFFFE FFFFFFFFh	5915B5h
FFFFFFFF FFFFFFFDh	B64EEh	FFFFFFFD FFFFFFFFh	046584h
FFFFFFFF FFFFFFFBh	01F445h	FFFFFFFB FFFFFFFFh	BE85E6h
FFFFFFFF FFFFFFF7h	B5A664h	FFFFFFF7 FFFFFFFFh	106255h
FFFFFFFF FFFFFFFEh	062551h	FFFFFFFE FFFFFFFFh	968A44h
FFFFFFFF FFFFFFFDh	BA044Ch	FFFFFFFD FFFFFFFFh	407D11h
FFFFFFFF FFFFFFFBh	196101h	FFFFFFFB FFFFFFFFh	36B4CCh
FFFFFFFF FFFFFFF7h	848CECh	FFFFFFF7 FFFFFFFFh	DB2776h

Table E.5 – Hash results for a walking zeros pattern (part 2 of 2)

64-bit input value	24-bit hashed value	64-bit input value	24-bit hashed value
FFFFFFFF FFFFFFFFh	647041h	FFFFFFEF FFFFFFFFh	DB2775h
FFFFFFFF FFFFDFh	7EAE6Ch	FFFFDF FFFFFFFFh	DB2773h
FFFFFFFF FFFFBFh	4B1236h	FFFFBF FFFFFFFFh	DB277Fh
FFFFFFFF FFFF7Fh	206A82h	FFFF7F FFFFFFFFh	DB2767h
FFFFFFFF FFFEFFFFh	F69BEAh	FFEFFFF FFFFFFFFh	DB2757h
FFFFFFFF FFFDFFFh	805E4Dh	FFFDFF FFFFFFFFh	DB2737h
FFFFFFFF FFFBFFFh	6DD503h	FFFBFF FFFFFFFFh	DB27F7h
FFFFFFFF FFFF7FFFh	6DE4E8h	FFFF7FFF FFFFFFFFh	DB2677h
FFFFFFFF FFFEFFFFh	6D873Eh	FFEFFFF FFFFFFFFh	DB2577h
FFFFFFFF FFFDFFFh	6D4092h	FFFDFFF FFFFFFFFh	DB2377h
FFFFFFFF FFFBFFFh	6CCFCAh	FFFBFFF FFFFFFFFh	DB2F77h
FFFFFFFF FFF7FFFFh	6FD17Ah	FFF7FFF FFFFFFFFh	DB3777h
FFFFFFFF FFEFFFFh	69EC1Ah	FFEFFFF FFFFFFFFh	DB0777h
FFFFFFFF FFDFFFFh	6596DAh	FFDFFFF FFFFFFFFh	DB6777h
FFFFFFFF FFBFFFFh	7D635Ah	FFBFFFF FFFFFFFFh	DBA777h
FFFFFFFF FF7FFFFh	4C885Ah	FF7FFFF FFFFFFFFh	DA2777h
FFFFFFFF FEFFFFh	2F5E5Ah	FEFFFF FFFFFFFFh	D92777h
FFFFFFFF FDFFFFh	E8F25Ah	FDFFFF FFFFFFFFh	DF2777h
FFFFFFFF FBFFFFh	BC8D2Dh	FBFFFF FFFFFFFFh	D32777h
FFFFFFFF F7FFFFh	1473C3h	F7FFFF FFFFFFFFh	CB2777h
FFFFFFFF EFFFFh	9EA968h	EFFFF FFFFFFFFh	FB2777h
FFFFFFFF DFFFFh	503B49h	DFFFF FFFFFFFFh	9B2777h
FFFFFFFF BFFFFh	16387Ch	BFFFF FFFFFFFFh	5B2777h
FFFFFFFF 7FFFFh	9A3E16h	7FFFF FFFFFFFFh	000000h

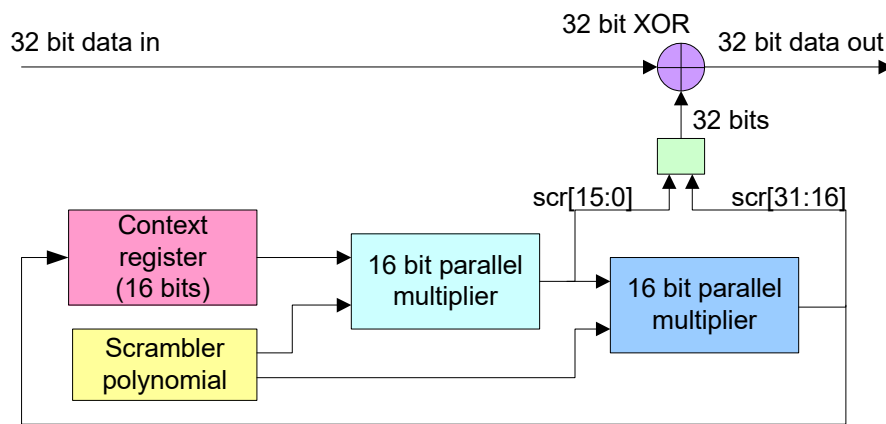
## Annex F (informative)

### Scrambling

#### F.1 Scrambling while in SAS dword mode

##### F.1.1 SAS dword mode scrambler implementation example

Figure F.1 shows an example of a SAS dword mode data scrambler. This example generates the value to XOR with the dword input with two 16-bit parallel multipliers. The maximum width for the multiplier is 16 bits as the generating polynomial is 16 bits.



**Figure F.1 – SAS dword mode Scrambler**

The generator polynomial is:

$$G(x) = x^{16} + x^{15} + x^{13} + x^4 + 1$$

For all implementations, the context register is initialized to produce a first dword output of C2D2768Dh for a dword input of all zeros.

##### F.1.2 SAS dword mode scrambler implementation in C

The following example C program generates the scrambled data dwords for transmission. The inputs are the data dword to scramble and control indication to reinitialize the residual value (e.g., following an SOF).

```
#include <stdio.h>

unsigned long scramble(int reset, unsigned long dword);
void main(void)
{
    int i;

    for (i = 0; i < 12; i++)
        printf(" %08X \n", scramble(i==0, 0)); /* scramble all 0s */
}

#define poly 0xA011
```

```

unsigned long scramble(int reset, unsigned long dword) {
static unsigned short scramble;
int i;

    if (reset)
        scramble = 0xFFFF;
    for (i = 0; i < 32; i++) /* serial shift register implementation */
    {
        dword ^= (scramble & 0x8000)? (1 << i):0;
        scramble = (scramble << 1) ^ ((scramble & 0x8000)? poly:0);
    }
    return dword;
}

```

### F.1.3 SAS dword mode scrambler implementation with XORs

These equations generate the scrambled dwords to XOR with dwords before transmission and dword reception to recover the original data. The ^ symbol represents an XOR operation. The initialized value for d[15:0] is F0F6h (i.e., 0xF0F6) in this example.

```

scr0=d15^d13^d4^d0;
scr1=d15^d14^d13^d5^d4^d1^d0;
scr2=d14^d13^d6^d5^d4^d2^d1^d0;
scr3=d15^d14^d7^d6^d5^d3^d2^d1;
scr4=d13^d8^d7^d6^d3^d2^d0;
scr5=d14^d9^d8^d7^d4^d3^d1;
scr6=d15^d10^d9^d8^d5^d4^d2;
scr7=d15^d13^d11^d10^d9^d6^d5^d4^d3^d0;
scr8=d15^d14^d13^d12^d11^d10^d7^d6^d5^d1^d0;
scr9=d14^d12^d11^d8^d7^d6^d4^d2^d1^d0;
scr10=d15^d13^d12^d9^d8^d7^d5^d3^d2^d1;
scr11=d15^d14^d10^d9^d8^d6^d3^d2^d0;
scr12=d13^d11^d10^d9^d7^d3^d1^d0;
scr13=d14^d12^d11^d10^d8^d4^d2^d1;
scr14=d15^d13^d12^d11^d9^d5^d3^d2;
scr15=d15^d14^d12^d10^d6^d3^d0;
scr16=d11^d7^d1^d0;
scr17=d12^d8^d2^d1;
scr18=d13^d9^d3^d2;
scr19=d14^d10^d4^d3;
scr20=d15^d11^d5^d4;
scr21=d15^d13^d12^d6^d5^d4^d0;
scr22=d15^d14^d7^d6^d5^d4^d1^d0;
scr23=d13^d8^d7^d6^d5^d4^d2^d1^d0;
scr24=d14^d9^d8^d7^d6^d5^d3^d2^d1;
scr25=d15^d10^d9^d8^d7^d6^d4^d3^d2;
scr26=d15^d13^d11^d10^d9^d8^d7^d5^d3^d0;
scr27=d15^d14^d13^d12^d11^d10^d9^d8^d6^d1^d0;
scr28=d14^d12^d11^d10^d9^d7^d4^d2^d1^d0;
scr29=d15^d13^d12^d11^d10^d8^d5^d3^d2^d1;
scr30=d15^d14^d12^d11^d9^d6^d3^d2^d0;
scr31=d12^d10^d7^d3^d1^d0;

```

### F.1.4 SAS dword mode scrambler examples

Table F.1 shows several SAS dword mode scrambler examples. Data is shown in dwords, from first to last.

**Table F.1 – SAS dword mode scrambler examples**

Frame contents	Scrambled output
<SOF> 06D0B992h 00B5DF59h 00000000h 00000000h 1234FFFFh 00000000h 00000000h 00000000h 00000000h 00000000h 08000012h 01000000h 00000000h 00000000h 3F4F1C26h <sup>a</sup> <EOF>	<SOF> C402CF1Fh 1F936C31h A508436Ch 3452D354h 98616AFDh BB1ABE1Bh FA56B73Dh 53F60B1Bh F0809C41h 7C7FC358h BF865291h 7A6FA7B6h 3163E6D6h CF79E22Ah <sup>a</sup> <EOF>
<SOF> 00000000h 00000000h 00000000h 00000000h 00000000h 00000000h 00000000h 00000000h 00000000h 00000000h 00000000h 00000000h 00000000h 00000000h 00000000h B00F2BCCh <sup>a</sup> <EOF>	<SOF> C2D2768Dh 1F26B368h A508436Ch 3452D354h 8A559502h BB1ABE1Bh FA56B73Dh 53F60B1Bh F0809C41h 747FC34Ah BE865291h 7A6FA7B6h 3163E6D6h 4039D5C0h <sup>a</sup> <EOF>
<sup>a</sup> The last dword represents a CRC dword.	

Table F.2 shows the first 64 dwords output by the scrambler.

**Table F.2 – Initial SAS dword mode scrambler output**

Dword	Scrambler output	Dword	Scrambler output
0	C2D2768Dh	32	F46D6948h
1	1F26B368h	33	7BCD8A93h
2	A508436Ch	34	1513AD7Eh
3	3452D354h	35	1E72FEEeh
4	8A559502h	36	A014AA3Bh
5	BB1ABE1Bh	37	23AAD4E7h
6	FA56B73Dh	38	B0DC9E67h
7	53F60B1Bh	39	E0A573FBh
8	F0809C41h	40	06CA944Fh
9	747FC34Ah	41	63E29212h
10	BE865291h	42	4578626Dh
11	7A6FA7B6h	43	53260C93h
12	3163E6D6h	44	3E592202h
13	F036FE0Ch	45	2B6ECA63h
14	1EF3EA29h	46	636A1F1Fh
15	EB342694h	47	35B5A9EDh
16	53853B17h	48	4AA2A0FDh
17	E94ADC4Dh	49	71AFE196h
18	5D200E88h	50	E1D57B62h
19	6901EDD0h	51	55A0568Ah
20	FA9E38DEh	52	82D18968h
21	68DB4B07h	53	234CB4FFh
22	450A437Bh	54	83481E7Fh
23	960DD708h	55	B21AE87Fh
24	3F35E698h	56	A9C5EACDh
25	FE7698A5h	57	6201ACC3h
26	C80EF715h	58	F60939CEh
27	666090AFh	59	395F767Dh
28	FAF0D5CBh	60	2FA55841h
29	2B82009Fh	61	836D4A7Ah
30	0E317491h	62	388D587Ah
31	76F46A1Eh	63	773DFF5Ch

## F.2 Scrambling while in SAS packet mode

### F.2.1 SAS packed mode scrambler implementation example

Figure F.2 shows an example of a SAS packet mode data scrambler. In this example an 8-bit pattern generator provides 8-bit scrambling patterns. Four successive 8-bit scrambled patterns are assembled into a dword that is XORed with the dword input. The dword resulting from the XOR is transmitted as described in 6.9.2 and 5.5.7.2.

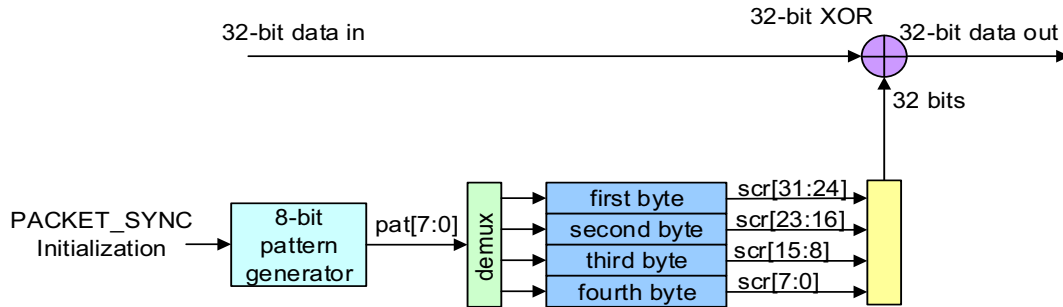


Figure F.2 – SAS packet mode scrambler

### F.2.2 SAS packet mode 8-bit pattern generator implementation in C

The following example C program:

- generates the 8-bit pattern used in SAS packet mode;
- includes a control indication to reinitialize the scrambler (i.e., following reception of a `PACKET_SYNC`); and
- sets a data byte input to the function to all zeroes to produce the required scrambler pattern.

This C program does not include the accumulation of successive bytes into a dword, as shown in figure F.2.

```
#include <stdio.h>
unsigned char patterngen(int reset, unsigned char byte);
int main(void) {
    int i;
    printf(" Step Pattern Generator Output\n");
    printf(" -----");
    for (i = 0; i < 128; i++) {
        if (i % 16 == 0) {
            printf("\n %3d to %3d ", i, i+15);
        }
        printf("%02X ", patterngen(i==0, 0)); // scramble all 0s
    }
    printf("\n");
}

#define gScram 0xA10125 // G(x) = x^23 + x^21 + x^16 + x^8 + x^5 + x^2 + 1
#define shiftreg_init 0x1DBFBC

unsigned char patterngen(int reset, unsigned char byte)
{
    static unsigned long shiftreg;
    static unsigned char patternout;
    int i, j;
```



```

patternout = 0;
if (reset)
    shiftreg = shiftreg_init;
// advance shift reg 8 times for one byte of pattern generator output
for (i = 0; i < 8; i++) {
    // output set by shift reg bit 22, put in bit 7 then shifts to bit 0
    patternout = (patternout >> 1) | (shiftreg & 0x00400000) >> 15;
    // shifting shift reg and applying xor per polynomial if bit 22 set
    shiftreg = (shiftreg << 1) ^ ((shiftreg & 0x00400000) ? gScram : 0);
}
byte ^= patternout;
return byte;
}

```

### F.2.3 SAS packet mode 8-bit pattern generator implementation block diagram

A linear feedback shift register is used within the 8-bit pattern generator shown in figure F.3. In this example, the value of bit 22 is the first output, corresponding to the LSB of the pat[7:0] bus shown in figure F.2. The shift register is clocked to produce subsequent bits comprising the first byte, second byte, third byte, etc. This example may be reorganized to produce multiple bits per clock.

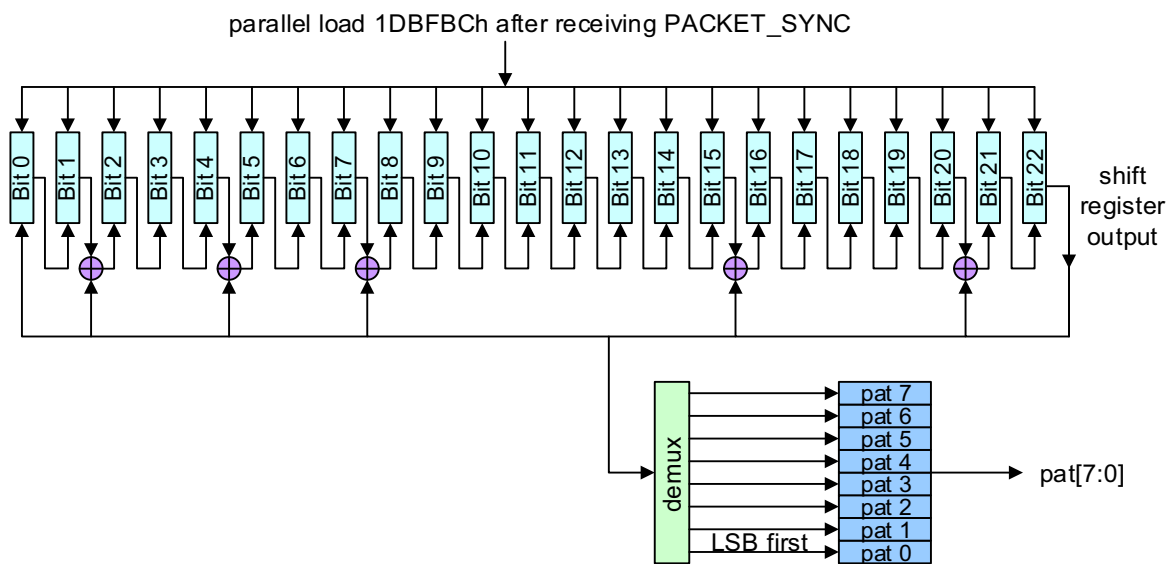


Figure F.3 – SAS packet mode 8-bit pattern generator

### F.2.4 SAS packet mode 8-bit pattern generator output

After PACKET\_SYNC initialization the first 128 steps of the 8-bit pattern generator produces the consecutive values shown in table F.3.

The 23-bit linear feedback shift register returns to its initial state if:

- $2^{23} - 1$  bits are processed; or
- a PACKET\_SYNC is received.

**Table F.3 – 8-bit pattern generator values produced after initialization of scrambler by PACKET\_SYNC**

Step	SAS packet mode pattern generator output															
0 to 15	6Ch	BDh	94h	98h	53h	C6h	D8h	CEh	50h	6Ah	75h	C1h	04h	4Fh	C3h	07h
16 to 31	75h	26h	C6h	06h	A3h	B0h	B4h	ABh	05h	11h	CCh	57h	4Eh	69h	42h	73h
32 to 47	1Dh	0Fh	B7h	03h	E0h	45h	BAh	5Eh	30h	EBh	D7h	43h	2Ch	5Dh	F5h	D0h
48 to 63	15h	41h	76h	8Eh	C3h	9Dh	D1h	57h	CDh	FFh	76h	A1h	7Ah	4Ch	64h	2Eh
64 to 79	87h	05h	A3h	24h	89h	FFh	A2h	4Bh	46h	7Ch	1Dh	62h	12h	19h	A5h	2Fh
80 to 95	E6h	B3h	CAh	33h	EDh	F3h	2Bh	88h	67h	3Eh	ABh	96h	E8h	9Eh	6Ah	5Dh
96 to 111	5Ch	1Ch	64h	1Dh	F5h	2Ch	51h	C8h	D8h	A8h	F4h	4Dh	96h	ACH	5Dh	7Ah
112 to 127	2Bh	F4h	2Fh	09h	08h	2Eh	0Eh	C9h	02h	4Bh	AFh	D9h	3Dh	4Eh	78h	E7h

## **Annex G**

(informative)

### **ATA architectural notes**

#### **G.1 STP differences from SATA**

Some of the differences of STP compared with SATA are:

- a) STP adds addressing of multiple SATA devices. Each SATA device is assigned a SAS address by its attached expander device with STP SATA bridge functionality. The STP initiator port is capable of addressing more than one STP target port;
- b) STP allows multiple STP initiator ports to share access to a SATA device behind an STP SATA bridge using affiliations (see 6.21.6);
- c) SAS interface power management is used instead of SATA interface power management;
- d) far-end analog loopback testing is not supported;
- e) far-end retimed loopback testing is not supported;
- f) near-end analog loopback testing is not supported;
- g) use of SATA\_CONT is required; and
- h) BIST Activate frames are not supported.

#### **G.2 STP differences from SATA**

The following features of SATA are excluded from SAS STP or handled differently in a SAS domain:

- a) extended differential voltages;
- b) enclosure services;
- c) staggered spin-up (see 5.21);
- d) SATA device activity indication;
- e) presence detect; and
- f) power management improvements.

#### **G.3 Affiliation policies**

##### **G.3.1 Affiliation policies overview**

SATA is based on a model that assumes a SATA device is controlled by a single SATA host and does not address the concept of multiple SATA hosts having the ability to access any given SATA device.

With STP SATA bridges, SATA devices are cast into an environment where multiple STP initiator ports, by sharing the SATA host port of the STP SATA bridge, have access to the same SATA device. The SATA protocol used inside STP connections does not account for the possibility that more than one STP initiator port is vying for access to the SATA device. Affiliations provide a way to ensure contention for a SATA device does not result in incoherent access to the SATA device when commands from different STP initiator ports collide at the SATA device.

To prevent a SATA device from confusing commands from one STP initiator port with commands from another STP initiator port, an STP initiator port requires a means to maintain exclusive access to a SATA device or STP target port for the duration of the processing of a command.

For example, consider the case where an STP initiator port establishes a connection to send a command (e.g., a read) and then closes the connection while the SATA device (e.g., a disk drive) retrieves the data (e.g., performs a seek operation to the track containing the data). If, after the connection is closed, another STP initiator port is allowed to establish a connection and send another command, then the SATA device no longer has a means to determine which STP initiator port should receive the data when the SATA device requests the connection to send the data for the first command. This is because, unlike SCSI target devices, SATA devices have no concept of multiple SATA hosts.

The consequences are worse for write commands since a possible result is wrong data written to media, with the original data being overwritten and permanently lost.

Affiliations provide a means for an STP initiator port to establish atomic access to a SATA device across the processing of a command or series of commands to the SATA device, without requiring the STP initiator port to maintain a connection open to the STP target port for the duration of command processing.

### **G.3.2 Affiliation policy for static STP initiator port to STP target port mapping**

Affiliations should not be used to enforce policies establishing fixed associations between STP initiator ports and STP target ports.

### **G.3.3 Affiliation policy with SATA queued commands and multiple STP initiator ports**

When sharing an affiliation context, STP initiator ports using queued commands when other STP initiator ports may be accessing the same STP target port should, at vendor specific intervals, allow commands to complete and release the affiliation to allow other STP initiator ports access to the STP target port.

### **G.3.4 Applicability of affiliation for STP target ports**

Affiliation may or may not be necessary for STP target ports depending on whether the STP target port tracks the STP initiator port's SAS address on each command received. If the STP target port has the means to manage and track commands from each STP initiator port independently, then affiliations are not necessary because the STP target port is capable of associating each information transfer with the appropriate STP initiator port and is capable of establishing a connection to the appropriate STP initiator port when sending information back for a command.

An STP target port capable of tracking commands may support a limited number of STP initiator ports (i.e., more than one, but less than one per command) and use multiple affiliations in order to manage that restriction.

An STP target port that behaves the same as a SATA device, in that it maintains only a single affiliation context to be shared among all STP initiator ports, provides a way for STP initiator ports to maintain exclusive access to the STP target port while commands remain outstanding. In this model, an STP target port is capable of establishing connections to an STP initiator port, but is only capable of remembering the SAS address of the last STP initiator port to establish a connection and therefore is only capable of requesting a connection back to that same STP initiator port.

See 9.4.3.12 for an explanation of how an STP target port reports support for affiliations.

## **G.4 SATA port selector considerations**

Not all the protocol elements for STP initiator ports to manage a SATA port selector (see SATA) in a SAS domain are defined in this standard. Additional coordination between STP initiator ports may be needed to avoid conflicting usage of the SATA port selector between STP initiator ports (e.g., between two SAS domains). Such additional coordination is outside the scope of this standard.

## G.5 SATA device not transmitting initial Register Device-to-Host FIS

Some SATA devices do not return the initial Register Device-to-Host FIS after a link reset sequence if they did not detect the COMINIT during the link reset sequence (e.g., if the SATA device originated the link reset sequence). While waiting for the initial Register Device-to-Host FIS, an STP SATA bridge responds as follows:

- a) in the SMP DISCOVER response (see 9.4.3.10):
  - A) the ATTACHED SAS DEVICE TYPE field is set to 000b;
  - B) the NEGOTIATED LOGICAL LINK RATE field and the NEGOTIATED PHYSICAL LINK RATE field are set to a value indicating the phy is enabled at a valid link rate;
  - C) the ATTACHED SATA DEVICE bit is set to one; and
  - D) the ATTACHED SAS ADDRESS field is set to the SAS address of the STP target port of the STP SATA bridge;
- and
- b) returns OPEN\_REJECT (NO DESTINATION) for connection requests to the SAS address of the STP target port.

If an STP initiator port detects this situation for a vendor specific amount of time, then a management application client should send an SMP PHY CONTROL function requesting a phy operation of LINK RESET or HARD RESET to originate a new link reset sequence. The SATA device is expected to detect the COMINIT during this link reset sequence and provide the initial Register Device-to-Host FIS.

## **Annex H**

(informative)

### **Minimum deletable primitive and scrambled idle segment insertion rate summary**

Table H.1 shows all the possible combinations of deletable primitive (see 6.2.5) insertion rates for physical link rate tolerance management (see 6.5) and rate matching (see 6.17) if the phy is in the SAS dword mode.

**Table H.1 – Minimum deletable primitive insertion rate examples while in the SAS dword mode**

Physical link rate	Connection rate	Deletable primitive insertion rate (per specified number of dwords)
12 Gbit/s	12 Gbit/s	8 per 1 024 (physical link rate tolerance management)
	6 Gbit/s	8 per 1 024 (physical link rate tolerance management) + 1 per 2 (rate matching)
	3 Gbit/s	8 per 1 024 (physical link rate tolerance management) + 3 per 4 (rate matching)
	1.5 Gbit/s	8 per 1 024 (physical link rate tolerance management) + 7 per 8 (rate matching)
6 Gbit/s	6 Gbit/s	4 per 512 (physical link rate tolerance management)
	3 Gbit/s	4 per 512 (physical link rate tolerance management) + 1 per 2 (rate matching)
	1.5 Gbit/s	4 per 512 (physical link rate tolerance management) + 3 per 4 (rate matching)
3 Gbit/s	3 Gbit/s	2 per 256 (physical link rate tolerance management)
	1.5 Gbit/s	2 per 256 (physical link rate tolerance management) + 1 per 2 (rate matching)
1.5 Gbit/s	1.5 Gbit/s	1 per 128 (physical link rate tolerance management)

While in SAS packet mode see table H.2 for all the possible combinations of:

- a) scrambled idle segment insertion rates for rate matching (see 6.17.3); and
- a) deletable primitive, deletable binary primitive, and extended deletable primitive insertion rates for physical link rate tolerance management (see 6.5).

Table H.2 – Minimum insertion rate examples while in the SAS packet mode

Physical link rate	Connection rate	Specified number of SPL packets	
		Physical link rate tolerance (deletable SPL packets <sup>a</sup> )	Rate matching (scrambled idle segment insertion rate)
22.5 Gbit/s	22.5 Gbit/s	4 per 512	none
	12 Gbit/s	4 per 512	+ 1 per 2
	6 Gbit/s	4 per 512	+ 3 per 4
	3 Gbit/s	4 per 512	+ 7 per 8
	1.5 Gbit/s	4 per 512	+ 15 per 16
<sup>a</sup> For physical link rate tolerance management (see 6.5.3) a deletable SPL packet is: a) a primitive segment that contains only deletable primitives and deletable binary primitives; or b) an extended deletable primitive.			



## Annex I

(informative)

### Zone permission configuration descriptor examples

This annex provides examples of using multiple zone permission configuration descriptors in the SMP CONFIGURE ZONE PERMISSION TABLE function (see 9.4.3.26) if the number of zone groups is 128.

Table I.1 shows an example initial value of the zone permission table.

**Table I.1 – Zone permission table example initial value**

Zone group	0 <sup>a</sup>	1 <sup>a</sup>	2 to 3	4 to 7 <sup>a</sup>	8	9	10	11	12 to 127
0 <sup>a</sup>	0	1	0	0	0	0	0	0	0
1 <sup>a</sup>	1	1	1	1	1	1	1	1	1
2 to 3	0	1	0	0	0	0	0	0	0
4 to 7 <sup>a</sup>	0	1	0	0	0	0	0	0	0
8	0	1	0	0	0	0	0	0	0
9	0	1	0	0	0	0	0	0	0
10	0	1	0	0	0	0	0	0	0
11	0	1	0	0	0	0	0	0	0
12 to 127	0	1	0	0	0	0	0	0	0
<sup>a</sup> Zone permission table entries for this zone group are not changeable.									

Table I.2 shows an example SMP CONFIGURE ZONE PERMISSION TABLE request where the STARTING ZONE GROUP field is set to 10 (i.e., 0Ah) and the zone permission configuration descriptor list contains two zone permission configuration descriptors.

**Table I.2 – CONFIGURE ZONE PERMISSION TABLE request example**

Byte\Bit	7	6	5	4	3	2	1	0
0	SMP FRAME TYPE (40h)							
1	FUNCTION (8Bh)							
2	Reserved							
3	REQUEST LENGTH (0Bh)							
4	(MSB)	EXPECTED EXPANDER CHANGE COUNT						(LSB)
5								
6	STARTING SOURCE ZONE GROUP (0Ah)							
7	NUMBER OF ZONE PERMISSION CONFIGURATION DESCRIPTORS (02h)							
Zone permission configuration descriptor (first) (source zone group 10)								
8	FFh							
...	(each byte set to FFh)							
23	0Eh							
Zone permission configuration descriptor (second) (source zone group 11)								
24	00h							
...	(each byte set to 00h)							
47	00h							
48	(MSB)	CRC						(LSB)
...								
51								

Table I.3 shows the zone permission table after processing the first zone permission configuration descriptor (i.e., source zone group 10).

**Table I.3 – Zone permission table after processing first zone permission configuration descriptor**

Zone group	0 <sup>a</sup>	1 <sup>a</sup>	2 to 3	4 to 7 <sup>a</sup>	8	9	10 <sup>b</sup>	11	12 to 127
0 <sup>a</sup>	0	1	0	0	0	0	0	0	0
1 <sup>a</sup>	1	1	1	1	1	1	1	1	1
2 to 3	0	1	0	0	0	0	1	0	0
4 to 7 <sup>a</sup>	0	1	0	0	0	0	0	0	0
8	0	1	0	0	0	0	1	0	0
9	0	1	0	0	0	0	1	0	0
10 <sup>b</sup>	0	1	1	0	1	1	1	1	1
11	0	1	0	0	0	0	1	0	0
12 to 127	0	1	0	0	0	0	1	0	0
<sup>a</sup> Zone permission table entries for this zone group are not changeable. <sup>b</sup> Changeable entries in this zone group are changed by the descriptor.									

Table I.4 shows the zone permission table after processing the second zone permission configuration descriptor (i.e., source zone group 11).

**Table I.4 – Zone permission table after processing second zone permission configuration descriptor**

Zone group	0 <sup>a</sup>	1 <sup>a</sup>	2 to 3	4 to 7 <sup>a</sup>	8	9	10	11 <sup>b</sup>	12 to 127
0 <sup>a</sup>	0	1	0	0	0	0	0	0	0
1 <sup>a</sup>	1	1	1	1	1	1	1	1	1
2 to 3	0	1	0	0	0	0	1	0	0
4 to 7 <sup>a</sup>	0	1	0	0	0	0	0	0	0
8	0	1	0	0	0	0	1	0	0
9	0	1	0	0	0	0	1	0	0
10	0	1	1	0	1	1	1	0	1
11 <sup>b</sup>	0	1	0	0	0	0	0	0	0
12 to 127	0	1	0	0	0	0	1	0	0
<sup>a</sup> Zone permission table entries for this zone group are not changeable. <sup>b</sup> Changeable entries in this zone group are changed by the descriptor.									

## **Annex J**

(informative)

### **SAS addressing**

#### **J.1 SAS addressing in SAS domains**

When a set of SAS phys transmit the same SAS address in the identification sequence but receive different SAS addresses, indicating they are attached to more than one SAS domain, they become part of separate SAS ports in separate SAS domains, and each SAS port shares the same SAS address. See figure 46 in 4.7.2 for an example of what happens if they are not in separate SAS domains.

The SAS addresses used by SAS ports in different SAS domains may be the same (e.g., when a set of phys transmit the same SAS address in the identification sequence but receive different SAS addresses, indicating they are attached to two separate SAS domains) so the SAS address serves as a port identifier (see 4.2.9) rather than a port name (see 4.2.8).

#### **J.2 Expander device SAS addresses**

When a set of expander phys transmit the same SAS address in the identification sequence but receive different SAS addresses, indicating they are attached to separate SAS ports or expander ports, they become part of separate expander ports in the same SAS domain.

The SMP port in an expander device has a port identifier that is the same as the device name of the expander device (see 4.5.2) and is used for addressing an expander device. Expander ports are not individually addressed.

## Annex K

(informative)

### Expander device handling of connections

#### K.1 Expander device handling of connections overview

This annex provides examples of how expander devices process connection requests.

Figure K.1 shows the topology used by examples in this annex.

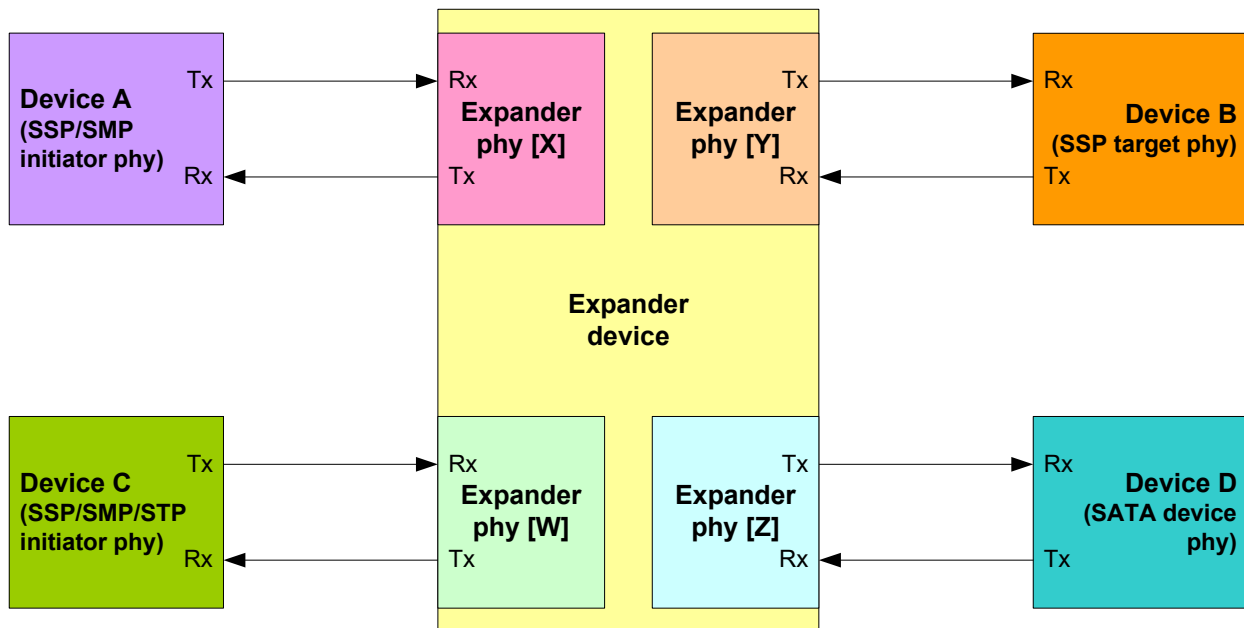


Figure K.1 – Example topology

Table K.1 defines the column headers used within the figures contained within this annex.

**Table K.1 – Column descriptions for connection examples**

Column header	Description
Phy [W] Rx	Expander phy [W] Receive from device C
Phy [W] Tx	Expander phy [W] Transmit to device C
Phy [W] XL state	Expander phy [W] XL state machine state (see 6.19)
Phy [W] XL req/rsp	Expander phy [W] XL requests and responses (see 4.5.6)
Phy [W] XL cnf/ind	Expander phy [W] XL confirmations and indications (see 4.5.6)
Phy [X] Rx	Expander phy [X] Receive from device A
Phy [X] Tx	Expander phy [X] Transmit to device A
Phy [X] XL state	Expander phy [X] XL state machine state (see 6.19)
Phy [X] XL req/rsp	Expander phy [X] XL requests and responses (see 4.5.6)
Phy [X] XL cnf/ind	Expander phy [X] XL confirmations and indications (see 4.5.6)
Phy [Y] XL cnf/ind	Expander phy [Y] XL confirmations and indications (see 4.5.6)
Phy [Y] XL req/rsp	Expander phy [Y] XL requests and responses (see 4.5.6)
Phy [Y] XL state	Expander phy [Y] XL state machine state (see 6.19)
Phy [Y] Tx	Expander phy [Y] Transmit to device B
Phy [Y] Rx	Expander phy [Y] Receive from device B
Phy [Z] XL cnf/ind	Expander phy [Z] XL confirmations and indications (see 4.5.6)
Phy [Z] XL req/rsp	Expander phy [Z] XL requests and responses (see 4.5.6)
Phy [Z] XL state	Expander phy [Z] XL state machine state (see 6.19)
Phy [Z] Tx	Expander phy [Z] Transmit to device D
Phy [Z] Rx	Expander phy [Z] Receive from device D

Figure K.2 shows the establishment of a successful connection between two end devices.

### Figure K.2 – Connection request - OPEN\_ACCEPT



### K.3 Connection request - OPEN\_REJECT by end device

Figure K.3 shows failure to establish a connection due to rejection of the connection request by an end device.

Expander phy [X]					Expander phy [Y]				
Rx	Tx	XL state	XL req/rsp	XL cnf/ind	XL cnf/ind	XL req/rsp	XL state	Tx	Rx
idle dwords	idle dwords	XL0:Idle					XL0:Idle	idle dwords	idle dwords
SOAF									
OPEN (A to B)									
EOAF									
idle dwords	AIP (NORMAL) and/or idle dwords	XL1: Request_Path	Request Path	Arbitrating (Normal)					
				Arb Won					
		XL2: Request_Open	Forward Open		Forward Open		XL5: Forward_Open	SOAF	
		XL3: Open_Cnf_Wait	Forward Dword (idle dwords)		Forward Dword (idle dwords)	Arb Status (Waiting On Device)	XL6: Open_Rsp_Wait	idle dwords (forwarded or generated)	
	AIP (WAITING ON DEVICE)			Arb Status (Waiting On Device)		Open Reject			OPEN_REJECT
	idle dwords								
	OPEN_REJECT								
	idle dwords	XL0:Idle					XL0:Idle	idle dwords	idle dwords

Figure K.3 – Connection request - OPEN\_REJECT by end device

K.4 Connection request - OPEN\_REJECT by expander device

Figure K.4 shows failure to establish a connection due to rejection of the connection request by an expander device.

Expander phy [X]					Expander phy [Y]				
Rx	Tx	XL state	XL req/rsp	XL cnf/ind	XL cnf/ind	XL req/rsp	XL state	Tx	Rx
idle dwords	idle dwords	XL0:Idle					XL0:Idle	idle dwords	idle dwords
SOAF									
OPEN (A to B)									
EOAF									
idle dwords	AIP (NORMAL) and/or idle dwords	XL1: Request_Path	Request Path	Arbitrating (Normal)					
	OPEN_REJECT	XL4:Open_Reject							
	idle dwords	XL0:Idle							

Figure K.4 – Connection request - OPEN\_REJECT by expander device

## K.5 Connection request - arbitration lost

Figure K.5 shows two end devices attempting to establish a connection at the same time. This example assumes that the OPEN (A to B) address frame has higher priority than the OPEN (B to A) address frame and therefore device A wins arbitration and device B loses arbitration.

Expander phy [X]					Expander phy [Y]				
Rx	Tx	XL state	XL req/rsp	XL cnf/ind	XL cnf/ind	XL req/rsp	XL state	Tx	Rx
idle dwords	idle dwords	XL0:Idle					XL0:Idle	idle dwords	idle dwords
SOAF								SOAF	SOAF
OPEN (A to B)								OPEN (B to A)	OPEN (B to A)
EOAF								EOAF	EOAF
idle dwords	AIP (NORMAL) and/or idle dwords	XL1: Request_Path	Request Path	Arbitrating (Normal)	Arbitrating (Normal)	Request Path	XL1: Request_Path	AIP (NORMAL) and/or idle dwords	idle dwords
				Arb Won	Arb Lost				
	AIP (WAITING ON DEVICE)	XL2: Request_Open	Forward Open		Forward Open		XL0:Idle	idle dwords	
							XL5: Forward_Open	SOAF	
					Forward Dword (idle dwords)			OPEN (A to B)	
								EOAF	
	idle dwords	XL3: Open_Cnf_Wait	Forward Dword (idle dwords)	Arb Status (Waiting On Device)		Arb Status (Waiting On Device)	XL6: Open_Rsp_Wait	idle dwords (forwarded or generated)	
						Open Accept			
						Forward Dword (connection dwords)			
connection dwords	connection dwords	XL7:Connected	Forward Dword (connection dwords)		Forward Dword (connection dwords)		XL7:Connected	idle dwords (forwarded)	connection dwords
								connection dwords	

Figure K.5 – Connection request - arbitration lost

## K.6 Connection request - backoff and retry

Figure K.6 shows a higher priority OPEN address frame (B to C) received by a phy that has previously forwarded an OPEN address frame (A to B) whose source (A) differs from the winning destination (C). In this case expander phy [X] is required to back off and retry path arbitration (see 6.19.8).

Expander phy [X]					Expander phy [Y]												
Rx	Tx	XL state	XL req/rsp	XL cnf/ind	XL cnf/ind	XL req/rsp	XL state	Tx	Rx								
idle dwords	idle dwords	XL0:Idle					XL0:Idle	idle dwords	idle dwords								
SOAF																	
OPEN (A to B)																	
EOAF																	
idle dwords	AIP (NORMAL) and/or idle dwords	XL1: Request_Path	Request Path	Arbitrating (Normal)													
				Arb Won													
		XL2: Request_Open	Forward Open		Forward Open		XL5: Forward_Open	SOAF	OPEN (A to B)								
										XL3: Open_Cnf_Wait	Forward Dword (idle dwords)		Forward Dword (idle dwords)		XL6: Open_Rsp_Wait	idle dwords (forwarded or generated)	EOAF
	AIP (WAITING ON DEVICE)	XL3: Open_Cnf_Wait	Forward Dword (idle dwords)	Arb Status (Waiting On Device)		Arb Status (Waiting On Device)		XL6: Open_Rsp_Wait	idle dwords (forwarded or generated)	SOAF							
idle dwords									OPEN (B to C)								
AIP (NORMAL) and/or idle dwords	XL1: Request_Path	Request Path		Arbitrating (Normal)		Request Path	XL1: Request_Path	AIP(NORMAL) and/or idle dwords									
				ArbWon													
				Forward Open						XL2: Request_Open							

Figure K.6 – Connection request - backoff and retry

## K.7 Connection request - backoff and reverse path

Figure K.7 shows a higher priority OPEN address frame (B to A) received by a phy that has previously forwarded an OPEN address frame (A to B) whose source (A) matches the winning destination (A). In this case expander phy [Y] forwards the higher priority OPEN to expander phy [X] (see 6.19.8).

Expander phy [X]					Expander phy [Y]						
Rx	Tx	XL state	XL req/rsp	XL cnf/ind	XL cnf/ind	XL req/rsp	XL state	Tx	Rx		
idle dwords	idle dwords	XL0:Idle					XL0:Idle	idle dwords	idle dwords		
SOAF											
OPEN (A to B)											
EOAF											
idle dwords	AIP (NORMAL) and/or idle dwords	XL1: Request_Path	Request Path	Arbitrating (Normal)							
				ArbWon							
			XL2: Request_Open	Forward Open							Forward Open
			Arb Status (Wait On Device)							Arb Status (Wait On Device)	
											Backoff Reverse Path
	SOAF	XL5: Forward_Open		Forward Open		Forward Open	XL2: Request_Open	AIP (NORMAL)			
	OPEN (B to A)										
	EOAF										
	idle dwords (forwarded or generated)		XL6: Open_Rsp_Wait		Arb Status (Waiting on Device)		Arb Status (Waiting on Device)		idle dwords (forwarded or generated)		

Figure K.7 – Connection request - backoff and reverse path

K.8 Connection close - single step

Figure K.8 shows an end device initiating the closing of a connection by transmitting a CLOSE primitive sequence, followed by another end device responding with a CLOSE primitive sequence at a later time.

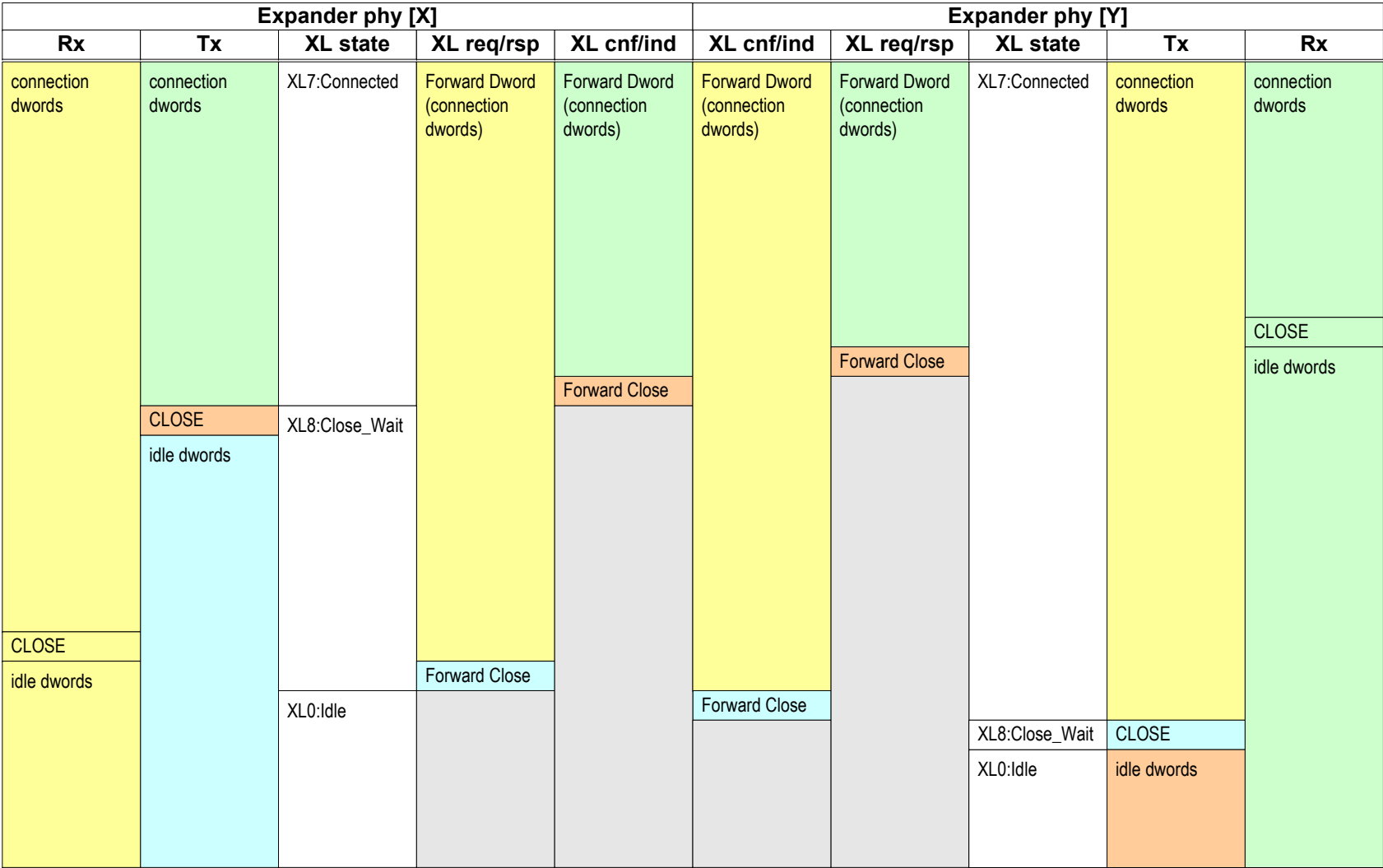


Figure K.8 – Connection close - single step

K.9 Connection close - simultaneous

Figure K.9 shows two end devices simultaneously transmitting a CLOSE primitive sequence to each other.

Expander phy [X]					Expander phy [Y]				
Rx	Tx	XL state	XL req/rsp	XL cnf/ind	XL cnf/ind	XL req/rsp	XL state	Tx	Rx
connection dwords	connection dwords	XL7:Connected	Forward Dword (connection dwords)	Forward Dword (connection dwords)	Forward Dword (connection dwords)	Forward Dword (connection dwords)	XL7:Connected	connection dwords	connection dwords
CLOSE									CLOSE
idle dwords			Forward Close	Forward Close	Forward Close	Forward Close			idle dwords
	CLOSE	XL8:Close_Wait	XL8:Close_Wait	CLOSE					
	idle dwords	XL0:Idle	XL0:Idle	idle dwords					

Figure K.9 – Connection close - simultaneous

K.10 BREAK handling during path arbitration when the BREAK\_REPLY method is disabled

Figure K.10 shows an expander device responding to the reception of a BREAK primitive sequence during path arbitration when the BREAK\_REPLY method of responding to BREAK primitive sequences is disabled (see 6.16.6).

Expander phy [X]					Expander phy [Y]				
Rx	Tx	XL state	XL req/rsp	XL cnf/ind	XL cnf/ind	XL req/rsp	XL state	Tx	Rx
idle dwords	idle dwords	XL0:Idle					XL0:Idle	idle dwords	idle dwords
SOAF									
OPEN(A to B)									
EOAF									
idle dwords	AIP (NORMAL) and/or idle dwords	XL1: Request_Path	Request Path	Arbitrating (Normal)					
BREAK									
idle dwords	BREAK	XL9:Break							
	idle dwords	XL0:Idle							

Figure K.10 – BREAK handling during path arbitration when the BREAK\_REPLY method is disabled



K.11 BREAK handling during connection when the BREAK\_REPLY method is disabled

Figure K.11 shows an expander device responding to the reception of a BREAK primitive sequence during a connection when the BREAK\_REPLY method of responding to BREAK primitive sequences is disabled (see 6.16.6).

Expander phy [X]					Expander phy [Y]					
Rx	Tx	XL state	XL req/rsp	XL cnf/ind	XL cnf/ind	XL req/rsp	XL state	Tx	Rx	
connection dwords	connection dwords	XL7:Connected	Forward Dword (connection dwords)	Forward Dword (connection dwords)	Forward Dword (connection dwords)	Forward Dword (connection dwords)	XL7:Connected	connection dwords	connection dwords	
				BREAK					idle dwords	
				Forward Break						
	BREAK	XL10: Break_Wait					XL9:Break	BREAK		
	idle dwords						XL0:Idle	idle dwords		
BREAK										
idle dwords										

Figure K.11 – BREAK handling during a connection when the BREAK\_REPLY method is disabled

K.12 BREAK handling during path arbitration when the BREAK\_REPLY method is enabled

Figure K.12 shows an expander device responding to the reception of a BREAK primitive sequence during path arbitration when the BREAK\_REPLY method of responding to BREAK primitive sequences is enabled (see 6.16.6).

Expander phy [X]					Expander phy [Y]				
Rx	Tx	XL state	XL req/rsp	XL cnf/ind	XL cnf/ind	XL req/rsp	XL state	Tx	Rx
idle dwords	idle dwords	XL0:Idle					XL0:Idle	idle dwords	idle dwords
SOAF									
OPEN(A to B)									
EOAF									
idle dwords	AIP (NORMAL) and/or idle dwords	XL1: Request_Path	Request Path	Arbitrating (Normal)					
BREAK									
idle dwords	BREAK_REPLY	XL9:Break							
	idle dwords	XL0:Idle							

Figure K.12 – BREAK handling during path arbitration when the BREAK\_REPLY method is enabled

K.13 BREAK handling during connection when BREAK\_REPLY method is enabled

Figure K.13 shows an expander device responding to the reception of a BREAK primitive sequence during a connection when the BREAK\_REPLY method of responding to BREAK primitive sequences is enabled (see 6.16.6).

Expander phy [X]					Expander phy [Y]				
Rx	Tx	XL state	XL req/rsp	XL cnf/ind	XL cnf/ind	XL req/rsp	XL state	Tx	Rx
connection dwords	connection dwords	XL7:Connected	Forward Dword (connection dwords)	Forward Dword (connection dwords)	Forward Dword (connection dwords)	Forward Dword (connection dwords)	XL7:Connected	connection dwords	connection dwords
									BREAK
				Forward Break		Forward Break			idle dwords
	BREAK	XL10: Break_Wait					XL9:Break	BREAK_REPLY	idle dwords
	idle dwords						XL0:Idle	idle dwords	
BREAK_REPLY									
idle dwords									

Figure K.13 – BREAK handling during a connection when the BREAK\_REPLY method is enabled

969 **K.14 STP connection - originated by STP initiator port**

Figure K.14 shows an STP initiator port originating a connection to an STP target port in an STP SATA bridge.

Expander phy [W] - STP target port in an STP SATA bridge					Expander phy [Z] - SATA host port in an STP SATA bridge				
Rx	Tx	XL state	XL req/rsp	XL cnf/ind	XL cnf/ind	XL req/rsp	XL state	Tx	Rx
idle dwords	idle dwords	XL0:Idle						SYNC/CONT	SATA device dwords
SOAF									
OPEN (C to D)									
EOAF									
idle dwords	AIP (NORMAL) and/or idle dwords	XL1: Request_Path	Request Path	Arbitrating (Normal)					
				Arb Won					
		XL2: Request_Open	Forward Open		Forward Open				
		XL3: Open_Cnf_Wait	Forward Dword (idle dwords)		Forward Dword (idle dwords)				
AIP (WAITING ON DEVICE)				Arb Status (Waiting On Device)					
idle dwords				Open Accept					
OPEN_ACCEPT				Forward Dword (SATA device dwords <sup>1</sup> )					
				Forward Dword (SATA device dwords)					
STP connection dwords	SATA device dwords <sup>1</sup>	XL7:Connected	Forward Dword (STP connection dwords)	Forward Dword (SATA device dwords <sup>1</sup> )	Forward Dword (STP connection dwords)			STP connection dwords	
	SATA device dwords			Forward Dword (SATA device dwords)					

<sup>1</sup> STP SATA bridge duplicates the dword stream that is being received from the SATA device before forwarding dwords - this ensures that a continued SATA primitive is correctly forwarded to the STP initiator port.

**Figure K.14 – STP connection - originated by STP initiator port**

## K.15 STP connection - originated by STP target port in an STP SATA bridge

Figure K.15 shows an STP target port in an STP SATA bridge originating a connection on behalf of a SATA device that is requesting to transmit a frame.

Expander phy [W] - STP target port in an STP SATA bridge					Expander phy [Z] - SATA host port in an STP SATA bridge				
Rx	Tx	XL state	XL req/rsp	XL cnf/ind	XL cnf/ind	XL req/rsp	XL state	Tx	Rx
idle dwords	idle dwords	XL0:Idle						SYNC/CONT	SYNC/CONT
					Arbitrating (Normal)	Request Path			X_RDY/CONT
					Arb Won				
						Forward Open			Forward Dword (idle dwords)
	SOAF	XL5: Forward_Open							
	OPEN (D to C)								
	EOAF								
	idle dwords (forwarded or generated)	XL6: Open_Rsp_Wait		Arb Status (Waiting On Device)	Arb Status (Waiting On Device)				
						OPEN_ACCEPT			
						STP connection dwords			
SATA device dwords <sup>a</sup>	XL7:Connected	Transmit Dword (STP connection dwords)	Forward Dword (SATA device dwords <sup>a</sup> )	Forward Dword (SATA device dwords)	Forward Dword (SATA device dwords <sup>a</sup> )	STP connection dwords	SATA device dwords		

<sup>a</sup> STP SATA bridge duplicates the dword stream that is being received from the SATA device before forwarding dwords. This ensures that a continued SATA primitive is correctly forwarded to the STP initiator port.

Figure K.15 – STP connection - originated by STP target port in an STP SATA bridge

K.16 STP connection close - originated by STP initiator port

Figure K.16 shows an STP initiator port closing a connection to an STP target port in an STP SATA bridge.

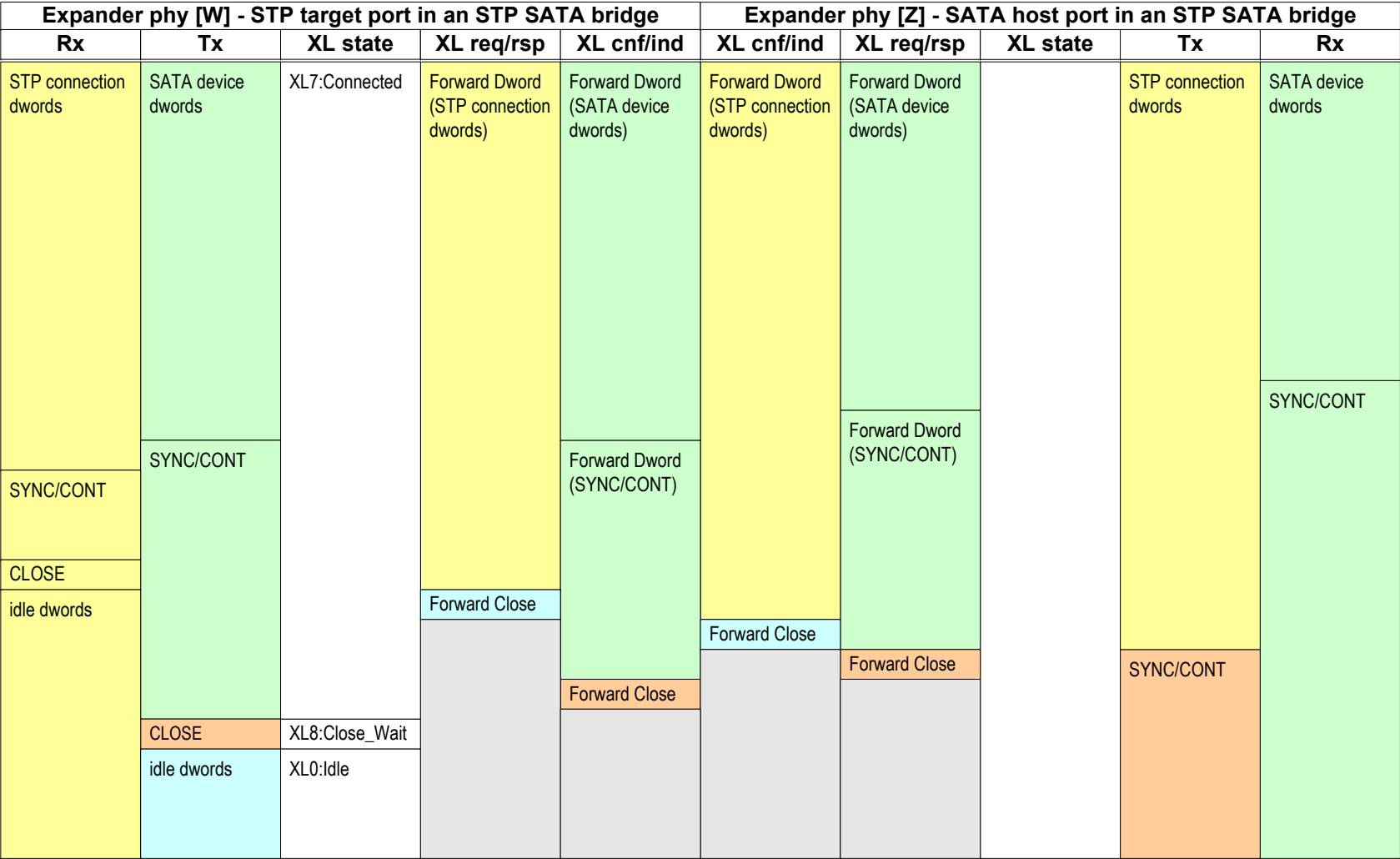


Figure K.16 – STP connection close - originated by STP initiator port

K.17 STP connection close - originated by STP target port in an STP SATA bridge

Figure K.17 shows an STP target port in an STP SATA bridge closing an STP connection.

Expander phy [W] - STP target port in an STP SATA bridge					Expander phy [Z] - SATA host port in an STP SATA bridge				
Rx	Tx	XL state	XL req/rsp	XL cnf/ind	XL cnf/ind	XL req/rsp	XL state	Tx	Rx
STP connection dwords	SATA device dwords	XL7:Connected	Forward Dword (STP connection dwords)	Forward Dword (SATA device dwords)	Forward Dword (STP connection dwords)	Forward Dword (SATA device dwords)		STP connection dwords	SATA device dwords
				Forward Dword (SYNC/CONT)		Forward Dword (SYNC/CONT)			SYNC/CONT
	SYNC/CONT		Forward Dword (SYNC/CONT)		Forward Dword (SYNC/CONT)	Forward Close		SYNC/CONT	
SYNC/CONT	CLOSE	XL8:Close_Wait		Forward Close					
	idle dwords	XL0:Idle							
	CLOSE								
idle dwords									

Figure K.17 – STP connection close - originated by STP target port in an STP SATA bridge

K.18 Connection request - XL1:Request\_Path to XL5:Forward\_Open transition

Figure K.18 shows the establishment of a connection following an XL1:Request\_Path to XL5:Forward\_Open transition by expander phy [Y].

Expander phy [X]					Expander phy [Y]				
Rx	Tx	XL state	XL req/rsp	XL cnf/ind	XL cnf/ind	XL req/rsp	XL state	Tx	Rx
idle dwords	idle dwords	XL0:Idle					XL0:Idle	Idle dwords	Idle dwords
SOAF									SOAF
OPEN (A to B)									OPEN (B to A)
EOAF									EOAF
idle dwords	AIP (NORMAL) and/or idle dwords	XL1: Request_Path	Request Path	Arbitrating (Normal)		Request Path	XL1: Request_Path		idle dwords
				Arb Won					
		XL2: Request_Open	Forward Open		Forward Open		XL5: Forward_Open	SOAF	
	AIP (WAITING ON DEVICE)	XL3: Open_Confirm_Wait	Forward Dword (idle dwords)	Arb Status (Waiting on Device)	Forward Dword (idle dwords)	Arb Status (Waiting on Device)	XL6: Open_Response_Wait	Idle dwords (forwarded or generated)	
	idle dwords		Forward Dword (connection dwords)	Open Accept		Forward Dword (connection dwords)	XL7:Connected	Idle dwords (forwarded)	connection dwords
	OPEN_ACCEPT								
connection dwords	connection dwords	XL7:Connected	Forward Dword (connection dwords)		Forward Dword (connection dwords)			connection dwords	

Figure K.18 – XL1:Request\_Path to XL5:Forward\_Open transition



## K.19 Pathway blocked and pathway recovery example

Figure K.19 shows a topology used to illustrate pathway recovery. Exp[1] and Exp[2] are expander devices. A, B, and C are end devices. A attempts to open a connection to B while B attempts to open a connection to A.

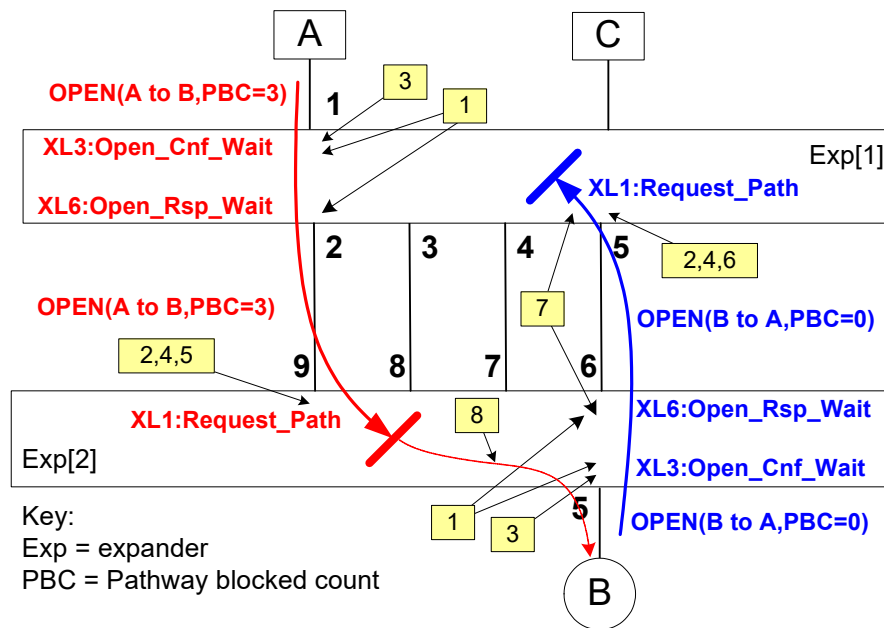


Figure K.19 – Partial pathway recovery

The sequence of events used to identify pathway blockage and to perform pathway recovery are as follows:

- 1) Exp[1].Phy[1,2] and Exp[2].Phy[5,6] each send Phy Status (Partial Pathway) responses to the ECM to indicate that they contain partial pathways;
- 2) Exp[1].Phy[5] and Exp[2].Phy[9] each receive Arbitrating (Waiting On Partial) confirmations from the ECM, which cause them to transmit an AIP (WAITING ON PARTIAL) primitive sequence;
- 3) AIP (WAITING ON PARTIAL) is received by Exp[1].Phy[2] and Exp[2].Phy[6] then forwarded to Exp[1].Phy[1] and Exp[2].Phy[5] as an Arb Status (Waiting On Partial) confirmation. Exp[1].Phy[1] and Exp[2].Phy[5] each send Phy Status (Blocked On Partial) responses to ECM as confirmation that they are blocked waiting on a partial pathway in another expander device;
- 4) Exp[1].Phy[5] and Exp[2].Phy[9] each receive Arbitrating (Blocked On Partial) confirmations from the ECM while all destination phys send Phy Status (Blocked On Partial) responses, which cause them to run their Partial Pathway Timeout timers;
- 5) the Partial Pathway Timeout timer expires in Exp[2].Phy[9]. This causes a request to the ECM to resolve pathway blockage. The pathway recovery priority for this phy is not lower than all phys within the destination port that are also blocked (i.e., the Request Path from Exp[2].Phy[9] has higher priority than the Request Path from Exp[2].Phy[5], which is receiving Arbitrating (Blocked On Partial)). The ECM does not provide an Arb Reject (Pathway Blocked) confirmation to Exp[2].Phy[9], so this expander phy waits for pathway resolution to occur elsewhere in the topology;
- 6) the Partial Pathway Timeout timer expires in Exp[1].Phy[5]. This causes a request to the ECM to resolve pathway blockage. The pathway recovery priority for this expander phy is lower than all expander phys within the destination port that are also blocked (i.e., the Request Path from Exp[1].Phy[5] is lower priority than the Request Path from Exp[1].Phy[1], which is receiving Arbitrating (Blocked On Partial)). The ECM provides an Arb Reject (Pathway Blocked) confirmation to Exp[1].Phy[5], which instructs this expander phy to reject the connection request using OPEN\_REJECT (PATHWAY BLOCKED);

NOTE 91 - The Partial Pathway Timeout timer in Exp[1].Phy[5] may expire before, after, or at the same time the Partial Pathway Timeout timer expires in Exp[2].Phy[9].

- 7) OPEN\_REJECT (PATHWAY BLOCKED) tears down partial pathway all the way to the originating end device (Device B);
- 8) Exp[2].Phy[9] receives Arb Won and the partial pathway is extended through Exp[2].Phy[5]; and
- 9) OPEN (A to B) is delivered to device B.

## Annex L

(informative)

### Primitive encoding, binary primitive coding, and extended binary primitive coding

#### L.1 Primitive encoding

Table L.1 describes a set of the K28.5-based primitive encodings whose 40-bit values (after 8b10b encoding with either starting running disparity) have a Hamming distance (i.e., the number of bits different in two patterns) of at least 7. All the primitive encodings in 6.2 except for TRAIN and TRAIN\_DONE were selected from this list. Unassigned encodings may be used by future versions of this standard.

**Table L.1 – Primitives with Hamming distance of at least 7 (part 1 of 3)**

Character				Assignment
1 <sup>st</sup>	2 <sup>nd</sup>	3 <sup>rd</sup>	4 <sup>th</sup>	
K28.5	D01.3	D01.3	D01.3	ALIGN (2)
K28.5	D01.4	D01.4	D01.4	ACK
K28.5	D01.4	D02.0	D31.4	RRDY (RESERVED 0)
K28.5	D01.4	D04.7	D24.0	NAK (RESERVED 1)
K28.5	D01.4	D07.3	D30.0	CREDIT_BLOCKED
K28.5	D01.4	D16.7	D07.3	NAK (RESERVED 2)
K28.5	D01.4	D24.0	D16.7	RRDY (NORMAL)
K28.5	D01.4	D27.4	D04.7	NAK (CRC ERROR)
K28.5	D01.4	D30.0	D02.0	RRDY (CLOSE)
K28.5	D01.4	D31.4	D29.7	NAK (RESERVED 0)
K28.5	D02.0	D01.4	D29.7	ERROR
K28.5	D02.0	D02.0	D02.0	HARD_RESET
K28.5	D02.0	D04.7	D01.4	CLOSE (RESERVED 1)
K28.5	D02.0	D07.3	D04.7	CLOSE (CLEAR AFFILIATION)
K28.5	D02.0	D16.7	D31.4	MUX (LOGICAL LINK 0)
K28.5	D02.0	D24.0	D07.3	BREAK
K28.5	D02.0	D29.7	D16.7	BREAK_REPLY
K28.5	D02.0	D30.0	D27.4	CLOSE (NORMAL)
K28.5	D02.0	D31.4	D30.0	CLOSE (RESERVED 0)
K28.5	D04.7	D01.4	D24.0	BROADCAST (EXPANDER)
K28.5	D04.7	D02.0	D01.4	BROADCAST (CHANGE)
K28.5	D04.7	D04.7	D04.7	BROADCAST (ASYNCHRONOUS EVENT)
K28.5	D04.7	D07.3	D29.7	BROADCAST (SES)
K28.5	D04.7	D16.7	D02.0	BROADCAST (RESERVED 3)
K28.5	D04.7	D24.0	D31.4	BROADCAST (RESERVED CHANGE 0)
K28.5	D04.7	D27.4	D07.3	BROADCAST (RESERVED CHANGE 1)
K28.5	D04.7	D29.7	D30.0	BROADCAST (RESERVED 4)

Table L.1 – Primitives with Hamming distance of at least 7 (part 2 of 3)

Character				Assignment
1 <sup>st</sup>	2 <sup>nd</sup>	3 <sup>rd</sup>	4 <sup>th</sup>	
K28.5	D04.7	D31.4	D27.4	MUX (LOGICAL LINK 1)
K28.5	D07.0	D03.4	D13.4	OOB_IDLE
K28.5	D07.0	D07.0	D07.0	ALIGN (1)
K28.5	D07.3	D01.4	D31.4	
K28.5	D07.3	D02.0	D04.7	PS_REQ (PARTIAL)
K28.5	D07.3	D04.7	D30.0	
K28.5	D07.3	D07.3	D07.3	PWR_ACK
K28.5	D07.3	D24.0	D29.7	PWR_DONE
K28.5	D07.3	D27.4	D16.7	PWR_GRANT
K28.5	D07.3	D29.7	D27.4	PWR_REQ
K28.5	D07.3	D30.0	D24.0	
K28.5	D07.3	D31.4	D02.0	
K28.5	D10.2	D10.2	D27.3	ALIGN (0)
K28.5	D16.7	D01.4	D02.0	
K28.5	D16.7	D02.0	D07.3	
K28.5	D16.7	D04.7	D31.4	
K28.5	D16.7	D16.7	D16.7	OPEN_ACCEPT
K28.5	D16.7	D24.0	D27.4	
K28.5	D16.7	D27.4	D30.0	PS_ACK
K28.5	D16.7	D29.7	D24.0	
K28.5	D16.7	D30.0	D04.7	
K28.5	D16.7	D31.4	D01.4	
K28.5	D24.0	D01.4	D16.7	EXTEND_CONNECTION (CLOSE)
K28.5	D24.0	D02.0	D29.7	
K28.5	D24.0	D04.7	D07.3	SOF
K28.5	D24.0	D07.3	D31.4	EOAF
K28.5	D24.0	D16.7	D27.4	EOF
K28.5	D24.0	D24.0	D24.0	
K28.5	D24.0	D27.4	D02.0	PS_NAK
K28.5	D24.0	D29.7	D04.7	EXTEND_CONNECTION (NORMAL)
K28.5	D24.0	D30.0	D01.4	SOAF
K28.5	D27.3	D27.3	D27.3	ALIGN (3)
K28.5	D27.4	D01.4	D07.3	AIP (RESERVED WAITING ON PARTIAL)
K28.5	D27.4	D04.7	D02.0	
K28.5	D27.4	D07.3	D24.0	AIP (WAITING ON CONNECTION)
K28.5	D27.4	D16.7	D30.0	AIP (RESERVED 1)
K28.5	D27.4	D24.0	D04.7	AIP (WAITING ON PARTIAL)
K28.5	D27.4	D27.4	D27.4	AIP (NORMAL)
K28.5	D27.4	D29.7	D01.4	AIP (RESERVED 2)

Table L.1 – Primitives with Hamming distance of at least 7 (part 3 of 3)

Character				Assignment
1 <sup>st</sup>	2 <sup>nd</sup>	3 <sup>rd</sup>	4 <sup>th</sup>	
K28.5	D27.4	D30.0	D29.7	AIP (WAITING ON DEVICE)
K28.5	D27.4	D31.4	D16.7	AIP (RESERVED 0)
K28.5	D29.7	D02.0	D30.0	OPEN_REJECT (RESERVED CONTINUE 0)
K28.5	D29.7	D04.7	D27.4	OPEN_REJECT (RESERVED STOP 1)
K28.5	D29.7	D07.3	D16.7	OPEN_REJECT (RESERVED INITIALIZE 1)
K28.5	D29.7	D16.7	D04.7	OPEN_REJECT (PATHWAY BLOCKED)
K28.5	D29.7	D24.0	D01.4	OPEN_REJECT (RESERVED CONTINUE 1)
K28.5	D29.7	D27.4	D24.0	OPEN_REJECT (RETRY)
K28.5	D29.7	D29.7	D29.7	OPEN_REJECT (NO DESTINATION)
K28.5	D29.7	D30.0	D31.4	OPEN_REJECT (RESERVED INITIALIZE 0)
K28.5	D29.7	D31.4	D07.3	OPEN_REJECT (RESERVED STOP 0)
K28.5	D30.0	D01.4	D04.7	DONE (ACK/NAK TIMEOUT)
K28.5	D30.0	D02.0	D16.7	
K28.5	D30.0	D07.3	D27.4	DONE (CREDIT TIMEOUT)
K28.5	D30.0	D16.7	D01.4	DONE (RESERVED 0)
K28.5	D30.0	D24.0	D02.0	PS_REQ (SLUMBER)
K28.5	D30.0	D27.4	D29.7	DONE (RESERVED TIMEOUT 0)
K28.5	D30.0	D29.7	D31.4	DONE (CLOSE)
K28.5	D30.0	D30.0	D30.0	DONE (NORMAL)
K28.5	D30.0	D31.4	D24.0	DONE (RESERVED TIMEOUT 1)
K28.5	D31.3	D01.3	D07.0	NOTIFY (RESERVED 1)
K28.5	D31.3	D07.0	D01.3	NOTIFY (POWER LOSS EXPECTED)
K28.5	D31.3	D31.3	D31.3	NOTIFY (ENABLE SPINUP)
K28.5	D31.4	D01.4	D30.0	OPEN_REJECT (RESERVED ABANDON 3)
K28.5	D31.4	D02.0	D27.4	OPEN_REJECT (ZONE VIOLATION)
K28.5	D31.4	D04.7	D29.7	OPEN_REJECT (CONNECTION RATE NOT SUPPORTED)
K28.5	D31.4	D07.3	D02.0	OPEN_REJECT (RESERVED ABANDON 2)
K28.5	D31.4	D16.7	D24.0	OPEN_REJECT (WRONG DESTINATION)
K28.5	D31.4	D27.4	D01.4	OPEN_REJECT (STP RESOURCES BUSY)
K28.5	D31.4	D29.7	D07.3	OPEN_REJECT (PROTOCOL NOT SUPPORTED)
K28.5	D31.4	D30.0	D16.7	OPEN_REJECT (RESERVED ABANDON 1)
K28.5	D31.4	D31.4	D31.4	OPEN_REJECT (BAD DESTINATION)

Table L.2 describes the K28.5-based primitive encodings that do not have Hamming distances of 7 from the other primitives.

**Table L.2 – Primitives without Hamming distance of 7**

Character				Assignment
1 <sup>st</sup>	2 <sup>nd</sup>	3 <sup>rd</sup>	4 <sup>th</sup>	
K28.5	D30.3	D30.3	D30.3	TRAIN
K28.5	D30.3	D30.3	D10.2	TRAIN_DONE

## L.2 Binary primitive coding

### L.2.1 Binary primitive codes overview

Binary primitives that are D.C. balanced and have a minimum Hamming distance (i.e., the number of bits different in two patterns) of at least seven are listed in L.2.

### L.2.2 Deletable binary primitives

Table L.3 describes the set of the deletable binary primitives. Unassigned binary primitives shown in table L.3 may be used in future versions of this standard as deletable binary primitives.

**Table L.3 – Deletable binary primitives** (part 1 of 3)

Byte				Assignment
0 (first) <sup>a</sup>	1	2	3 (last)	
01h	01h	FDh	FDh	
01h	3Dh	0Dh	FDh	
01h	3Dh	F1h	3Dh	
01h	3Dh	FDh	C1h	
01h	CDh	31h	FDh	
01h	CDh	CDh	3Dh	
01h	D5h	5Dh	CDh	
01h	D5h	ADh	F1h	
01h	D9h	F5h	55h	
01h	D9h	F9h	A9h	
01h	E5h	F5h	69h	
01h	E5h	F9h	95h	
01h	E9h	5Dh	F1h	
01h	E9h	ADh	CDh	
01h	F1h	3Dh	3Dh	APTA_COEFFICIENT_1 (DECREMENT)
01h	F1h	C1h	FDh	APTA_COEFFICIENT_1 (INCREMENT)
01h	FDh	65h	99h	APTA_COEFFICIENT_1 (MAXIMUM)
01h	FDh	69h	65h	APTA_COEFFICIENT_1 (MINIMUM)
01h	FDh	95h	A5h	APTA_COEFFICIENT_1 (UPDATED)
<sup>a</sup> The PRIMITIVE SYNCHRONIZE SELECT field, CONTROL1 field, CONTROL2 field, or CONTROL3 field (see table 58) is set to 01b.				

Table L.3 – Deletable binary primitives (part 2 of 3)

Byte				Assignment
0 (first) <sup>a</sup>	1	2	3 (last)	
01h	FDh	99h	59h	APTA_COEFFICIENT_1 (RESERVED 1)
05h	4Dh	6Dh	D5h	
05h	4Dh	9Dh	E9h	
05h	55h	79h	79h	
05h	55h	B5h	9Dh	
05h	69h	75h	ADh	
05h	69h	B9h	75h	
05h	7Dh	C5h	4Dh	
05h	7Dh	C9h	B1h	
05h	8Dh	F5h	B1h	
05h	8Dh	F9h	4Dh	
05h	99h	3Dh	D9h	
05h	99h	CDh	E5h	
05h	A5h	3Dh	E5h	
05h	A5h	CDh	D9h	
05h	BDh	51h	D5h	
05h	BDh	5Dh	29h	
05h	BDh	A1h	E9h	
05h	BDh	ADh	15h	
09h	55h	F1h	E5h	
09h	59h	CDh	D9h	
09h	65h	3Dh	D9h	
09h	7Dh	5Dh	15h	
09h	7Dh	ADh	29h	
09h	8Dh	6Dh	E9h	
09h	8Dh	9Dh	D5h	
09h	95h	FDh	19h	
09h	99h	55h	BDh	
09h	99h	A9h	7Dh	
09h	A5h	59h	7Dh	
09h	A5h	A5h	BDh	
09h	A9h	F1h	D9h	
09h	A9h	FDh	25h	
09h	B1h	6Dh	D5h	APTA_COEFFICIENT_2 (DECREMENT)
09h	B1h	9Dh	E9h	APTA_COEFFICIENT_2 (INCREMENT)
09h	BDh	35h	4Dh	APTA_COEFFICIENT_2 (MAXIMUM)
09h	BDh	39h	B1h	APTA_COEFFICIENT_2 (MINIMUM)
09h	BDh	C5h	71h	APTA_COEFFICIENT_2 (UPDATED)
09h	BDh	C9h	8Dh	APTA_COEFFICIENT_2 (RESERVED 1)
0Dh	0Dh	3Dh	3Dh	
0Dh	0Dh	C1h	FDh	
0Dh	19h	F5h	69h	

<sup>a</sup> The PRIMITIVE SYNCHRONIZE SELECT field, CONTROL1 field, CONTROL2 field, or CONTROL3 field (see table 58) is set to 01b.

Table L.3 – Deletable binary primitives (part 3 of 3)

Byte				Assignment
0 (first) <sup>a</sup>	1	2	3 (last)	
0Dh	19h	F9h	95h	
0Dh	25h	F5h	55h	
0Dh	25h	F9h	A9h	
0Dh	31h	31h	FDh	
0Dh	31h	CDh	3Dh	
0Dh	C1h	0Dh	FDh	APTA_COEFFICIENT_3 (DECREMENT)
0Dh	C1h	F1h	3Dh	APTA_COEFFICIENT_3 (INCREMENT)
0Dh	C1h	FDh	C1h	APTA_COEFFICIENT_3 (MAXIMUM)
0Dh	CDh	55h	59h	APTA_COEFFICIENT_3 (MINIMUM)
0Dh	CDh	59h	A5h	APTA_COEFFICIENT_3 (UPDATED)
0Dh	CDh	A5h	65h	APTA_COEFFICIENT_3 (RESERVED 1)
0Dh	CDh	A9h	99h	
0Dh	D5h	C5h	A9h	APTA_COEFFICIENT_1_2 (DECREMENT)
0Dh	D5h	C9h	55h	APTA_COEFFICIENT_1_2 (INCREMENT)
0Dh	D9h	61h	CDh	APTA_COEFFICIENT_1_2 (MAXIMUM)
0Dh	D9h	6Dh	31h	APTA_COEFFICIENT_1_2 (MINIMUM)
0Dh	D9h	91h	F1h	APTA_COEFFICIENT_1_2 (UPDATED)
0Dh	D9h	9Dh	0Dh	APTA_COEFFICIENT_1_2 (RESERVED 1)
0Dh	E5h	61h	F1h	APTA_COEFFICIENT_2_3 (DECREMENT)
0Dh	E5h	6Dh	0Dh	APTA_COEFFICIENT_2_3 (INCREMENT)
0Dh	E5h	91h	CDh	APTA_COEFFICIENT_2_3 (MAXIMUM)
0Dh	E5h	9Dh	31h	APTA_COEFFICIENT_2_3 (MINIMUM)
0Dh	E9h	C5h	95h	APTA_COEFFICIENT_2_3 (UPDATED)
0Dh	E9h	C9h	69h	APTA_COEFFICIENT_2_3 (RESERVED 1)
0Dh	F1h	55h	65h	APTA_ADJUST (COMPLETE)
0Dh	F1h	59h	99h	APTA_ADJUST (READY)
0Dh	F1h	A5h	59h	APTA_ADJUST (START)
0Dh	F1h	A9h	A5h	APTA_ADJUST (TERMINATE)
0Dh	FDh	01h	3Dh	APTA_ADJUST (RESERVED 1)
0Dh	FDh	0Dh	C1h	APTA_ADJUST (RESERVED 2)
0Dh	FDh	F1h	01h	APTA_ADJUST (RESERVED 3)
<sup>a</sup> The PRIMITIVE SYNCHRONIZE SELECT field, CONTROL1 field, CONTROL2 field, or CONTROL3 field (see table 58) is set to 01b.				



**L.2.3 Binary primitives for use outside SAS logical link connections**

Table L.4 describes the set of the binary primitives used outside SAS logical link connections. Unassigned binary primitives shown in table L.4 may be used in future versions of this standard as binary primitives that are limited to use outside SAS logical link connections.

**Table L.4 – Binary primitives used outside SAS logical link connections (part 1 of 2)**

Byte				Assignment
0 (first) <sup>a</sup>	1	2	3 (last)	
51h	0Dh	B9h	D9h	
51h	1Dh	CDh	A9h	
51h	25h	71h	F5h	
51h	55h	E9h	1Dh	
51h	79h	8Dh	35h	
51h	99h	59h	6Dh	
51h	9Dh	35h	95h	
51h	C9h	75h	C9h	
51h	CDh	E1h	A5h	
51h	E5h	05h	F9h	
51h	F5h	7Dh	01h	
55h	09h	ADh	6Dh	
55h	0Dh	D5h	1Dh	
55h	21h	B5h	B9h	
55h	35h	D9h	49h	
55h	49h	C5h	F1h	
55h	61h	E1h	CDh	
55h	6Dh	61h	39h	
55h	8Dh	09h	F5h	
55h	91h	EDh	91h	
55h	B1h	29h	5Dh	
55h	C1h	99h	79h	
55h	C5h	35h	2Dh	
55h	C9h	79h	15h	
55h	D5h	85h	C5h	
55h	DDh	0Dh	19h	
55h	EDh	A9h	41h	
59h	41h	59h	BDh	
59h	4Dh	B5h	31h	
59h	51h	EDh	61h	
59h	5Dh	49h	C5h	
59h	69h	D1h	55h	
59h	95h	31h	E9h	
59h	A5h	D5h	A1h	
59h	ADh	89h	39h	
59h	B1h	55h	59h	
<sup>a</sup> The PRIMITIVE SYNCHRONIZE SELECT field, CONTROL-1 field, CONTROL-2 field, or CONTROL-3 field (see table 58) is set to 01b.				

**Table L.4 – Binary primitives used outside SAS logical link connections** (part 2 of 2)

Byte				Assignment
0 (first) <sup>a</sup>	1	2	3 (last)	
59h	B9h	2Dh	89h	
59h	F1h	61h	35h	
59h	F1h	89h	D1h	
5Dh	35h	C1h	95h	
5Dh	39h	15h	2Dh	
5Dh	59h	31h	59h	
5Dh	61h	1Dh	E1h	
5Dh	81h	65h	E5h	
5Dh	85h	B9h	85h	
5Dh	9Dh	51h	31h	
5Dh	ADh	41h	C9h	
<sup>a</sup> The PRIMITIVE SYNCHRONIZE SELECT field, CONTROL-1 field, CONTROL-2 field, or CONTROL-3 field (see table 58) is set to 01b.				

**L.2.4 Binary primitives for use inside SAS logical link connections**

Table L.5 describes the set of the binary primitives used inside SAS logical link connections. Unassigned binary primitives shown in table L.5 may be used in future versions of this standard as binary primitives that are limited to use inside SAS logical link connections.

**Table L.5 – Binary primitives used inside SAS logical link connections** (part 1 of 3)

Byte				Use	Assignment
0 (first) <sup>a</sup>	1	2	3 (last)		
31h	19h	71h	DDh	SSP	B_EOF (0)
31h	1Dh	CDh	55h	SSP	B_EOF (0) (RESERVED 1)
31h	25h	A9h	7Dh	SSP	
31h	45h	D5h	D9h	SSP	B_EOF (1)
31h	49h	A5h	BDh	SSP	B_EOF (1) (RESERVED 1)
31h	4Dh	79h	2Dh	SSP	B_EOF (1) (RESERVED 2)
31h	75h	4Dh	39h	SSP	
31h	8Dh	55h	E5h	SSP	
31h	9Dh	B5h	29h	SSP	
31h	A5h	79h	C9h	SSP	
31h	C5h	3Dh	55h	SSP	
31h	C9h	E9h	D1h	SSP	
31h	D1h	8Dh	6Dh	SSP	
31h	E1h	B1h	E5h	SSP	B_EOF (2)
31h	EDh	2Dh	A1h	SSP	B_EOF (2) (RESERVED 1)
Key:					
SSP = SAS logical links, inside SSP connections					
SMP = SAS logical links, inside SMP connections					
<sup>a</sup> The PRIMITIVE SYNCHRONIZE SELECT field, CONTROL-1 field, CONTROL-2 field, or CONTROL-3 field (see table 58) is set to 01b.					

Table L.5 – Binary primitives used inside SAS logical link connections (part 2 of 3)

Byte				Use	Assignment
0 (first) <sup>a</sup>	1	2	3 (last)		
31h	EDh	D1h	31h	SSP	B_EOF (2) (RESERVED 2)
31h	F9h	19h	95h	SSP	
31h	F9h	21h	79h	SSP	
35h	09h	5Dh	B9h	SSP	
35h	29h	EDh	85h	SSP	
35h	35h	85h	ADh	SSP	
35h	51h	49h	F5h	SSP	B_EOF (3)
35h	5Dh	25h	C9h	SSP	B_EOF (3) (RESERVED 1)
35h	5Dh	B9h	11h	SSP	B_EOF (3) (RESERVED 2)
35h	6Dh	5Dh	41h	SSP	
35h	75h	E1h	61h	SSP	
35h	91h	D1h	E9h	SSP	
35h	A1h	D9h	1Dh	SSP	
35h	B9h	71h	25h	SSP	
35h	CDh	81h	5Dh	SSP	
35h	D1h	75h	19h	SSP	
35h	E9h	95h	89h	SSP	
39h	15h	E1h	B9h	SMP	
39h	21h	75h	3Dh	SMP	
39h	4Dh	09h	F9h	SMP	
39h	59h	D5h	25h	SMP	
39h	65h	9Dh	0Dh	SMP	
39h	81h	CDh	B5h	SMP	
39h	91h	B5h	D1h	SMP	
39h	ADh	E1h	45h	SMP	
39h	B5h	01h	DDh	SMP	
39h	B9h	49h	E1h	SMP	
39h	D5h	D9h	81h	SMP	
Unassigned use group					
3Dh	01h	EDh	59h		
Key:					
SSP = SAS logical links, inside SSP connections					
SMP = SAS logical links, inside SMP connections					
<sup>a</sup> The PRIMITIVE SYNCHRONIZE SELECT field, CONTROL-1 field, CONTROL-2 field, or CONTROL-3 field (see table 58) is set to 01b.					

**Table L.5 – Binary primitives used inside SAS logical link connections** (part 3 of 3)

Byte				Use	Assignment
0 (first) <sup>a</sup>	1	2	3 (last)		
3Dh	15h	55h	4Dh		
3Dh	2Dh	49h	35h		
3Dh	35h	2Dh	91h		
3Dh	39h	A1h	1Dh		
3Dh	41h	39h	CDh		
3Dh	45h	75h	A1h		
3Dh	89h	3Dh	61h		
3Dh	8Dh	0Dh	8Dh		
3Dh	D5h	29h	29h		
3Dh	E1h	41h	ADh		
Key:					
SSP = SAS logical links, inside SSP connections					
SMP = SAS logical links, inside SMP connections					
<sup>a</sup> The PRIMITIVE SYNCHRONIZE SELECT field, CONTROL-1 field, CONTROL-2 field, or CONTROL-3 field (see table 58) is set to 01b.					

**L.2.5 Binary primitives for use inside and outside SAS logical link connections**

Table L.6 describes the set of the binary primitives used inside SAS logical link connections and outside SAS logical link connections. Unassigned binary primitives shown in table L.6 may be used in future versions of this standard as binary primitives that are limited to use inside SAS logical link connections and outside SAS logical link connections.

**Table L.6 – Binary primitives used inside and outside SAS logical link connections** (part 1 of 2)

Byte				Assignment
0 (first) <sup>a</sup>	1	2	3 (last)	
A1h	15h	DDh	69h	
A1h	19h	6Dh	F1h	
A1h	29h	9Dh	D9h	
A1h	3Dh	3Dh	25h	
A1h	49h	D1h	EDh	
A1h	5Dh	29h	5Dh	
A1h	85h	EDh	C5h	
A1h	91h	35h	EDh	
A1h	A1h	EDh	39h	
A1h	B5h	75h	51h	
A1h	B9h	E1h	95h	
A1h	DDh	B1h	C1h	
A1h	E9h	5Dh	0Dh	
A1h	F5h	89h	35h	
A5h	2Dh	45h	E9h	
<sup>a</sup> The PRIMITIVE SYNCHRONIZE SELECT field, CONTROL-1 field, CONTROL-2 field, or CONTROL-3 field (see table 58) is set to 01b.				

Table L.6 – Binary primitives used inside and outside SAS logical link connections (part 2 of 2)

Byte				Assignment
0 (first) <sup>a</sup>	1	2	3 (last)	
A5h	45h	65h	3Dh	
A5h	51h	E1h	D9h	
A5h	61h	ADh	E1h	
A5h	65h	D5h	85h	
A5h	8Dh	99h	39h	
A5h	ADh	29h	D1h	
A5h	B1h	89h	CDh	
A5h	C5h	15h	F1h	
A5h	C9h	E9h	25h	
A5h	DDh	C5h	11h	
A5h	F1h	B1h	29h	
A9h	59h	35h	99h	
A9h	6Dh	89h	C5h	
A9h	89h	49h	DDh	
A9h	C5h	8Dh	59h	
A9h	D9h	45h	69h	
ADh	1Dh	79h	09h	
ADh	31h	59h	A5h	
ADh	69h	D9h	11h	
ADh	71h	1Dh	49h	
ADh	85h	E1h	69h	
ADh	99h	D1h	45h	
ADh	A5h	15h	1Dh	
<sup>a</sup> The PRIMITIVE SYNCHRONIZE SELECT field, CONTROL-1 field, CONTROL-2 field, or CONTROL-3 field (see table 58) is set to 01b.				

## L.2.6 Unassigned binary primitives

Table L.7 describes a set of the unassigned binary primitives. Binary primitives shown in table L.7 may be assigned in future versions of this standard.

**Table L.7 – Unassigned binary primitives** (part 1 of 4)

Byte				Assignment
0 (first) <sup>a</sup>	1	2	3 (last)	
Unassigned group 6				
61h	15h	3Dh	B9h	
61h	19h	99h	F5h	
61h	25h	DDh	35h	
61h	29h	7Dh	69h	
61h	31h	A5h	DDh	
61h	45h	E9h	E9h	
61h	5Dh	45h	B5h	
61h	69h	C9h	9Dh	
61h	6Dh	15h	CDh	
61h	6Dh	E5h	51h	
61h	75h	59h	D1h	
61h	89h	BDh	1Dh	
61h	9Dh	C1h	D9h	
61h	B5h	51h	ADh	
61h	DDh	1Dh	61h	
61h	F5h	8Dh	89h	
65h	11h	DDh	8Dh	
65h	2Dh	A1h	B5h	
65h	41h	F5h	65h	
65h	5Dh	D1h	29h	
65h	69h	3Dh	91h	
65h	81h	79h	F1h	
65h	85h	4Dh	6Dh	
65h	85h	B5h	C9h	
65h	99h	65h	A9h	
65h	A9h	8Dh	71h	
65h	BDh	31h	19h	
65h	F1h	95h	15h	
65h	F9h	01h	E5h	
69h	39h	C1h	6Dh	
69h	45h	ADh	95h	
69h	61h	91h	F9h	
69h	95h	E5h	25h	
69h	A1h	29h	EDh	
69h	ADh	4Dh	91h	
69h	B9h	B9h	41h	

<sup>a</sup> The PRIMITIVE SYNCHRONIZE SELECT field, CONTROL-1 field, CONTROL-2 field, or CONTROL-3 field (see table 58) is set to 01b.

Table L.7 – Unassigned binary primitives (part 2 of 4)

Byte				Assignment
0 (first) <sup>a</sup>	1	2	3 (last)	
69h	C1h	55h	D5h	
69h	C1h	DDh	29h	
69h	D1h	69h	59h	
69h	E9h	71h	A1h	
6Dh	19h	5Dh	51h	
6Dh	59h	8Dh	A1h	
6Dh	69h	69h	45h	
6Dh	91h	19h	3Dh	
6Dh	B1h	C5h	C1h	
6Dh	E5h	C1h	19h	
Unassigned group 9				
91h	4Dh	1Dh	9Dh	
91h	51h	B1h	7Dh	
91h	6Dh	25h	6Dh	
91h	75h	B9h	A1h	
91h	99h	EDh	49h	
91h	C1h	75h	B5h	
91h	F5h	51h	1Dh	
91h	F9h	45h	C5h	
95h	05h	59h	EDh	
95h	05h	A5h	F5h	
95h	19h	89h	F9h	
95h	29h	D1h	B5h	
95h	35h	E5h	19h	
95h	4Dh	F1h	89h	
95h	59h	4Dh	2Dh	
95h	61h	79h	D1h	
95h	79h	B5h	05h	
95h	89h	71h	79h	
95h	9Dh	1Dh	45h	
95h	9Dh	69h	A1h	
95h	A1h	4Dh	75h	
95h	C1h	ADh	1Dh	
95h	F1h	85h	B1h	
99h	09h	B9h	ADh	
99h	45h	61h	DDh	
99h	4Dh	EDh	05h	
99h	5Dh	85h	E1h	
99h	61h	C5h	79h	
99h	71h	09h	EDh	
99h	A5h	3Dh	29h	
99h	A9h	39h	55h	
<sup>a</sup> The PRIMITIVE SYNCHRONIZE SELECT field, CONTROL-1 field, CONTROL-2 field, or CONTROL-3 field (see table 58) is set to 01b.				

Table L.7 – Unassigned binary primitives (part 3 of 4)

Byte				Assignment
0 (first) <sup>a</sup>	1	2	3 (last)	
99h	B1h	A5h	65h	
99h	B9h	D1h	29h	
99h	C1h	D9h	4Dh	
99h	D9h	81h	9Dh	
9Dh	2Dh	8Dh	49h	
9Dh	39h	E1h	C1h	
9Dh	55h	11h	B9h	
9Dh	81h	D5h	99h	
9Dh	C5h	B1h	51h	
9Dh	C9h	25h	A9h	
Unassigned group C				
C1h	19h	F5h	A5h	
C1h	25h	6Dh	ADh	
C1h	31h	D9h	B9h	
C1h	4Dh	59h	75h	
C1h	51h	BDh	C9h	
C1h	7Dh	15h	39h	
C1h	A5h	9Dh	4Dh	
C1h	A9h	25h	F5h	
C1h	ADh	C9h	E1h	
C1h	DDh	81h	6Dh	
C1h	EDh	39h	89h	
C5h	1Dh	E1h	55h	
C5h	35h	91h	E5h	
C5h	45h	89h	DDh	
C5h	45h	FDh	11h	
C5h	55h	4Dh	E1h	
C5h	79h	E9h	09h	
C5h	91h	C5h	79h	
C5h	A1h	71h	9Dh	
C5h	A9h	F5h	41h	
C5h	B5h	1Dh	91h	
C5h	CDh	45h	8Dh	
C9h	21h	E9h	5Dh	
C9h	49h	E9h	B1h	
C9h	65h	D1h	2Dh	
C9h	75h	E5h	81h	
C9h	81h	BDh	71h	
C9h	95h	41h	F5h	
C9h	99h	CDh	15h	
<sup>a</sup> The PRIMITIVE SYNCHRONIZE SELECT field, CONTROL-1 field, CONTROL-2 field, or CONTROL-3 field (see table 58) is set to 01b.				



Table L.7 – Unassigned binary primitives (part 4 of 4)

Byte				Assignment
0 (first) <sup>a</sup>	1	2	3 (last)	
C9h	B5h	B1h	15h	
C9h	C5h	29h	3Dh	
C9h	DDh	25h	51h	
C9h	F9h	19h	25h	
CDh	09h	15h	DDh	
CDh	0Dh	71h	E1h	
CDh	3Dh	09h	A9h	
CDh	51h	A5h	2Dh	
CDh	E1h	81h	75h	
CDh	F1h	31h	C1h	
<sup>a</sup> The PRIMITIVE SYNCHRONIZE SELECT field, CONTROL-1 field, CONTROL-2 field, or CONTROL-3 field (see table 58) is set to 01b.				

### L.3 Extended binary primitive coding

Table L.8 describes a set of 128-bit codes for extended binary primitives. Unassigned codes may be used by future versions of this standard.

Table L.8 – Extended binary primitives (part 1 of 3)

Dword	Byte				Assignment
	0 (first) <sup>a</sup>	1	2	3 (last)	
0	02h	02h	FEh	FEh	
1	02h	3Eh	0Eh	FEh	
2	02h	3Eh	F2h	3Eh	
3	02h	3Eh	FEh	C2h	
0	02h	CEh	32h	FEh	
1	02h	CEh	CEh	3Eh	
2	02h	D6h	5Eh	CEh	
3	02h	D6h	AEh	F2h	
0	02h	DAh	F6h	56h	
1	02h	DAh	FAh	AAh	
2	02h	E6h	F6h	6Ah	
3	02h	E6h	FAh	96h	
0	02h	EAh	5Eh	F2h	
1	02h	EAh	AEh	CEh	
2	02h	F2h	3Eh	3Eh	
3	02h	F2h	C2h	FEh	
<sup>a</sup> The PRIMITIVE SYNCHRONIZE SELECT field, CONTROL-1 field, CONTROL-2 field, or CONTROL-3 field (see table 58) is set to 10b.					

Table L.8 – Extended binary primitives (part 2 of 3)

Dword	Byte				Assignment
	0 (first) <sup>a</sup>	1	2	3 (last)	
0	02h	FEh	66h	9Ah	
1	02h	FEh	6Ah	66h	
2	02h	FEh	96h	A6h	
3	02h	FEh	9Ah	5Ah	
0	06h	4Eh	6Eh	D6h	
1	06h	4Eh	9Eh	EAh	
2	06h	56h	7Ah	7Ah	
3	06h	56h	B6h	9Eh	
0	06h	6Ah	76h	AEh	
1	06h	6Ah	BAh	76h	
2	06h	7Eh	C6h	4Eh	
3	06h	7Eh	CAh	B2h	
0	06h	8Eh	F6h	B2h	
1	06h	8Eh	FAh	4Eh	
2	06h	9Ah	3Eh	DAh	
3	06h	9Ah	CEh	E6h	
0	06h	A6h	3Eh	E6h	
1	06h	A6h	CEh	DAh	
2	06h	BEh	52h	D6h	
3	06h	BEh	5Eh	2Ah	
0	06h	BEh	A2h	EAh	
1	06h	BEh	AEh	16h	
2	0Ah	56h	F2h	E6h	
3	0Ah	5Ah	CEh	DAh	
0	0Ah	66h	3Eh	DAh	
1	0Ah	7Eh	5Eh	16h	
2	0Ah	7Eh	AEh	2Ah	
3	0Ah	8Eh	6Eh	EAh	
0	0Ah	8Eh	9Eh	D6h	
1	0Ah	96h	FEh	1Ah	
2	0Ah	9Ah	56h	BEh	
3	0Ah	9Ah	AAh	7Eh	
0	0Ah	A6h	5Ah	7Eh	
1	0Ah	A6h	A6h	BEh	
2	0Ah	AAh	F2h	DAh	
3	0Ah	AAh	FEh	26h	
0	0Ah	B2h	6Eh	D6h	
1	0Ah	B2h	9Eh	EAh	
2	0Ah	BEh	36h	4Eh	
3	0Ah	BEh	3Ah	B2h	
<sup>a</sup> The PRIMITIVE SYNCHRONIZE SELECT field, CONTROL-1 field, CONTROL-2 field, or CONTROL-3 field (see table 58) is set to 10b.					

Table L.8 – Extended binary primitives (part 3 of 3)

Dword	Byte				Assignment
	0 (first) <sup>a</sup>	1	2	3 (last)	
0	0Ah	BEh	C6h	72h	
1	0Eh	32h	CEh	3Eh	
2	0Eh	C2h	0Eh	FEh	
3	0Eh	C2h	F2h	3Eh	
0	0Eh	C2h	FEh	C2h	
1	0Eh	CEh	56h	5Ah	
2	0Eh	CEh	5Ah	A6h	
3	0Eh	CEh	A6h	66h	
0	0Eh	CEh	AAh	9Ah	
1	0Eh	D6h	C6h	AAh	
2	0Eh	D6h	CAh	56h	
3	0Eh	DAh	62h	CEh	
0	0Eh	DAh	6Eh	32h	
1	0Eh	DAh	92h	F2h	
2	0Eh	DAh	9Eh	0Eh	
3	0Eh	E6h	62h	F2h	
0	0Eh	E6h	6Eh	0Eh	
1	0Eh	E6h	92h	CEh	
2	0Eh	E6h	9Eh	32h	
3	0Eh	EAh	C6h	96h	
0	0Eh	EAh	CAh	6Ah	
1	0Eh	F2h	56h	66h	
2	0Eh	F2h	5Ah	9Ah	
3	0Eh	F2h	A6h	5Ah	
0	0Eh	F2h	AAh	A6h	END_TRAIN
1	0Eh	FEh	F2h	02h	
2	0Eh	FEh	0Eh	C2h	
3	0Eh	FEh	02h	3Eh	
0	A6h	FAh	03h	C1h	PACKET_SYNC_LOST
1	50h	3Ch	D1h	5Ch	
2	F9h	C2h	91h	6Ch	
3	EDh	8Dh	A5h	3Bh	
0	AEh	7Ch	E1h	48h	LINK_RATE_MANAGEMENT
1	B6h	F6h	C6h	D2h	
2	9Dh	A9h	FEh	40h	
3	30h	14h	4Fh	34h	
0	E2h	89h	E6h	CAh	PACKET_SYNC
1	17h	8Eh	64h	6Bh	
2	6Fh	2Ch	DDh	99H	
3	EAh	0Fh	04h	43h	
<sup>a</sup> The PRIMITIVE SYNCHRONIZE SELECT field, CONTROL-1 field, CONTROL-2 field, or CONTROL-3 field (see table 58) is set to 10b.					

## Annex M

(informative)

### Standards bodies contact information

Table M.1 shows standards bodies and their web sites.

**Table M.1 – Standards bodies**

Abbreviation	Standards body	Web site
ANSI® <sup>a</sup>	American National Standards Institute	<a href="http://www.ansi.org">http://www.ansi.org</a>
DIN	German Institute for Standardization	<a href="http://www.din.de">http://www.din.de</a>
IEC® <sup>b</sup>	International Electrotechnical Commission	<a href="http://www.iec.ch">http://www.iec.ch</a>
IEEE® <sup>c</sup>	Institute of Electrical and Electronics Engineers	<a href="http://www.ieee.org">http://www.ieee.org</a>
INCITS	International Committee for Information Technology Standards	<a href="http://www.incits.org">http://www.incits.org</a>
ISO® <sup>d</sup>	International Organization for Standardization	<a href="http://www.iso.ch">http://www.iso.ch</a>
ITIC	Information Technology Industry Council	<a href="http://www.itic.org">http://www.itic.org</a>
JIS	Japanese Industrial Standards Committee	<a href="http://www.jisc.co.jp">http://www.jisc.co.jp</a>
T10	INCITS T10 SCSI storage interfaces	<a href="http://www.t10.org">http://www.t10.org</a>
T11	INCITS T11 Fibre Channel interfaces	<a href="http://www.t11.org">http://www.t11.org</a>
T13	INCITS T13 ATA storage interface	<a href="http://www.t13.org">http://www.t13.org</a>
<sup>a</sup> ANSI is a registered trademark of the American National Standards Institute. <sup>b</sup> IEC is a registered trademark of the International Electrotechnical Commission. <sup>c</sup> IEEE is a registered trademark of the Institute of Electrical Electronics Engineers, Inc. <sup>d</sup> ISO is a registered trademark of the International Organization for Standardization.		

## **Annex N**

(informative)

### **Successful low phy power condition handshake sequence**

This annex contains an example of the sequencing required between attached phys to successfully enter into a partial phy power condition (see 4.10.1.3). Although this annex only contains an example:

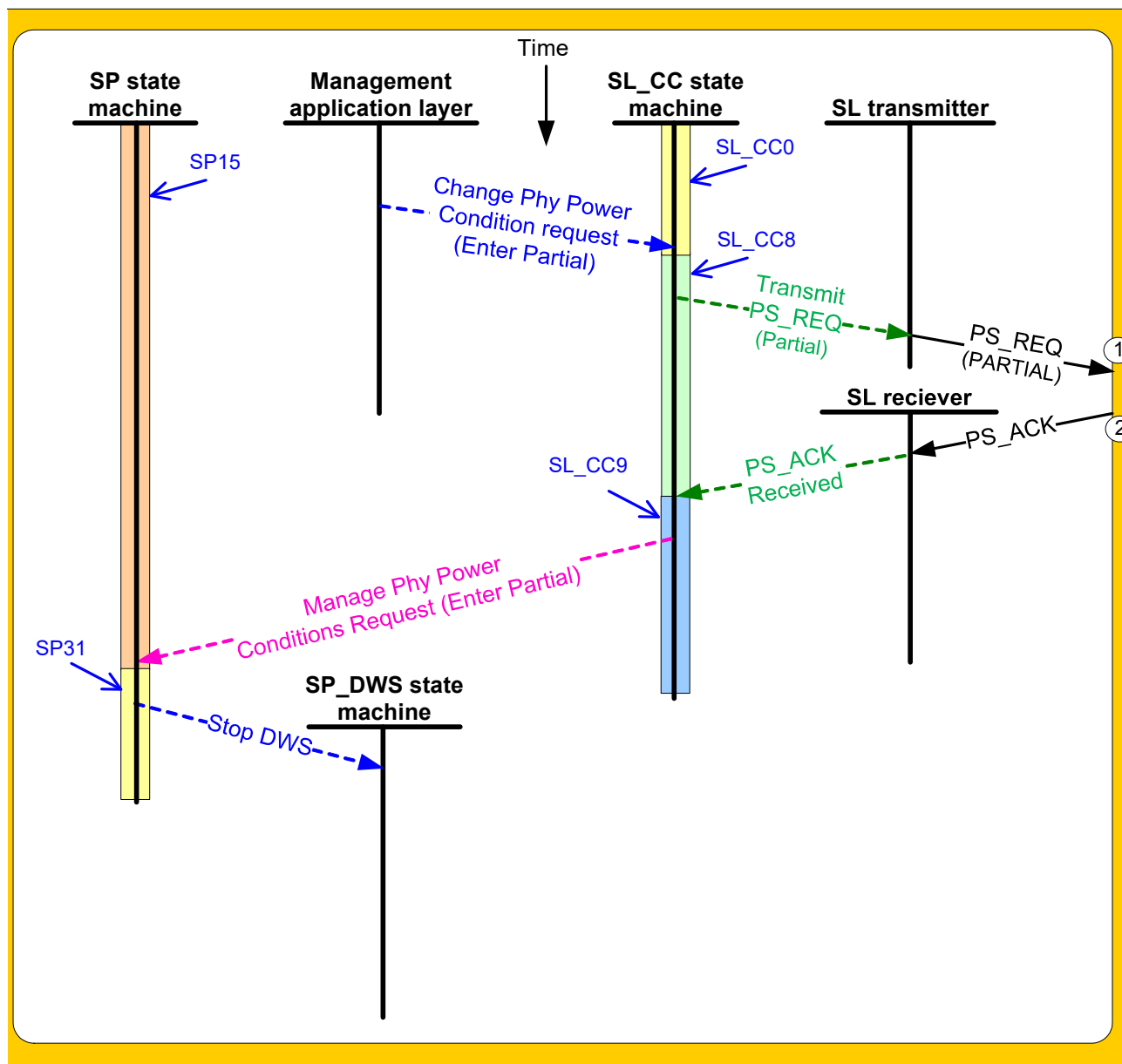
- a) of entering into a partial phy power condition a similar sequence is used to enter into a slumber phy power condition (see 4.10.1.4); and
- b) of phys in SAS dword mode, phys in SAS packet mode have the same sequencing except that:
  - A) the Stop PS message is substituted for the Stop DWS message; and
  - B) the SP\_PS state machine is substituted for the SP\_DWS state machine.

This example assumes both phys:

- a) are in SAS dword mode;
- b) have the partial phy power condition enabled (see 4.10.1.5 and 4.10.1.6);
- c) have received IDENTIFY address frame that has the PARTIAL CAPABLE bit set to one (see 6.10.2);
- d) are in the active phy power condition (see 4.10.1.2);
- e) have multiplexing disabled (see 5.13.2); and
- f) have optical mode disabled.

Figure N.1 shows an example of a requesting SAS device's sequencing of a successful request to enter a partial phy power condition.

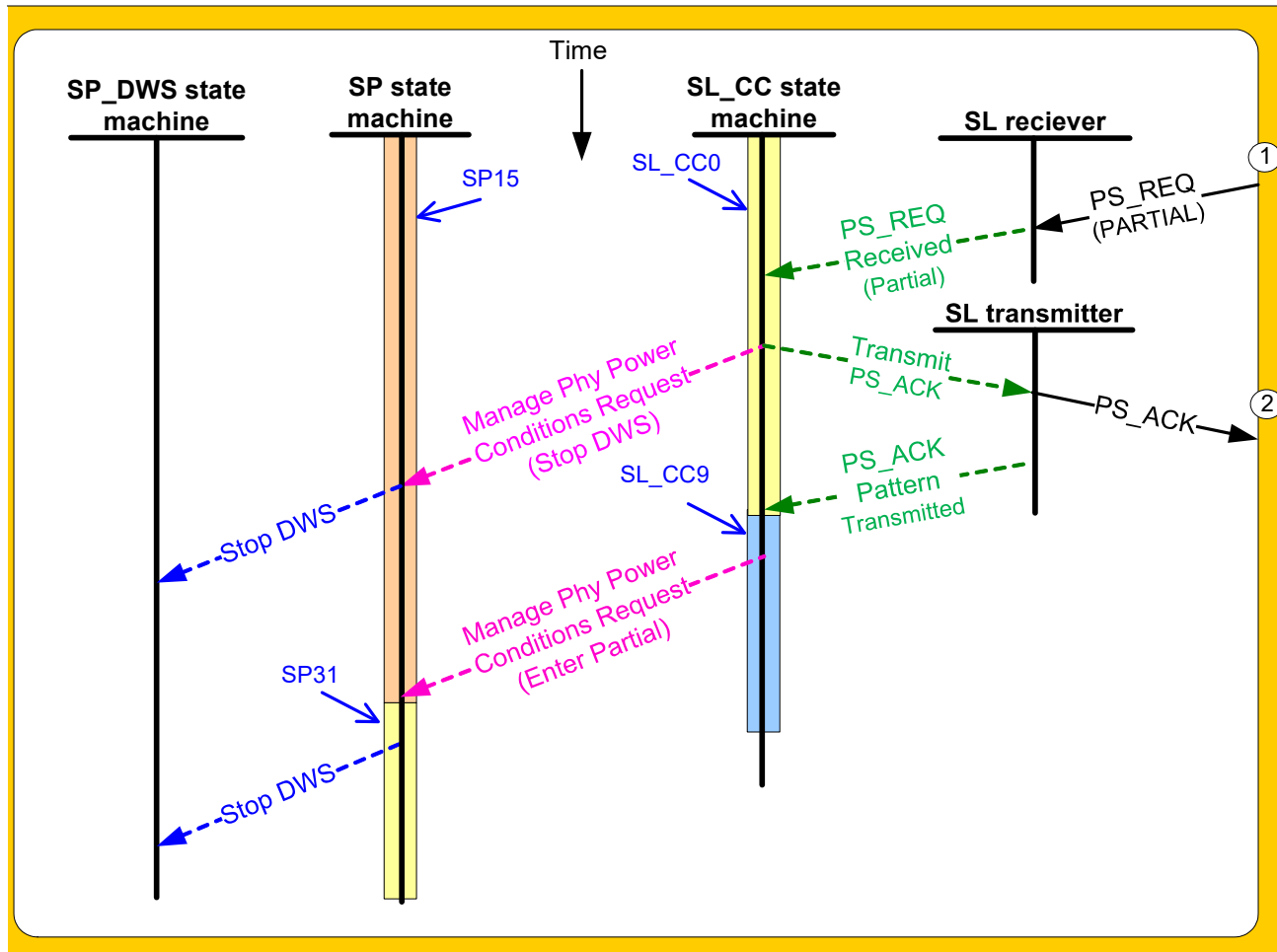
Figure N.2 shows an example of a SAS device's sequencing of a successful request to enter a partial phy power condition.



Key:

Processing SP state name or SL state name →

**Figure N.1 – Example of a requesting SAS device's sequencing of a successful request for entering a partial phy power condition**



Key:

Processing SP state name or SL state name →

NOTE - A Stop DWS message is sent from the SP state machine to the SP\_DWS state machine twice. The first prevents false errors as the link goes idle while the second provides consistency in the behavior of SP31:SAS\_PS\_Low\_Phy\_Power (see 5.14.5.2) in both the SAS device requesting the low phy power condition and the SAS device receiving the low phy power condition request.

**Figure N.2 – Example of a SAS device's sequencing for a successful request to enter a partial phy power condition**

## **Annex O**

(informative)

### **Terminology mapping to SPL-3**

The removal of peripheral device type from this standard resulted in changes in terminology (see table O.1) and field names (see table O.2) between this standard and SPL-3.

**Table O.1 – Terminology name mapping to SPL-3**

<b>Term used in this standard</b>	<b>Term used in SPL-3</b>
SAS device type	device type
attached SAS device type	attached device type
device type	peripheral device type

**Table O.2 – Field name mapping to SPL-3**

<b>Field name used in this standard</b>	<b>Field name used in SPL-3</b>
SAS DEVICE TYPE	DEVICE TYPE
ATTACHED SAS DEVICE TYPE	ATTACHED DEVICE TYPE



## Bibliography

ISO/IEC 14776-150, *Serial Attached SCSI (SAS)*

INCITS 457-2010, *Serial Attached SCSI-2 (SAS-2)*

ISO/IEC 9899:2011, *Programming languages -C*

ISO 80000-2, *Quantities and units - Part 2: Mathematical signs and symbols to be used in the natural sciences and technology*

OMG Unified Modeling Language (UML) Specification. Version 1.5, March 2003<sup>1</sup>

---

1. For more information on the UML specification, contact the Object Modeling Group (see <http://www.omg.org>).