To: T10 Technical Committee
From: Rob Elliott, HP (elliott@hp.com)
Date: 24 February 2005
Subject: 05-086r0 SAS-1.1 Link layer timeout race conditions

Revision history
Revision 0 (24 February 2005) First revision

Related documents
05-040r0 SAS-1.1 Break_Wait handling (Tim Hoglund, LSI Logic)(incorporated in sas1r08)
sas1r08 - Serial Attached SCSI 1.1 revision 8

Overview
05-040 patched a race condition involving OPENs, OPEN_REJECTs, and BREAKs that can happen between any pair of SL/SL, SL/XL, or XL/XL state machines.

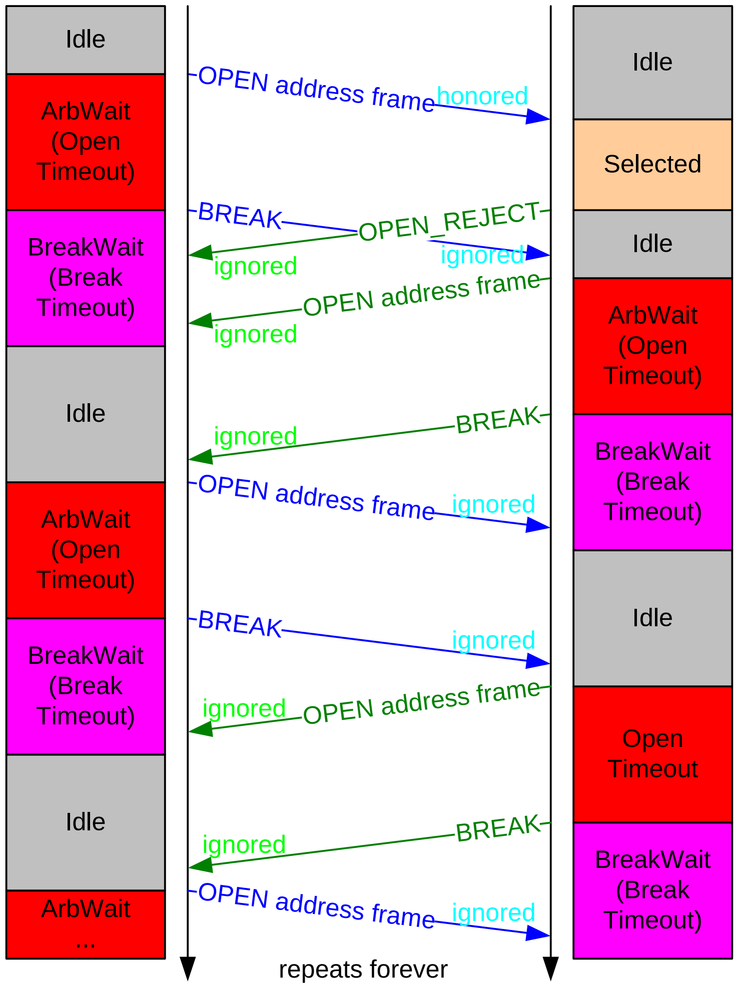


Figure 1 — OPEN_REJECT racing BREAK

In the scenario, phy A sends an OPEN then sends BREAK (perhaps after an Open Timeout, or for any other reason) just as phy B sends OPEN_REJECT. When this happens, phy A ignores the OPEN_REJECT because its link layer state machine is sitting in the SL_CC or XL Break_Wait state (which is just looking for an incoming BREAK). Phy B ignores the BREAK because it is in the idle state. If phy B then sends an OPEN, it is ignored by phy A which is still in the Break_Wait state. Phy A experiences a BREAK Timeout and returns to idle, but then Phy B experiences an Open Timeout and sends BREAK; phy A ignores the BREAK. The two phys ping-pong OPENs and BREAKs forever.

The suggested solution was to make the Break_Wait state proceed to the Idle state upon receiving OPEN_REJECT (in addition to receiving BREAK).

However, Mark Evans (Maxtor) noted there are other reasons the state machine might be in Break_Wait and experience similar problems. When a closing a connection, if the incoming CLOSE arrives just after the Close Timeout timer expires triggering an outgoing BREAK, the phys enter the same loop.

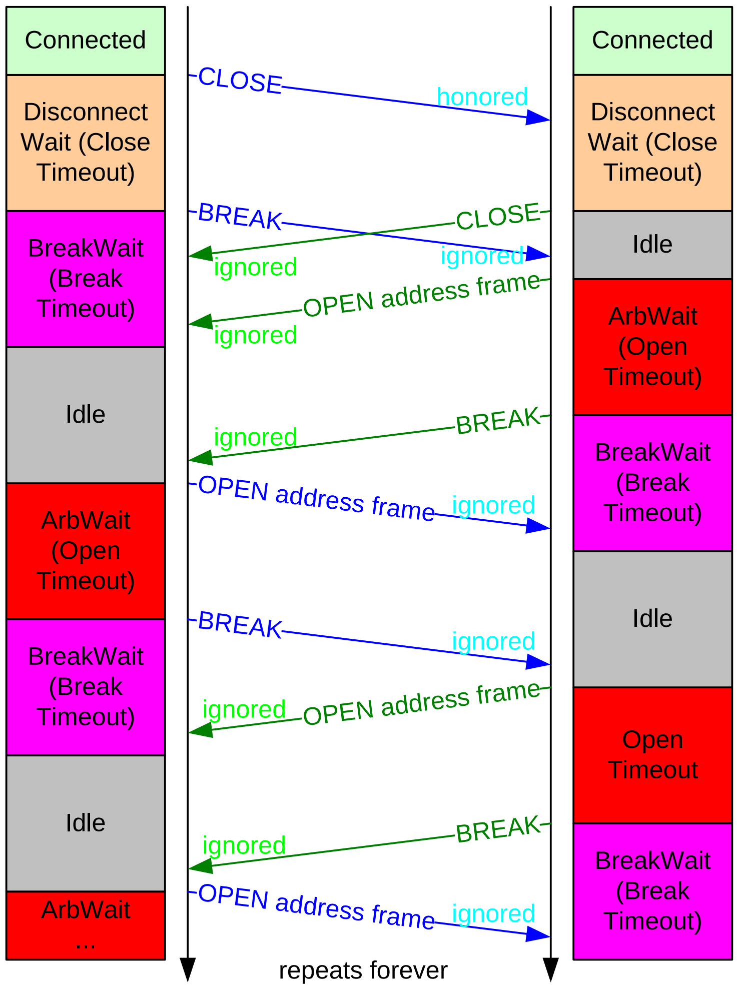


Figure 2 — CLOSE racing BREAK

The SL_CC and XL state machines have numerous reasons to enter the Break_Wait state including CLOSE Timeouts (see table below), all of which may cause this problem.

Minor related problem: With the change made by 05-040, an OPEN_REJECT sends the state machine back to Idle even if Break_Wait was entered because of a missing CLOSE. The transition needs to be conditioned upon “and this state was entered from SL_CC1:ArbSel”.

Most of these cases can be avoided if phys respond to OPENs, DONEs, and CLOSEs faster than 1 ms, to avoid triggering expiration of the 1 ms timer in the other phy. The CLOSE scenario only happens if phy B waits 1 ms to reply to CLOSE; this is unlikely.

In any situation where the BREAK is triggered by an arbitrary upper layer request, however, there is no guarantee about what the other phy is doing at the time. The OPEN_REJECT scenario will usually be avoided if phy B doesn't wait 1 ms to send OPEN_REJECT; however, if phy A sends BREAK earlier than 1 ms, it could still cross the OPEN_REJECT.

Table 1 — Causes for entry into Break-Wait state

Cause for entry into Break_Wait state	To avoid race condition
7.14 SL_CC:	
7.14.4.1 any SL_CC state: any detection of any internal error	no remedy
7.14.4.3.5 SL_CC1:ArbSel: **Stop Arb request** from upper layers	no remedy; BreakWait must be sensitive to OPEN_ACCEPT and OPEN_REJECT
7.14.4.3.5 SL_CC1:ArbSel: **Open Timeout timer** expires	send OPEN_ACCEPT or OPEN_REJECT within << 1 ms of receiving OPEN or, for expanders, sending an AIP
7.14.4.5.3 SL_CC3:Connected: **Request Break message** received from SSP, SMP, or STP:	
a) 7.16.7.5 SSP_D: **DONE Timeout timer** expires*:*	send DONE within << 1 ms
b) 7.17.8 STP reasons not specified	no remedy; BreakWait must be sensitive to CLOSE
c) 7.18.4.3.2/3/4 SMP_IP1/2/3: **SMP Transmit Break** from upper layers	no remedy; BreakWait must be sensitive to CLOSE
d) 7.18.4.3.4 SMP_IP3: frame is too small, too big, has a invalid dword or ERROR, has a CRC error	no remedy; BreakWait must be sensitive to CLOSE
e) 7.18.4.4.2/3 SMP_TP1/2: **SMP Transmit Break** from upper layers	no remedy; BreakWait must be sensitive to CLOSE
f) 7.18.4.4.2 SMP_TP1: frame is too small, too big, has a invalid dword or ERROR, has a CRC error	send BREAK within << 1 ms
7.14.4.6.3 SL_CC4:DisconnectWait: **Close Timeout timer** expires	in SSP, STP, and SMP, send CLOSE within << 1 ms of noticing conditions that require it
a) SSP: after DONE/DONE exchange	
b) SMP_IP: after receiving the EOF for the SMP RESPONSE frame	
c) SMP_TP: after transmitting the EOF for the SMP RESPONSE frame	
d) STP: after receiving CLOSE	
7.15 XL, each after receiving Forward Break from the ECM:	
a) 7.15.6 XL3:Open_Confirm_Wait	
b) 7.15.9 XL6:Open_Response_Wait	
c) 7.15.10 XL7:Connected	
d) 7.15.11 XL8:Close_Wait	Break_Wait must be sensitive to CLOSE

A combination of time limits and additional transitions from Break_Wait are proposed. The time limits just standardize the behavior of current implementations and ensure that, if applications don't arbitrary cause

BREAKs, no race conditions will occur. The additional transitions ensure that applications can cause BREAKs without causing a problem.

Suggested changes

7.12 Connections

7.12.1 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, or transfer data. SSP target ports open SSP connections to transfer data or transmit status.

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. AIP, OPEN_ACCEPT and OPEN_REJECT are the responses to an OPEN address frame. BREAK is used to abort connection requests and to unilaterally break a connection. CLOSE is used for orderly closing a connection.

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 in an expander device attached to a SATA device. The SATA device is not aware of SAS connection management.

A wide port may have separate connections on each of its phys.

7.12.x Timeout timers and time limits

Phys maintain the timeout timers listed in table 2.

Table 2 — Phy timeout timers

Timer	Initial value	Description
Open Timeout timer	1 ms	Maximum time from transmitting an OPEN address frame or receiving AIP until receiving AIP, OPEN_ACCEPT or OPEN_REJECT. See 7.12.2.
Close Timeout timer	1 ms	Maximum time from transmitting CLOSE until receiving CLOSE. See 7.12.7.
Break Timeout timer	1 ms	Maximum time from transmitting BREAK until receiving BREAK. See 7.12.6 and 7.12.8.

Phys honor the time limits listed in table 3.

Table 3 — Phy time limits

Time limit	Value	Description
Open Response time limit	0,5 ms	Maximum time from receiving an OPEN address frame or transmitting AIP until transmitting OPEN_ACCEPT or OPEN_REJECT. See 7.12.2.
Close Response time limit	0,5 ms	Maximum time from receiving a CLOSE until transmitting CLOSE. See 7.12.7.
Break Response time limit	0,5 ms	Maximum time from receiving a BREAK until transmitting BREAK. See 7.12.6 and 7.12.8.

7.12.2 Opening a connection

7.12.2.1 Connection request

The OPEN address frame (see 7.8.3) is used to open a connection from a source port to a destination port using one source phy and one destination phy.

To make a connection request, the source port shall transmit an OPEN address frame through an available phy. The source phy shall transmit idle dwords after the OPEN address frame until it receives a response or aborts the connection request with BREAK.

After transmitting an OPEN address frame, the source phy shall initialize and start ~~a 1 ms~~ the Open Timeout timer (see 7.12.x). 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, the source phy may assume the destination port does not exist and shall transmit BREAK to abort the connection request (see 7.12.6).

The OPEN address frame flows through expander devices onto intermediate physical links. If an expander device on the pathway is unable to forward the connect request because none of the prospective physical links support the requested connection rate, the expander device shall return OPEN_REJECT (CONNECTION RATE NOT SUPPORTED) within an OPEN Response time limit (see 7.13.6) of receiving the OPEN address frame or transmitting an AIP. If the OPEN address frame reaches the destination, it shall return either OPEN_ACCEPT or OPEN_REJECT within an OPEN Response time limit (see 7.12.6) of receiving the OPEN address frame . Rate matching shall be used on any physical links in the pathway with negotiated physical link rates that are faster than the requested connection rate (see 7.13).

7.12.2.2 Connection responses

Table 4 lists the responses to an OPEN address frame being transmitted.

Table 4 — Connection responses

Response	Description
AIP	Arbitration in progress. When an expander device is trying to open a connection to the selected destination port, it returns an AIP to the source phy. The source phy shall reinitialize and restart its Open Timeout timer when it receives an AIP. AIP is sent by an expander device while it is internally arbitrating for access to an expander port.
OPEN_ACCEPT	Connection request accepted. This is sent by the destination phy.
OPEN_REJECT	Connection request rejected. This is sent in response by the destination phy or by an expander device. The different versions are described in 7.2.5.11. See 4.5 for I_T nexus loss handling.
OPEN address frame	If AIP has been previously detected, this indicates an overriding connection request. If AIP has not yet been detected, this indicates two connection requests crossing on the physical link. Arbitration fairness determines which one wins (see 7.12.3).
BREAK	The destination port or expander port may reply with BREAK indicating the connection is not being established.
Open Timeout timer expires	The source phy shall abort the connection request by transmitting BREAK (see 7.12.6). See 4.5 for I_T nexus loss handling.

After an OPEN_REJECT (CONNECTION RATE NOT SUPPORTED) has been received by a SAS target port, the SAS target device shall set the connection rate for future requests for that I_T_L_Q nexus to:

a) the last value received in a connection request from the SAS initiator port;
b) 1,5 Gbps; or
c) the connection rate in effect when the command was received.

7.12.3 Arbitration fairness

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

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

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

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

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

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

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

NOTE 1 - NOTE 1 Connection responses that are conclusively from the destination phy (see table 69 and table 70 in 7.2.5.11) are included in the list. Connection responses that are only from or may be from expander phys are not included.

A SAS port should not stop the Arbitration Wait Time timer and set it to zero when it receives an incoming OPEN address frame that has priority over the outgoing OPEN address frame according to table 5, regardless of whether it 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 7.12.4.

If two connection requests pass on a physical link, the phy shall determine the winner by comparing OPEN address frame field contents using the arbitration priority described in table 5.

Table 5 — Arbitration priority for OPEN address frames passing on a physical link

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

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

7.12.4 Arbitration and resource management in an expander device

7.12.4.1 Arbitration overview

The ECM shall arbitrate and assign or deny path resources for Request Path requests from each expander 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 pathway(s) to tear down to allow at least one connection to complete.

The ECM responds to connection requests by returning an Arb Won, Arb Lost, or Arb Reject confirmation to the requesting expander phy.

Several of the Request Path arguments are used for arbitration. The Arbitration Wait Time, Source SAS Address, and Connection Rate arguments are filled in from the received OPEN address frame and are used to by the ECM to compare Request Path requests. The Retry Priority Status argument is used to prevent the Arbitration Wait Time argument from being considered during an arbitration which occurs after a Backoff Retry response is sent by an expander phy (see 7.15.4).

An expander phy shall set the Retry Priority Status argument to IGNORE AWT when it requests a path after:

a) it has forwarded an OPEN address frame to the physical link;
b) an OPEN address frame is received with higher arbitration priority (see 7.12.3); 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 7.15.4 and 7.15.9).

Otherwise, the expander phy shall set the Retry Priority Status argument to NORMAL.

If two or more Request Path requests contend and all of the Request Path requests include a Retry Priority Status argument set to NORMAL, the ECM shall select the winner by comparing the OPEN address frame contents described in table 6.

Table 6 — Arbitration priority for contending Request Path requests in the ECM when all requests have Retry Priority Status arguments of NORMAL

Bits 83-68 (83 is MSB)	Bits 67-4	Bits 3-0 (0 is LSB)
ARBITRATION WAIT TIME field value	SOURCE SAS ADDRESS field value	CONNECTION RATE field value

If two or more Request Path requests contend and one or more of the Request Path requests include a Retry Priority Status argument set to IGNORE AWT, the ECM shall select the winner from the set of Request Path requests with Retry Priority Status arguments of IGNORE AWT by comparing the OPEN address frame contents described in table 7.

Table 7 — Arbitration priority for contending Request Path requests in the ECM among requests with Retry Priority Status arguments of IGNORE AWT

Bits 67-4 (67 is MSB)	Bits 3-0 (0 is LSB)
SOURCE SAS ADDRESS field value	CONNECTION RATE field value

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

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

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

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

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

a) the connection request maps to an expander phy that:
    A) supports the connection rate; and
    B) is not reporting a Phy Status (Partial Pathway), Phy Status (Blocked Partial Pathway), or Phy Status (Connection) response, unless that expander phy is arbitrating for the expander phy making this connection request;
b) there are sufficient routing resources to complete the connection request;
c) no higher priority connection requests are present with this expander phy as the destination; and

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

7.12.4.2 Arbitration status

Arbitration status shall be conveyed between expander devices and by expander devices to SAS endpoints using AIP primitives. 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 phy is set to the last type of AIP received.

7.12.4.3 Partial Pathway Timeout timer

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

NOTE 2 - The partial pathway timeout value allows flexibility in specifying how long an expander device waits before attempting pathway recovery. The recommended default value (see 10.4.3.5) was chosen to cover a wide range of topologies. Selecting small partial pathway timeout value values within a large topology may compromise performance because of the time a device waits after receiving OPEN_REJECT (PATHWAY BLOCKED) before it retries the connection request. Similarly, selecting large partial pathway timeout value values within a small topology may compromise performance due to waiting longer than necessary to detect pathway blockage.

When the Partial Pathway Timeout timer is not running, an expander phy shall initialize and start the Partial Pathway Timeout timer when all expander phys within the requested destination port contain a blocked partial pathway (i.e., are returning Phy Status (Blocked Partial Pathway)).

NOTE 3 - The Partial Pathway Timeout timer is not initialized and started if one or more of the expander phys within a requested destination port are being used for a connection.

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

7.12.4.4 Pathway recovery

Pathway recovery provides a means to abort connection requests in order to prevent deadlock using pathway recovery priority comparisons. Pathway recovery priority compares the OPEN address frames of the blocked connection requests as described in table 8.

Table 8 — Pathway recovery priority

Bits 71-64 (71 is MSB)	Bits 63-0 (0 is LSB)
PATHWAY BLOCKED COUNT field value	SOURCE SAS ADDRESS field value

When the Partial Pathway Timeout timer for an arbitrating expander 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 instruct the arbitrating expander phy to reject the connection request by transmitting OPEN_REJECT (PATHWAY BLOCKED) when:

a) the Partial Pathway Timeout timer expires; and
b) the pathway recovery priority of the arbitrating expander phy (i.e., the expander phy requesting the connection) is less than or equal to the pathway recovery priority of any of the expander 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 phy sent a Request Path request to the ECM or those of the Forward Open indication if the phy received a Forward Open indication from the ECR.

7.12.5 Expander devices and connection requests

7.12.5.1 All expander devices

Before an expander device transmits AIP, it may have transmitted an OPEN address frame on the same physical link. Arbitration fairness dictates which OPEN address frame wins (see 7.12.3).

After an expander device transmits an AIP, it shall not transmit an OPEN address frame unless it has higher arbitration priority than the incoming connection request.

Expander devices shall transmit no more than three consecutive AIPs without transmitting an idle dword. Expander devices may transmit three consecutive AIPs to provide better tolerance of errors. Expander devices shall transmit at least one AIP every 128 dwords while transmitting AIP (NORMAL), AIP (WAITING ON PARTIAL), or AIP (WAITING ON CONNECTION).

NOTE 4 - Future versions of this standard may require that expander devices transmit three consecutive AIPs.

Expander devices shall transmit an AIP (e.g., an AIP (NORMAL)) within 128 dwords of receiving an OPEN address frame.

7.12.5.2 Edge expander devices

When an edge expander device receives a connection request, it shall compare the destination SAS address to the SAS addresses of the devices to which each of its phys is attached. For all phys which have table routing attributes (see 4.6.7.1) and are attached to edge expander devices, it shall compare the destination SAS address to all the enabled routed SAS addresses in the expander route table.

If it finds a match in one or more phys, then the expander device shall arbitrate for access to one of the matching phys and forward the connection request.

If it does not find a match, but at least one phy has the subtractive routing attribute and is attached to an expander device (i.e., it is attached to an edge expander device or a fanout expander device and the phy is using the subtractive routing method), and the request did not come from that expander device, the connection request shall be forwarded to the expander device through any of the subtractive routing phys.

If it does not find a match and no phy using the subtractive routing method is available, the edge expander device shall reply with OPEN_REJECT (NO DESTINATION).

If the destination phy is in the same expander port as the source phy and is using the subtractive routing method, the edge expander device shall reply with OPEN_REJECT (NO DESTINATION).

If the destination phy is in the same expander port as the source phy and is using the direct routing method or the table routing method, the edge expander device shall reply with either OPEN_REJECT (NO DESTINATION) or OPEN_REJECT (BAD DESTINATION); it should reply with OPEN_REJECT (NO DESTINATION).

NOTE 5 - Edge expander devices compliant with previous versions of this standard were required to return OPEN_REJECT (BAD DESTINATION).

7.12.5.3 Fanout expander devices

When a fanout expander device receives a connection request, it shall compare the destination SAS address to the SAS addresses of the devices to which each of its phys is attached. For all phys that are attached to edge expander devices, the fanout expander device shall compare the destination SAS addresses to all the enabled SAS addresses in the expander route table.

If the fanout expander device finds a match in one or more phys, it shall arbitrate for access to one of the matching phys and forward the connection request.

If the fanout expander device does not find a match, it shall reply with OPEN_REJECT (NO DESTINATION). If the destination phy is in the same expander port as the source phy, it shall reply with either OPEN_REJECT (NO DESTINATION) or OPEN_REJECT (BAD DESTINATION); it should reply with OPEN_REJECT (NO DESTINATION).

NOTE 6 - Fanout expander devices compliant with previous versions of this standard were required to return OPEN_REJECT (BAD DESTINATION).

7.12.6 Aborting a connection request

BREAK may be used to abort a connection request. The source phy shall transmit a BREAK after the Open Timeout timer expires or if it chooses to abort its request for any other reason.

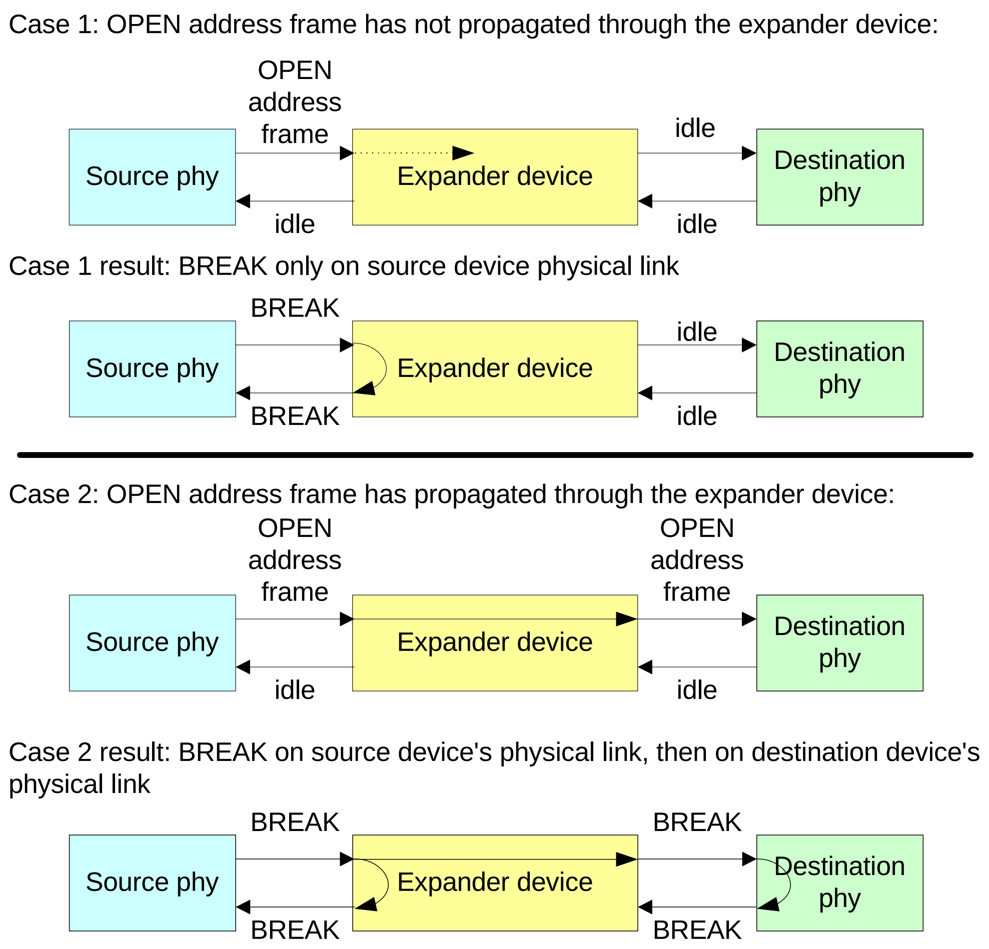
After transmitting BREAK, the source phy shall initialize ~~a~~ and start the Break Timeout timer (see 7.12.x) ~~to 1 ms and start the Break Timeout timer~~.

Table 9 lists the responses to a BREAK being transmitted before a connection response has been received.

Table 9 — Abort connection responses

Response	Description
BREAK	This confirms that the connection request has been aborted.
Break Timeout timer expires	The originating phy shall assume the connection request has been aborted.

When a phy sourcing a BREAK is attached to an expander device, the BREAK response to the source phy is generated by the expander phy to which the source phy is attached, not the other SAS phy in the connection. If the expander device has transmitted a connection request to the destination, it shall also transmit BREAK to the destination. If the expander device has not transmitted a connection request to the destination, it shall not transmit BREAK to the destination. After transmitting BREAK back to the originating phy, the expander device shall ensure that an open response does not occur (i.e., the expander device shall not forward dwords from the destination any more). Figure 3 shows an example of BREAK usage.



Case 1: OPEN address frame has not propagated through the expander device:

Case 1 result: BREAK only on source device physical link

Case 2: OPEN address frame has propagated through the expander device:

Case 2 result: BREAK on source device's physical link, then on destination device's physical link

Figure 3 — Aborting a connection request with BREAK

Figure 4 shows the sequence for a connection request where the Open Timeout timer expires.

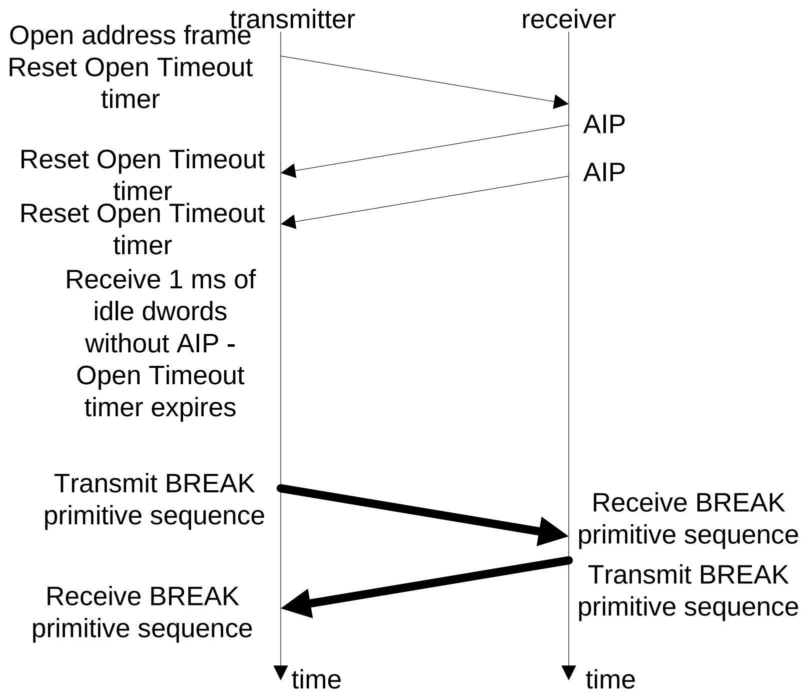

transmitter
receiver
Open address frame
Reset Open Timeout timer
AIP
AIP
Reset Open Timeout timer
Reset Open Timeout timer
Receive 1 ms of idle dwords without AIP - Open Timeout timer expires
Transmit BREAK primitive sequence
Receive BREAK primitive sequence
Transmit BREAK primitive sequence
Receive BREAK primitive sequence
time
time

Figure 4 — Connection request timeout example

7.12.7 Closing a connection

CLOSE is used to close a connection of any protocol. See 7.16.6 for details on closing SSP connections, 7.17.6 for details on closing STP connections, and 7.18.3 for details on closing SMP connections.

After transmitting CLOSE, the source phy shall initialize ~~a~~and start the Close Timeout timer (see 7.12.x) ~~to 1 ms and start the Close Timeout timer~~.

Table 10 lists the responses to a CLOSE being transmitted.

Table 10 — Close connection responses

Response	Description
CLOSE	This confirms that the connection has been closed.
Close Timeout timer expires	The originating phy shall attempt to break the connection (see 7.12.8).

No additional dwords for the connection shall follow the CLOSE. Expander devices shall close the full-duplex connection upon detecting a CLOSE in each direction.

When a phy has both transmitted and received CLOSE, it shall consider the connection closed.

Figure 5 shows example sequences for closing a connection.

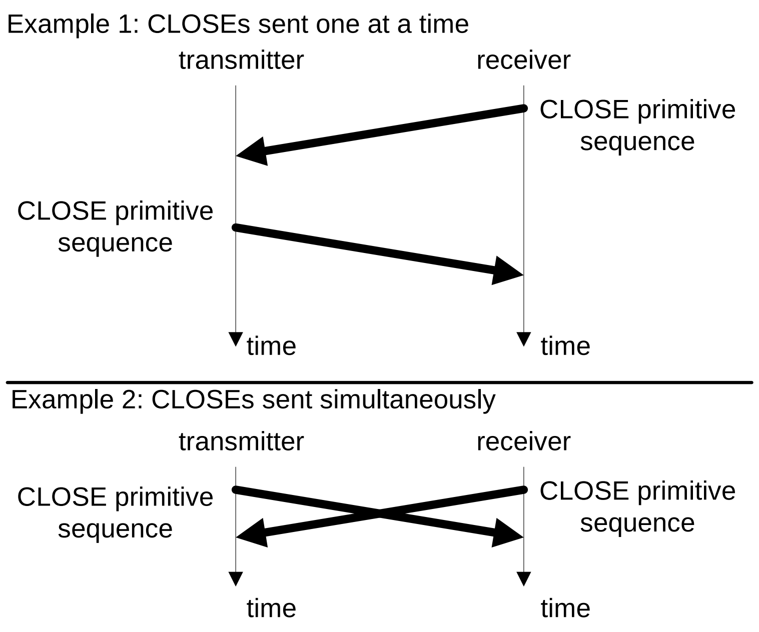


Figure 5 — Closing a connection example

7.12.8 Breaking a connection

In addition to aborting a connection request, BREAK may also be used to break a connection, in cases where CLOSE is not available. After transmitting BREAK, the originating phy shall ignore all incoming dwords except for BREAKs.

After transmitting BREAK, the source phy shall initialize ~~a~~and start the Break Timeout timer (see 7.12.x) ~~to 1 ms and start the Break Timeout timer~~.

Table 11 lists the responses to a BREAK being transmitted after a connection has been established.

Table 11 — Break connection responses

Response	Description
BREAK	This confirms that the connection has been broken.
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 due to loss of dword synchronization (see 6.9).

In addition to the actions described in this subclause and in 7.12.6, the following shall be the responses by an SSP phy to a broken connection:

a) 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;
b) Transmitted frames for which an ACK has been received prior to a broken connection shall be considered successfully transmitted; and
c) Transmitted frames for which an ACK or NAK has not been received prior to a broken connection shall be considered not successfully transmitted.

...

7.14.9 SL_CC (connection control) state machine

7.14.9.1 SL_CC state machine overview

The state machine consists of the following states:

a) SL_CC0:Idle (see 7.14.9.2)(initial state);
b) SL_CC1:ArbSel (see 7.14.9.3);
c) SL_CC2:Selected (see 7.14.9.4);
d) SL_CC3:Connected (see 7.14.9.5);
e) SL_CC4:DisconnectWait (see 7.14.9.6);
f) SL_CC5:BreakWait (see 7.14.9.7);
g) SL_CC6:Break (see 7.14.9.8); and
h) SL_CC7:CloseSTP (see 7.14.9.9).

The 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 7.9.5).

The SL_CC state machine receives the following messages from the SSP link layer state machine (see 7.16.7), the STP link layer state machine, and SMP link layer state machine (see 7.18.4):

a) Request Break; and
b) Request Close.

The SL_CC 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;
b) Enable Disable STP; and
c) Enable Disable SMP.

The SL_CC state machine receives the following messages from the SL_IR state machines (see 7.9.5):

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 shall be ignored.

Any detection of an internal error shall cause the SL_CC state machine to transition to the SL_CC5:BreakWait state.

The SL_CC state machine shall maintain the timers listed in table 12.

Table 12 — SL_CC timers

Timer	Initial value
Open Timeout timer	1 ms
Close Timeout timer	1 ms
Break Timeout timer	1 ms

The SL_CC state machine shall comply with the time limits listed in table 13.

Table 13 — SL_CC time lmiits

Time limit	Value	Description
Open Response time limit	0,5 ms	Maximum time from receiving an OPEN address frame until transmitting OPEN_ACCEPT or OPEN_REJECT.
Close Response time limit	0,5 ms	Maximum time from receiving a CLOSE until transmitting CLOSE.
Break Response time limit	0,5 ms	Maximum time from receiving a BREAK until transmitting BREAK.

7.14.9.2 SL_CC0:Idle state

7.14.9.2.1 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 send Enable Disable SSP (Disable), Enable Disable SMP (Disable), and Enable Disable STP (Disable) messages to the SSP, SMP, and STP link layer state machines.

This state shall request idle dwords be transmitted by repeatedly sending Transmit Idle Dword messages to the SL transmitter (see 7.4).

If a BROADCAST Received (Change) message is received, this state shall send a Change Received confirmation to the management layer.

If a Transmit Broadcast request is received with any argument, this state shall send a Transmit BROADCAST message with the same argument to the SL transmitter.

7.14.9.2.2 Transition SL_CC0:Idle to SL_CC1:ArbSel

This transition shall occur after receiving both an Enable Disable SAS Link (Enable) 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.

7.14.9.2.3 Transition SL_CC0:Idle to SL_CC2:Selected

This transition shall occur after receiving both an Enable Disable SAS Link (Enable) confirmation and an OPEN Address Frame Received message.

7.14.9.3 SL_CC1:ArbSel state

7.14.9.3.1 State description

This state is used to make a connection request.

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.

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 is received this state shall send a Change Received confirmation to the management layer.

If an AIP Received message is received after requesting the OPEN address frame be transmitted, 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 this state receives an OPEN_REJECT Received (No Destination) message after transmitting the OPEN address frame, this state shall send an Open Failed (No Destination) confirmation to the port layer.

If this state receives an OPEN_REJECT Received (Bad Destination) message after transmitting the OPEN address frame, this state shall send an Open Failed (Bad Destination) confirmation to the port layer.

If this state receives an OPEN_REJECT Received (Wrong Destination) message after transmitting the OPEN address frame, this state shall send an Open Failed (Wrong Destination) confirmation to the port layer.

If this state receives an OPEN_REJECT Received (Connection Rate Not Supported) message after transmitting the OPEN address frame, this state shall send an Open Failed (Connection Rate Not Supported) confirmation to the port layer.

If this state receives an OPEN_REJECT Received (Protocol Not Supported) message after transmitting the OPEN address frame, this state shall send an Open Failed (Protocol Not Supported) confirmation to the port layer.

If this state receives an OPEN_REJECT Received (Retry) message after transmitting the OPEN address frame, this state shall send an Open Failed (Retry) confirmation to the port layer.

If this state receives an OPEN_REJECT Received (Pathway Blocked) message after transmitting the OPEN address frame, this state shall send an Open Failed (Pathway Blocked) confirmation to the port layer.

7.14.9.3.2 Transition SL_CC1:ArbSel to SL_CC0:Idle

This transition shall occur after sending an Open Failed confirmation.

7.14.9.3.3 Transition SL_CC1:ArbSel to SL_CC2:Selected

This transition shall occur after transmitting the OPEN address frame 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 the arbitration fairness rules (see 7.12.3) indicate the received OPEN address frame overrides the outgoing OPEN address frame.

The arbitration fairness comparison shall use the value of the arbitration wait time argument to the Open Connection request for the outgoing OPEN address frame and the value of the ARBITRATION WAIT TIME field received in the incoming OPEN address frame.

7.14.9.3.4 Transition SL_CC1:ArbSel to SL_CC3:Connected

This transition shall occur if this state receives an OPEN_ACCEPT Received message after transmitting the OPEN address frame.

If the PROTOCOL field in the transmitted OPEN address frame was set to STP, then this state shall send a Connection Opened (STP, Source Opened) confirmation to the port layer before the transition. This transition shall include an Open STP Connection argument. At this point an STP connection has been opened between the source phy and the destination phy.

If the PROTOCOL field in the transmitted OPEN address frame was set to SSP, then this state shall send a Connection Opened (SSP, Source Opened) confirmation to the port layer before the transition. This transition shall include an Open SSP Connection argument. At this point an SSP connection has been opened between the source phy and the destination phy.

If the PROTOCOL field in the transmitted OPEN address frame was set to SMP, then this state shall send a Connection Opened (SMP, Source Opened) confirmation to the port layer before the transition. This transition shall include an Open SMP Connection argument. At this point an SMP connection has been opened between the source phy and the destination phy.

7.14.9.3.5 Transition SL_CC1:ArbSel to SL_CC5:BreakWait

This transition shall occur if a BREAK Received message has not been received and after:

a) a Stop Arb request is received and after sending an Open Failed (Port Layer Request) confirmation to the port layer; or
b) there is no response to the OPEN address frame before the Open Timeout timer expires and after sending an Open Failed (Open Timeout Occurred) confirmation to the port layer.

7.14.9.3.6 Transition SL_CC1:ArbSel to SL_CC6:Break

This transition shall occur after:

a) receiving a BREAK Received message; and
b) sending an Open Failed (Break Received) confirmation to the port layer.

7.14.9.4 SL_CC2:Selected state

7.14.9.4.1 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 within an OPEN Response time limit using the following rules:

1) If the OPEN address frame DESTINATION SAS ADDRESS field does not match the SAS address of this port, this state shall send a Transmit OPEN_REJECT (Wrong Destination) message to the SL transmitter;
2) If the OPEN address frame INITIATOR PORT bit, PROTOCOL field, FEATURES field, and/or INITIATOR CONNECTION TAG field are set to values that are not supported (e.g., a connection request from an SMP target port), this state shall send a Transmit OPEN_REJECT (Protocol Not Supported) message to the SL transmitter;
3) If the OPEN address frame CONNECTION RATE field is set to a connection rate that is not supported, this state shall send a Transmit OPEN_REJECT (Connection Rate Not Supported) message to the SL transmitter;
4) If the OPEN address frame PROTOCOL field is set to STP, the source SAS address is not that of the STP initiator port with an affiliation established or the source SAS address is not that of an STP initiator port with task file register set resources (see 7.17.4), this state shall send a Transmit OPEN_REJECT (STP Resources Busy) message to the SL transmitter;
5) If an Accept_Reject Opens (Reject SSP) request, Accept_Reject Opens (Reject SMP) request, or Accept_Reject Opens (Reject STP) request is received and the requested protocol is the corresponding protocol, this state shall send a Transmit OPEN_REJECT (Retry) message to the SL transmitter;
6) If the requested protocol is SSP and this state has not received an Accept_Reject Opens (Reject SSP) request then this state shall send a Transmit OPEN_ACCEPT message to the SL transmitter and send a Connection Opened (SSP, Destination Opened) confirmation to the port layer;

7) If the requested protocol is SMP and this state has not received an Accept_Reject Opens (Reject SMP) request then this state shall send a Transmit OPEN_ACCEPT message to the SL transmitter and send a Connection Opened (SMP, Destination Opened) confirmation to the port layer; or
8) If the requested protocol is STP and this state has not received an Accept_Reject Opens (Reject STP) request then this state shall send a Transmit OPEN_ACCEPT message to the SL transmitter and send a Connection Opened (STP, Destination Opened) confirmation to the port layer.

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

7.14.9.4.2 Transition SL_CC2:Selected to SL_CC0:Idle

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

7.14.9.4.3 Transition SL_CC2:Selected to SL_CC3:Connected

This transition shall occur after sending a Connection Opened confirmation.

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

7.14.9.4.4 Transition SL_CC2:Selected to SL_CC6:Break

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

7.14.9.5 SL_CC3:Connected state

7.14.9.5.1 State description

This state enables the SSP, STP, or SMP link layer state machine to transmit dwords during a connection. See 7.13 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 7.18.4).

If this state is entered from SL_CC1:ArbSel state or the SL_CC2:Selected state with an argument of Open SSP Connection then this state shall send an Enable Disable SSP (Enable) message to the SSP link layer state machines (see 7.16.7).

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 7.17.8).

This state shall request 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) is received during an STP connection, this state shall clear any affiliation (see 7.17.4).

7.14.9.5.2 Transition SL_CC3:Connected to SL_CC4:DisconnectWait

This transition shall occur if a Request Close message is received.

7.14.9.5.3 Transition SL_CC3:Connected to SL_CC5:BreakWait

This transition shall occur after sending a Connection Closed (Break Requested) confirmation to the port layer if:

a) a Request Break message is received; and
b) a BREAK Received message has not been received.

7.14.9.5.4 Transition SL_CC3:Connected to SL_CC6:Break

This transition shall occur if a BREAK Received message is received and after sending a Connection Closed (Break Received) confirmation to the port layer.

7.14.9.5.5 Transition SL_CC3:Connected to SL_CC7:CloseSTP

This transition shall occur if a CLOSE Received message is received during an STP connection.

7.14.9.6 SL_CC4:DisconnectWait state

7.14.9.6.1 State description

This state closes the connection and releases all resources associated with the connection.

This state shall:

1) send a Transmit CLOSE (Normal) message or Transmit CLOSE (Clear Affiliation) message to the SL transmitter. If the state machine has received a CLOSE Received message, this state shall send theTransmit CLOSE message within a CLOSE Response timeout; 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) is received during an STP connection, this state shall clear any affiliation (see 7.17.4).

7.14.9.6.2 Transition SL_CC4:DisconnectWait to SL_CC0:Idle

This transition shall occur after:

a) sending a Transmit CLOSE message to the SL transmitter;
b) receiving a CLOSE Received message; and
c) sending a Connection Closed (Normal) confirmation to the port layer.

7.14.9.6.3 Transition SL_CC4:DisconnectWait to SL_CC5:BreakWait

This transition shall occur if:

a) a BREAK Received message has not been received;
b) no CLOSE Received message is received in response to a Transmit CLOSE message before the Close Timeout timer expires; and
c) after sending a Connection Closed (Close Timeout) confirmation to the port layer.

7.14.9.6.4 Transition SL_CC4:DisconnectWait to SL_CC6:Break

This transition shall occur after receiving a BREAK Received message and after sending a Connection Closed (Break Received) confirmation to the port layer.

7.14.9.7 SL_CC5:BreakWait state

7.14.9.7.1 State description

This state closes the connection if one is established and releases all resources associated with the connection.

This state shall:

1) send a Transmit BREAK message to the SL transmitter; and
2) initialize and start the Break Timeout timer.

7.14.9.7.2 Transition SL_CC5:BreakWait to SL_CC0:Idle

This transition shall occur after:

a) receiving a BREAK Received message;

b) receiving an OPEN_REJECT Received message, if this state was entered from the SL_CC1:ArbWait state; ~~or~~
c) receiving a CLOSE Received message, if this state was entered from the SL_CC4:DisconnectWait state; or
d) the Break Timeout timer expires.

7.14.9.8 SL_CC6:Break state

7.14.9.8.1 State description

This state closes any connection and releases all resources associated with this connection.

This state shall send a Transmit BREAK message to the SL transmitter within a BREAK Response time limit.

7.14.9.8.2 Transition SL_CC6:Break to SL_CC0:Idle

This transition shall occur after sending a Transmit BREAK message to the SL transmitter.

7.14.9.9 SL_CC7:CloseSTP state

7.14.9.9.1 State description

This state closes an STP connection and releases all resources associated with the connection.

This state shall:

1) send a Transmit CLOSE (Normal) message or Transmit CLOSE (Clear Affiliation) message to the SL transmitter within a CLOSE Response time limit; and
2) send a Connection Closed (Normal) confirmation to the port layer.

7.14.9.9.2 Transition SL_CC7:CloseSTP to SL_CC0:Idle

This transition shall occur after sending a Connection Closed (Normal) confirmation to the port layer.

7.15 XL (link layer for expander phys) state machine

7.15.1 XL state machine overview

The XL state machine controls the flow of dwords on the physical link and establishes and maintains connections with another XL state machine as facilitated by the expander function - specifically the ECM and ECR.

This state machine consists of the following states:

a) XL0:Idle (see 7.15.3)(initial state);
b) XL1:Request_Path (see 7.15.4);
c) XL2:Request_Open (see 7.15.5);
d) XL3:Open_Confirm_Wait (see 7.15.6);
e) XL4:Open_Reject (see 7.15.7);
f) XL5:Forward_Open (see 7.15.8);
g) XL6:Open_Response_Wait (see 7.15.9);
h) XL7:Connected (see 7.15.10);
i) XL8:Close_Wait (see 7.15.11);
j) XL9:Break (see 7.15.12); and
k) XL10:Break_Wait (see 7.15.13).

The XL 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 7.9.5).

The XL state machine receives the following messages from the SL_IR state machine:

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 shall be ignored.

The XL state machine shall maintain the timers listed in table 14.

Table 14 — XL timers

Timer	Initial value
Partial Pathway Timeout timer	Partial pathway timeout value (see 7.12.4.3)
Break Timeout timer	1 ms

The XL state machine shall comply with the time limits listed in table 15.

Table 15 — XL time limits

Time limit	Value	Description
Open Response time limit	0,5 ms	Maximum time from receiving an OPEN address frame or transmitting an AIP until transmitting OPEN_REJECT.
Break Response time limit	0,5 ms	Maximum time from receiving a BREAK Received messageuntil transmitting BREAK (see 7.15.9).

Figure 6 shows several states in the XL state machine.

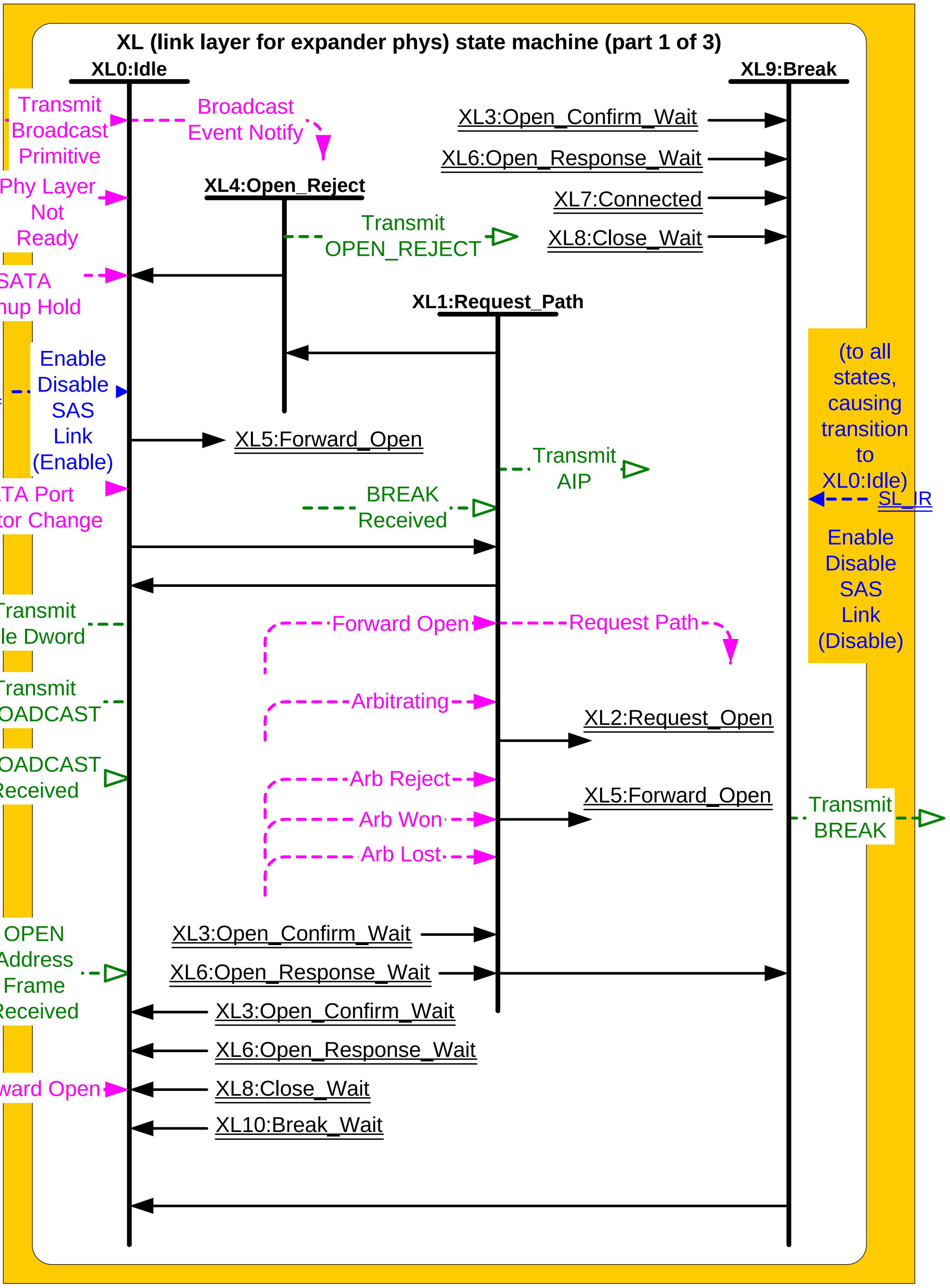


Figure 6 — XL (link layer for expander phys) state machine (part 1)

Figure 7 shows additional states in the XL state machine.

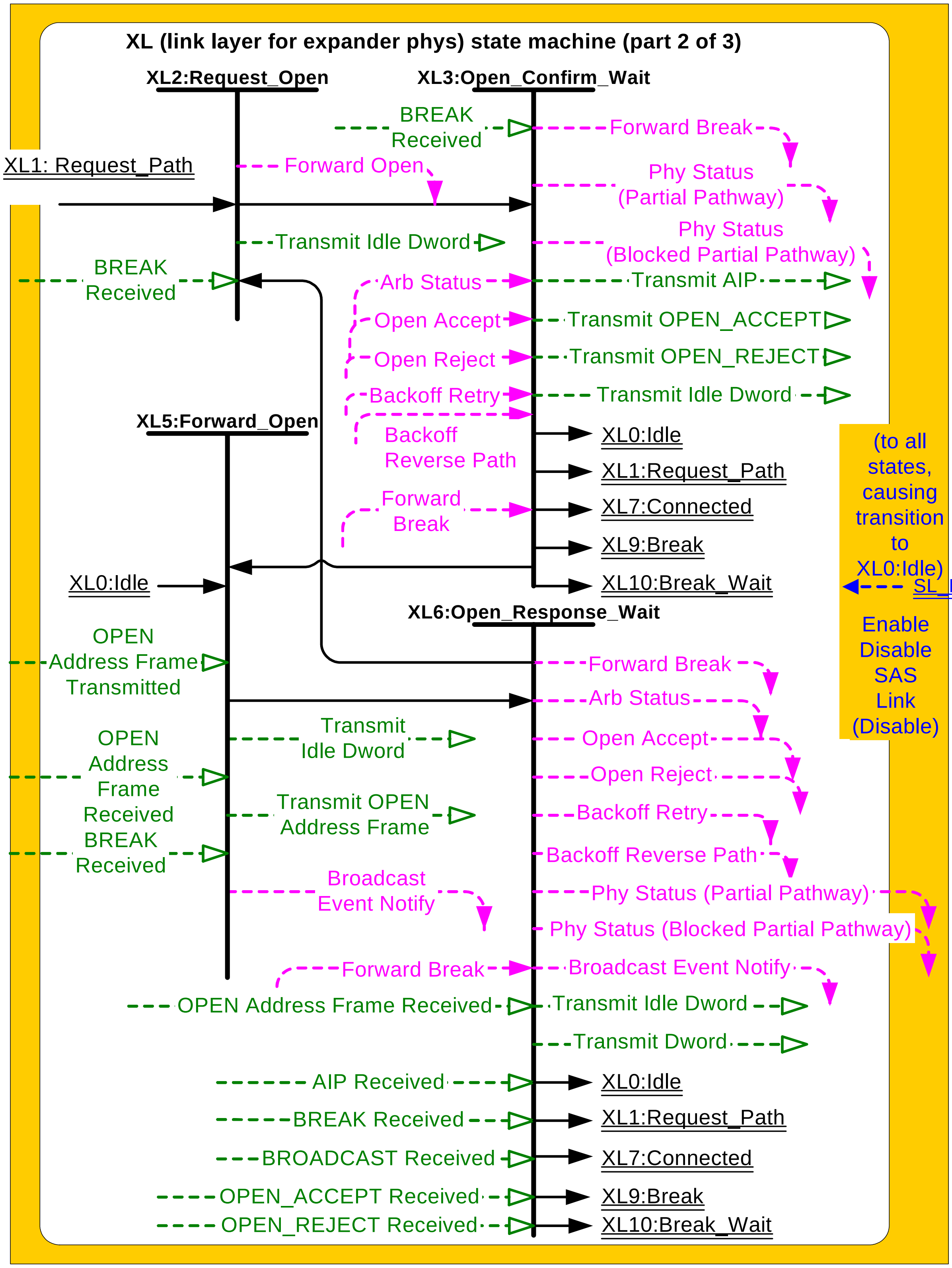


Figure 7 — XL (link layer for expander phys) state machine (part 2)

Figure 8 shows additional states in the XL state machine.

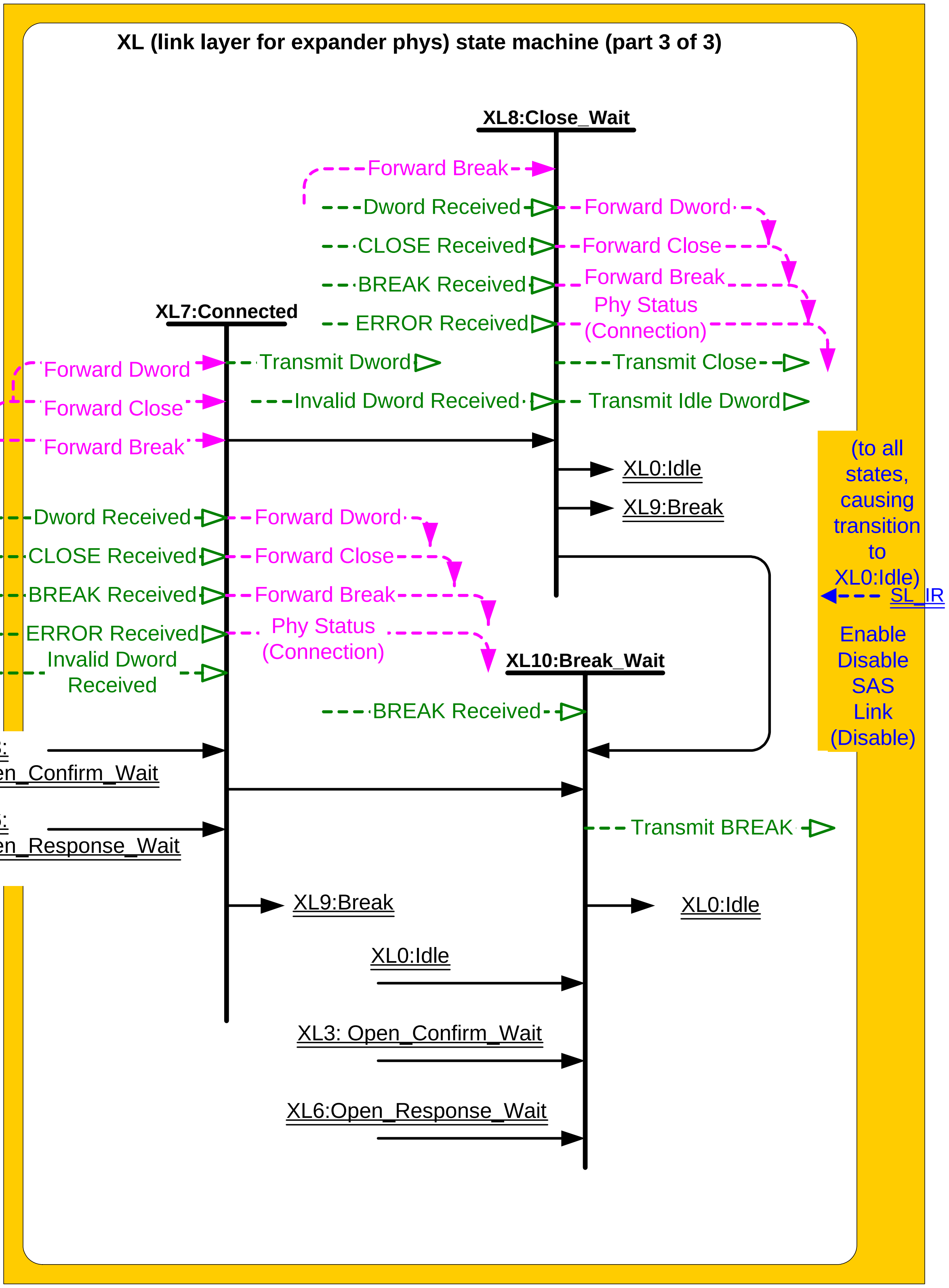


Figure 8 — XL (link layer for expander phys) state machine (part 3)

7.15.2 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 BROADCAST with an argument indicating the specific type (e.g., Transmit BROADCAST (Change));
e) Transmit CLOSE with an argument indicating the specific type (e.g., Transmit CLOSE (Normal));
f) Transmit OPEN_ACCEPT;
g) Transmit OPEN_REJECT, with an argument indicating the specific type (e.g., Transmit OPEN_REJECT (No Destination));
h) Transmit OPEN Address Frame; and
i) Transmit Dword.

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

a) OPEN Address Frame Transmitted.

The XL transmitter shall ensure clock skew management requirements are met (see 7.3) while originating dwords.

The XL transmitter shall ensure clock skew management requirements are met (see 7.3) during and after switching from forwarding dwords to originating dwords, including, for example:

a) when transmitting BREAK;
b) when transmitting CLOSE;
c) when transmitting an idle dword after closing a connection (i.e., after receiving BREAK or CLOSE);
d) while transmitting a SATA frame to a SAS physical link, when transmitting the first SATA_HOLDA in response to detection of SATA_HOLD; and
e) while receiving dwords of a SATA frame from a SAS physical link, when transmitting SATA_HOLD.

NOTE 7 - The XL transmitter may always insert an ALIGN or NOTIFY before transmitting a BREAK, CLOSE, or SATA_HOLDA to meet clock skew management requirements.

The XL transmitter shall insert an ALIGN or NOTIFY before switching from originating dwords to forwarding dwords, 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 physical link, when transmitting the last dword from the SATA flow control buffer in response to release of SATA_HOLD;
d) while transmitting a SATA frame to a SAS physical link, when transmitting the last SATA_HOLDA in response to release of SATA_HOLD (e.g., if the SATA flow control buffer is empty); and
e) while receiving dwords of a SATA frame from a SAS physical link, when transmitting the last SATA_HOLD.

NOTE 8 - This ensures that clock skew management requirements are met, even if the forwarded dword stream does not include an ALIGN or NOTIFY until the last possible dword.

The XL transmitter shall ensure rate matching requirements are met during a connection (see 7.13).

The XL transmitter shall ensure STP initiator phy throttling requirements are met (see 7.17.2) when:

a) transmitting dwords in the direction of an STP target port while originating dwords (e.g., while transmitting SATA_HOLD, SATA_HOLDA, or unloading the SATA flow control buffer);
b) switching from forwarding dwords to originating dwords; and
c) switching from originating dwords to forwarding dwords.

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

The XL receiver sends the following messages to the XL state machine indicating primitive sequences, frames, and dwords received from the SP_DWS receiver (see 6.9.2):

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

a) ignore the invalid dword or ERROR; or
b) discard the address frame.

7.15.3 XL0:Idle state

7.15.3.1 State description

This state is the initial state and is the state that is used when there is no connection pending or established.

If a Phy Layer Not Ready confirmation is received, this state shall send a Broadcast Event Notify (Phy Not Ready) request to the BPP.

If a SATA Spinup Hold confirmation is received, 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, this state shall send a Broadcast Event Notify (Identification Sequence Complete) request to the BPP.

If a SATA Port Selector Change confirmation is received, this state shall send a Broadcast Event Notify (SATA Port Selector Change) request to the BPP.

If a BROADCAST Received message is received, this state shall send a Broadcast Event Notify request to the BPP with the argument indicating the specific BROADCAST primitive received (e.g., CHANGE Received).

If a Transmit Broadcast indication is received, 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 idle dwords be transmitted by repeatedly sending Transmit Idle Dword messages to the XL transmitter.

7.15.3.2 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 with the transition.

7.15.3.3 Transition XL0:Idle to XL5:Forward_Open

This transition shall occur if:

a) an Enable Disable SAS Link (Enable) message has been received; and

b) a Forward Open indication is received.

This transition shall include a set of arguments containing the arguments received in the Forward Open indication.

If an OPEN Address Frame Received message is received, this state shall include an OPEN Address Frame Received argument with the transition.

7.15.4 XL1:Request_Path state

7.15.4.1 State description

This state is used to arbitrate for connection resources and to specify the destination of the connection.

If an Arbitrating (Normal) confirmation is received, this state shall repeatedly send Transmit AIP (Normal) and Transmit Idle Dword messages to the XL transmitter in accordance with AIP transmission rules (see 7.12.5.1).

If an Arbitrating (Waiting On Partial) or Arbitrating (Blocked On Partial) confirmation is received, this state shall repeatedly send Transmit AIP (Waiting On Partial) and Transmit Idle Dword messages to the XL transmitter in accordance with AIP transmission rules (see 7.12.5.1).

If an Arbitrating (Waiting On Partial) confirmation is received, this state shall repeatedly send a Phy Status (Partial Pathway) message to the ECM.

If an Arbitrating (Blocked On Partial) confirmation is received, this state shall repeatedly send a Phy Status (Blocked Partial Pathway) message to the ECM.

If an Arbitrating (Waiting On Connection) confirmation is received, this state shall repeatedly send Transmit AIP (Waiting On Connection) and Transmit Idle Dword messages to the XL transmitter in accordance with AIP transmission rules (see 7.12.5.1).

If an Arbitrating (Waiting On Connection) confirmation is received, this state shall repeatedly send a Phy Status (Connection) message to the ECM.

If this state is entered from the XL6:Open_Response_Wait state, the Retry Priority Status argument shall be set to IGNORE AWT; otherwise, the Retry Priority Status argument shall be set to NORMAL.

Upon entry into this state, this state shall send a Request Path request to the ECM with the following 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;
i) partial pathway timeout status; and
j) retry priority status.

This state maintains the Partial Pathway Timeout timer.

If the Partial Pathway Timeout timer is not already running, 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, 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, timeout status is conveyed to the expander connection manager via the partial pathway timeout status argument in the Request Path request.

7.15.4.2 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.

7.15.4.3 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.

7.15.4.4 Transition XL1:Request_Path to XL4:Open_Reject

This transition shall occur if:

a) a BREAK Received message has not been received; and
b) an Arb Reject confirmation is received.

This transition shall include an Arb Reject argument corresponding to the Arb Reject confirmation.

7.15.4.5 Transition XL1:Request_Path to XL5:Forward_Open

This transition shall occur if a Forward Open indication is received, a BREAK Received message has not been 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 (Bad Connection Rate); or
j) Arb Reject (Pathway Blocked).

This transition shall include an OPEN Address Frame Received argument containing the arguments received in the Forward Open indication.

7.15.4.6 Transition XL1:Request_Path to XL9:Break

This transition shall occur after receiving a BREAK Received message.

7.15.5 XL2:Request_Open state

7.15.5.1 State description

This state is used to forward an OPEN address frame through the ECR to a destination phy.

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

Upon entry into this state, this state shall send a Forward Open request to the ECR, received by the destination phy as a Forward Open indication. The arguments to the Forward Open request are:

a) initiator port bit;
b) protocol;
c) features;
d) connection rate;
e) initiator connection tag;
f) destination SAS address;

g) source SAS address;
h) compatible features;
i) pathway blocked count;
j) arbitration wait time; and
k) more compatible features.

7.15.5.2 Transition XL2:Request_Open to XL3:Open_Confirm_Wait

This transition shall occur after sending a Forward Open request.

If a BREAK Received message is received, this state shall include a BREAK Received argument with the transition.

7.15.6 XL3:Open_Confirm_Wait state

7.15.6.1 State description

This state waits for confirmation to an OPEN address frame sent on a destination phy.

This state shall send the following messages to the XL transmitter:

a) Transmit AIP (Normal) when an Arb Status (Normal) confirmation is received;
b) Transmit AIP (Waiting On Partial) when an Arb Status (Waiting On Partial) confirmation is received;
c) Transmit AIP (Waiting On Connection) when an Arb Status (Waiting On Connection) confirmation is received;
d) Transmit AIP (Waiting On Device) when an Arb Status (Waiting On Device) confirmation is received;
e) Transmit OPEN_ACCEPT when an Open Accept confirmation is received;
f) Transmit OPEN_REJECT 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, 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, 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. 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.

7.15.6.2 Transition XL3:Open_Confirm_Wait to XL0:Idle

This transition shall occur after sending a Transmit OPEN_REJECT message 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.

7.15.6.3 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.

7.15.6.4 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.

7.15.6.5 Transition XL3:Open_Confirm_Wait to XL7:Connected

This transition shall occur after sending a Transmit OPEN_ACCEPT message 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.

7.15.6.6 Transition XL3:Open_Confirm_Wait to XL9:Break

This transition shall occur after sending a Forward Break request.

7.15.6.7 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.

7.15.7 XL4:Open_Reject state

7.15.7.1 State description

This state is used to reject a connection request.

This state shall send one of the following messages within an OPEN Response time limit (see 7.15.1) after sending the last Transmit AIP message to the XL transmitter:

a) a Transmit OPEN_REJECT (No Destination) message when an Arb Reject (No Destination) argument is received with the transition into this state;
b) a Transmit OPEN_REJECT (Bad Destination) message when an Arb Reject (Bad Destination) argument is received with the transition into this state;
c) a Transmit OPEN_REJECT (Connection Rate Not Supported) message when an Arb Reject (Bad Connection Rate) argument is received with the transition into this state; or
d) a Transmit OPEN_REJECT (Pathway Blocked) message when an Arb Reject (Pathway Blocked) argument is received with the transition into this state.

7.15.7.2 Transition XL4:Open_Reject to XL0:Idle

This transition shall occur after OPEN_REJECT has been transmitted.

7.15.8 XL5:Forward_Open state

7.15.8.1 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, this state shall send a Broadcast Event Notify request to the BPP with the argument indicating the specific BROADCAST primitive received (e.g., CHANGE Received).

Upon entry into this state, this state shall send a Transmit OPEN Address Frame message to the XL transmitter with the fields set to the values specified with the transition into this state.

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

7.15.8.2 Transition XL5:Forward_Open to XL6:Open_Response_Wait

This transition shall occur after receiving an OPEN Address Frame Transmitted message.

If an OPEN Address Frame Received message is received, this state shall include an OPEN Address Frame Received argument with the transition.

If a BREAK Received message is received, this state shall include a BREAK Received argument with the transition.

7.15.9 XL6:Open_Response_Wait state

7.15.9.1 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 clock skew 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 this state shall send a Broadcast Event Notify request to the BPP with the argument indicating the specific BROADCAST primitive received (e.g., CHANGE Received).

This state shall send the following responses through the ECR to a source phy, 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 when an AIP Received message has not been received, 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 7.12.3), and 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, after releasing path resources;
d) a Backoff Retry response when an AIP Received message has been received and an OPEN Address Frame Received message is received or an OPEN Address Frame Received argument is included in the transition into this state, and 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, after releasing path resources;
e) a Backoff Reverse Path response when an AIP Received message has not been received, 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 7.12.3), and the destination SAS address and connection rate of the received OPEN address frame are equal to the source SAS address and connection rate of the transmitted OPEN address frame; and
f) a Backoff Reverse Path response when an AIP Received message has been received, an OPEN Address Frame Received message is received or an OPEN Address Frame Received argument is included in the transition into this state, and the destination SAS address and connection rate of the received OPEN address frame are equal to the source SAS address and connection rate of the transmitted OPEN address frame.

This state shall send the following responses through the ECR to a source phy, 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, 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. 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.

7.15.9.2 Transition XL6:Open_Response_Wait to XL0:Idle

This transition shall occur after sending an Open Reject response.

7.15.9.3 Transition XL6:Open_Response_Wait to XL1:Request_Path

This transition shall occur after sending a Backoff Retry response, after releasing path resources.

7.15.9.4 Transition XL6:Open_Response_Wait to XL2:Request_Open

This transition shall occur after sending a Backoff Reverse Path response.

7.15.9.5 Transition XL6:Open_Response_Wait to XL7:Connected

This transition shall occur after sending an Open Accept response.

7.15.9.6 Transition XL6:Open_Response_Wait to XL9:Break

This transition shall occur after sending a Forward Break response.

7.15.9.7 Transition XL6:Open_Response_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.

7.15.10 XL7:Connected state

7.15.10.1 State description

This state provides a full-duplex circuit between two phys within an expander device.

This state shall send Transmit Dword messages to the XL transmitter to transmit all dwords received with Forward Dword indications.

This state shall send Forward Dword requests to the ECR containing each valid dword except BREAK and CLOSE primitives received with Dword Received messages.

If:

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

the expander phy shall:

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

If:

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

the expander phy shall:

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

If a CLOSE Received message is received, this state shall send a Forward Close request to the ECR with the argument from the CLOSE Received message.

If a BREAK Received message is received, this state shall send a Forward Break request to the ECR.

This state shall repeatedly send a Phy Status (Connection) response to the ECM.

7.15.10.2 Transition XL7:Connected to XL8:Close_Wait

This transition shall occur after receiving a Forward Close indication if a BREAK Received message has not been received.

7.15.10.3 Transition XL7:Connected to XL9:Break

This transition shall occur after sending a Forward Break request.

7.15.10.4 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.

7.15.11 XL8:Close_Wait state

7.15.11.1 State description

This state closes a connection and releases path resources.

Upon entry into this state, this state shall send a Transmit CLOSE message to the XL transmitter with the argument from the Forward Close indication, then shall request idle dwords be transmitted by repeatedly sending Transmit Idle Dword messages to the XL transmitter.

If a Dword Received message is received containing a valid dword except a BREAK or CLOSE primitive, this state shall send Forward Dword requests to the ECR containing that dword.

If:

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

the expander phy shall:

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

If:

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

the expander phy shall:

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

If a CLOSE Received message is received, this state shall release path resources and send a Forward Close request to the ECR with the argument from the CLOSE Received message.

If a BREAK Received message is received, this state shall send a Forward Break request to the ECR.

This state shall repeatedly send a Phy Status (Connection) response to the ECM.

7.15.11.2 Transition XL8:Close_Wait to XL0:Idle

This transition shall occur after sending a Forward Close request.

7.15.11.3 Transition XL8:Close_Wait to XL9:Break

This transition shall occur after sending a Forward Break request.

7.15.11.4 Transition XL8:Close_Wait to XL10:Break_Wait

This transition shall occur after receiving a Forward Break indication if a BREAK Received message has not been received.

7.15.12 XL9:Break state

7.15.12.1 State description

This state closes any connection and releases path resources.

This state shall send a Transmit BREAK message to the XL transmitter within a BREAK Response time limit (see 7.15.1) of receiving a BREAK Received message.

7.15.12.2 Transition XL9:Break to XL0:Idle

This transition shall occur after sending a Transmit BREAK message to the XL transmitter.

7.15.13 XL10:Break_Wait state

7.15.13.1 State description

This state closes any connection and releases path resources.

This state shall send a Transmit BREAK message to the XL transmitter. After transmitting the BREAK this state shall initialize and start the Break Timeout timer.

7.15.13.2 Transition XL10:Break_Wait to XL0:Idle

This transition shall occur after:

a) a BREAK Received message is received;
b) an OPEN_REJECT Received message is received, if this state was entered from the XL6:Open_Response_Wait state; ~~or~~
c) an CLOSE Received message is received, if this state was entered from the XL8:Close_Wait state; or
d) the Break Timeout timer expires.

7.16 SSP link layer

7.16.1 Opening an SSP connection

An SSP phy that accepts an OPEN address frame shall transmit at least one RRDY in that connection within 1 ms of transmitting an OPEN_ACCEPT. If the SSP phy is not able to grant credit, it shall respond with OPEN_REJECT (RETRY) and not accept the connection request.

7.16.2 Full duplex

SSP is a full duplex protocol. An SSP phy may receive an SSP frame or primitive in a connection while it 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).

When a connection is open and an SSP phy has no more SSP frames to transmit on that connection, it transmits a DONE to start closing the connection (see 8.2.2.3.5). The other direction may still be active, so the DONE may be followed by one or more of the following primitives: CREDIT_BLOCKED, RRDY, ACK, or NAK.

7.16.3 SSP frame transmission and reception

During an SSP connection, SSP frames are preceded by SOF and followed by EOF as shown in figure 9.

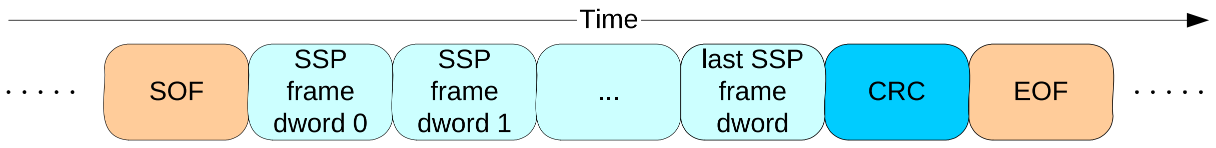


Figure 9 — SSP frame transmission

The last data dword after the SOF prior to the EOF always contains a CRC (see 7.5). The SSP link layer state machine checks that the frame is not too short and that the CRC is valid (see 7.16.7.7).

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

NOTE 9 - It is not required that frame recipients generate NAK (CRC ERROR) from invalid dwords and ERRORs (see 7.16.7.2).

The transport layer (see 9.2.4) either retries sending SSP frames that encounter a link layer error (e.g., are NAKed or create an ACK/NAK timeout), or aborts the SCSI command associated with the SSP frame that encountered a link layer error.

7.16.4 SSP flow control

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

SSP phys shall not increment credit past 255 frames.

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

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

If credit is zero, SSP phys that are going to be unable to provide 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 7.16.7).

If credit is nonzero, SSP phys that are going to be unable to provide additional credit for 1 ms, even if they receive frames per the existing credit, may transmit CREDIT_BLOCKED.

After sending CREDIT_BLOCKED, an SSP phy shall not transmit any additional RRDYs in the connection.

7.16.5 Interlocked frames

Table 16 shows which SSP frames shall be interlocked and which are non-interlocked.

Table 16 — SSP frame interlock requirements

SSP frame type	Interlock requirement
COMMAND	Interlocked
TASK	Interlocked
XFER_RDY	Interlocked
DATA	Non-interlocked
RESPONSE	Interlocked
See 9.2 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 credit is available. After transmitting an interlocked frame, an SSP phy shall not transmit another SSP frame until it has been acknowledged with ACK or NAK, even if credit is available.

Before transmitting a non-interlocked frame, an SSP phy shall wait for:

a) all non-interlocked frames with different tags; and
b) all interlocked frames;

to be acknowledged with ACK or NAK, even if credit is available.

After transmitting a non-interlocked frame, an SSP phy may transmit another non-interlocked frame with the same tag if credit is available. The phy shall not transmit:

a) a non-interlocked frame with a different tag; or
b) an interlocked frame;

until all SSP frames have been acknowledged with ACK or NAK, even if credit is available.

Interlocking does not prevent transmitting and receiving interlocked frames simultaneously (e.g., an SSP initiator phy could be transmitting a COMMAND frame while receiving XFER_RDY, DATA, or RESPONSE frames for a different command).

An SSP phy may transmit primitives responding to traffic it is receiving (e.g., an ACK or NAK to acknowledge an SSP frame, an RRDY to grant more receive credit, or a CREDIT_BLOCKED to specify that no more RRDYs are going to be transmitted in the connection) while waiting for an interlocked frame it transmitted to be acknowledged. These primitives may also be interspersed within an SSP frame.

Figure 10 shows an example of interlocked frame transmission.

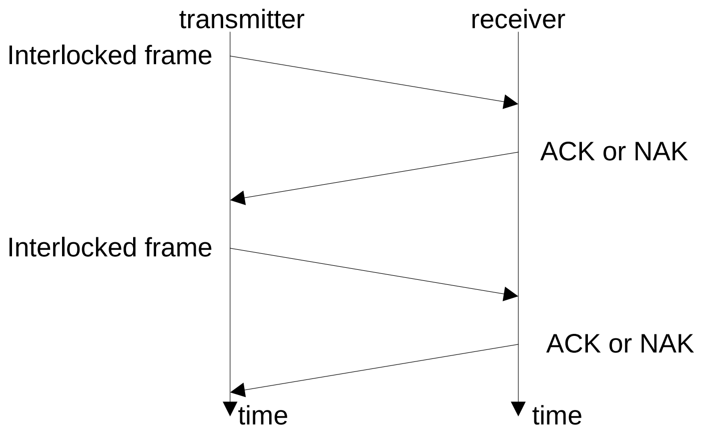


Figure 10 — Interlocked frames

Figure 11 shows an example of non-interlocked frame transmission with the same tags.

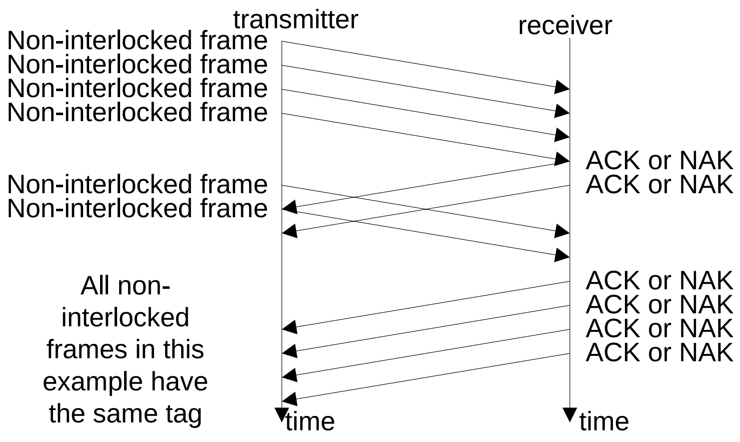


Figure 11 — Non-interlocked frames with the same tag

Figure 12 shows an example of non-interlocked frame transmission with different tags.

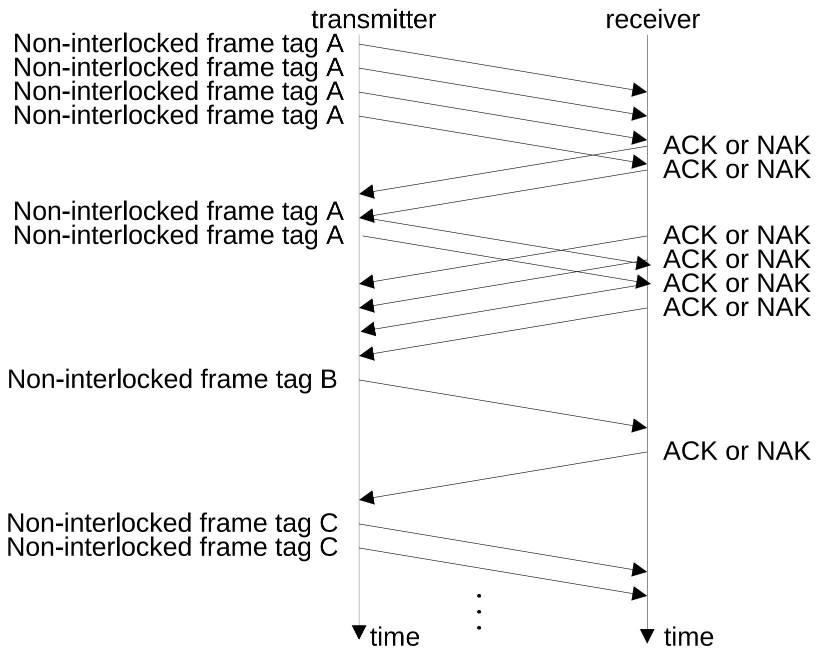

transmitter
receiver
Non-interlocked frame tag A
Non-interlocked frame tag A
Non-interlocked frame tag A
Non-interlocked frame tag A
ACK or NAK
ACK or NAK
Non-interlocked frame tag A
Non-interlocked frame tag A
ACK or NAK
ACK or NAK
ACK or NAK
ACK or NAK
Non-interlocked frame tag B
ACK or NAK
Non-interlocked frame tag C
Non-interlocked frame tag C
time
time

Figure 12 — Non-interlocked frames with different tags

7.16.6 Closing an SSP connection

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

a) DONE (NORMAL) specifies normal completion; the transmitter has no more SSP frames to transmit;
b) DONE (CREDIT TIMEOUT) specifies that the transmitter still has SSP frames to transmit but did not receive an RRDY granting frame credit within 1 ms, or the transmitter has received a CREDIT_BLOCKED and has consumed all RRDYs received; and
c) 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 in 1 ms unless the recipient replies with DONE and the connection is closed.

If the transmitter has no more SSP frames to transmit and receives a CREDIT_BLOCKED, 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 7.16.7.5).~~

After transmitting DONE, the transmitting phy shall not transmit any more SSP frames during this connection. However, the phy may transmit ACK, NAK, RRDY, and CREDIT_BLOCKED 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 CLOSE (NORMAL) (see 7.12.7).

Figure 13 shows the sequence for a closing an SSP connection.

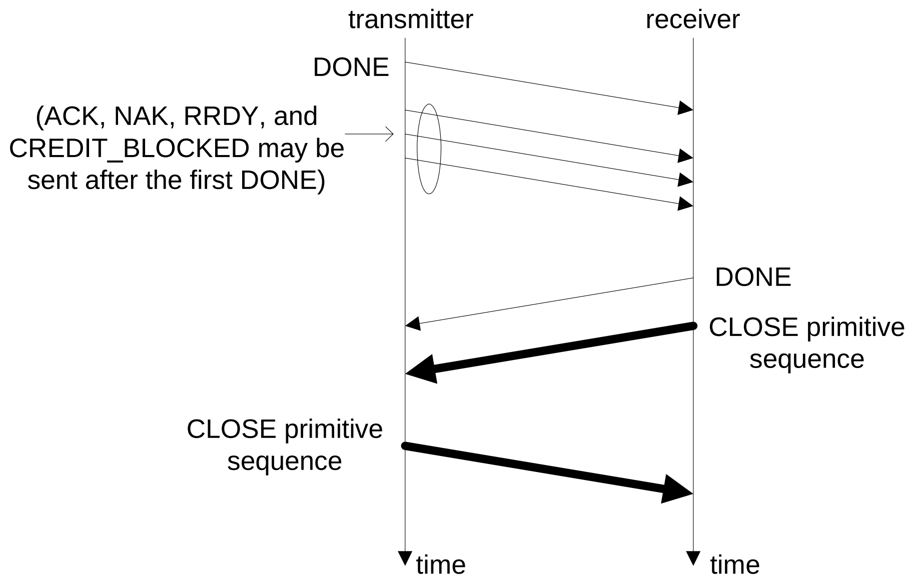


Figure 13 — Closing an SSP connection example

SSP phys maintain the timeout timers listed in table 17.

Table 17 — SSP timeout timers

Timer	Initial value	Description
Done Timeout timer	1 ms	Maximum time from transmitting DONE or receiving a frame after transmitting DONE until receiving a frame or DONE.

SSP phys honor the time limits listed in table 18.

Table 18 — SSP time limits

Time limit	Value	Description
Done Response time limit	0,5 ms	Maximum time from receiving a DONE or transmitting a frame after receiving DONE until transmitting DONE.

7.16.7 SSP (link layer for SSP phys) state machines

7.16.7.1 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 7.16.7.3);
b) SSP_TCM (transmit frame credit monitor) state machine (see 7.16.7.4);
c) SSP_D (DONE control) state machine (see 7.16.7.5);
d) SSP_TF (transmit frame control) state machine (see 7.16.7.6);
e) SSP_RF (receive frame control) state machine (see 7.16.7.7);
f) SSP_RCM (receive frame credit monitor) state machine (see 7.16.7.8);
g) SSP_RIM (receive interlocked frame monitor) state machine (see 7.16.7.9);
h) SSP_TC (transmit credit control) state machine (see 7.16.7.10); and
i) SSP_TAN (transmit ACK/NAK control) state machine (see 7.16.7.11).

All the SSP state machines shall start after receiving an Enable Disable SSP (Enable) message from the SL state machines (see 7.14).

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; or
c) receiving a Request Break message from the SSP_D state machine indicating that a BREAK has been transmitted.

If a state machine consists of multiple states 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 19.

Table 19 — SSP link layer timers

Timer	**Initial value**
ACK/NAK Timeout timer	1 ms
DONE Timeout timer	1 ms
Credit Timeout timer	1 ms

The SSP state machines shall comply with the time limits listed in table 20.

Table 20 — SSP time limits

Time limit	**Value**	**Description**
Done Response time limit	0,5 ms	Maximum time from receiving DONE or transmitting a frame until transmitting DONE.

Figure 14 shows the SSP state machines and states related to frame transmission.

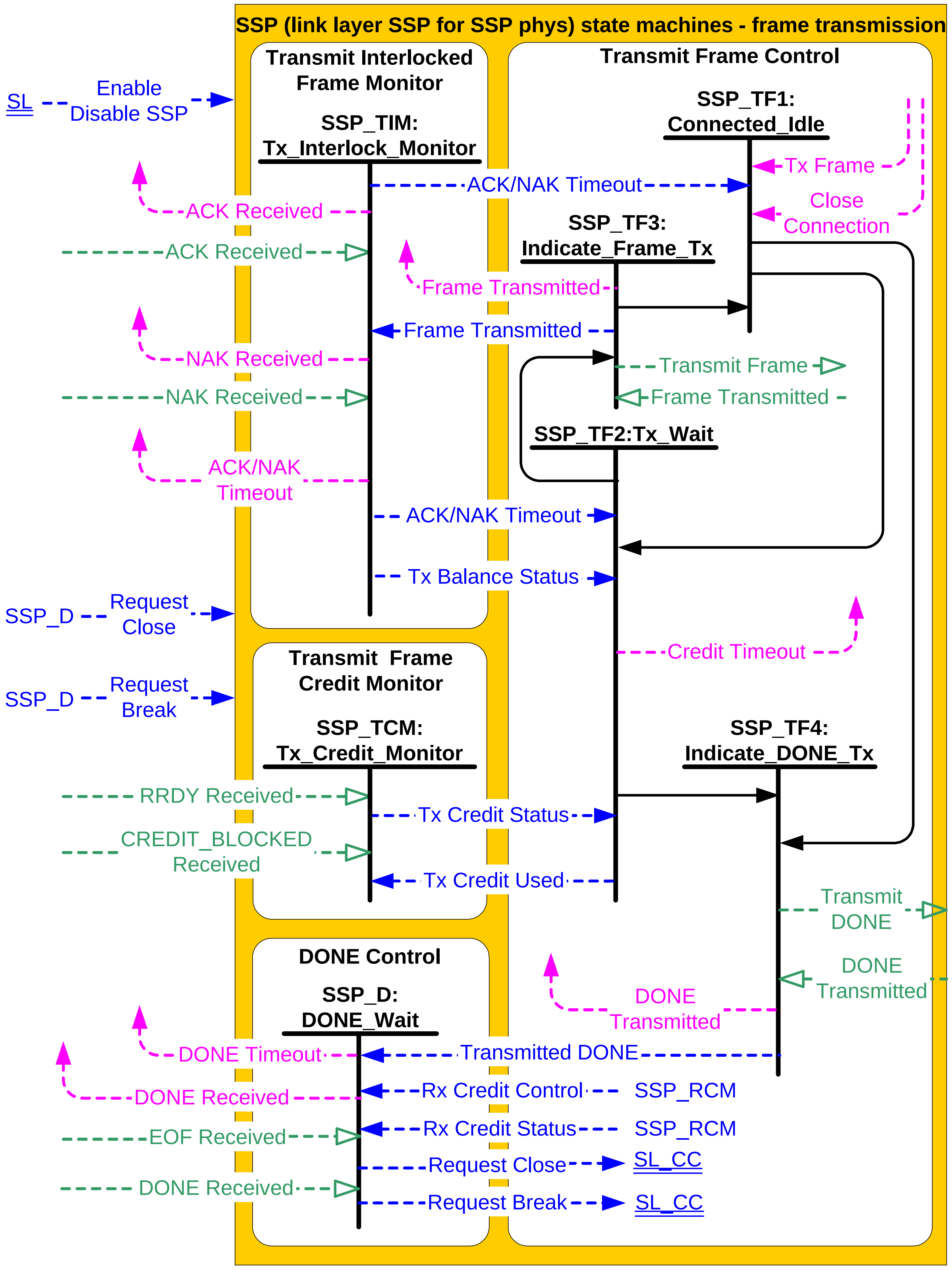


Figure 14 — SSP (link layer for SSP phys) state machines (part 1 - frame transmission)

Figure 15 shows the SSP state machines and states related to frame reception.

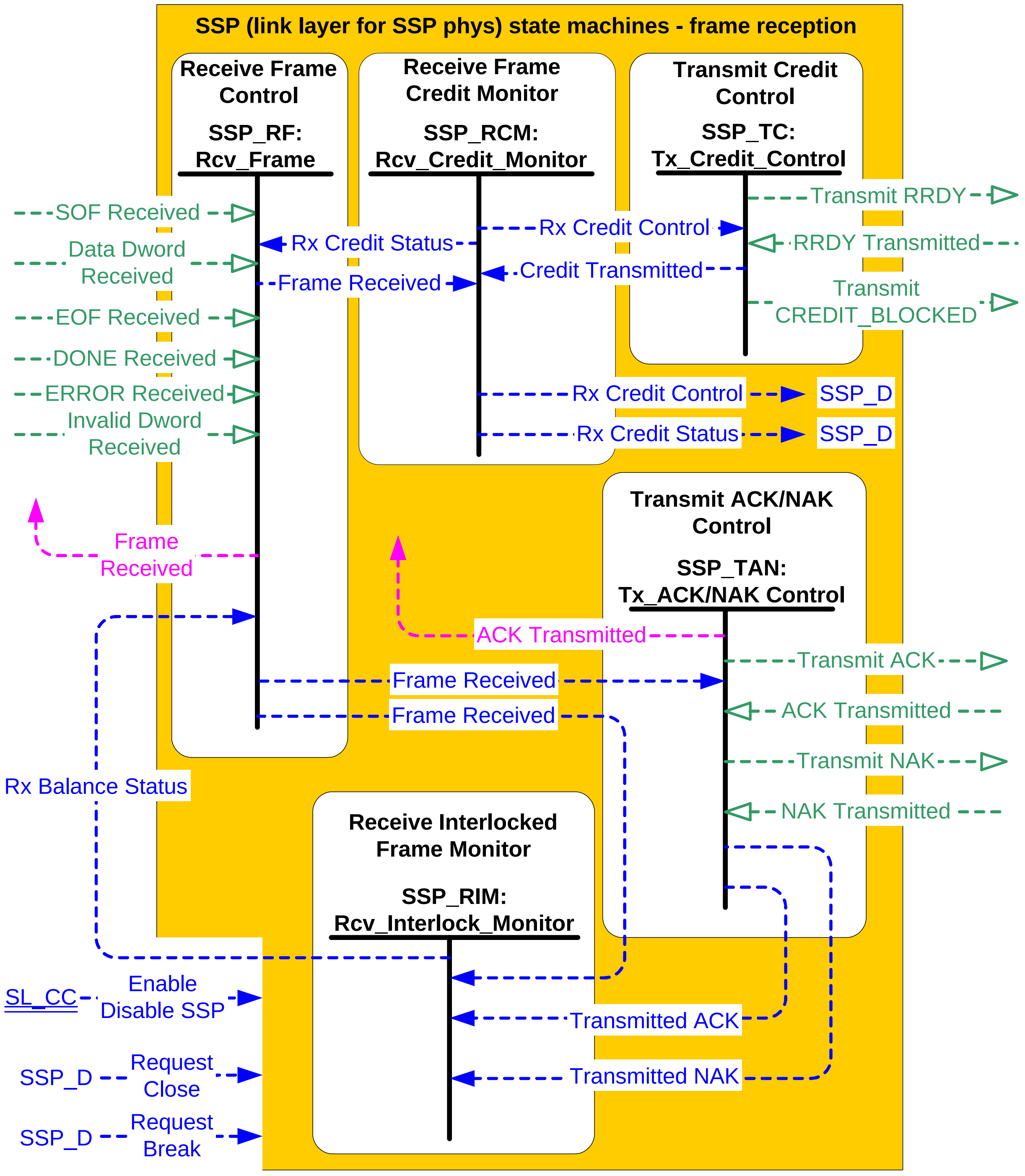


Figure 15 — SSP (link layer for SSP phys) state machines (part 2 - frame reception)

7.16.7.2 SSP transmitter and receiver

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

a) Transmit RRDY with an argument indicating the specific type (e.g., Transmit RRDY (Normal));
b) Transmit CREDIT_BLOCKED;
c) Transmit ACK;

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

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

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

When 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 6.9.2):

a) ACK Received;
b) NAK Received;
c) RRDY Received;
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) ERROR Received; and
j) Invalid Dword Received.

The SSP receiver shall ignore all other dwords.

7.16.7.3 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 and monitors the number of ACKs and NAKs received. This state machine ensures that an ACK or NAK is received for each frame transmitted and indicates a timeout if they are not.

The Frame Transmitted message shall be used by this state machine to count the number of frames transmitted.

When the number of Frame Transmitted messages received equals the number of ACK Received and NAK Received messages received then the ACK/NAK count is balanced and this state machine shall send the Tx Balance Status (Balanced) message to the SSP_TF2:Tx_Wait state. When the number of Frame Transmitted messages received does not equal the number of ACK Received and NAK Received messages received then this 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.

If the ACK/NAK count is not balanced and an ACK Received message is received this state machine shall:

a) use the ACK Received message to count the number of ACKs and NAKs received; and
b) send an ACK Received confirmation to the port layer each time the ACK Received message is received.

If the ACK/NAK count is not balanced and an NAK Received message is received this state machine shall:

a) use the NAK Received message to count the number of ACKs and NAKs received; and
b) send an NAK Received confirmation to the port layer each time the NAK Received message is received.

If the ACK/NAK count is balanced, 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 re-initialized each time an ACK or NAK is counted. If the ACK/NAK Timeout timer expires, 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 shall be set to the number of ACKs and NAKs received.

7.16.7.4 SSP_TCM (transmit frame credit monitor) state machine

The SSP_TCM state machine’s function is to ensure that transmit 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 frame credits available. This state machine shall add one transmit frame credit for each RRDY Received message received and subtract one transmit frame credit for each Tx Credit Used message received.

The CREDIT_BLOCKED Received message indicates that transmit 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.

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

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

When transmit frame credit is not available and transmit frame credit is blocked, this state machine shall send the Tx Credit Status (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 frame credit to not available and transmit frame credit shall be set to not blocked.

7.16.7.5 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 DONEs may be transmitted and received in any order.

If the DONE Received message has been received before the Transmitted DONE message is received, this state machine shall send the Request Close message to the SL_CC state machine (see 7.14) and all the SSP state machines after receiving the Transmitted DONE message.

If a DONE Received message, the Transmitted DONE (Normal) message, or the Transmitted DONE (Credit Timeout) message has not been received and the Rx Credit Status (Extended) message or the Rx Credit Control (Blocked) message has been received, then this state shall initialize and start the DONE Timeout timer after receiving the Transmitted DONE (Normal) message or the Transmitted DONE (Credit Timeout) message.

If the DONE Received message has not been received and the Transmitted DONE (Normal) message or the Transmitted DONE (Credit Timeout) message has been received, this state machine shall initialize and start the DONE Timeout timer each time:

a) the Rx Credit Status (Extended) message is received; or
b) the Rx Credit Control (Blocked) message is received.

If the Transmitted DONE (Normal) message or the Transmitted DONE (Credit Timeout) message has been received, the DONE Timeout timer shall be reinitialized each time the EOF Received message is received.

If the Transmitted DONE (Normal) message or the Transmitted DONE (Credit Timeout) message has been received, the DONE Timeout timer shall be stopped after:

a) the Rx Credit Status (Exhausted) message is received; and
b) the Rx Credit Control (Blocked) message has not been received.

NOTE 10 - Stopping the timer ensures that, if credit remains exhausted long enough that the Credit Timeout timer of the other phy in the connection expires, the other phy is able to transmit a DONE (CREDIT TIMEOUT).

If the DONE Received message has not been received and the Transmitted DONE (ACK/NAK Timeout) message has been received:

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 is received.

If the DONE Received message is received before the DONE Timeout timer expires, this state machine shall send the Request Close message to the SL_CC state machine and all the SSP state machines.

If the DONE Received message is not received before the DONE Timeout timer expires, this state machine shall send a Request Break message to the SL_CC state machine and all the SSP state machines.

Any time a DONE Received message is received 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 (Close Connection) and DONE Received (Credit Timeout)) may be used by the application layer to decide when to reuse tags.

NOTE 11 - 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.

For 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 BREAK.

Since the timers may expire at slightly different times (e.g., due to timer resolution differences), the DONE (ACK/NAK TIMEOUT) may be transmitted before, concurrently with, or after the BREAK. Nevertheless, the phys handle the link layer error (i.e., the ACK/NAK timeout or the DONE timeout) the same way (see 9.2.4.5).

7.16.7.6 SSP_TF (transmit frame control) state machine

7.16.7.6.1 SSP_TF state machine overview

The SSP_TF state machine's function is to control when the SSP transmitter transmits SOF, frame dwords, EOF, and DONE. This state machine consists of the following states:

a) SSP_TF1:Connected_Idle (see 7.16.7.6.2)(initial state);
b) SSP_TF2:Tx_Wait (see 7.16.7.6.3);
c) SSP_TF3:~~Indicate_Frame_Tx~~Tx_Frame (see 7.16.7.6.4); and
d) SSP_TF4:~~Indicate_DONE_Tx~~Tx_DONE (see 7.16.7.6.5).

Editor's Note 1: change TF3 and TF4 state names in figure

7.16.7.6.2 SSP_TF1:Connected_Idle state

7.16.7.6.2.1 State description

This state waits for a request to transmit a frame or to close the connection.

7.16.7.6.2.2 Transition SSP_TF1:Connected_Idle to SSP_TF2:Tx_Wait

This transition shall occur after a Tx Frame request is received or a Close Connection request is received.

If a Tx Frame (Balance Required) request was received this transition shall include a Transmit Frame Balance Required argument.

If a Tx Frame (Balance Not Required) request was received this transition shall include a Transmit Frame Balance Not Required argument.

If a Close Connection request was received this transition shall include a Close Connection argument.

7.16.7.6.2.3 Transition SSP_TF1:Connected_Idle to SSP_TF4:~~Indicate_DONE_Tx~~Tx_Done

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

7.16.7.6.3 SSP_TF2:Tx_Wait state

7.16.7.6.3.1 State description

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

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

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

7.16.7.6.3.2 Transition SSP_TF2:Tx_Wait to SSP_TF3:~~Indicate_Frame_Tx~~Tx_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:

a) the last Tx Balance Status message received had an argument of Balanced; and
b) the last Tx Credit Status message received had an argument of 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 after sending a Tx Credit Used message to the SSP_TCM state machine.

7.16.7.6.3.3 Transition SSP_TF2:Tx_Wait to SSP_TF4:~~Indicate_DONE_Tx~~Tx_Done

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

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

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

This transition shall occur 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.

7.16.7.6.4 SSP_TF3:~~Indicate_Frame_Tx~~Tx_Frame state

7.16.7.6.4.1 State description

This state shall request a frame transmission by sending a Transmit Frame message to the SSP transmitter. Each time a Transmit Frame message is sent to the SSP transmitter, one SSP frame (i.e., SOF, frame contents, and EOF) is transmitted.

In this state receiving a Frame Transmitted message indicates that the frame has been transmitted.

7.16.7.6.4.2 Transition SSP_TF3:~~Indicate_Frame_Tx~~Tx_Frame to SSP_TF1:Connected_Idle

This transition shall occur after:

a) receiving a Frame Transmitted message;
b) sending an Frame Transmitted message to the SSP_TIM state machine; and
c) sending a Frame Transmitted confirmation to the port layer.

7.16.7.6.5 SSP_TF4:~~Indicate_DONE_Tx~~Tx_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 (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 Transmit DONE (Credit Timeout) message if this state was entered from the SSP_TF2:Tx_Wait state with an argument of Credit Timeout.

If the SSP state machines have received a DONE Received message, this state shall send the Transmit DONE message within a DONE Response time limit.

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;
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.

7.16.7.7 SSP_RF (receive frame control) state machine

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

This state machine:

a) checks the frame to determine if the frame should be accepted or discarded;

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

If this state receives a subsequent SOF Received message after receiving an SOF Received message but before receiving an EOF Received message (i.e., SOF, data dwords, SOF, data dwords, and EOF instead of SOF, data dwords, EOF, SOF, data dwords, and EOF) , then this state shall discard the Data Dword Received messages received before the subsequent SOF Received message.

This state shall discard the frame if:

a) this state receives more than 263 Data Dword Received messages after an SOF Received message and before an EOF Received message;
b) this state receives fewer than 7 Data Dword Received messages after an SOF Received message and before an EOF Received message,
c) this state receives an Rx Credit Status (Credit Exhausted) message; or
d) this state receives 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, then this state machine shall:

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_TAN1:Idle state.

If the frame is not discarded and the frame CRC is bad, this state machine shall:

a) send a Frame Received message to the SSP_RCM state machine;
b) send a Frame Received message to the SSP_RIM state machine; and
c) send a Frame Received (Unsuccessful) message to the SSP_TAN1:Idle state.

If the frame is not discarded and the frame CRC is good, this state machine shall send a Frame Received (Successful) message to the SSP_TAN1:Idle state and:

a) send a Frame Received message to the SSP_RCM state machine;
b) send a Frame Received message to the SSP_RIM state machine; and
c) send a Frame Received (Successful) message to the SSP_TAN1:Idle state, and:
    A) if the last Rx Balance Status message received had an argument of Balanced, send a Frame Received (ACK/NAK Balanced) confirmation to the port layer; or
    B) if the last Rx Balance Status message received had an argument of Not Balanced, send a Frame Received (ACK/NAK Not Balanced) confirmation to the port layer.

7.16.7.8 SSP_RCM (receive frame credit monitor) state machine

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

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

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

This state machine may send the Rx Credit Control (Blocked) message to the SSP_TC state machine and the SSP_D state machine 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 5 frames the maximum number of Rx Credit Control requests with the Available argument outstanding is 5).

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

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

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

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

7.16.7.9 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. This state machine consists of one state.

This state machine monitors the number of frames received versus the number of ACKs and NAKs transmitted.

This state machine shall use the ACK Transmitted message and the NAK Transmitted message to keep track of the number of ACKs and NAKs transmitted. This state machine shall use the Frame Received message to keep a track of the number of frames received.

Any time the number of the ACK Transmitted messages and the number of NAK Transmitted messages received equals the number of Frame Received messages received this state machine shall send an Rx Balance Status (Balanced) message to the SSP_RF state machine.

Any time the number of the ACK Transmitted messages and the number of NAK Transmitted messages received does not equal the number of Frame Received messages received this state machine shall send an Rx Balance Status (Not Balanced) message to the SSP_RF state machine.

When the SL state machines send the Enable Disable SSP (Enable) message, the number of the ACKs and NAKs transmitted shall be set to the number of frames received.

7.16.7.10 SSP_TC (transmit credit control) state machine

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

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

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

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

7.16.7.11 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.

Any time this state machine receives a Frame Received (Successful) message it shall send a Transmit ACK message to the SSP transmitter.

Any time this state machine receives a Frame Received (Unsuccessful) message it 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.

Any time this state machine receives an ACK Transmitted message it shall:

a) send a Transmitted ACK message to the SSP_RIM state machine; and
b) send an ACK Transmitted confirmation to the port layer.

Any time this state receives a NAK Transmitted argument it shall send a Transmitted NAK message to the SSP_RIM state machine.