SRIO VIP Compliance

V1.0

NOTICE

THIS DOCUMENT IS PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS DOCUMENT COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THE DOCUMENT. MOBIVEIL, INC. OR IT’s LICENSOR MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE TECHNOLOGY DESCRIBED IN THIS DOCUMENT AT ANY TIME.

Revision History

Revision	Date	By	Change
0.1	04/26/2013	MV	Initial version.

Glossary

IO	Input Output
CRF	Critical Request Flow
DS	Data Streaming
LFC	Logical Flow Control

Contents

Tables

Figures

1 INTRODUCTION

1.1 Purpose

This document contains the SRIO VIP compliance checklists adhering to Rev. 1.3, Rev. 2.x, and 3.x of the RapidIO Interconnect Specification.

1.2 References

- *"Serial RapidIO specification 1.3"*
- *"Serial RapidIO specification 2.x"*
- *"Serial RapidIO specification 3.x"*
- *"SRIO VIP Microarchitecture"*

2 OVERVIEW

SRIO VIP Compliance Checklist provides the list of RapidIO specification features to be verified. The Item No contains a number/letter combination which uniquely identifies the checklist item. A text description of the aspect of the RapidIO specification is kept in the Compliance Item column. A reference to the specific section of the specification which contains the requirement occurs in the Specification Reference column. The test-case name column contains the specific testcase, for the given compliance item. Some parts of the specification are covered default by all the cases, but still require check list items to be assigned to them. For those checklist items, the testcase column left as empty. One testcase may cover more than 1 compliance item.

3 Part1 - Input/Output Logical Specification compliance

3.1 Functional/Performance Features Compliance

Table 1: Functional/Performance Features Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	Read-modify-write atomic operations are useful for synchronization between processors or other system elements.	Part1, Sec 1.3.1	srio_ll_atomic_req_trans.sv
2	The RapidIO architecture supports 50- and 66-bit addresses as well as 34-bit local addresses for smaller systems.	Part1, Sec 1.3.1	
3	Multiple transactions must be allowed concurrently in the system, otherwise a majority of the potential system throughput is wasted.	Part1, Sec 1.3.3	srio_ll_io_concurrent_trans.sv
4	The protocols handle out-of-order packet transmission and reception.	Part1, Sec 1.3.2	srio_ll_io_out_of_order_trans.sv
	The destination processing element shall guarantee that all outstanding non-coherent operations from that source are completed before servicing a subsequent non-coherent request from that source.	Part1, Sec 2.3.2.1	
	Transaction IDs may also be used to indicate sequence information if ordered reception of transactions is required by the destination processing element and the interconnect fabric can reorder packets.	Part1, Sec 3.1	
	The receiving device must accept and not complete the subsequent out-of-order requests until the missing transactions in the sequence have been received and completed.	Part1, Sec 3.1	

3.2 Operation Ordering Compliance

Table 2: Operation Ordering Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	A transaction request flow is defined as an ordered sequence of non-maintenance request transactions from a given source (as indicated by the source identifier) to a given destination (as indicated by the transaction destination identifier), where a maintenance request is a special system support request.	Part1, Sec 2.3.1	srio_ll_io_operation_ordering.sv
2	There may be multiple transaction request flows between a given source and destination pair. When multiple flows exist between a source and destination pair, the flows are distinguished by a flow indicator (flowID).		
3	RapidIO allows multiple transaction request flows between any source and destination pair		
4	When multiple transaction request flows exist between a given source and destination pair, transactions of a higher priority flow may pass transactions of a lower priority flow, but transactions of a lower priority flow may not pass transactions of a higher priority flow.		
5	Write request transactions in a transaction request flow shall be completed at the logical layer of the destination in the same order that the transactions were delivered to the logical layer of the destination.		
6	A read request transaction with source A and destination B shall force the completion at the logical layer of B of all write requests in the same transaction request flow that were received by the logical layer of B before the read request transaction.		

3.3 Basic Functionality Compliance

Table 3: Basic Functionality Compliance

Item No	Compliance Item	Spec reference	Testcase Name
1	A request transaction sent into the system is marked with a transaction ID that is unique for each requestor and responder processing element pair. This transaction ID allows a response to be easily matched to the original request when it is returned to the requestor.	Part1, Sec 3.1	srio_ll_request_test.sv
2	All data payloads that are less than 8 bytes shall be padded and have their bytes aligned to their proper byte position within the double-word.	Part1, Sec 3.3	
3	**NREAD - Read Specified address**	Part1, Sec 4.1	srio_ll_nread_req_test.sv
3a	Destination Operation CAR Bit[16] - PE can support a read operation	Part1, Sec 5.4.8	
3b	Type 2 packets never contain a data payload.	Part1, Sec 4.1.5	
3c	The type 13 packet format returns status, data (if required), and the requestor's transaction ID.	Part1, Sec 4.2.3	
3d	Verify all possible wdptr, rdsize	Part1, Sec 4.1.2	
	Verify payload size 1 to 256 bytes		
3e	For a packet whose length, exclusive of CRC, is 80 bytes or less, a single CRC is appended at the end of the logical fields. Unpadded Packet of Length 80 Bytes or Less Padded Packet of Length 80 Bytes or Less	Part6, Sec 2.4.1	
	For packets whose length, exclusive of CRC, is greater than 80 bytes, a CRC is added after the first 80 bytes and a second CRC is appended at the end of the logical layer fields. Unpadded Packet of Length Greater than 80 Bytes Padded Packet of Length Greater than 80 Bytes		
	If the CRC appended to the end of the logical layer fields does not cause the end of the resulting packet to align to a 32-bit boundary, a two byte pad of all logic 0s is postpended to the packet		
	The pad of logic 0s allows the CRC check to always be done at the 32-bit boundary. A corrupt pad may or may not cause a CRC error to be detected, depending upon the implementation.		
3f	Up to 2^n outstanding unacknowledged request and/or response packets. where N is the number of bits in the ackID field.	Part6, Sec 6.6.2	

Item No	Compliance Item	Spec reference	Testcase Name
	To eliminate the ambiguity between 0 and 2N outstanding packets, a maximum of 2N-1 outstanding unacknowledged packets shall be allowed at any one time.		
3g	Cover the below possible combination when CRF is RSVD = 0 PRIO sets packet priority as follows: 00 - lowest priority 01 - medium priority 10 - high priority 11- highest priority	Part6, Sec 2.2	
	Cover the below possible combination when CRF is supported PRIO||CRF sets packet priority: 00 0 - lowest priority 00 1 - critical flow lowest priority 01 0 - medium priority 01 1 - critical flow medium priority 10 0 - high priority 10 1 - critical flow high priority 11 0 - highest priority 11 1 - critical flow high priority		
4	**ATOMIC (Read-Modify-Write)**		srio_ll_atomic_req_trans.sv
4a	Destination Operations CAR Bit[22] to Bit[28]- PE can support an a atomic operation.	Part1, Sec 5.4.8	
4b	The allowed specified data sizes are one word (4 bytes), one half-word (2 bytes) or one byte, with the size of the transaction specified in the same way as for an NWRITE transaction. Double-word (8-byte) and 3, 5, 6, and 7 byte ATOMIC transactions may not be specified.	Part1, Sec 3.3.4	
	The atomic operation is a combination read and write operation.		
	The destination reads the data at the specified address, returns the read data to the requestor, performs the required operation to the data, and then writes the modified data back to the specified address without allowing any intervening activity to that address		

Item No	Compliance Item	Spec reference	Testcase Name
4c	ATOMIC set - Read-write 1s to specified address	Part1, Sec 4.1	
	ATOMIC Clear - Read-write 1s to specified address		
	ATOMIC increment -Read-increment-write to specified address		
	ATROMIC decrement- Read-decrement-write to specified address		
	ATOMIC test-and-swap Read-test=0-swap-write to specified address . Read and return the data, compare to 0, write with supplied data if compare is true		
	ATOMIC swap Read-write to specified address. Read and return the data, unconditionally write with supplied data.		
	ATOMIC compare-and-swap Read-test=first data-write second data to specified address. Read and return the data, if the read data is equal to the first 8 bytes of data payload, write the second 8 bytes of data to the memory location.		
4d	The data payload size for the response to an ATOMIC transaction is 8 bytes.	Part1, Sec 4.1.5	
	Double-word (8-byte), 3, 5, 6, and 7 byte ATOMIC transactions are not allowed.		
4e	The ATOMIC test-and-swap transaction is limited to one double-word (8 bytes) of data payload.	Part1, Sec 4.1.7	
	Double-word (8-byte) and 3, 5, 6, and 7 byte ATOMIC test-and-swap transactions are not allowed.		
	The ATOMIC compare-and-swap operation is different from the other ATOMIC operations in that it requires two double-words (16 bytes) of data payload.		
4f	Up to 2^n outstanding unacknowledged request and/or response packets. where N is the number of bits in the ackID field.	Part6, Sec 6.6.2	
	To eliminate the ambiguity between 0 and 2N outstanding packets, a maximum of 2N-1 outstanding unacknowledged packets shall be allowed at any one time.		
4g	Cover the below possible combination when CRF is RSVD = 0 PRIO sets packet priority as follows: 00 - lowest priority 01 - medium priority 10 - high priority 11- highest priority	Part6, Sec 2.2	

Item No	Compliance Item	Spec reference	Testcase Name
4t	Cover the below possible combination when CRF is supported PRIO||CRF sets packet priority: 00 0 - lowest priority 00 1 - critical flow lowest priority 01 0 - medium priority 01 1 - critical flow medium priority 10 0 - high priority 10 1 - critical flow high priority 11 0 - highest priority 11 1 - critical flow high priority	Part6, Sec 2.2	
5	**NWRITE- Write specified address**	Part1, Sec 4.1	srio_ll_nwrite_req_test.sv
5a	Destination Operations CAR Bit[17] - PE can support a write operation	Part1, Sec 5.4.8	
5b	The NWRITE transaction allows multiple double-word, word, half-word and byte writes with properly padded and aligned (to the 8-byte boundary) data payload.	Part1, Sec 3.3.2	
	NWRITE transaction do not receive response		
5c	NWRITE transaction use the type 5 format	Part1, Sec 4.1.7	
	The data payload may not exceed that size but may be smaller if desired		
	Type 5 packets always contain a data payload		
	NWRITE request packets do not require a response. Therefore, the transaction ID (srcTID) field for a NWRITE request is undefined and may have an arbitrary value.		
5d	Verify all possible wdptr, wrsize	Part1, Sec 4.1.2	
	Verify payload size 1 to 256 bytes		
5e	Non-contiguous and unaligned writes are not supported. It is the requestor's responsibility to break up a write operation into multiple transactions if the block is not aligned	Part1, Sec 3.3.2	
5f	For a packet whose length, exclusive of CRC, is 80 bytes or less, a single CRC is appended at the end of the logical fields. Unpadded Packet of Length 80 Bytes or Less Padded Packet of Length 80 Bytes or Less	Part6, Sec 2.4.1	
	For packets whose length, exclusive of CRC, is greater than 80 bytes, a CRC is added after the first 80 bytes and a second CRC is appended at the end of the logical layer fields. Unpadded Packet of Length Greater than 80 Bytes Padded Packet of Length Greater than 80 Bytes		
	If the CRC appended to the end of the logical layer fields does not cause the end of the resulting packet to align to a 32-bit boundary, a two byte pad of all logic 0s is postpended to the packet		

Item No	Compliance Item	Spec reference	Testcase Name
	The pad of logic 0s allows the CRC check to always be done at the 32-bit boundary. A corrupt pad may or may not cause a CRC error to be detected, depending upon the implementation.		
5g	Up to 2^n outstanding unacknowledged request and/or response packets. where N is the number of bits in the ackID field.	Part6, Sec 6.6.2	
	To eliminate the ambiguity between 0 and 2N outstanding packets, a maximum of 2N-1 outstanding unacknowledged packets shall be allowed at any one time.		
5h	Cover the below possible combination when CRF is RSVD = 0 PRIO sets packet priority as follows: 00 - lowest priority 01 - medium priority 10 - high priority 11- highest priority	Part6, Sec 2.2	
	Cover the below possible combination when CRF is supported PRIO||CRF sets packet priority: 00 0 - lowest priority 00 1 - critical flow lowest priority 01 0 - medium priority 01 1 - critical flow medium priority 10 0 - high priority 10 1 - critical flow high priority 11 0 - highest priority 11 1 - critical flow high priority		
6	**NWRITE_R - Write specified address, notify source of completion**	Part1, Sec 4.1	srio_ll_nwrite_r_req_test.sv
6a	Destination Operations CAR Bit[19] - PE can support a write-with-response operation	Part1, Sec 5.4.8	
6a	The write-with-response operation, consisting of the NWRITE_R and RESPONSE transactions (typically a DONE response)	Part1, Sec 3.3.3	
6b	NWRITE_R transaction use the type 5 format	Part1, Sec 4.1.7	
	Type 5 packets always contain a data payload.		
	The data payload may not exceed that size but may be smaller if desired.		
6c	Verify all possible wdptr, wrsize	Part1, Sec 4.1.2	
	Verify payload size 1 to 256 bytes		
6d	For a packet whose length, exclusive of CRC, is 80 bytes or less, a single CRC is appended at the end of the logical fields. Unpadded Packet of Length 80 Bytes or Less Padded Packet of Length 80 Bytes or Less		

Item No	Compliance Item	Spec reference	Testcase Name
	For packets whose length, exclusive of CRC, is greater than 80 bytes, a CRC is added after the first 80 bytes and a second CRC is appended at the end of the logical layer fields. Unpadded Packet of Length Greater than 80 Bytes Padded Packet of Length Greater than 80 Bytes .	Part6, Sec 2.4.1	
	If the CRC appended to the end of the logical layer fields does not cause the end of the resulting packet to align to a 32-bit boundary, a two byte pad of all logic 0s is postpended to the packet		
	The pad of logic 0s allows the CRC check to always be done at the 32-bit boundary. A corrupt pad may or may not cause a CRC error to be detected, depending upon the implementation.		
6e	Up to 2^n outstanding unacknowledged request and/or response packets. where N is the number of bits in the ackID field.	Part6, Sec 6.6.2	
	To eliminate the ambiguity between 0 and 2N outstanding packets, a maximum of 2N-1 outstanding unacknowledged packets shall be allowed at any one time.		
6f	Cover the below possible combination when CRF is RSVD = 0 PRIO sets packet priority as follows: 00 - lowest priority 01 - medium priority 10 - high priority 11- highest priority	Part6, Sec 2.2	
	Cover the below possible combination when CRF is supported PRIO||CRF sets packet priority: 00 0 - lowest priority 00 1 - critical flow lowest priority 01 0 - medium priority 01 1 - critical flow medium priority 10 0 - high priority 10 1 - critical flow high priority 11 0 - highest priority 11 1 - critical flow high priority		
7	**SWRITE - Write Specified address**	Part1, Sec 4.1	srio_ll_swrite_req_test.sv
7a	Destination Operations CAR Bit[18] - PE can support a streaming-write operation	Part1, Sec 5.4.8	
7b	The SWRITE transaction is a double-word-only transaction	Part1, Sec 3.3.2	
	SWRITE transaction do not receive responses		

Item No	Compliance Item	Spec reference	Testcase Name
7c	SWRITE transaction is the only transaction to use format type 6,	Part1, Sec 4.1.8	
	The type 6 packet is a special-purpose type that always contains data		
	Type 6 transactions may contain any number of double-words up to the maximum (256)		
	There are also no size or transaction ID fields.		
7d	For a packet whose length, exclusive of CRC, is 80 bytes or less, a single CRC is appended at the end of the logical fields. Unpadded Packet of Length 80 Bytes or Less Padded Packet of Length 80 Bytes or Less	Part6, Sec 2.4.1	
	For packets whose length, exclusive of CRC, is greater than 80 bytes, a CRC is added after the first 80 bytes and a second CRC is appended at the end of the logical layer fields. Unpadded Packet of Length Greater than 80 Bytes Padded Packet of Length Greater than 80 Bytes.		
	If the CRC appended to the end of the logical layer fields does not cause the end of the resulting packet to align to a 32-bit boundary, a two byte pad of all logic 0s is postpended to the packet		
	The pad of logic 0s allows the CRC check to always be done at the 32-bit boundary. A corrupt pad may or may not cause a CRC error to be detected, depending upon the implementation.		
7e	Up to 2^n outstanding unacknowledged request and/or response packets. where N is the number of bits in the ackID field.	Part6, Sec 6.6.2	
	To eliminate the ambiguity between 0 and 2N outstanding packets, a maximum of 2N-1 outstanding unacknowledged packets shall be allowed at any one time.		
7f	Cover the below possible combination when CRF is RSVD = 0 PRIO sets packet priority as follows: 00 - lowest priority 01 - medium priority 10 - high priority 11- highest priority	Part6, Sec 2.2	
	Cover the below possible combination when CRF is supported PRIO||CRF sets packet priority: 00 0 - lowest priority 00 1 - critical flow lowest priority 01 0 - medium priority 01 1 - critical flow medium priority 10 0 - high priority 10 1 - critical flow high priority 11 0 - highest priority 11 1 - critical flow high priority		

Item No	Compliance Item	Spec reference	Testcase Name
8	Maintenance - System exploration, initialization and maintenance operation. Read or write device configuration registers and perform other system maintenance tasks	Part1, Sec 3.2	srio_ll_maintenace_req_test.sv
8a	MAINTENANCE requests receive a MAINTENANCE response rather than a normal response for both read and write operations.	Part1, Sec 3.4.1	
	Supported accesses are in 32 bit quantities and may optionally be in double-word and multiple double-word quantities to a maximum of 64 bytes.		
8b	The type 8 MAINTENANCE packet format is used to access the RapidIO capability and status registers (CARs and CSRs) and data structures	Part1, Sec 4.1.10	
	Unlike other request formats, the type 8 packet format serves as both the request and the response format for maintenance operations		
	Type 8 packets contain no addresses and only contain data payloads for write requests and read responses.		
	All configuration register read accesses are performed in word (4-byte), and optionally double-word (8-byte) or specifiable multiple double-word quantities up to a limit of 64 bytes		
	All register write accesses are also performed in word (4-byte), and optionally double-word (8-byte) or multiple double-word quantities up to a limit of 64 bytes.		
	Both the maintenance read and the maintenance write request generate the appropriate maintenance response.		
	Maintenance read responses with a status of ERROR may optionally include data in the response.		
	The data payload may not exceed that size but may be smaller if desired. Both the maintenance read and the maintenance write request generate the appropriate		
	Type 8 packet transaction field 0b0000 - Specifies a maintenance read request 0b0001 - Specifies a maintenance write request 0b0010 - Specifies a maintenance read response 0b0011 - Specifies a maintenance write response 0b0100 - Specifies a maintenance port-write request		
8c	Port-Write Target deviceID CSR This register contains the target Dev8 or Dev16 deviceID to be used when a device generates a Maintenance port-write operation to report errors to a system host	Part8, Sec 2.5.11	
8d	The maintenance port-write operation is a write operation that does not have guaranteed delivery and does not have an associated response.	Part1, Sec 4.1.10	
	The srcTID and config_offset fields are reserved for port-write requests.		
8e	Port-write Dev32 Target Device ID CSR This register contains the Dev32 target deviceID to be used when a device generates a Maintenance port-write operation to report errors to a system host and the Dev32_PW bit is set.	Part8, Sec 2.5.13	

Item No	Compliance Item	Spec reference	Testcase Name
8f	Port-Write Transmission Control The Port-Write transmission control CSR determines whether port-write notification is enabled or disabled for the device.	Part8, Sec 2.5.14	
8g	The sending device sets the Port-write Pending status bit in the Port *n* Error and Status CSR.	Part8, Sec 1.4	
	A 16 byte data payload of the Maintenance Port-write packet contains the contents of several CSRs, the port on the device that encountered the error condition		
	Software indicates that it has seen the port-write operation by clearing the Port-write Pending status bit.		
8h	Destination Operations CAR Bit[29] - PE can support a port-write operation.	Part1, Sec 5.4.8	
8i	Up to 2^n outstanding unacknowledged request and/or response packets. where N is the number of bits in the ackID field.	Part6, Sec 6.6.2	
	To eliminate the ambiguity between 0 and 2N outstanding packets, a maximum of 2N-1 outstanding unacknowledged packets shall be allowed at any one time.		
8j	Cover the below possible combination when CRF is RSVD = 0 PRIO sets packet priority as follows: 00 - lowest priority 01 - medium priority 10 - high priority 11- highest priority	Part6, Sec 2.2	
	Cover the below possible combination when CRF is supported PRIO||CRF sets packet priority: 00 0 - lowest priority 00 1 - critical flow lowest priority 01 0 - medium priority 01 1 - critical flow medium priority 10 0 - high priority 10 1 - critical flow high priority 11 0 - highest priority 11 1 - critical flow high priority		

3.4 Detectable Error Compliance

Table 4: Detectable Error Compliance

Item No	Compliance Item	Spec reference	Testcase Name
1	**NREAD**		srio_ll_nread_req_err_test.sv
1a	Data size for read transactions, used in conjunction with the word pointer (wdptr) bit.	Part1, Sec 4.1.2	
	Bit[8] An unsolicited/unexpected Response packet was received in Logical/ Transport Layer Error Detect CSR.	Part8, Sec 2.5.3	
	• Received NREAD response data payload size other than the NREAD request size		
1b	Type 2 packets never contain a data payload.	Part1, Sec 4.1.5	
	• Received NREAD request with data payload.		
1c	Bit[14] (Received packet exceeds 276 Bytes) in Port n Error Detect CSR. Received packet that exceeds the maximum allowed size.	Part8, Sec 2.5.15	
	Packet that exceeds the maximum packet size. Reception of packet (that exceeds the maximum packet size) causes the receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	• Received NREAD data payload greater than 256 bytes.		
1d	Bit[9] (Unsupported Transaction) Logical/Transport Layer Error Detect CSR- A transaction is received that is not supported in the Destination Operations.	Part1, Sec 5.4.8 Part8, Sec 2.5.3	
	• Change the Destination Operation CAR bit[16] to zero. NREAD request received.		
1e	Cyclic Redundancy Code used to detect transmission errors in the packet.	Part6, Sec 2.2	

Item No	Compliance Item	Spec reference	Testcase Name
1f	Packet with an incorrect CRC value The receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	Bit[13] (Received packet with bad CRC) in Port n Error Detect CSR	Part8, Sec 2.5.15	
	• Received NREAD request packet with incorrect CRC.		
	• Received NREAD response packet with incorrect CRC.		
1g	Bit[12] (Received packet with unexpected ackID) Port n Error Detect CSR	Part8, Sec 2.5.15	
	Packet with an unexpected ackID value Reception of packet (with an unexpected ackID value) causes the receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	ackID values on each link are assigned sequentially for each subsequent transmitted packet, an error in the ackID field is easily detected.	Part6, Sec 2.4	
	Received NREAD request packet with incorrect ackid		
	Received NREAD response packet with incorrect ackid		
1h	A packet carrying a response shall have a priority at least one priority level higher than the priority of the associated request.	Part6, Sec 6.11	
	• Received NREAD request with priority value 3.		
1i	Bit[4] (Illegal Transaction code - Received a supported request/response packet with undefined field values) Logical/Transport Layer Error Detect CSR.	Part8,Sec 2.5.3	
	• Received NREAD response with reserved bit set in transaction field.		
	• Received NREAD response with reserved bit set in status field.		
	• Received NREAD response with data payload, but transaction filed encoding value as Response transaction with no data payload.		
1j	Bit[0] (IO error response) in Logical/Transport Layer Error Detect CSR	Part8, Sec 2.5.3	
	• Received NREAD response with status field value as ERROR		
1k	A RESPONSE packet with an “ERROR” status never has a data payload.	Part1, Sec 4.2.3	
	• Received NREAD response with data payload for ERROR status		
1l	TYPE 13 Response Issued by a processing element when it completes a request by a remote element.	Part1, Sec 4.2	
	• Received NREAD response without NREAD request.		

Item No	Compliance Item	Spec reference	Testcase Name
1m	Bit[7] (Packet Response timeout) bit Logical/Transport Layer Error Detect CSR. A required response has not been received within the specified time out interval	Part8, Sec 2.5.3	
	• Allow response time out for NREAD request.		
1n	Received NREAD response without crf bit set for NREAD request with crf bit set		
1o	Packet containing invalid characters or valid non-data characters	Part6, Sec 6.13.2.4	
	• Received NREAD data with non-data characters		
	• Received NREAD data with invalid characters		
2	**ATOMIC**		srio_ll_atomic_req_err_test.sv
2a	Bit[9] (Unsupported Transaction) Logical/Transport Layer Error Detect CSR- A transaction is received that is not supported in the Destination Operations.	Part8, Sec 2.5.3	
	• Change the Destination Operations CAR Bit[22] to Bit[28]- value to 0. Atomic request received.		
2b	The allowed specified data sizes are one word (4 bytes), one half-word (2 bytes) or one byte Double-word (8-byte) and 3, 5, 6, and 7 byte ATOMIC transactions may not be specified.	Part1, Sec 3.3.4	
	• Received ATOMIC transaction with following size Double-word (8-byte) and 3, 5, 6, and 7 byte.		
2c	Type 2 packets never contain a data payload.	Part1, Sec 4.1.5	
	• Received ATOMIC transaction with data payload		
2d	The ATOMIC test-and-swap transaction is limited to one double-word (8 bytes) of data payload.	Part1 Sec 4.1.7	
	• Received ATOMIC test-and-swap transaction data payload greater than 1 double-word		
2e	The ATOMIC compare-and-swap operation is different from the other ATOMIC operations in that it requires two double-words (16 bytes) of data payload	Part1, Sec 4.1.7	
	• Received ATOMIC compare-and-swap operation data payload less than 2 double words and greater than 2 double words.		
2f	Cyclic Redundancy Code used to detect transmission errors in the packet.	Part6, Sec 2.2	
	Packet with an incorrect CRC value The receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	Bit[13] (Received packet with bad CRC) in Port n Error Detect CSR	Part8, Sec 2.5.16	
	• Received ATOMIC request packet with incorrect CRC		
	• Received ATOMIC response packet with incorrect CRC		

Item No	Compliance Item	Spec reference	Testcase Name
2g	Bit[12] (Received packet with unexpected ackID) Port n Error Detect CSR	Part8, Sec 2.5.15	
	Packet with an unexpected ackID value Reception of packet (with an unexpected ackID value) causes the receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	ackID values on each link are assigned sequentially for each subsequent transmitted packet, an error in the ackID field is easily detected.	Part6, Sec 2.4	
	• Received ATOMIC request packet with incorrect ackid		
	• Received ATOMIC response packet with incorrect ackid		
2h	Bit[7] (Packet Response timeout) bit Logical/Transport Layer Error Detect CSR. A required response has not been received within the specified time out interval	Part8, Sec 2.5.4	
	• Allow response timeout for ATOMIC request		
2i	TYPE 13 Response Issued by a processing element when it completes a request by a remote element.	Part1, Sec 4.2	
	• Received Atomic response without Atomic request.		
2j	Bit[4] (Illegal Transaction code - Received a supported request/response packet with undefined field values) Logical/Transport Layer Error Detect CSR.	Part8, Sec 2.5.3	
	• Received Atomic response with reserved bit set in transaction field		
	• Received Atomic response with reserved bit set in status field		
	• Received Atomic response with data payload, but transaction filed encoding value as Response transaction with no data payload.		
2k	A packet carrying a response shall have a priority at least one priority level higher than the priority of the associated request.	Part6, Sec 6.11	
	• Received ATOMC request with priority value 3.		
2l	Bit[0] (IO error response) in Logical/Transport Layer Error Detect CSR	Part8, Sec 2.5.3	
	• Received Atomic response with status field value as ERROR		
2m	A RESPONSE packet with an “ERROR” status or a response that is not expected to have a data payload never has a data payload.	Part1, Sec 4.2.3	
	• Received Atomic response with data payload for ERROR status.		
2n	Received ATOMIC response without crf bit set for ATOMIC request with crf bit set		
3	NWRITE		srio_ll_nwrite_req_err_test.sv

Item No	Compliance Item	Spec reference	Testcase Name
3a	Type 5 packets always contain a data payload	Part1, Sec 4.1.7	
	• Received NWRITE without data payload		
3b	The data payload may not exceed that size but may be smaller if desired.	Part1,Sec 4.1.7	
	• Received NWRITE data payload exceed the size.		
3c	Bit[14] (Received packet exceeds 276 Bytes) in Port n Error Detect CSR. Received packet that exceeds the maximum allowed size.	Part8, Sec 2.5.15	
	Packet that exceeds the maximum packet size. Reception of packet (that exceeds the maximum packet size) causes the receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	Received NWRITE data payload greater than 256 bytes.		
3d	Transaction Fields and Encodings for Type 5 Packet	Part1, Sec 4.1.7	
	• Received Type 5 request with reserved bit set in transaction fields • 0b0000–0011 - Reserved • 0b1111 - Reserved		
3e	NWRITE and SWRITE transactions do not receive responses.	Part1, Sec 3.3.2	
	• Received response for NWRITE request		
3f	A packet carrying a response shall have a priority at least one priority level higher than the priority of the associated request.	Part6, Sec 6.11	
	• Received NWRITE request with priority value 3		
3g	Cyclic Redundancy Code used to detect transmission errors in the packet	Part6, Sec 2.2	
	Packet with an incorrect CRC value The receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	Bit[13] (Received packet with bad CRC) in Port n Error Detect CSR	Part8, Sec 2.5.15	
	• Received NWRITE request packet with incorrect CRC.		

Item No	Compliance Item	Spec reference	Testcase Name
3h	Bit[12] (Received packet with unexpected ackID) Port n Error Detect CSR	Part8, Sec 2.5.15	
	Packet with an unexpected ackID value Reception of packet (with an unexpected ackID value) causes the receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	ackID values on each link are assigned sequentially for each subsequent transmitted packet, an error in the ackID field is easily detected.	Part6, Sec 2.4	
	• Received NWRITE request packet with incorrect ackid		
3i	Bit[9] (Unsupported Transaction) Logical/Transport Layer Error Detect CSR- A transaction is received that is not supported in the Destination Operations	Part8, Sec 2.5.3	
	Change the Destination Operations CAR Bit[17] to 0. Received NWRITE request.		
3j	Packet containing invalid characters or valid non-data characters	Part6, Sec 6.13.2.4	
	• Received NWRITE packet with valid non-data characters		
	• Received NWRITE packet with valid non-data characters		
4	**NWRITE_R**		srio_ll_nwrite_r_req_err.sv
4a	Type 5 packets always contain a data payload	Part1, Sec 4.1.7	
	• Received NWRITE_R without data payload		
4b	The data payload may not exceed that size but may be smaller if desired.	Part1, Sec 4.1.7	
	• Received NWRITE_R data payload exceed the size.		
4c	Bit[14] (Received packet exceeds 276 Bytes) in Port n Error Detect CSR. Received packet that exceeds the maximum allowed size.	Part8, Sec 2.5.15	
	Packet that exceeds the maximum packet size. Reception of packet (that exceeds the maximum packet size) causes the receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	• Received NWRITE_R data payload greater than 256 bytes.		
4d	A packet carrying a response shall have a priority at least one priority level higher than the priority of the associated request.	Part6, Sec 6.11	
	Received NWRITE_R request with priority value 3		

Item No	Compliance Item	Spec reference	Testcase Name
4e	Cyclic Redundancy Code used to detect transmission errors in the packet	Part6, Sec 2.2	
	Packet with an incorrect CRC value The receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	Bit[13] (Received packet with bad CRC) in Port n Error Detect CSR	Part8, Sec 2.5.15	
	• Received NWRITE_R request packet with incorrect CRC.		
	• Received NWRITE_R request packet with incorrect CRC.		
4f	Bit[12] (Received packet with unexpected ackID) Port n Error Detect CSR	Part8, Sec 2.5.15	
	Packet with an unexpected ackID value. Reception of packet (with an unex-pected ackID value) causes the receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	ackID values on each link are assigned sequentially for each subsequent transmitted packet, an error in the ackID field is easily detected.	Part6, Sec 2.4	
	• Received NWRITE_R request packet with incorrect ackid		
	• Received NWRITE_R response packet with incorrect ackid		
4g	Bit[9] (Unsupported Transaction) Logical/Transport Layer Error Detect CSR- A transaction is received that is not supported in the Destination Operations	Part8, Sec 2.5.3	
	• Change the Destination Operations CAR Bit[19] to 0. Received NWRITE_R request.		
4h	Bit[0] (IO error response) in Logical/Transport Layer Error Detect	Part8, Sec 2.5.3	
	• Received NWRITE_R response with ERROR status		
4i	TYPE 13 Response Issued by a processing element when it completes a request by a remote element.	Part1, Sec 4.2	
	• Received NWRITE_R response without NWRITE_R request.		
4j	Bit[4] (Illegal Transaction code - Received a supported request/response packet with undefined field values) Logical/Transport Layer Error Detect CSR.	Part8, Sec 2.5.3	
	• Received NWRITE_R response with reserved bit set in transaction field		
	• Received NWRITE_R response with reserved bit set in status field		
	• Received NWRITE_R response with data payload, but transaction filed encod-ing value as Response transaction with no data payload		

Item No	Compliance Item	Spec reference	Testcase Name
4k	Bit[7] (Packet Response timeout) bit Logical/Transport Layer Error Detect CSR. A required response has not been received within the specified time out interval	Part8, Sec 2.5.3	
	• Allow response timeout for NWRITE_R request		
4l	Received NWRITE_R response without crf bit set for NWRITE_R request with crf bit set		
4m	Packet containing invalid characters or valid non-data characters	Part6, Sec 6.13.2.4	
	Received NWRITE_R packet with invalid characters.		
	Received NWRITE_R packet with valid non-data characters.		
5	**Maintenance Transaction**		srio_ll_maintenace_req_err.sv
5a	Supported accesses are in 32 bit quantities and may optionally be in double-word and multiple double-word quantities to a maximum of 64 bytes.	Part1, Sec 3.4.1	
	• Received maintenance transaction with payload greater than 64 bytes.		
5b	Unlike other request formats, the type 8 packet format serves as both the request and the response format for maintenance operations.	Part1, Sec 4.1.10	
	• Received Type 13 response for Maintenance request		
5c	The data payload may not exceed that size but may be smaller if desired.	Part1, Sec 4.1.10	
	• Received Maintenance packet payload greater than the size specified.		
5d	Encodings for Type 8 Packets	Part1, Sec 4.1.10	
	• Received Maintenance request with reserved filed set in transaction field encoding. • 0b0101–1111 - Reserved		
5e	Encoding for Type 8 Response packet status field	Part1, Sec 4.1.10	
	• Received Maintenance response with reserved bit set in status value.		
	• Received Maintenance response with ERROR bit set in status value.		
	• Received Maintenance read response for Maintenance write request.		
	• Received Maintenance write response for Maintenance read request.		
	• Received Maintenance response with incorrect targetTID		
5f	Type 8 packets contain no addresses and only contain data payloads for write requests and read responses.	Part1, Sec 4.1.10	
	• Received Maintenance write without data payload		
	• Received Maintenance read response without data payload		

Item No	Compliance Item	Spec reference	Testcase Name
5g	Cyclic Redundancy Code used to detect transmission errors in the packet	Part6, Sec 2.2	
	Packet with an incorrect CRC value The receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	Bit[13] (Received packet with bad CRC) in Port n Error Detect CSR		
	Received Maintenance request packet with incorrect CRC		
	Received Maintenance response packet with incorrect CRC		
5h	Bit[12] (Received packet with unexpected ackID) Port n Error Detect CSR	Part8, Sec 2.5.15	
	Packet with an unexpected ackID value Reception of packet (with an unexpected ackID value) causes the receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery	Part6, Sec 6.13.2.4	
	ackID values on each link are assigned sequentially for each subsequent transmitted packet, an error in the ackID field is easily detected	Part6, Sec 2.4	
	• Received maintenance request with incorrect ackid		
	• Received maintenance response with incorrect ackid		
5i	srcTID - The type 8 request packet's transaction ID (reserved for port-write requests)	Part1, Sec 4.1.10	
	• Received Maintenance request packet, with srcTID value set		
5j	Bit[0] (IO error response) in Logical/Transport Layer Error Detect	Part8, Sec 2.5.3	
	• Received Maintenance response with ERROR status		
5k	Bit[7] (Packet Response timeout) bit Logical/Transport Layer Error Detect CSR. A required response has not been received within the specified time out interval	Part8, Sec 2.5.3	
	• Allow response timeout for Maintenance request		
5l	Packet containing invalid characters or valid non-data characters	Part6, Sec 6.13.2.4	
	Received maintenance packet with invalid characters		
	Received maintenance packet with valid non-data characters		
6	SWRITE Transaction		srio_ll_swrite_req_err.sv
6a	The type 6 packet is a special-purpose type that always contains data	Part1, Sec 4.1.8	
	• Received swrite transaction without data payload		

Item No	Compliance Item	Spec reference	Testcase Name
6b	Bit[14] (Received packet exceeds 276 Bytes) in Port n Error Detect CSR. Received packet that exceeds the maximum allowed size.	Part8, Sec 2.5.15	
	Packet that exceeds the maximum packet size. Reception of packet (that exceeds the maximum packet size) causes the receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	• Received SWRITE data payload greater than 256 bytes.		
6c	NWRITE and SWRITE transactions do not receive responses	Part1, Sec 3.3.2	
	• Received response for SWRITE request		
6d	The data payload always contains a minimum of one complete double-word.	Part1, Sec 4.1.8	
	• Received SWRITE transaction with less than one double word data payload		
6e	Cyclic Redundancy Code used to detect transmission errors in the packet	Part6, Sec 2.2	
	Packet with an incorrect CRC value The receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery	Part6, Sec 6.13.2.4	
	Bit[13] (Received packet with bad CRC) in Port n Error Detect CSR	Part8, Sec 2.5.15	
	• Received swrite packet with incorrect crc		
6f	The SWRITE transaction is a double-word-only transaction	Part1, Sec 3.3.2	
	• Received swrite transaction with data payload not aligned to double word		
6g	Bit[12] (Received packet with unexpected ackID) Port n Error Detect CSR	Part8, Sec 2.5.15	
	Packet with an unexpected ackID value Reception of packet (with an unexpected ackID value) causes the receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	ackID values on each link are assigned sequentially for each subsequent transmitted packet, an error in the ackID field is easily detected.	Part6, Sec 2.4	
	• Received SWRITE request packet with incorrect ackid		

Item No	Compliance Item	Spec reference	Testcase Name
6h	Bit[9] (Unsupported Transaction) Logical/Transport Layer Error Detect CSR. A transaction is received that is not supported in the Destination Operations	Part8, Sec 2.5.3	
	• Change the Destination Operations CAR Bit[18] to 0. • Received SWRITE request.		
6i	A packet carrying a response shall have a priority at least one priority level higher than the priority of the associated request.	Part6, Sec 6.11	
	• Received SWRITE request with priority value 3		
6j	Packet containing invalid characters or valid non-data characters	Part6, Sec 6.13.2.4	
	• Received SWRITE packet with invalid characters.		
	• Received SWRITE packet with valid non-data characters.		

3.5 Random Compliance

Table 5: Random Compliance

Item No	Compliance Item	Spec reference	Testcase Name
1	Multiple transactions must be allowed concurrently in the system, otherwise a majority of the potential system throughput is wasted	Part1, Sec 1.3.3	srio_ll_io_random_test.sv
2	Randomly generate error transaction followed by normal transaction for all the IO		srio_ll_io_normal_err_test.sv
3	Randomly generate wrsize/rdsize, all priority, crf combination for all the IO transaction		srio_ll_io_random_test.sv
4	Randomly generate back to back transaction		srio_ll_io_random_back_to_back_test.sv
5	Generate packet accepted control symbol in random interval for all the IO transaction		srio_ll_io_delay_ack_random.sv

4 Part2 - Message Passing Logical Specification Compliance

4.1 Performance feature compliance

Table 6: Performance feature compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	Multiple transactions must be allowed concurrently in the system, otherwise a majority of the potential system throughput is wasted	Part2, Sec 1.3.3	srio_ll_msg_mseg_concurrent_req.sv srio_ll_msg_sseg_concurrent_req.sv
2	The number of outstanding transactions that may be supported is implementation dependent.	Part2, Sec 3.1	srio_ll_msg_mseg_outstand_req.sv srio_ll_msg_sseg_outstand_req.sv

4.2 Operation ordering compliance

Table 7: Operation ordering compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	It is important to recognize that systems may contain a mix of transactions that are maintained under the message passing model as well as under another model. As an example, I/O traffic may be interspersed with message traffic.	Part2, Sec 2.4.1	srio_ll_io_message_req_test.sv
2	The protocols handle out-of-order packet transmission and reception.	Part2, Sec 1.3.2	srio_ll_msg_mseg_outoforder_resp_test.sv
3	All packets except the last have the same data payload size, the receiver is able to calculate the local memory storage addresses if the packets are received out of order, allowing operation with an interconnect fabric that does not guarantee packet delivery ordering.	Part2, Sec 2.3.1	
4	A message operation may consist of several transactions. It is possible for these transactions to arrive at a target mailbox in an arbitrary order. A message transaction contains explicit tagging information to allow the message to be reconstructed as it arrives at the target processing element	Part2, Sec 2.4.2	

Item No	Compliance Item	Spec Reference	Testcase Name
5	Each request transaction sent into the system is marked with a transaction ID that is unique for each requestor and responder processing element pair. This transaction ID allows a response to be easily matched to the original request when it is returned to the requestor.	Part2, Sec 3.1	

4.3 Basic Functionality Compliance

Table 8: Basic functionality compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	**Multi Segment Message**		srio_ll_msg_mseg_req_test.sv
	Many times a message is required to be larger than a single packet allows, so the source needs to break up the message into multiple packets before transmitting it. At times it may also be useful to have more than one message being transmitted at a time.	Part2, Sec 2.3	
	RapidIO defines all data message packets as containing the same amount of data with the exception of the last one, which can contain a smaller data payload if desired.	Part2, Sec 2.3.1	
	The packets are formatted with three fields One field specifies the size of the data payload for all except the last packet for the data message operation. The second field specifies the size of the data payload for that packet. The third field contains the packet sequence order information.		
	For multiple packet messages, a letter field and a mailbox field allow a source to simultaneously have up to four data message operations (or “letters”) in progress to each of four different mailboxes, allowing up to sixteen concurrent data message operations between a sender and a receiver.		srio_ll_msg_mseg_concurrent_req.sv
	The mailbox field can be used to indicate the priority of a data message, allowing a higher priority message to interrupt a lower priority one at the sender, or it can be used as a simple mailbox identifier for a particular receiver if the receiver allows multiple mailbox addresses.		srio_ll_msg_mseg_priority_mailbox_test.sv
	If the mailbox number is used as a priority indicator, mailbox number 0 is the highest priority and mailbox 3 is the lowest.		
	As for multiple packet messages, if the mailbox number is used as a priority indicator, mailbox number 0 is the highest priority and mailbox 63 is the lowest.		

Item No	Compliance Item	Spec Reference	Testcase Name
1a	Responses to message and doorbell packets never contain data.	Part2, Sec 4.3.3	srio_ll_msg_mseg_reg_test.sv
1b	Completing a data message operation can consist of up to 16 individual MESSAGE transactions.	Part2, Sec 3.3.2	
	Message transaction data payloads are always multiples of doubleword quantities.		
1c	The type 11 packet is the MESSAGE transaction format	Part2, Sec 4.2.5	
	Type 11 packets always have a data payload.		
	The combination of the letter, mbox, and the msgseg or xmbox fields uniquely identifies the message packet in the system for each requestor and responder processing element pair in the same way as the transaction ID is used for other request types.		
1d	The message length and segment fields allow the individual packets of a data message to be sent or received out of order.	Part2, Sec 3.2.2	srio_ll_msg_mseg_outstand_req.sv
1e	Up to 2N outstanding unacknowledged request and/or response packets where N is the number of bits in the ackID field	Part6, Sec 6.6.2	
	To eliminate the ambiguity between 0 and 2N outstanding packets, a maximum of 2N-1 outstanding unacknowledged packets shall be allowed at any one time.		
1f	Cover the below possible combination when CRF is RSVD = 0 PRIO sets packet priority as follows: 00 - lowest priority 01 - medium priority 10 - high priority 11- highest priority	Part6, Sec 2.2	srio_ll_msg_mseg_reg_test.sv
	Cover the below possible combination when CRF is supported PRIO||CRF sets packet priority: 00 0 - lowest priority 00 1 - critical flow lowest priority 01 0 - medium priority 01 1 - critical flow medium priority 10 0 - high priority 10 1 - critical flow high priority 11 0 - highest priority 11 1 - critical flow high priority		
2	**Single Segment Message**		srio_ll_msg_sseq_req_test.sv
2a	A source may generate a single message operation of up to 16 individual packets containing as much as 256 data bytes per packet	Part2, Sec 2.3.1	
	A variety of data payload sizes exist, allowing a source to choose a smaller size data payload if needed for an application.		

Item No	Compliance Item	Spec Reference	Testcase Name
2b	For single packet messages, the letter and mailbox fields instead allow four concurrent letters to sixty-four possible mailboxes		srio_ll_msg_sseg_concurrent_req.sv
2c	The data message operation, consisting of the MESSAGE and RESPONSE transactions (typically a DONE response)	Part2, Sec 3.3.2	srio_ll_msg_sseq_req_test.sv
2d	The type 11 packet is the MESSAGE transaction format	Part2, Sec 4.2.5	
	Type 11 packets always have a data payload.		
	The combination of the letter, mbox, and the msgseg or xmbox fields uniquely identifies the message packet in the system for each requestor and responder processing element pair in the same way as the transaction ID is used for other request types.		
2e	Responses to message and doorbell packets never contain data.	Part2, Sec 4.3.3	
2f	Up to 2N outstanding unacknowledged request and/or response packets where N is the number of bits in the ackID field	Part6, Sec 6.6.2	srio_ll_msg_sseg_outstand_req.sv
	To eliminate the ambiguity between 0 and 2N outstanding packets, a maximum of 2N-1 outstanding unacknowledged packets shall be allowed at any one time.		
2g	Cover the below possible combination when CRF is RSVD = 0 PRIO sets packet priority as follows: 00 - lowest priority 01 - medium priority 10 - high priority 11- highest priority	Part6, Sec 2.2	srio_ll_msg_sseq_req_test.sv
	Cover the below possible combination when CRF is supported PRIO||CRF sets packet priority: 00 0 - lowest priority 00 1 - critical flow lowest priority 01 0 - medium priority 01 1 - critical flow medium priority 10 0 - high priority 10 1 - critical flow high priority 11 0 - highest priority 11 1 - critical flow high priority		
3	**Doorbell Message**		srio_ll_doorbell_req.sv
3a	All information supplied in a doorbell message is embedded in the packet header so the doorbell message never has a data payload.	Part2, Sec 2.3.2	
3b	The doorbell operation, consisting of the DOORBELL and RESPONSE transactions (typically a DONE response)	Part2, Sec 3.3.1	

Item No	Compliance Item	Spec Reference	Testcase Name
3c	The type 10 packet format is the DOORBELL transaction format	Part2, Sec 4.2.4	
	Type 10 packets never have data payloads.		
3d	Up to 2N outstanding unacknowledged request and/or response packets where N is the number of bits in the ackID field	Part6, Sec 6.6.2	
	To eliminate the ambiguity between 0 and 2N outstanding packets, a maximum of 2N-1 outstanding unacknowledged packets shall be allowed at any one time.		
3e	Cover the below possible combination when CRF is RSVD = 0 PRIO sets packet priority as follows: 00 - lowest priority 01 - medium priority 10 - high priority 11- highest priority	Part6, Sec 2.2	
	Cover the below possible combination when CRF is supported PRIO||CRF sets packet priority: 00 0 - lowest priority 00 1 - critical flow lowest priority 01 0 - medium priority 01 1 - critical flow medium priority 10 0 - high priority 10 1 - critical flow high priority 11 0 - highest priority 11 1 - critical flow high priority		

4.4 Detectable Error Compliance

Table 9: Detectable Errors Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	**Single Segment**		srio_ll_msg_sseg_req_err.sv
1a	Type 11 packets always have a data payload.	Part2, Sec 4.2.5	
	• Received Single segment message request without data payload		
1b	Specific Field Definitions and Encodings for Type 11 Packets msglen - A value of 0 indicates a single-packet message.	Part2, Sec 4.2.5	
	• Non zero value in msg_len field for single segment message		
1c	ssize - Standard message packet data size.	Part2, Sec 4.2.5	
	• Received single segment message with payload other than ssize field value		
	• Received single segment message with reserved value set in ssize field		
1d	TYPE 13 Response Issued by a processing element when it completes a request by a remote element.	Part2, Sec 4.3	
	• Received Single segment message response without type11 request.		
	• Received reserved filed set in response transaction field for Single segment message.		
	• Received 0b0000 set in response transaction field		
	• Received reserved filed set in response status(0b1000–1011) field for Single segment message.		
1e	Responses to message and doorbell packets never contain data	Part2, Sec 4.3.3	
	• Received response with data payload for single segment message.		
1f	target_info Field for Message Responses	Part2, Sec 4.3.3	
	• Received response with incorrect target_info for single segment message		
1g	Bit[1] Message error response Logical/Transport Layer Error Detect CSR Received a response of ‘ERROR’ for an MSG Logical Layer Request	Part8, Sec 2.5.3	
	• Received error filed set in response status(0b0111) field for Single segment message.		
1h	Message response timeout A required response has not been received within the specified time out interval	Part8, Sec 1.3.2	
	• Allow response timeout for Single segment message request		

Item No	Compliance Item	Spec Reference	Testcase Name
1i	status - RETRY Requested transaction is not accepted; re-transmission of the request is needed to complete the transaction	Part2, Sec 4.3.1	
	• Received RETRY filed set in response status(0b0011) field for Single segment message.		
1j	Bit[14] (Received packet exceeds 276 Bytes) in Port n Error Detect CSR. Received packet that exceeds the maximum allowed size.	Part8, Sec 2.5.15	
	Packet that exceeds the maximum packet size. Reception of packet (that exceeds the maximum packet size) causes the receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	• Received Single segment data payload greater than 256 bytes.		
1k	Bit[9] (Unsupported Transaction) Logical/Transport Layer Error Detect CSR- A transaction is received that is not supported in the Destination Operations	Part8, Sec 2.5.3	
	Change the Destination Operation CAR bit[20] to zero. Single segment message received.		
1l	Cyclic Redundancy Code used to detect transmission errors in the packet.	Part6, Sec 2.2	
	Packet with an incorrect CRC value The receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	Bit[13] (Received packet with bad CRC) in Port n Error Detect CSR	Part8, Sec 2.5.15	
	• Received Single segment message packet with incorrect CRC.		
	• Received Single segment message response packet with incorrect CRC.		
1m	Bit[12] (Received packet with unexpected ackID) Port n Error Detect CSR	Part8, Sec 2.5.15	
	Packet with an unexpected ackID value Reception of packet (with an unexpected ackID value) causes the receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	ackID values on each link are assigned sequentially for each subsequent transmitted packet, an error in the ackID field is easily detected.	Part6, Sec 2.4	
	• Received Single segment request packet with incorrect ackid		
	• Received Single segment response packet with incorrect ackid		
1n	A packet carrying a response shall have a priority at least one priority level higher than the priority of the associated request.	Part6, Sec 6.11	
	• Received Single segment request with priority value 3.		

Item No	Compliance Item	Spec Reference	Testcase Name
1o	Received Single segment response without crf bit set for Single segment request with crf bit set		
1m	Packet containing invalid characters or valid non-data characters	Part6, Sec 6.13.2.4	
	• Received single segment message packet with valid non-data characters		
	• Received single segment message packet with invalid characters		
	Multi Segment		srio_ll_msg_mseg_req_err.sv
2a	MESSAGE transaction data payloads are always multiples of doubleword quantities.	Part2, Sec 3.3.2	
	• Received multi segment message transaction data payload non multiple of double words.		
2b	Completing a data message operation can consist of up to 16 individual MESSAGE transactions.	Part2, Sec 3.3.2	
	• Received multi segment message transaction with more than 16 segments.		
2c	msglen - field - Specifies the number of transactions that comprise the data message operation	Part2, Sec 3.3.2	
	• Received incorrect value in the msgleng field for Multi segment message.		
2d	msgseg - Identifies which part of the data message operation is contained in this transaction	Part2, Sec 4.2.5	
	• Received incorrect value in the msgseg field for Multi segment message.		
2e	Bit[14] (Received packet exceeds 276 Bytes) in Port n Error Detect CSR. Received packet that exceeds the maximum allowed size.	Part8, Sec 2.5.15	
	Packet that exceeds the maximum packet size. Reception of packet (that exceeds the maximum packet size) causes the receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	• Received Multi segment data payload greater than 256 bytes.		
2f	Type 11 packets always have a data payload.	Part2, Sec 4.2.5	
	• Received Multi segment message request without data payload		
2g	ssize - Standard message packet data size.	Part2, Sec 4.2.5	
	• Received Multi segment message with payload other than ssize field value		
	• Received Multi segment message with reserved value set in ssize field		

Item No	Compliance Item	Spec Reference	Testcase Name
2h	TYPE 13 Response Issued by a processing element when it completes a request by a remote element.	Part2, Sec 4.3	
	• Received Multi segment message response without type11 request.		
	• Received reserved filed set in response transaction field for Multi segment message.		
	• Received 0b0000 set in response transaction field		
	• Received reserved filed set in response status(0b1000–1011) field for Multi segment message.		
2i	Responses to message and doorbell packets never contain data	Part2, Sec 4.3.3	
	• Received response with data payload for multi segment message.		
2j	target_info Field for Message Responses	Part2, Sec 4.3.3	
	• Received response with incorrect target_info for multi segment message		
2k	Bit[1] Message error response Logical/Transport Layer Error Detect CSR Received a response of 'ERROR' for an MSG Logical Layer Request	Part8, Sec 2.5.3	
	• Received error filed set in response status(0b0111) field for Multi segment message.		
2l	Message response timeout A required response has not been received within the specified time out interval	Part8, Sec 1.3.2	
	• Allow response timeout for Multi segment message request		
2m	status - RETRY Requested transaction is not accepted; re-transmission of the request is needed to complete the transaction	Part2, Sec 4.3.1	
	• Received RETRY filed set in response status(0b0011) field for Multi segment message.		
2n	Bit[9] (Unsupported Transaction) Logical/Transport Layer Error Detect CSR-A transaction is received that is not supported in the Destination Operations	Part8, Sec 2.5.3	
	• Change the Destination Operation CAR bit[20] to zero. Multi segment message received.		

Item No	Compliance Item	Spec Reference	Testcase Name
2o	Cyclic Redundancy Code used to detect transmission errors in the packet.	Part6, Sec 2.2	
	Packet with an incorrect CRC value The receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	Bit[13] (Received packet with bad CRC) in Port n Error Detect CSR	Part8, Sec 2.5.15	
	• Received Multi segment message packet with incorrect CRC.		
	• Received Multi segment message response packet with incorrect CRC.		
2p	Bit[12] (Received packet with unexpected ackID) Port n Error Detect CSR	Part8, Sec 2.5.15	
	Packet with an unexpected ackID value Reception of packet (with an unexpected ackID value) causes the receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	ackID values on each link are assigned sequentially for each subsequent transmitted packet, an error in the ackID field is easily detected.	Part6, Sec 2.4	
	• Received Multi segment request packet with incorrect ackid		
	• Received Multi segment response packet with incorrect ackid		
2q	A packet carrying a response shall have a priority at least one priority level higher than the priority of the associated request.	Part6, Sec 6.11	
	• Received Multi segment request with priority value 3.		
2r	Received Multi segment response without crf bit set for Multi segment request with crf bit set		
2s	Packet containing invalid characters or valid non-data characters	Part6, Sec 6.13.2.4	
	• Received multi segment packet with invalid characters		
	• Received multi segment packet with valid non-data characters		
3	**Doorbell message**		srio_ll_doorbell_req_err.sv
3a	The DOORBELL transaction contains the info field to hold information and does not have a data payload	Part2,Sec 3.3.1	
	• Received door bell transaction with data payload		
3b	Responses to message and doorbell packets never contain data.	Part2, Sec 4.3.3	
	• Received response with data for Doorbell message		

Item No	Compliance Item	Spec Reference	Testcase Name
3c	TYPE 13 Response Issued by a processing element when it completes a request by a remote element.	Part2, Sec 4.3	
	• Received doorbell response without type10 request.		
	• Received reserved filed set in response transaction field for doorbell message.		
	• Received 0b0000 set in response transaction field		
	• Received reserved filed set in response status(0b1000–1011) field for doorbell message.		
3d	Bit[1] Message error response Logical/Transport Layer Error Detect CSR Received a response of 'ERROR' for an MSG Logical Layer Request	Part8, Sec 2.5.3	
	• Received error filed set in response status(0b0111) field for door bell message.		
3e	Message response timeout A required response has not been received within the specified time out interval	Part8, Sec 1.3.2	
	• Allow response timeout for doorbell message request		
3f	status - RETRY Requested transaction is not accepted; re-transmission of the request is needed to complete the transaction	Part2, Sec 4.3.1	
	• Received RETRY filed set in response status(0b0011) field for doorbell message.		
3g	Bit[9] (Unsupported Transaction) Logical/Transport Layer Error Detect CSR-A transaction is received that is not supported in the Destination Operations	Part8, Sec 2.5.3	
	• Change the Destination Operation CAR bit[21] to zero. Door bell message.		
3h	Cyclic Redundancy Code used to detect transmission errors in the packet.	Part6, Sec 2.2	
	Packet with an incorrect CRC value The receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	Bit[13] (Received packet with bad CRC) in Port n Error Detect CSR	Part8, Sec 2.5.15	
	• Received Doorbell message packet with incorrect CRC.		
	• Received Doorbell response packet with incorrect CRC.		

Item No	Compliance Item	Spec Reference	Testcase Name
3i	Bit[12] (Received packet with unexpected ackID) Port n Error Detect CSR	Part8, Sec 2.5.15	
	Packet with an unexpected ackID value Reception of packet (with an unexpected ackID value) causes the receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	ackID values on each link are assigned sequentially for each subsequent transmitted packet, an error in the ackID field is easily detected.	Part6, Sec 2.4	
	• Received Doorbell request packet with incorrect ackid		
	• Received Doorbell response packet with incorrect ackid		
3j	A packet carrying a response shall have a priority at least one priority level higher than the priority of the associated request.	Part6, Sec 6.11	
	• Received Doorbell request with priority value 3.		
3k	Received Doorbell response without crf bit set for Doorbell request with crf bit set		

4.5 Random Case Compliance

Table 10: Random Case Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	Randomly generate both mseg/sseg in a single testcase		srio_ll_message_req_test.sv
2	Randomly generate IO/Message (Single seg/Multi seg/Doorbell) transaction in a single testcase		srio_ll_io_message_doorbell_req_test.sv
3	Randomly generate mseg message with msgleng (Message length) starting from 1 to 16		srio_ll_msg_mseg_req_random_test.sv
4	Randomly generate sseg normal transaction followed by error transaction		srio_ll_msg_sseg_req_normal_err_test.sv

Item No	Compliance Item	Spec Reference	Testcase Name
5	Randomly generate mseg normal transaction followed by error transaction		srio_ll_msg_mseg_req_normal_err_test.sv
6	Randomly generate mseg/sseg normal, error transaction		srio_ll_message_req_normal_err_test.sv
7	Randomly generate back-to-back message transaction		srio_ll_message_req_random_back_to_back_trans.sv
8	Generate packet accepted control symbol in random interval for all the message transaction		srio_ll_msg_delay_ack_random_test.sv

5 Part 3 - Common Transport Specification Compliance

5.1 Transport Format Compliance

Table 11: Transport Format Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	*Common Transport Specification* adds a transport type (tt) field to the logical specification packet that allows four different transport packet types to be specified.	Part3, Sec 2.4	
2	The three fields (tt, destinationID, and sourceID) added to the logical packets allow for three different sizes of the device ID fields: Dev32 (32-bit), Dev16 (16-bit), and a Dev8 (8-bit),		
3	The Dev8 fields allow a maximum of 256 devices to be attached to the fabric. The Dev16 fields allow systems with up to 65,536 devices. The Dev32 fields allow systems with up to 4,294,967,296 devices.		

5.2 Detectable Error Compliance

Table 12: Detectable Error Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	Reserved field set in Transaction Fields for all io/msg/data stream Packets. Illegal transaction decode bit set in Logical/Transport Layer Error Detect CSR	Part8, Spec 2.3.2.2	srio_ll_nread_req_err_test.sv srio_ll_atomic_req_err_test.sv srio_ll_nwrite_req_err_test.sv srio_ll_nwrite_r_req_err.sv srio_ll_maintenace_req_err.sv srio_ll_swrite_req_err.sv srio_ll_msg_sseg_req_err.sv srio_ll_msg_mseg_req_err.sv srio_ll_doorbell_req_err.sv
2	Force unsupported value in the tt field for all io/msg/data stream Packet Unsupported Transaction bit set in Logical/Transport Layer Error		

6 Part5 - Globally Shared Memory Logical Specification

6.1 GSM Operations Compliance

Table 13: GSM Operations Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	**Read** • READ-HOME • READ_OWNER • RESPONSE A read operation always returns one coherence granule-sized data payload.	Part5, Sec 3.2	
2	**READ-HOME** Read shared copy of home memory for coherence granule	Part5, Sec 3.3.1	srio_ll_gsm_rd_home_test.sv
	The READ_HOME transaction is used by a processing element that needs to read a shared copy of a coherence granule from a remote home memory on another processing element		
	Read Operation to Remote Shared Coherence Granule If the requested data exists in the memory directory as shared, the data can be returned immediately from memory with a DONE RESPONSE transaction and the requesting processing element's device ID is added to the sharing mask		
2a	Type 2 request packets never include data.	Part5, Sec 4.2.6	
	Transaction Field Encodings for Type 2 Packets 0b0000 READ_HOME		

Item No	Compliance Item	Spec Reference	Testcase Name
2b	Resolving an Outstanding READ_HOME Transaction	Part5, Sec 7.2	
	Outstanding request - READ_HOME Incoming Request - READ_HOME Resolution - Generate “ERROR” response		
	Outstanding request - READ_HOME Incoming Request - IREAD_HOME Resolution - Generate “ERROR” response		
	Outstanding request - READ_HOME Incoming Request - READ_OWNER Resolution - Generate “NOT_OWNER” response		
	Outstanding request - READ_HOME Incoming Request - READ_TO_OWN_HOME Resolution - Generate “ERROR” response		
	Outstanding request - READ_HOME Incoming Request - READ_TO_OWN_OWNER Resolution Generate “NOT_OWNER” response		
	Outstanding request - READ_HOME Incoming Request - DKILL_HOME Resolution - Generate “ERROR” response		
	Outstanding request - READ_HOME Incoming Request - DKILL_SHARER Resolution - • If outstanding request, wait for all expected responses. • If final response is “DONE”, return data if necessary and forward DKILL_SHARER to processor then generate a “DONE” response. • If final response is “RETRY”, cancel the read at the processor and forward DKILL_SHARED to processor then generate a “DONE” response. • If no outstanding request, cancel the read at the processor and forward DKILL_SHARER to processor then generate a “DONE” response (this case should be very rare).		
	Outstanding request - READ_HOME Incoming Request - CASTOUT Resolution - Generate “ERROR” response		
	Outstanding request - READ_HOME Incoming Request - TLBIE Resolution -No collision, forward to processor then generate “DONE” response (software must maintain TLB entry coherence)		
	Outstanding request - READ_HOME Incoming Request - TLBSYNC Resolution - No collision, forward to processor then generate “DONE” response (software must maintain TLB entry coherence)		
	Outstanding request - READ_HOME Incoming Request - IKILL_HOME Resolution - Generate “ERROR” response		

Item No	Compliance Item	Spec Reference	Testcase Name
	Outstanding request - READ_HOME Incoming Request - IKILL_SHARER Resolution - No collision, forward to processor then generate “DONE” response (software must maintain instruction cache coherence)		
	Outstanding request - READ_HOME Incoming Request - FLUSH Resolution : Generate “ERROR” response		
	Outstanding request - READ_HOME Incoming Request - IO_READ_HOME Resolution: Generate “ERROR” response		
	Outstanding request - READ_HOME Incoming Request - IO_READ_OWNER Resolution: Generate “NOT_OWNER” response		
	Cover the below possible combination when CRF is RSVD = 0 PRIO sets packet priority as follows: 00 - lowest priority 01 - medium priority 10 - high priority 11- highest priority	Part6, Sec 2.2	
	Cover the below possible combination when CRF is supported PRIO||CRF sets packet priority: 00 0 - lowest priority 00 1 - critical flow lowest priority 01 0 - medium priority 01 1 - critical flow medium priority 10 0 - high priority 10 1 - critical flow high priority 11 0 - highest priority 11 1 - critical flow high priority		
3	**READ_OWNER** Read shared copy of remotely owned coherence granule.		srio_ll_gsm_rd_owner_test.sv
3a	The READ_OWNER transaction is used by a home memory processing element that needs to read a shared copy of a coherence granule that is owned by a remote processing element.	Part5, Sec 3.3.1	
3b	Type 1 request packets never include data	Part5, Sec 4.2.5	
	Type 1 request are the only request types that can cause an intervention, so the secondary domain, secondary ID, and secondary transaction ID fields are required.		
	Type 1 packets are issued only by a home memory controller to allow the third party intervention data transfer.		
	transaction - 0b0000 - READ_OWNER		

Item No	Compliance Item	Spec Reference	Testcase Name
3c	**Resolving an Outstanding READ_OWNER Transaction**	Part5, Sec 7.4	
	Outstanding Request - READ_OWNER Incoming Request - READ_HOME Resolution - Generate “RETRY” response		
	Outstanding Request - READ_OWNER Incoming Request - IREAD_HOME Resolution - Generate “RETRY” response		
	Outstanding Request - READ_OWNER Incoming Request -READ_OWNER Resolution - Generate “ERROR” response		
	Outstanding Request - READ_OWNER Incoming Request - READ_TO_OWN_HOME Resolution - Generate “RETRY” response		
	Outstanding Request - READ_OWNER Incoming Request - DKILL_HOME Resolution - Generate “RETRY” response		
	Outstanding Request - READ_OWNER Incoming Request - DKILL_SHARER Resolution - Generate “ERROR” response		
	Outstanding Request - READ_OWNER Incoming Request - CASTOUT Resolution - No collision, update directory state, generate “DONE” response (CASTOUT bypasses address collision detection)		
	Outstanding Request - READ_OWNER Incoming Request - TLBIE Resolution - No collision, forward to processor then generate “DONE” response (software must maintain TLB entry coherence)		
	Outstanding Request - READ_OWNER Incoming Request - TLBSYNC Resolution - No collision, forward to processor then generate “DONE” response (software must maintain TLB entry coherence)		
	Outstanding Request - READ_OWNER Incoming Request -IKILL_HOME Resolution - No collision, forward to processor, send IKILL_SHARER to all participants except requestor (software must maintain instruction cache coherence)		

Item No	Compliance Item	Spec Reference	Testcase Name
	Outstanding Request - READ_OWNER Incoming Request -IKILL_SHARER Resolution - Generate “ERROR” response		
	Outstanding Request - READ_OWNER Incoming Request - FLUSH Resolution - Generate “RETRY” response		
	Outstanding Request - READ_OWNER Incoming Request - IO_READ_HOME Resolution - Generate “RETRY” response		
	Outstanding Request - READ_OWNER Incoming Request - IO_READ_OWNER Resolution - Generate “ERROR” response		
3d	Read Operation to Remote Modified Coherence Granule If the requested data exists in the memory directory as modified, the up-to-date (current) data must be obtained from the owner. The home memory then sends a READ_OWNER request to the processing element that owns the coherence granule. The owner passes a copy of the data to the original requestor and to memory, memory is updated, and the directory state is changed from modified and owner to shared by the previous owner and the requesting processing element’s device ID	Part5, Sec 3.3.1	
3e	Read Operation to Local Modified Coherence Granule If the processing element requesting a modified coherence granule happens to be the home for the memory, some of the transactions can be eliminated	Part5, Sec 3.3.1	
3f	Cover the below possible combination when CRF is RSVD = 0 PRIO sets packet priority as follows: 00 - lowest priority 01 - medium priority 10 - high priority 11- highest priority	Part6, Sec 2.2	
	Cover the below possible combination when CRF is supported PRIO||CRF sets packet priority: 00 0 - lowest priority 00 1 - critical flow lowest priority 01 0 - medium priority 01 1 - critical flow medium priority 10 0 - high priority 10 1 - critical flow high priority 11 0 - highest priority 11 1 - critical flow high priority		

Item No	Compliance Item	Spec Reference	Testcase Name
4	**Instruction read** • IREAD_HOME • READ_OWNER • RESPONSE The behavior of the instruction read operation is nearly identical to a data read operation with the only difference being the way that the apparent coherence paradox is managed.	Part5, Sec 3.3.2	
4a	An instruction read operation always returns one coherence granule-sized data payload.		
4b	**IREAD_HOME** Read shared copy of home memory for instruction cache	Part5, Sec 4.2	srio_ll_gsm_iread_home_test.sv
	Transaction Field Encodings for Type 2 Packets 0b1000 - IREAD_HOME	Part5, Sec 4.2.6	
4c	**Resolving an Outstanding IREAD_HOME**	Part5, Sec 7.3	
	Outstanding Request - IREAD_HOM Incoming Request - READ_HOME Resolution - Generate “ERROR” response		

Item No	Compliance Item	Spec Reference	Testcase Name
	Outstanding Request - IREAD_HOME Incoming Request - IREAD_HOME Resolution - Generate “ERROR” response		
	Outstanding Request - IREAD_HOME Incoming Request - READ_OWNER Resolution - No collision, forward to processor then generate “DONE” response		
	Outstanding Request - IREAD_HOME Incoming Request - READ_TO_OWN_HOME Resolution - Generate “ERROR” response		
	Outstanding Request - IREAD_HOME Incoming Request - READ_TO_OWN_OWNER Resolution - No collision, forward to processor then generate “DONE” response		
	Outstanding Request - IREAD_HOME Incoming Request - DKILL_HOME Resolution - Generate “ERROR” response		
	Outstanding Request - IREAD_HOME Incoming Request - DKILL_SHARER Resolution - No collision, forward to processor then generate “DONE” response		
	Outstanding Request - IREAD_HOME Incoming Request - CASTOUT Resolution- Generate “ERROR” response		
	Outstanding Request - IREAD_HOME Incoming Request - TLBIE Resolution - No collision, forward to processor then generate “DONE” response		
	Outstanding Request - IREAD_HOME Incoming Request - TLBSYNC Resolution- No collision, forward to processor then generate “DONE” response		
	Outstanding Request - IREAD_HOME Incoming Request - IKILL_HOME Resolution- Generate “ERROR” response		
	Outstanding Request - IREAD_HOME Incoming Request - IKILL_SHARER Resolution- No collision, forward to processor then generate “DONE” response		
	Outstanding Request - IREAD_HOME Incoming Request - FLUSH Resolution- Generate “ERROR” response		
	Outstanding Request - IREAD_HOME Incoming Request - IO_READ_HOME Resolution - Generate “ERROR” response		

Item No	Compliance Item	Spec Reference	Testcase Name
	Outstanding Request - IREAD_HOME Incoming Request - IO_READ_OWNER Resolution - No collision, forward to processor then generate “DONE” response		
4d	**Instruction Read Operation to Remote Shared Coherence Granule** If the requested instructions exists in the memory directory as shared, the instructions can be returned immediately from memory and the requesting processing element’s device ID is added to the sharing mask.	Part5, Sec 3.3.2	
4e	**Instruction Read Operation to Remote Modified Coherence Granule** If the requested data exists in the memory directory as modified, the up-to-date (current) data must be obtained from the owner. The home memory then sends a READ_OWNER request to the processing element that owns the coherence granule. The owner passes a copy of the data to the original requestor and to memory, memory is updated, and the directory state is changed from modified and owner to shared by the previous owner and the requesting processing element’s device ID.		
4f	**Instruction Read Operation to Local Modified Coherence Granule** If the processing element requesting a modified coherence granule happens to be the home for the memory the READ_OWNER transaction is used to obtain the coherence granule		
4g	**Instruction Read Operation Paradox Case** The home memory sends a READ_OWNER transaction back to the requesting processing element with the source and secondary ID set to the home memory ID, which indicates that the response behavior should be an INTERVENTION transaction rather than an INTERVENTION and a DATA_ONLY transaction		
4h	Cover the below possible combination when CRF is RSVD = 0 PRIO sets packet priority as follows: 00 - lowest priority 01 - medium priority 10 - high priority 11- highest priority	Part6, Sec 2.2	
	Cover the below possible combination when CRF is supported PRIO||CRF sets packet priority: 00 0 - lowest priority 00 1 - critical flow lowest priority 01 0 - medium priority 01 1 - critical flow medium priority 10 0 - high priority 10 1 - critical flow high priority 11 0 - highest priority 11 1 - critical flow high priority		

Item No	Compliance Item	Spec Reference	Testcase Name
5	**Read-for-Ownership Operations** READ_TO_OWN_HOME READ_TO_OWN_OWNER DKILL_SHARER	Part5, Sec 3.3.3	
	A read-for-ownership operation always returns one coherence granule-sized data payload.		
5a	**READ_TO_OWN_HOME** Read for store of home memory for coherence granule.	Part5, Sec 4.2	srio_ll_gsm_rd_to_own_home_test.sv
	The READ_TO_OWN_HOME transaction is used by a processing element that needs to read a writable copy of a coherence granule from a remote home memory on another processing element. This transaction causes a copy of the data to be returned to the requestor, from memory if the data is shared, or from the owner if it is modified	Part5, Sec 3.3.3	
5b	Type 2 request packets never include data.	Part5, Sec 4.2.6	
	0b0001 - READ_TO_OWN_HOME		

Item No	Compliance Item	Spec Reference	Testcase Name
5c	**Resolving an Outstanding READ_TO_OWN_HOME Transaction**	Part5, Sec 7.5	
	Outstanding Request - READ_TO_OWN_HOME Incoming Request - READ_HOME Resolution- Generate “ERROR” response		
	Outstanding Request - READ_TO_OWN_HOME Incoming Request - IREAD_HOME Resolution- Generate “ERROR” response		
	Outstanding Request - READ_TO_OWN_HOME Incoming Request - READ_OWNER Resolution - If outstanding request, wait for all expected responses. If final response is “DONE”, return data if necessary and forward READ_OWNER to processor and generate an “DONE_INTERVENTION” with data response and a “DATA_ONLY” to originator If final response is “RETRY” generate an “ERROR” response If no outstanding request generate an “NOT_OWNER”response.		
	Outstanding Request - READ_TO_OWN_HOME Incoming Request - READ_TO_OWN_HOME Resolution - Generate “ERROR” response		
	Outstanding Request - READ_TO_OWN_HOME Incoming Request - READ_TO_OWN_OWNER Resolution - If outstanding request, wait for all expected responses If final response is “DONE”, return data if necessary and forward READ_TO_OWN_OWNER to processor and generate an “DONE_INTERVENTION” with data response and a “DATA_ONLY” to originator If final response is “RETRY” generate an “ERROR” response		
	Outstanding Request - READ_TO_OWN_HOME Incoming Request - DKILL_HOME Resolution - Generate “ERROR” response		
	Outstanding Request - READ_TO_OWN_HOME Incoming Request - DKILL_SHARER Resolution - If outstanding request, wait for all expected responses If final response is “DONE” generate an “ERROR” response If final response is “RETRY” generate a “DONE” response and continue the READ_TO_OWN_HOME. If no outstanding request generate a “DONE” response		
	Outstanding Request - READ_TO_OWN_HOME Incoming Request - CASTOUT Resolution - Generate “ERROR” response		
	Outstanding Request - READ_TO_OWN_HOME Incoming Request - TLBIE Resolution - No collision, forward to processor then generate “DONE” response		

Item No	Compliance Item	Spec Reference	Testcase Name
	Outstanding Request- READ_TO_OWN_HOME Incoming Request **-** TLBSYNC Resolution - No collision, forward to processor then generate “DONE” response		
	Outstanding Request - READ_TO_OWN_HOME Incoming Request -IKILL_HOME Resolution - Generate “ERROR” response		
	Outstanding Request - READ_TO_OWN_HOME Incoming Request - IKILL_SHARER Resolution - No collision, forward to processor then generate “DONE” response		
	Outstanding Request - READ_TO_OWN_HOME Incoming Request - FLUSH Resolution - If outstanding request, wait for all expected responses If final response is “DONE”, return data if necessary and forward FLUSH to processor and generate a “DONE” with data response If final response is “RETRY” generate an “ERROR” response If no outstanding request generate an “ERROR” response		
	Outstanding Request - READ_TO_OWN_HOME Incoming Request - IO_READ_HOME Resolution - Generate “ERROR” response		
	Outstanding Request - READ_TO_OWN_HOME Incoming Request - IO_READ_OWNER Resolution - If outstanding request, wait for all expected responses If final response is “DONE”, return data if necessary and forward IO_READ_OWNER to processor then generate a “DONE” with data response If final response is “RETRY” generate an “ERROR” response If no outstanding request generate an “NOT_OWNER” response		
5d	Cover the below possible combination when CRF is RSVD = 0 PRIO sets packet priority as follows: 00 - lowest priority 01 - medium priority 10 - high priority 11- highest priority	Part6, Sec 2.2	

Item No	Compliance Item	Spec Reference	Testcase Name
	Cover the below possible combination when CRF is supported PRIO||CRF sets packet priority: 00 0 - lowest priority 00 1 - critical flow lowest priority 01 0 - medium priority 01 1 - critical flow medium priority 10 0 - high priority 10 1 - critical flow high priority 11 0 - highest priority 11 1 - critical flow high priority	Part6, Sec 2.2	
6	**READ_TO_OWN_OWNER** Read for store of remotely owned coherence granule	Part5, Sec 4.2	srio_ll_gsm_rd_to_own_owner_test.sv
	The READ_TO_OWN_OWNER transaction is used by a home memory processing element that needs to read a writable copy of a coherence granule that is owned by a remote processing element.	Part5, Sec 3.3.3	
	READ_TO_OWN_OWNER transaction used for data cache flush operations, which return ownership of a coherence granule back to the home memory if it is modified and invalidate all copies if the granule is shared.	Part5, Sec 3.3.9	
	The READ_TO_OWN_OWNER transaction is used by a home memory processing element that needs to retrieve ownership of a coherence granule that is owned by a remote processing element.	Part5, Sec 3.3.9	
	Type1 packet format used	Part5, Sec 4.2.5	
	Type 1 request packets never include data		
	Encodings for Type 1 Packets 0b0001 - READ_TO_OWN_OWNER		

Item No	Compliance Item	Spec Reference	Testcase Name
6a	Resolving an Outstanding READ_TO_OWN_OWNER Transaction	Part5, Sec 7.6	
	Outstanding Request - READ_TO_OWN_OWNER Incoming Request - READ_HOME Resolution - Generate “RETRY” response		
	Outstanding Request - READ_TO_OWN_OWNER Incoming Request - IREAD_HOME Resolution - Generate “RETRY” response		
	Outstanding Request - READ_TO_OWN_OWNER Incoming Request - READ_OWNER Resolution - Generate “ERROR” response		
	Outstanding Request - READ_TO_OWN_OWNER Incoming Request -READ_TO_OWN_HOME Resolution - Generate “RETRY” response		
	Outstanding Request - READ_TO_OWN_OWNER Incoming Request - READ_TO_OWN_OWNER Resolution - Generate “ERROR” response		
	Outstanding Request - READ_TO_OWN_OWNER Incoming Request - DKILL_HOME Resolution - Generate “RETRY” response		
	Outstanding Request - READ_TO_OWN_OWNER Incoming Request - DKILL_SHARER Resolution - Generate “ERROR” response		
	Outstanding Request - READ_TO_OWN_OWNER Incoming Request -CASTOUT Resolution - No collision, update directory state, generate “DONE” response		
	Outstanding Request - READ_TO_OWN_OWNER Incoming Request -TLBIE Resolution - No collision, forward to processor then generate “DONE” response		
	Outstanding Request -READ_TO_OWN_OWNER Incoming Request - TLBSYNC Resolution - No collision, forward to processor then generate “DONE” response		
	Outstanding Request -READ_TO_OWN_OWNER Incoming Request - IKILL_HOME Resolution - No collision, forward to processor, send IKILL_SHARER to all participants except requestor		
	Outstanding Request - READ_TO_OWN_OWNER Incoming Request - IKILL_SHARER Resolution - Generate “ERROR” response		
	Outstanding Request - READ_TO_OWN_OWNER Incoming Request - FLUSH Resolution- Generate “RETRY” response		

Item No	Compliance Item	Spec Reference	Testcase Name
	Outstanding Request - READ_TO_OWN_OWNER Incoming Request - IO_READ_HOME Resolution -Generate “RETRY” response		
	Outstanding Request - READ_TO_OWN_OWNER Incoming Request -IO_READ_OWNER Resolution -Generate “ERROR” response		
6b	Cover the below possible combination when CRF is RSVD = 0 PRIO sets packet priority as follows: 00 - lowest priority 01 - medium priority 10 - high priority 11- highest priority	Part6, Sec 2.2	
	Cover the below possible combination when CRF is supported PRIO||CRF sets packet priority: 00 0 - lowest priority 00 1 - critical flow lowest priority 01 0 - medium priority 01 1 - critical flow medium priority 10 0 - high priority 10 1 - critical flow high priority 11 0 - highest priority 11 1 - critical flow high priority		
7	**DKILL_SHARER** Invalidate cached copy of coherence granule	Part5, Sec 4.2	srio_ll_gsm_dkill_sharer_test.sv
	The DKILL_SHARER transaction is used by the home memory processing element to invalidate shared copies of the coherence granule in remote processing elements	Part5, Sec 3.3.3	
	DKILL_SHARER transactions used for data cache flush operations	Part5, Sec 3.3.9	
	Type 2 request packets never include data	Part5, Sec 4.2.6	
	Transaction Field Encodings for Type 2 Packets 0b1011 - DKILL_SHARER	Part5, Sec 4.2.6	

Item No	Compliance Item	Spec Reference	Testcase Name
7a	**Resolving an Outstanding DKILL_SHARER Transaction**	Part5, Sec 7.8	
	Outstanding Request - DKILL_SHARER Incoming Request - READ_HOME Resolution - Generate “RETRY” response		
	Outstanding Request - DKILL_SHARER Incoming Request - IREAD_HOME Resolution - Generate “RETRY” response		
	Outstanding Request - DKILL_SHARER Incoming Request - READ_OWNER Resolution -Generate “ERROR” response		
	Outstanding Request - DKILL_SHARER Incoming Request - READ_TO_OWN_HOME Resolution - Generate “RETRY” response		
	Outstanding Request - DKILL_SHARER Incoming Request - READ_TO_OWN_OWNER Resolution - Generate “ERROR” response		
	Outstanding Request - DKILL_SHARER Incoming Request - DKILL_HOME Resolution - Generate “RETRY” response		
	Outstanding Request - DKILL_SHARER Incoming Request -DKILL_SHARER Resolution - Generate “ERROR” response		
	Outstanding Request - DKILL_SHARER Incoming Request -CASTOUT Resolution - Generate “ERROR” response		
	Outstanding Request - DKILL_SHARER Incoming Request -TLBIE Resolution - No collision, forward to processor then generate “DONE” response		
	Outstanding Request - DKILL_SHARER Incoming Request -TLBSYNC Resolution - No collision, forward to processor then generate “DONE”		
	Outstanding Request - DKILL_SHARER Incoming Request - IKILL_HOME Resolution - No collision, forward to processor, send IKILL_SHARER to all participants except requestor		
	Outstanding Request - DKILL_SHARER Incoming Request -IKILL_SHARER Resolution - Generate “ERROR” response		
	Outstanding Request - DKILL_SHARER Incoming Request -FLUSH Resolution - Generate “RETRY” response		

Item No	Compliance Item	Spec Reference	Testcase Name
	Outstanding Request - DKILL_SHARER Incoming Request - IO_READ_HOME Resolution - If processing element is HOME: generate a “RETRY” response If processing element is not HOME: If outstanding request, wait for all expected responses If final response is “DONE” forward IO_READ to processor then generate a “DONE” with data response If final response is “RETRY” generate an “ERROR” response If no outstanding request generate an “ERROR” response		
	Outstanding Request - DKILL_SHARER Incoming Request - IO_READ_OWNER Resolution - Generate “ERROR” response		
7b	Cover the below possible combination when CRF is RSVD = 0 PRIO sets packet priority as follows: 00 - lowest priority 01 - medium priority 10 - high priority 11- highest priority	Part6, Sec 2.2	
	Cover the below possible combination when CRF is supported PRIO||CRF sets packet priority: 00 0 - lowest priority 00 1 - critical flow lowest priority 01 0 - medium priority 01 1 - critical flow medium priority 10 0 - high priority 10 1 - critical flow high priority 11 0 - highest priority 11 1 - critical flow high priority		
8	**DKILL_HOME** Invalidate to home memory of coherence granule	Part5, Sec 4.2	srio_ll_gsm_dkill_home_test.sv
	The DKILL_HOME transaction is used by a processing element to invalidate a data coherence granule that has home memory in a remote processing element.	Part5, Sec 3.3.4	
	Type 2 request packets never include data	Part5, Sec 4.2.6	
	Transaction Field Encodings for Type 2 Packets 0b0011 - DKILL_HOME		
8a	**Resolving an Outstanding DKILL_HOME Transaction**	Part5,Sec 7.7	
	Outstanding Request - DKILL_HOME Incoming Request - READ_HOME Resolution - Generate ERROR response		

Item No	Compliance Item	Spec Reference	Testcase Name
	Outstanding Request - DKILL_HOME Incoming Request - IREAD_HOME Resolution - Generate ERROR response		
	Outstanding Request - DKILL_HOME Incoming Request - READ_OWNER Resolution - If outstanding request, wait for all expected responses. If final response is “DONE”, return data if necessary and forward READ_OWNER to processor and generate a “DONE_INTERVENTION” with data response and a “DATA_ONLY” to originator If final response is “RETRY” generate an “ERROR” response If no outstanding request generate an “ERROR” response		
	Outstanding Request - DKILL_HOME Incoming Request - READ_TO_OWN_HOME Resolution - Generate “ERROR” response		
	Outstanding Request - DKILL_HOME Incoming Request - READ_TO_OWN_OWNER Resolution - If outstanding request, wait for all expected responses If final response is “DONE” forward READ_TO_OWN_OWNER to processor and generate a “DONE_INTERVENTION” with data response and a “DATA_ONLY” to originator If final response is “RETRY” generate an “ERROR” response If no outstanding request generate an “ERROR” response		
	Outstanding Request - DKILL_HOME Incoming Request -DKILL_HOME Resolution - Generate “ERROR” response		
	Outstanding Request - DKILL_HOME Incoming Request -DKILL_SHARER Resolution - If outstanding request, wait for all expected responses If final response is “DONE” generate an “ERROR” response If final response is “RETRY” cancel the data cache invalidate at the processor and forward DKILL_SHARER to processor then generate a “DONE” response If no outstanding request, cancel the data cache invalidate at the processor and forward DKILL_SHARER to processor then generate a “DONE” response		
	Outstanding Request - DKILL_HOME Incoming Request -CASTOUT Resolution - Generate “ERROR” response		
	Outstanding Request - DKILL_HOME Incoming Request - TLBIE Resolution - No collision, forward to processor then generate “DONE” response		

Item No	Compliance Item	Spec Reference	Testcase Name
	Outstanding Request - DKILL_HOME Incoming Request -TLBSYNC Resolution - No collision, forward to processor then generate “DONE” response		
	Outstanding Request - DKILL_HOME Incoming Request -IKILL_HOME Resolution - Generate “ERROR” response		
	Outstanding Request - DKILL_HOME Incoming Request -IKILL_SHARER Resolution - No collision, forward to processor then generate “DONE” response		
	Outstanding Request - DKILL_HOME Incoming Request -FLUSH Resolution - Generate “ERROR” response		
	Outstanding Request - DKILL_HOME Incoming Request -IO_READ_HOME Resolution - Generate “ERROR” response		
	Outstanding Request - DKILL_HOME Incoming Request - IO_READ_OWNER Resolution - If outstanding request, wait for all expected responses If final response is “DONE” forward IO_READ_OWNER to processor then generate a “DONE” with data response If final response is “RETRY” generate an “ERROR” response If no outstanding request generate an “ERROR” response		
8b	Data Cache Invalidate Operation to Remote Shared Coherence Granule If the requestor is not on the same processing element as the home memory of the coherence granule, a DKILL_HOME transaction is sent to the remote home memory processing element. This causes the home memory for the shared coherence granule to send a DKILL_SHARER to all processing elements marked as sharing the granule in the memory directory state except for the requestor. The final memory state shows that the coherence granule is modified and owned by the requesting processing element’s device ID.	Part5, Sec 3.3.4	
8c	Data Cache Invalidate Operation to Local Shared Coherence Granule If the requestor is on the same processing element as the home memory of the coherence granule, the home memory sends a DKILL_SHARER transaction to all processing elements marked as sharing the coherence granule in the memory directory. The final memory state shows the coherence granule modified and owned by the local processor		
9	**Castout Operations** Return ownership of coherence granule to home memory	Part5, Sec 4.2	srio_ll_gsm_castout_test.sv
	The CASTOUT and RESPONSE transactions are used in a castout operation by a processing element to relinquish its ownership of a coherence granule and return it to the home memory	Part5, Sec 3.3.5	

Item No	Compliance Item	Spec Reference	Testcase Name
	The CASTOUT can be treated as a low-priority transaction unless there is an address collision with an incoming request, at which time it must become a high-priority transaction	Part5, Sec 3.3.5	
	A CASTOUT transaction does not participate in address collision detection at the home memory to prevent deadlocks or cache paradoxes caused by packet-to-packet timing in the interconnect fabric		
9a	Type 5 packets always contain data	Part5, Sec 4.2.8	
	Transaction Field Encodings for Type 5 Packets 0b0000 - CASTOUT		
9b	**Resolving an Outstanding CASTOUT Transaction**	Part5, Sec 7.11	
	Outstanding Request **-** CASTOUT Incoming Request **-** READ_HOME Resolution - Generate “ERROR” response		
	Outstanding Request **-** CASTOUT Incoming Request **-** IREAD_HOME Resolution - Generate “ERROR” response		
	Outstanding Request - CASTOUT Incoming Request - READ_OWNER Resolution - Generate “RETRY” response		
	Outstanding Request - CASTOUT Incoming Request - READ_TO_OWN_HOME Resolution - Generate “ERROR” response		
	Outstanding Request - CASTOUT Incoming Request - READ_TO_OWN_OWNER Resolution - Generate “RETRY” response		
	Outstanding Request - CASTOUT Incoming Request - DKILL_HOME Resolution- Generate “ERROR” response		
	Outstanding Request -CASTOUT Incoming Request - DKILL_SHARER Resolution - Generate “ERROR” response		
	Outstanding Request -CASTOUT Incoming Request - CASTOUT Resolution - Generate “ERROR” response		
	Outstanding Request -CASTOUT Incoming Request - TLBIE Resolution - No collision, forward to processor then generate “DONE” response		
	Outstanding Request -CASTOUT Incoming Request - TLBSYNC Resolution -No collision, forward to processor then generate “DONE” response.		

Item No	Compliance Item	Spec Reference	Testcase Name
	Outstanding Request -CASTOUT Incoming Request - IKILL_HOME Resolution - Generate “ERROR” response		
	Outstanding Request -CASTOUT Incoming Request - IKILL_SHARER Resolution - No collision, forward to processor then generate “DONE” response		
	Outstanding Request -CASTOUT Incoming Request - FLUSH Resolution - Generate “ERROR” response		
	Outstanding Request -CASTOUT Incoming Request -IO_READ_HOME Resolution - Generate “ERROR” response		
	Outstanding Request -CASTOUT Incoming Request - IO_READ_OWNER Resolution - Generate “RETRY” response		
10	TLBIE Invalidate TLB entry	Part5, Sec 4.2	srio_ll_gsm_tlbie_test.sv
10a	The TLBIE and RESPONSE transactions are used for TLB invalidate-entry operations	Part5, Sec 3.3.6	
	The TLBIE transaction is sent to all participants in the coherence domain except for the original requestor		
10b	TLBIE are address-only transactions so the rdsize and wdptr fields are ignored and shall be set to logic 0.	Part6, Sec 4.2.6	
	Transaction Field Encodings for Type 2 Packets 0b0110 TLBIE		
10c	**Resolving an Outstanding TLBIE or TLBSYNC Transaction**	Part6, Sec 7.12	
	Outstanding Request **-** TLBIE Incoming Request - Any Resolution - No collision, process request		
11	TLBSYNC Synchronize TLB invalidates	Part5, Sec 4.2	srio_ll_gsm_tlbsync_test.sv
11a	The TLBSYNC and RESPONSE transactions are used for TLB invalidate-entry synchronization operations	Part5, Sec 3.3.7	
	The TLBSYNC transaction is sent to all participants in the coherence domain except for the original requestor.		
11b	TLBSYNC is a transaction-type-only transaction so both the address, xamsbs, rdsize, and wdptr fields shall be set to logic 0	Part5, Sec 4.2.6	
	Transaction Field Encodings for Type 2 Packets 0b0111 TLBSYNC		
11c	**Resolving an Outstanding TLBIE or TLBSYNC Transaction**	Part6, Sec 7.12	

Item No	Compliance Item	Spec Reference	Testcase Name
	Outstanding Request - TLBSYNC Incoming Request - Any Resolution - No collision, process request		
12	**IKILL_HOME** Invalidate to home memory of coherence granule	Part5, Sec 4.2	srio_ll_gsm_ikill_home_test.sv
	The IKILL_HOME, IKILL_SHARER, and RESPONSE transactions are used during instruction cache invalidate operations to invalidate shared copies of an instruction coherence granule in remote processing elements.	Part5, Sec 3.3.8	
12a	**Instruction Cache Invalidate Operation to Remote Sharable Coherence Granule**		
	If the requestor is not on the same processing element as the home memory of the coherence granule, an IKILL_HOME transaction is sent to the remote home memory processing element. This causes the home memory for the shared coherence granule to send an IKILL_SHARER to all processing element participants in the coherence domain because the memory directory state only properly tracks data, not instruction, accesses		
12b	**Transaction Field Encodings for Type 2 Packets** 0b0101 IKILL_HOME	Part6, Sec 4.2.6	
12c	**Resolving an Outstanding IKILL_HOME Transaction**	Part5, Sec 7.9	
	Outstanding request - IKILL_HOME Incoming Request - READ_HOME Resolution - Generate “ERROR” response		
	Outstanding request - IKILL_HOME Incoming Request - IREAD_HOME Resolution - Generate “ERROR” response		
	Outstanding request - IKILL_HOME Incoming Request - READ_OWNER Resolution - No collision, process normally		
	Outstanding request - IKILL_HOME Incoming Request - READ_TO_OWN_HOME Resolution - Generate “ERROR” response		
	Outstanding request - IKILL_HOME Incoming Request - READ_TO_OWN_OWNER Resolution - No collision, process normally		
	Outstanding request - IKILL_HOME Incoming Request - DKILL_HOME Resolution - Generate “ERROR” response		
	Outstanding request - IKILL_HOME Incoming Request - DKILL_SHARER Resolution - No collision, process normally		
	Outstanding request - IKILL_HOME Incoming Request - CASTOUT Resolution - No collision, process normally		

Item No	Compliance Item	Spec Reference	Testcase Name
	Outstanding request - IKILL_HOME Incoming Request - TLBIE Resolution - No collision, forward to processor then generate “DONE” response		
	Outstanding request - IKILL_HOME Incoming Request - TLBSYNC Resolution - No collision, forward to processor then generate “DONE” response		
	Outstanding request - IKILL_HOME Incoming Request -IKILL_HOME Resolution - Generate “ERROR” response		
	Outstanding request - IKILL_HOME Incoming Request -IKILL_SHARER Resolution -No collision, forward to processor then generate “DONE” response		
	Outstanding request - IKILL_HOME Incoming Request - FLUSH Resolution - Generate “ERROR” response		
	Outstanding request - IKILL_HOME Incoming Request - IO_READ_HOME Resolution - No collision, process normally		
13	IKILL_SHARER Invalidate cached copy of coherence granule	Part5, Sec 4.2	srio_ll_gsm_ikill_sharer_test.sv
13a	The IKILL_HOME, IKILL_SHARER, and RESPONSE transactions are used during instruction cache invalidate operations to invalidate shared copies of an instruction coherence granule in remote processing elements	Part5, Sec 3.3.8	
	Instruction Cache Invalidate Operation to Local Sharable Coherence Granule		
	If the requestor is on the same processing element as the home memory of the coherence granule, the home memory sends an IKILL_SHARER transaction to all processing element participants in the coherence domain		
13b	Transaction Field Encodings for Type 2 Packets	Part5, Sec 4.2.6	
	0b1010 IKILL_SHARER		
13c	**Resolving an Outstanding IKILL_SHARER Transaction**	Part5, Sec 7.10	
	Outstanding request -IKILL_SHARER Incoming Request - READ_HOME Resolution - No collision, process normally		
	Outstanding request -IKILL_SHARER Incoming Request - IREAD_HOME Resolution - No collision, process normally		
	Outstanding request -IKILL_SHARER Incoming Request - READ_OWNER Resolution - Generate “ERROR” response		

Item No	Compliance Item	Spec Reference	Testcase Name
	Outstanding request -IKILL_SHARER Incoming Request - READ_TO_OWN_HOME Resolution - No collision, process normally		
	Outstanding request -IKILL_SHARER Incoming Request - READ_TO_OWN_OWNER Resolution - Generate “ERROR” response		
	Outstanding request -IKILL_SHARER Incoming Request - DKILL_HOME Resolution - No collision, process normally		
	Outstanding request -IKILL_SHARER Incoming Request - DKILL_SHARER Resolution - Generate “ERROR” response		
	Outstanding request -IKILL_SHARER Incoming Request - CASTOUT Resolution - No collision, process normally		
	Outstanding request -IKILL_SHARER Incoming Request - TLBIE Resolution - No collision, forward to processor then generate “DONE” response		
	Outstanding request -IKILL_SHARER Incoming Request - TLBSYNC Resolution - No collision, forward to processor then generate “DONE” response		
	Outstanding request -IKILL_SHARER Incoming Request - IKILL_HOME Resolution - No collision, forward to processor, send IKILL_SHARER to all participants except requestor		
	Outstanding request -IKILL_SHARER Incoming Request - IKILL_SHARER Resolution - Generate “ERROR” response		
	Outstanding request -IKILL_SHARER Incoming Request - FLUSH Resolution - No collision, process normally		
	Outstanding request -IKILL_SHARER Incoming Request - IO_READ_HOME Resolution - If processing element is HOME: generate a “RETRY” response If processing element is not HOME If outstanding request, wait for all expected responses. If final response is “DONE” forward IO_READ to processor then generate a “DONE” with data response If final response is “RETRY” generate an “ERROR” response If no outstanding request generate an “ERROR” response		

Item No	Compliance Item	Spec Reference	Testcase Name
	Outstanding request -IKILL_SHARER Incoming Request -IO_READ_OWNER Resolution - Generate “ERROR” response		
14	**FLUSH** Force return of ownership of coherence granule to home memory, no update to coherence granule	Part5, Sec 4.2	srio_ll_gsm_flush_test.sv
	Flush operations may also have no data payload in order to support cache manipulation instructions.	Part5, Sec 3.3	
	The FLUSH, DKILL_SHARER, READ_TO_OWN_OWNER, and RESPONSE transactions are used for data cache flush operations	Part5, Sec 3.3.9	
	The FLUSH transaction is used by a processing element to return the ownership and current data of a coherence granule to home memory. The data payload for the FLUSH transaction is typically the size of the coherence granule for the system but may be multiple double-words or one double-word or less. FLUSH transactions without a data payload are used to support cache manipulation operations. The memory directory state is changed to owned by home memory and shared (or modified, depending upon the processing element’s normal default state) by the local processing element.		
	The FLUSH transaction is able to specify multiple double-word and sub-double-word data payloads;		
	Multiple double-word FLUSH transactions cannot exceed the number of double-words in the coherence granule		
	Flush Operation to Remote Shared Coherence Granule		
	If a flush operation is to a remote shared coherence granule, the FLUSH transaction is sent to the home memory, which sends a DKILL_SHARER transaction to all of the processing elements marked in the sharing list except for the requesting processing element. The processing elements that receive the DKILL_SHARER transaction invalidate the specified address if it is found shared in their caching hierarchy		
	Flush Operation to Remote Modified Coherence Granule		
	If the coherence granule is owned by a remote processing element, the home memory sends a READ_TO_OWN_OWNER transaction to it with the secondary (intervention) ID set to the home memory ID instead of the requestor ID. The owner then invalidates the coherence granule in its caching hierarchy and returns the coherence granule data		
	Flush Operation to Local Shared Coherence Granule		
	If the requestor and the home memory for the coherence granule are in the same processing element, DKILL_SHARER transactions are sent to all participants marked in the sharing list		
	Flush Operation to Local Modified Coherence Granule		
	If the requestor and the home memory for the coherence granule are in the same processing element but the coherence granule is owned by a remote processing element, a READ_TO_OWN_OWNER transaction is sent to the owner		

Item No	Compliance Item	Spec Reference	Testcase Name
14a	Type 2 format used for FLUSH without data Encoding - 0b1001 FLUSH without data	Part5, Sec 4.2.6	
14b	The FLUSH with data and CASTOUT transactions use type 5 packets Encoding - 0b0001 FLUSH with data	Part5, Sec 4.2.8	
14c	**Resolving an Outstanding FLUSH Transaction**	Part5, Sec 7.13	
	Outstanding request -FLUSH Incoming Request - READ_HOME Resolution -Generate “ERROR” response		
	Outstanding request -FLUSH Incoming Request - IREAD_HOME Resolution -Generate “ERROR” response		
	Outstanding request -FLUSH Incoming Request - READ_OWNER Resolution -Generate “NOT_OWNER” response		
	Outstanding request -FLUSH Incoming Request - READ_TO_OWN_HOME Resolution - Generate “ERROR” response		
	Outstanding request -FLUSH Incoming Request - READ_TO_OWN_OWNER Resolution - Generate “NOT_OWNER” response		
	Outstanding request -FLUSH Incoming Request - DKILL_HOME Resolution -Generate “ERROR” response		
	Outstanding request -FLUSH Incoming Request - DKILL_SHARER Resolution - If outstanding request, wait for all expected responses If final response is “DONE” generate an “ERROR” response If final response is “RETRY” cancel the flush at the processor and forward DKILL_SHARER to processor then generate a “DONE” response If no outstanding request, cancel the data cache invalidate at the processor and forward DKILL_SHARER to processor then generate a “DONE” response.		
	Outstanding request -FLUSH Incoming Request - CASTOUT Resolution - Generate “ERROR” response		
	Outstanding request -FLUSH Incoming Request - TLBIE Resolution - No collision, forward to processor then generate “DONE” response		

Item No	Compliance Item	Spec Reference	Testcase Name
	Outstanding request -FLUSH Incoming Request - TLBSYNC Resolution - No collision, forward to processor then generate “DONE” response		
	Outstanding request -FLUSH Incoming Request - IKILL_HOME Resolution - Generate “ERROR” response		
	Outstanding request -FLUSH Incoming Request - IKILL_SHARER Resolution - No collision, forward to processor then generate “DONE” response		
	Outstanding request -FLUSH Incoming Request - FLUSH Resolution - Generate “ERROR” response		
	Outstanding request -FLUSH Incoming Request - IO_READ_HOME Resolution - Generate “ERROR” response		
	Outstanding request -FLUSH Incoming Request - IO_READ_OWNER Resolution - Generate “NOT_OWNER” response		
14d	**Address Collision Resolution for Non-participant FLUSH**	Part5, Sec 7.13	
	Outstanding request -FLUSH Incoming Request - READ_HOME Resolution -Generate “ERROR” response		
	Outstanding request -FLUSH Incoming Request - IREAD_HOME Resolution -Generate “ERROR” response		
	Outstanding request -FLUSH Incoming Request - READ_OWNER Resolution -Generate “ERROR” response		
	Outstanding request -FLUSH Incoming Request - READ_TO_OWN_HOME Resolution - Generate “ERROR” response		
	Outstanding request -FLUSH Incoming Request - READ_TO_OWN_OWNER Resolution -Generate “ERROR” response		
	Outstanding request -FLUSH Incoming Request - DKILL_HOME Resolution -Generate “ERROR” response		

Item No	Compliance Item	Spec Reference	Testcase Name
	Outstanding request -FLUSH Incoming Request - DKILL_SHARER Resolution - Generate “ERROR” response		
	Outstanding request -FLUSH Incoming Request - CASTOUT Resolution - Generate “ERROR” response		
	Outstanding request -FLUSH Incoming Request - TLBIE Resolution - No collision, forward to processor then generate “DONE” response		
	Outstanding request -FLUSH Incoming Request - TLBSYNC Resolution - No collision, forward to processor then generate “DONE” response		
	Outstanding request -FLUSH Incoming Request - IKILL_HOME Resolution - Generate “ERROR” response		
	Outstanding request -FLUSH Incoming Request - IKILL_SHARER Resolution - No collision, forward to processor then generate “DONE” response		
	Outstanding request -FLUSH Incoming Request- FLUSH Resolution - Generate “ERROR” response		
	Outstanding request -FLUSH Incoming Request - IO_READ_HOME Resolution - Generate “ERROR” response		
	Outstanding request -FLUSH Incoming Request - IO_READ_OWNER Resolution - Generate “ERROR” response		
15	IO_READ_HOME Read for I/O of home memory for coherence granule	Part5, Sec 4.2	srio_ll_gsm_io_rd_home_test.sv
15a	The IO_READ_HOME, IO_READ_OWNER, and RESPONSE transactions are used during I/O read operations by a processing element that needs a current copy of cache-coherent data from the memory system, but does not need to be added to the sharing list in the memory directory state.	Part5, Sec 3.3.10	
	The IO_READ_HOME transaction is used by a requestor that is not in the same processing element as the home memory for the coherence granule		
	I/O Read Operation to Remote Shared Coherence Granule		
	If the requested data exists in the memory directory as shared, the data can be returned immediately from memory and the sharing mask is not modified		
	I/O Read Operation to Remote Modified Coherence Granule		

Item No	Compliance Item	Spec Reference	Testcase Name
	If the requested data exists in the memory directory as modified, the home memory sends an IO_READ_OWNER transaction to the processing element that owns the coherence granule. The owner passes a copy of the data to the requesting processing element (intervention) but retains ownership of and responsibility for the coherence granule		
	I/O Read Operation to Local Modified Coherence Granule		
15b	**Transaction Field Encodings for Type 2 Packets** 0b0010 IO_READ_HOME		
15c	**Resolving an Outstanding IO_READ_HOME Transaction**	Part5, Sec 7.14	
	Outstanding request -IO_READ_HOME Incoming Request - READ_HOME Resolution - Generate “ERROR” response		
	Outstanding request -IO_READ_HOME Incoming Request -IREAD_HOME Resolution - Generate “ERROR” response		
	Outstanding request -IO_READ_HOME Incoming Request -READ_OWNER Resolution - Generate “NOT_OWNER” response		
	Outstanding request -IO_READ_HOME Incoming Request -READ_TO_OWN_HOME Resolution - Generate “ERROR” response		
	Outstanding request -IO_READ_HOME Incoming Request -READ_TO_OWN_OWNER Resolution - Generate “NOT_OWNER” response		
	Outstanding request -IO_READ_HOME Incoming Request -DKILL_HOME Resolution - Generate “ERROR” response		
	Outstanding request -IO_READ_HOME Incoming Request -DKILL_SHARER Resolution - If outstanding request, wait for all expected responses If final response is “DONE”, return data if necessary and forward DKILL_SHARER to processor then generate a “DONE” response. If final response is “RETRY” forward DKILL_SHARED to processor then generate a “DONE” response If no outstanding request forward DKILL_SHARER to processor then generate a “DONE” response		
	Outstanding request -IO_READ_HOME Incoming Request -CASTOUT Resolution - Generate “ERROR” response		
	Outstanding request -IO_READ_HOME Incoming Request -TLBIE Resolution - No collision, forward to processor then generate “DONE” response		

Item No	Compliance Item	Spec Reference	Testcase Name
	Outstanding request -IO_READ_HOME Incoming Request -TLBSYNC Resolution - No collision, forward to processor then generate “DONE” response		
	Outstanding request -IO_READ_HOME Incoming Request -IKILL_HOME Resolution - Generate “ERROR” response		
	Outstanding request -IO_READ_HOME Incoming Request -IKILL_SHARER Resolution - No collision, forward to processor then generate “DONE” response		
	Outstanding request -IO_READ_HOME Incoming Request -FLUSH Resolution		
	Outstanding request -IO_READ_HOME Incoming Request -IO_READ_HOME Resolution - Generate “ERROR” response		
	Outstanding request -IO_READ_HOME Incoming Request -IO_READ_OWNER Resolution - Generate “NOT_OWNER” response		
15d	**Address Collision Resolution for Non-participant IO_READ_HOME**	Part5, Sec 7.14	
	Outstanding request -IO_READ_HOME Incoming Request -READ_HOME Resolution - Generate “ERROR” response		
	Outstanding request -IO_READ_HOME Incoming Request -IREAD_HOME Resolution - Generate “ERROR” response		
	Outstanding request -IO_READ_HOME Incoming Request -READ_OWNER Resolution - Generate “ERROR” response		
	Outstanding request -IO_READ_HOME Incoming Request -READ_TO_OWN_HOME Resolution - Generate “ERROR” response		
	Outstanding request -IO_READ_HOME Incoming Request -READ_TO_OWN_OWNER Resolution - Generate “ERROR” response		
	Outstanding request -IO_READ_HOME Incoming Request -DKILL_HOME Resolution - Generate “ERROR” response		
	Outstanding request -IO_READ_HOME Incoming Request -DKILL_SHARER Resolution - Generate “ERROR” response		

Item No	Compliance Item	Spec Reference	Testcase Name
	Outstanding request -IO_READ_HOME Incoming Request -CASTOUT Resolution - Generate “ERROR” response		
	Outstanding request -IO_READ_HOME Incoming Request -TLBIE Resolution - No collision, forward to processor then generate “DONE” response		
	Outstanding request -IO_READ_HOME Incoming Request -TLBSYNC Resolution - No collision, forward to processor then generate “DONE” response		
	Outstanding request -IO_READ_HOME Incoming Request -IKILL_HOME Resolution -Generate “ERROR” response		
	Outstanding request -IO_READ_HOME Incoming Request -IKILL_SHARER Resolution -No collision, forward to processor then generate “DONE” response		
	Outstanding request -IO_READ_HOME Incoming Request -FLUSH Resolution -Generate “ERROR” response		
	Outstanding request -IO_READ_HOME Incoming Request -IO_READ_HOME Resolution -Generate “ERROR” response		
	Outstanding request -IO_READ_HOME Incoming Request -IO_READ_OWNER Resolution -Generate “ERROR” response		

Item No	Compliance Item	Spec Reference	Testcase Name
16	IO_READ_OWNER Read for I/O of remotely owned coherence granule	Part5, Sec 4.2	srio_ll_gsm_io_rd_owner_test.sv
	The IO_READ_HOME, IO_READ_OWNER, and RESPONSE transactions are used during I/O read operations by a processing element that needs a current copy of cache-coherent data from the memory system, but does not need to be added to the sharing list in the memory directory state.	Part5, Sec 3.3.10	
	The IO_READ_OWNER transaction is used by a home memory processing element that needs to read a copy of a coherence granule owned by a remote processing element.		
16a	Type 1 packet used for IO_READ_OWNER	Part5, Sec 4.2.5	
	Encoding 0b0010 IO_READ_OWNER		
16b	**Resolving an Outstanding IO_READ_OWNER Transaction**	Part5, Sec 7.15	
	Outstanding request -IO_READ_OWNER Incoming Request -READ_HOME Resolution - Generate “RETRY” response		
	Outstanding request -IO_READ_OWNER Incoming Request -IREAD_HOME Resolution - Generate “RETRY” response		
	Outstanding request -IO_READ_OWNER Incoming Request -READ_OWNER Resolution - Generate “ERROR” response		
	Outstanding request -IO_READ_OWNER Incoming Request -READ_TO_OWN_HOME Resolution - Generate “RETRY” response		
	Outstanding request -IO_READ_OWNER Incoming Request -READ_TO_OWN_OWNER Resolution - Generate “ERROR” response		
	Outstanding request -IO_READ_OWNER Incoming Request -DKILL_HOME Resolution - Generate “RETRY” response		
	Outstanding request -IO_READ_OWNER Incoming Request -DKILL_SHARER Resolution - Generate “ERROR” response		
	Outstanding request -IO_READ_OWNER Incoming Request -CASTOUT Resolution - No collision, update directory state and memory, generate DONE response (		
	Outstanding request -IO_READ_OWNER Incoming Request -TLBIE Resolution - No collision, forward to processor then generate “DONE” response		

Item No	Compliance Item	Spec Reference	Testcase Name
	Outstanding request -IO_READ_OWNER Incoming Request -TLBSYNC Resolution - No collision, forward to processor then generate “DONE” response		
	Outstanding request -IO_READ_OWNER Incoming Request -IKILL_HOME Resolution - No collision, forward to processor, send IKILL_SHARER to all participants except requestor		
	Outstanding request -IO_READ_OWNER Incoming Request -IKILL_SHARER Resolution - Generate “ERROR” response		
	Outstanding request -IO_READ_OWNER Incoming Request -FLUSH Resolution - Generate “RETRY” response		
	Outstanding request -IO_READ_OWNER Incoming Request -IO_READ_HOME Resolution - Generate “RETRY” response		
	Outstanding request -IO_READ_OWNER Incoming Request -IO_READ_OWNER Resolution - Generate “ERROR” response		
16c	**Address Collision Resolution for Non-participant IO_READ_OWNER**	Part5, Sec 7.15	
	Outstanding request -IO_READ_OWNER Incoming Request -READ_HOME Resolution - Generate “ERROR” response		
	Outstanding request -IO_READ_OWNER Incoming Request -IREAD_HOME Resolution - Generate “ERROR” response		
	Outstanding request -IO_READ_OWNER Incoming Request -READ_OWNER Resolution - Generate “ERROR” response		
	Outstanding request -IO_READ_OWNER Incoming Request -READ_TO_OWN_HOME Resolution - Generate “ERROR” response		
	Outstanding request -IO_READ_OWNER Incoming Request -READ_TO_OWN_OWNER Resolution - Generate “ERROR” response		
	Outstanding request -IO_READ_OWNER Incoming Request -DKILL_HOME Resolution - Generate “ERROR” response		
	Outstanding request -IO_READ_OWNER Incoming Request -DKILL_SHARER Resolution - Generate “ERROR” response		

Item No	Compliance Item	Spec Reference	Testcase Name
	Outstanding request -IO_READ_OWNER Incoming Request -CASTOUT Resolution - Generate “ERROR” response		
	Outstanding request -IO_READ_OWNER Incoming Request -TLBIE Resolution - No collision, forward to processor then generate “DONE” response		
	Outstanding request -IO_READ_OWNER Incoming Request -TLBSYNC Resolution - No collision, forward to processor then generate “DONE” response		
	Outstanding request -IO_READ_OWNER Incoming Request -IKILL_HOME Resolution - No collision, forward to processor, send IKILL_SHARER to all participants except requestor		
	Outstanding request -IO_READ_OWNER Incoming Request -IKILL_SHARER Resolution - Generate “ERROR” response		
	Outstanding request -IO_READ_OWNER Incoming Request -FLUSH Resolution - Generate “ERROR” response		
	Outstanding request -IO_READ_OWNER Incoming Request -IO_READ_HOME Resolution - Generate “ERROR” response		
	Outstanding request -IO_READ_OWNER Incoming Request -IO_READ_OWNER Resolution - Generate “ERROR” response		

6.2 Detectable Error Compliance

Table 14: Detectable Error Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	READ_HOME		srio_ll_gsm_rd_home_err_test.sv

Item No	Compliance Item	Spec Reference	Testcase Name
1a	Type 2 request packets never include data	Part5, Sec 4.2.6	
	• Received READ_HOME request with data payload		
1b	Bit[14] (Received packet exceeds 276 Bytes) in Port n Error Detect CSR. Received packet that exceeds the maximum allowed size.	Part8, Sec 2.5.15	
	Packet that exceeds the maximum packet size. Reception of packet (that exceeds the maximum packet size) causes the receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery	Part6, Sec 6.13.2.4	
	• Received READ_HOME data payload greater than 256 bytes		
1c	Bit[9] (Unsupported Transaction) Logical/Transport Layer Error Detect CSR- A transaction is received that is not supported in the Destination Operations	Part8, Sec 2.5.3	
	• Change the Destination Operation CAR bit[1] to zero. READ_HOME request received.	Part5, Sec 5.4.2	
1d	Cyclic Redundancy Code used to detect transmission errors in the packet	Part6, Sec 2.2	
	Packet with an incorrect CRC value The receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	Bit[13] (Received packet with bad CRC) in Port n Error Detect CSR	Part8, Sec 2.5.15	
	• Received READ_HOME request packet with incorrect CRC.		
	• Received READ_HOME response packet with incorrect CRC		
1e	Bit[12] (Received packet with unexpected ackID) Port n Error Detect CSR	Part8, Sec 2.5.15	
	Packet with an unexpected ackID value Reception of packet (with an unexpected ackID value) causes the receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	ackID values on each link are assigned sequentially for each subsequent transmitted packet, an error in the ackID field is easily detected.	Part6, Sec 2.4	
	• Received READ_HOME request packet with incorrect ackid		
	• Received READ_HOME response packet with incorrect ackid		
1f	A packet carrying a response shall have a priority at least one priority level higher than the priority of the associated request.	Part6, Sec 6.11	
	• Received READ_HOME request with priority value 3.		

Item No	Compliance Item	Spec Reference	Testcase Name
1g	Bit[4] (Illegal Transaction code - Received a supported request/response packet with undefined field values) Logical/Transport Layer Error Detect CSR.	Part8, Sec 2.5.3	
	• Received READ_HOME response with reserved bit set in transaction field.		
	• Received READ_HOME response with DATA_ONLY encoding field set in status field		
	• Received READ_HOME response with NOT_OWNER encoding field set in status field		
	• Received READ_HOME response with RETRY encoding field set in status field		
	• Received READ_HOME response with INTERVENTION encoding field set in status field		
	• Received READ_HOME response with DONE_INTERVENTION encoding field set in status field		
	• Received READ_HOME response with data payload, but transaction filed encoding value as Response transaction with no data payload.		
1h	Bit[2] (GSM error response) in Logical/Transport Layer Error Detect CSR	Part8, Sec 2.5.3	
	• Received READ_HOME response with ERROR encoding field set in status field		
1i	A RESPONSE packet with an “ERROR” status never has a data payload	Part5, Sec 4.3.3	
	• Received READ_HOME response with data payload for ERROR status		
1j	TYPE 13 Response Issued by a processing element when it completes a request by a remote element	Part5, Sec 4.3	
	• Received READ_HOME response without READ_HOME request.		
1k	Bit[7] (Packet Response timeout) bit Logical/Transport Layer Error Detect CSR. A required response has not been received within the specified time out interval	Part8, Sec 2.5.3	
	• Allow response time out for READ_HOME request		
	• Received READ_HOME response without crf bit set for READ_HOME request with crf bit set		
2	**READ_OWNER**		srio_ll_gsm_rd_owner_err_test.sv
2a	Type 1 request packets never include data	Part5, Sec 4.2.5	
	• Received READ_OWNER request with data payload		
2b	Bit[14] (Received packet exceeds 276 Bytes) in Port n Error Detect CSR. Received packet that exceeds the maximum allowed size.	Part8, Sec 2.5.15	

Item No	Compliance Item	Spec Reference	Testcase Name
	Packet that exceeds the maximum packet size. Reception of packet (that exceeds the maximum packet size) causes the receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery	Part6, Sec 6.13.2.4 s	
	• Received READ_OWNER data payload greater than 256 bytes		
2c	Bit[9] (Unsupported Transaction) Logical/Transport Layer Error Detect CSR- A transaction is received that is not supported in the Destination Operations	Part8, Sec 2.5.3	
	• Change the Destination Operation CAR bit[1] to zero. READ_OWNER request received.	Part5, Sec 5.4.2	
2d	Cyclic Redundancy Code used to detect transmission errors in the packet	Part6, Sec 2.2	
	Packet with an incorrect CRC value The receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	Bit[13] (Received packet with bad CRC) in Port n Error Detect CSR	Part8, Sec 2.5.15	
	• Received READ_OWNER request packet with incorrect CRC.		
	• Received READ_OWNER response packet with incorrect CRC		
2e	Bit[12] (Received packet with unexpected ackID) Port n Error Detect CSR	Part8, Sec 2.5.15	
	Packet with an unexpected ackID value Reception of packet (with an unexpected ackID value) causes the receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	ackID values on each link are assigned sequentially for each subsequent transmitted packet, an error in the ackID field is easily detected.	Part6, Sec 2.4	
	• Received READ_OWNER request packet with incorrect ackid		
	• Received READ_OWNER response packet with incorrect ackid		
2f	A packet carrying a response shall have a priority at least one priority level higher than the priority of the associated request	Part6, Sec 6.11	
	Received READ_OWNER request with priority value 3.		
2g	Bit[4] (Illegal Transaction code - Received a supported request/response packet with undefined field values) Logical/Transport Layer Error Detect CSR.	Part8, Sec 2.5.3	
	Received READ_HOME response with reserved bit set in transaction field		

Item No	Compliance Item	Spec Reference	Testcase Name
	• Received READ_OWNER response with DATA_ONLY encoding field set in status field		
	• Received READ_OWNER response with NOT_OWNER encoding field set in status field		
	• Received READ_OWNER response with RETRY encoding field set in status field		
	• Received READ_OWNR response with INTERVENTION encoding field set in status field		
	• Received READ_OWNER response with DONE_INTERVENTION encoding field set in status field		
	• Received READ_OWNER response with data payload, but transaction filed encoding value as Response transaction with no data payload		
2h	Bit[2] (GSM error response) in Logical/Transport Layer Error Detect CSR	Part8, Sec 2.5.3	
	• Received READ_OWNER response with ERROR encoding field set in status field		
2i	A RESPONSE packet with an “ERROR” status never has a data payload	Part5, Sec 4.3.3	
	• Received READ_OWNER response with data payload for ERROR status		
2j	TYPE 13 Response Issued by a processing element when it completes a request by a remote element	Part5, Sec 4.3	
	Received READ_OWNER response without READ_OWNER request		
2k	Bit[7] (Packet Response timeout) bit Logical/Transport Layer Error Detect CSR. A required response has not been received within the specified time out interval	Part8, Sec 2.5.3	
	Allow response time out for READ_OWNER request		
2l	Received READ_OWNER response without crf bit set for READ_OWNER request with crf bit set		
3	IREAD_HOME		srio_ll_gsm_iread_home_err_test.sv
3a	Type 1 request packets never include data	Part5, Sec 4.2.5	
	• Received IREAD_HOME request packet with data payload		
3b	Bit[14] (Received packet exceeds 276 Bytes) in Port n Error Detect CSR. Received packet that exceeds the maximum allowed size.	Part8, Sec 2.5.15	
	Packet that exceeds the maximum packet size. Reception of packet (that exceeds the maximum packet size) causes the receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	• Received IREAD_HOME data payload greater than 256 bytes		
3c	Bit[9] (Unsupported Transaction) Logical/Transport Layer Error Detect CSR- A transaction is received that is not supported in the Destination Operations	Part8, Sec 2.5.3	

Item No	Compliance Item	Spec Reference	Testcase Name
	• Change the Destination Operation CAR bit[1] to zero. IREAD_HOME request received.		
3d	Cyclic Redundancy Code used to detect transmission errors in the packet	Part6, Sec 2.2	
	Packet with an incorrect CRC value The receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	Bit[13] (Received packet with bad CRC) in Port n Error Detect CSR	Part8, Sec 2.5.15	
	• Received IREAD_HOME request packet with incorrect CRC.		
3e	Bit[12] (Received packet with unexpected ackID) Port n Error Detect CSR	Part8, Sec 2.5.15	
	Packet with an unexpected ackID value Reception of packet (with an unexpected ackID value) causes the receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery	Part6, Sec 6.13.2.4	
	ackID values on each link are assigned sequentially for each subsequent transmitted packet, an error in the ackID field is easily detected	Part6, Sec 2.4	
	• Received IREAD_HOME request packet with incorrect ack id value		
4	READ_TO_OWN_HOME		srio_ll_gsm_rd_to_own_home_err_test.sv
4a	Type 2 request packets never include data.	Part5, Sec 4.2.6	
	• Received read_to_own_home request packet with data payload		
4b	Bit[9] (Unsupported Transaction) Logical/Transport Layer Error Detect CSR-A transaction is received that is not supported in the Destination Operations	Part8, Sec 2.5.3	
	• Change the Destination Operation CAR bit[2] to zero. READ_TO_OWN_HOME request received.		
5	READ_TO_OWN_OWNER		srio_ll_gsm_rd_to_own_owner_err_test.sv
5a	Type 1 request packets never include data.	Part5, Sec 4.2.5	
	• Received read_to_own_OWNER request packet with data payload		
5b	Bit[9] (Unsupported Transaction) Logical/Transport Layer Error Detect CSR-A transaction is received that is not supported in the Destination Operations	Part8, Sec 2.5.3	
	Change the Destination Operation CAR bit[2] to zero. READ_TO_OWN_OWNER request received.		
6	DKILL_HOME		srio_ll_gsm_dkill_home_err_test.sv
6a	Type 2 request packets never include data	Part5, Sec 4.2.6	
	• Received DKILL_HOME request packet with data payload		

Item No	Compliance Item	Spec Reference	Testcase Name
6b	Bit[9] (Unsupported Transaction) Logical/Transport Layer Error Detect CSR-A transaction is received that is not supported in the Destination Operations	Part8, Sec 2.5.3	
	• Change the Destination Operation CAR bit[3] to zero. DKILL_HOME request received.		
7	DKILL_SHARER		srio_ll_gsm_dkill_sharer_err_test.sv
7a	Type 2 request packets never include data	Part5, Sec 4.2.6	
	• Received DKILL_SHARER request packet with data payload		
7b	Bit[9] (Unsupported Transaction) Logical/Transport Layer Error Detect CSR-A transaction is received that is not supported in the Destination Operations	Part8, Sec 2.5.3	
	• Change the Destination Operation CAR bit[3] to zero. DKILL_SHARER request received.		
8	CASTOUT	Part5, Sec 4.2.8	srio_ll_gsm_castout_err_test.sv
8a	Type 5 packets always contain data		
	• Received CASTOUT request packet with out data payload		
8b	Bit[9] (Unsupported Transaction) Logical/Transport Layer Error Detect CSR-A transaction is received that is not supported in the Destination Operations	Part8, Sec 2.5.3	
	• Change the Destination Operation CAR bit[4] to zero. CASTOUT request received.		
9	TLBIE	Part5, Sec 4.2.6	srio_ll_gsm_tlbie_err_test.sv
9a	Type 2 request packets never include data		
	• Received TLBIE request packet with data payload		
9b	Bit[9] (Unsupported Transaction) Logical/Transport Layer Error Detect CSR-A transaction is received that is not supported in the Destination Operations	Part8, Sec 2.5.3	
	• Change the Destination Operation CAR bit[8] to zero. TLBIE request received		
10	TLBSYNC	Part5, Sec 4.2.6	srio_ll_gsm_tlbsync_err_test.sv
10a	Type 2 request packets never include data		
	• Received TLBSYNC request packet with data payload		
10b	Bit[9] (Unsupported Transaction) Logical/Transport Layer Error Detect CSR-A transaction is received that is not supported in the Destination Operations	Part8, Sec 2.5.3	
	• Change the Destination Operation CAR bit[9] to zero. TLBSYNC request received		
11	**IKILL_HOME**	Part5, Sec 4.2.6	srio_ll_gsm_ikill_home_err_test.sv
11a	Type 2 request packets never include data		

Item No	Compliance Item	Spec Reference	Testcase Name
	• Received IKILL_HOME request packet with data payload		
11b	Bit[9] (Unsupported Transaction) Logical/Transport Layer Error Detect CSR-A transaction is received that is not supported in the Destination Operations	Part8, Sec 2.5.3	
	• Change the Destination Operation CAR bit[7] to zero. IKILL_HOME request received		
12	**IKILL_SHARER**	Part5, Sec 4.2.6	srio_ll_gsm_ikill_sharer_err_test.sv
12a	Type 2 request packets never include data		
	• Received IKILL_SHARER request packet with data payload		
12b	Bit[9] (Unsupported Transaction) Logical/Transport Layer Error Detect CSR-A transaction is received that is not supported in the Destination Operations	Part8, Sec 2.5.3	
	• Change the Destination Operation CAR bit[7] to zero. IKILL_SHARER request received		
13	**FLUSH**	Part5, Sec 4.2.6	srio_ll_gsm_flush_err_test.sv
13a	Type 2 request packets never include data		
	• Received FLUSH type 2 request packet with data payload		
13b	Type 5 packets always contain data	Part5, Sec 4.2.8	
	• Received FLUSH type 5 request packet without data payload		
13c	Bit[9] (Unsupported Transaction) Logical/Transport Layer Error Detect CSR-A transaction is received that is not supported in the Destination Operations	Part8, Sec 2.5.3	
	• Change the Destination Operation CAR bit[5] to zero. FLUSH request received		
14	**IO_READ_HOME**		srio_ll_gsm_io_rd_home_err_test.sv
14a	Type 2 request packets never include data	Part5, Sec 4.2.6	
	• Received IO_READ_HOME request packet with data payload		
14b	Bit[9] (Unsupported Transaction) Logical/Transport Layer Error Detect CSR-A transaction is received that is not supported in the Destination Operations	Part8, Sec 2.5.3	
	• Change the Destination Operation CAR bit[5] to zero. IO_READ_HOME request received		
15	**IO_READ_OWNER**	Part5, Sec 4.2.5	srio_ll_gsm_io_rd_owner_err_test.sv
15a	Type 1 request packets never include data.		
	Received IO_READ_OWNER request packet with data payload		

Item No	Compliance Item	Spec Reference	Testcase Name
15b	Bit[9] (Unsupported Transaction) Logical/Transport Layer Error Detect CSR-A transaction is received that is not supported in the Destination Operations	Part8, Sec 2.5.3	
	• Change the Destination Operation CAR bit[5] to zero. IO_READ_OWNER request received		

6.3 Random case compliance

Table 15: Random case compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	Randomly generate all GSM transaction		srio_ll_gsm_random_test.sv
2	Randomly generate normal transaction followed by error transaction		srio_ll_gsm_random_normal_err_test.sv

7 Part6 - LP-Serial Physical Layer Specification

7.1 Functional/performance Feature Compliance

Table 16: Functional/Performance Feature Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	Embeds the transmission clock with data using an 8B/10B or 64B/67B encoding scheme.	Part6, Sec 1.1	
2	Supports links with from one lane, up to sixteen ganged lanes where each lane is a pair of unidirectional serial paths with one path in each direction		
3	Employs retry and error recovery protocols for link level reliability.		srio_pl_retry_stopped_state_test.sv srio_pl_error_stopped_state_test.sv
4	Supports division of the physical layer bandwidth into up to 9 virtual channels with independent flow control.		srio_pl_multiple_vc_support_test.sv
5	Supports Time Synchronization across RapidIO links with several different levels of accuracy.		srio_pl_time_synchronization_baud_rate1_test.sv
6	Supports transmission rates of 1.25, 2.5, 3.125, 5, 6.25 and 10.3125 Gbaud (data rates of 1, 2, 2.5, 4, 5 and 9.85 Gbps) per lane.		

7.2 Packet Field Compliance

Table 17: Packet Field Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	ackID Acknowledge ID is the packet identifier for link-level packet acknowledgment	Part6, Sec 2.2	
	The control symbol 24 is being used, the ackID value shall be 5 bits long and shall be left justified in the ackID field (ackID[0-4]) with the right-most bit of the field (ackID[5]) set to 0b0.		
	The control symbol 48 is being used, the ackID value shall be 6 bits long which fills the ackID field.		
	The control symbol 64 is being used, the complete ackID value shall be 12 bits long with the least significant 6 bits carried in the ackID field of the packet, and the most significant 6 bits carried in the start-of-packet control symbol.		
	The value of ackID assigned to the first packet transmitted after a reset shall be 0.	Part6, Sec 6.6.2	
	The values of ackID assigned to subsequent packets shall be in increasing numerical order, wrapping back to 0 on overflow		
2	VC	Part6, Sec 2.2	
	The VC bit specifies the usage of the PRIO and CRF fields.		
	When VC = 0, the PRIO and CRF fields contain the priority bits for a virtual channel 0 packet.		
	When VC = 1 the PRIO and CRF fields contain the Virtual Channel ID for a VC 1-8 packet.		
3	prio Depending on the value of the VC field, PRIO specifies packet priority or contains the most significant bits of the Virtual Channel ID (VCID)	Part6, Sec 2.2	
4	CRF Depending on the value of the VC field, CRF differentiates between virtual channel 0 flows of equal priority or contains the least significant bit of the Virtual Channel ID	Part6, Sec 2.2	

7.3 Control Symbols compliance

Table 18: Control symbols compliance

1	Control symbols are used for link maintenance, packet delimiting, packet acknowledgment, error reporting, and error recovery	Part6, Sec 3.1	

2	**short control symbol (**24 bits long)	Part6, Sec 3.1	
2a	It was designed for links operating at Baud Rate Class 1 and receivers that do not employ decision feedback equalization (DFE)		
2b	LP-Serial control symbols carry at least two independent functions.		
2c	Stype 0 Control symbol	Part6, Sec 3.4	
	0b000 packet-accepted		
	0b001 packet-retry		
	0b010 packet-not-accepted		
	0b011 timestamp		
	0b100 status		
	0b101 VC_status		
	0b110 link-response		
	0b111 implementation-defined		
2d	Stype1 Control Symbols	Part6, Sec 3.5	
	0b000 Start-of-packet		
	0b001 Stomp		
	0b010 End-of-packet		
	0b011 Restart-from-retry		
	0b100 Link-request		
	0b101 Timing		
	0b110 Reserved		
	0b111 NOP		
3	**long control symbol (**48 bits long)	Part6, Sec3.1	
3a	It was designed for links operating at Baud Rate Class 2 and receivers employing DFE		
3b	LP-Serial control symbols carry at least two independent functions.		
3c	Stype 0 Control symbol	Part6, Sec 3.4	
	0b000 packet-accepted		
	0b001 packet-retry		
	0b010 packet-not-accepted		
	0b011 timestamp		
	0b100 status		
	0b101 VC_status		
	0b110 link-response		
	0b111 implementation-defined		

3d	Stype1 Control Symbols	Part6, Sec 3.5	
	0b000 Start-of-packet		
	0b001 Stomp		
	0b010 End-of-packet		
	0b011 Restart-from-retry		
	0b100 Link-request		
	0b101 Timing		
	0b110 Reserved		
	0b111 NOP		
4	**Control Symbol 64**	Part6, Sec 3.1	
4a	It was designed for links operating at Baud Rate Class 3		
4b	LP-Serial control symbols carry at least two independent functions.		
4c	The cmd field is not used for Control Symbol 64;	Part6, Sec 3.3	
4d	Stype 0 Control symbol	Part6, Sec 3.4	
	0b0000 packet-accepted		
	0b0001 packet-retry		
	0b0010 packet-not-accepted		
	0b0011 timestamp		
	0b0100 status		
	0b0101 VC_status		
	0b0110 link-response		
	0b0111 implementation-defined		
	0b1000 reserved		
	0b1001 reserved		
	0b1010 reserved		
	0b1011 loop-response		
	0b1100 reserved		
	0b1101 VoQ-backpressure		
	0b1110 reserved		
	0b1111 reserved		

4e	Stype1 Control symbol	Part6, Sec 3.5	
	0x00–0x07 Reserved		
	0x08 Stomp		
	0x09–0x0F Reserved		
	0x10 End-of-packet-unpadded		
	0x11 End-of-packet-padded		
	0x12–0x17 Reserved		
	0x18 Restart-from-retry		
	0x19–0x21 Reserved		
	0x22 Link-request/Reset-port		
	0x23 Link-request/Reset-device		
	0x24 Link-request/Port-status		
	0x25–0x27 Reserved		
	0x28 Multicast-event		
	0x29–0x2A Reserved		
	0x2B Loop-timing-request		
	0x2C–0x37 Reserved		
	0x38 NOP (Ignore) **		
	0x39–0x7F Reserved		
	0b10, ackID[0:5] Start-of-packet-unpadded		
	0b11, ackID[0:5] Start-of-packet-padded		
5	**Control Symbol Selection**	Part6, Sec 6.5.1	
	For a LP-Serial link running at Baud Rate Class 1 the control symbol type used on the link is determined by the idle sequence being used on the link.		
	If the link is operating with idle sequence 1 (IDLE1), the Control Symbol 24 shall be used.		
	If the link is operating with idle sequence 2 (IDLE2), the Control Symbol 48 shall be used.		
	A LP-Serial link running at Baud Rate Class 2 shall always use Control Symbol 48 and idle sequence 2.		
	A LP-Serial link running at Baud Rate Class 3 shall always use Control Symbol 64 and idle sequence 3.		

6	**Control Symbol Delimiting**	Part6, Sec 6.5.2	
	LP-Serial control symbols (Control Symbol 24 and Control Symbol 48) on 8B/10B encoded links are delimited for transmission by 8B/10B special characters		
	For 64B/67B encoded link no such delimiting exists for Control Symbol 64.		
	Control Symbol 24 are delimited by a single 8B/10B special character that marks the beginning of the control symbol and immediately precedes the first character of the control symbol. Since control symbol length is constant and known, an end delimiters is neither needed nor provided.		
	Control Symbol 48 are delimited by two 8B/10B special characters. The first special character marks the beginning of the control symbol (the start delimiter) and immediately precedes the first character of the control symbol. The second special character marks the end of the control symbol (the end delimiter) and immediately follows the last character of the control symbol. The end delimiter special character replicates the value of the start delimiter special character.		
	If the control symbol contains a packet delimiter, the special character PD (K28.3) (‘h7c) is used.		
	If the control symbol does not contain a packet delimiter, the special character SC (K28.0) (‘h1c) is used.		
7	**Link Initialization**	Part6, Sec 6.5.3.1	
	The link is initialized after the port has successfully completed the following link initialization process and entered the link_initialized state (link_initialized variable asserted).		
	When a port is in the port_initialized state, but not in the link_initialized state, the port shall transmit only idle sequences, status, VC-status, link-request and link-response control symbols and, if IDLE2 is the idle sequence in use on the link, SYNC sequences.		
	The initialized port shall transmit idle sequence and at least one status control symbol per 1024 code-groups or codewords transmitted per lane until the port has received an error free status control symbol from the connected port.		
	The transmission of status control symbols indicates to the connected port that the port has completed initialization. The transmission of an idle sequence is required for the connected port to complete initialization.		
	After the initialized port has received an error free status control symbol from the connected port, the port shall transmit idle sequence and at least 15 additional status control symbols.		
	After the initialized port has received an error free status control symbol, the port shall wait until it has received a total of seven error free status control symbols with no intervening errors. This requirement provides a degree of link verification before packets and other control symbols are transmitted		
	If any VC other than VC0 is implemented and enabled, the port shall transmit a single VC_Status control symbol for each such VC. This initializes the flow control status for each implemented and enabled VC other than VC0.		

7a	**Status Control Symbol**	Part6, Sec 3.4.5	
	The status control symbol indicates receive status information about the port sending the control symbol		
	The control symbol contains the ackID_status and the buf_status fields.		
	The ackID_status field allows the receiving port to determine if it and the sending port are in sync with respect to the next ackID value the sending port expects to receive.		
	The buf_status field indicates to the receiving port the number of maximum length packet buffers the sending port has available for reception on VC0		
	Status is the default stype0 encoding and is used when the control symbol does not convey another stype0 function		
	Port n Initialization Status bit[16:19] Received status control symbol Count the number of consecutive error-free status control symbols received. This counter shall not increment once the link-initialized state has been achieved.	Part6, Sec 7.6.8	
	Port n Initialization Status bit[21:27] Transmitted status control symbol Count the number of status control symbol transmitted. This counter shall continue to increment until the link initialized state has been achieved. If the port can determine the status of the link partner through the contents of the IDLE2 or IDLE3 sequence, this counter shall also continue to increment until the link partner indicates that it has achieved the “port initialized” state.		
	If status control symbol is transmitted it shall be transmitted first, before any of the VC status control symbol	Part6, Sec 6.13.2.6	
7b	**VC-Status Control Symbol**		
	The VC-status control symbol indicates to the receiving port the available buffer space that the sending port has available for packet reception on the virtual channel (VC) specified in the control symbol.	Part6, Sec 3.4.6	
	The VC-status control symbol is used only for virtual channels 1 through 8 (VC1 through VC8) and may be transmitted only when the specified VC is implemented and enabled.		
	The VC-status control symbol may be transmitted at any time and should be transmitted whenever the number of maximum length packet buffers available for reception on a VC has changed and has not been otherwise communicated to the connected port.		
	Any VC-status control symbols that are transmitted shall be transmitted after the status control symbol and in order of increasing VCID	Part6, Sec 6.13.2.6	
	VC-status control symbol for a VC operating in RT mode shall indicate the number of receive buffers available for that VC inclusive of the buffer consumption of all packets received and accepted by the port for that VC before the event that caused the port to enter the Input Error-stopped state.		
	The VC-status control symbol for a VC operating in CT mode shall indicate the number of receive buffers available for that VC inclusive of the buffer consumption of all packets received and accepted by the port for that VC before the link-request/port-status control symbol was received.		

	The VC-status control symbols shall be transmitted before any packet acknowledgment control symbols are transmitted for packets received after the link-request/port-status control symbol was received	Part6, Sec 6.13.2.6	
7c	Once a port is in the link_initialized state, loss of port initialization (port_initialized variable deasserted) shall cause the port to exit the link_initialized state (link_initialized variable deasserted)		srio_pl_link_init_to_uninitialized_test.sv
	The link is then uninitialized from the point of view of that port. Once the port has exited the link_initialized state, the port shall not resume the normal transmission of packets and control symbols until the port has re-entered both the port_initialized and link_initialized states		
	A port that is not in the port_initialized state shall ignore and discard any packet or control symbol that it receives from the connected port.		
	A port that is in the port_initialized state but not in the link_initialized state shall ignore and discard any packet or any control symbol, other than status, VC-status, link-request or link-response control symbols, that it receives from the connected port.		
	A LP-Serial port shall not enter the Input error-stopped state or the Output error-stopped state unless the port is in the link_initialized state		
	The loss of link initialization (the state machine link_initialized variable is deasserted) shall not cause a port already in the Input error-stopped state or the Output error-stopped state to exit either of those states.		
8	**Buffer Status Maintenance**	Part6, Sec 6.5.3.2	
	When a LP-Serial port is in the normal operational state, it shall transmit a control symbol containing the buf_status field for VC0 at least once every 1024 code-groups transmitted per lane		
8a	When a LP-Serial port is in the normal operational state and any VC other than VC0 is active (VCs 1-8), the port shall transmit a control symbol containing the buf_status field for each active VC at least once every VC refresh period		srio_pl_multiple_vc_support_test.sv
	To comply with this requirement, the port shall transmit a VC_status control symbol for each active VC, other than VC0, if no other control symbol containing the buf_status field for that VC is available for transmission during the VC refresh interval.		
	The VC refresh period can be configured through the VC Refresh Interval register		
	Port *n* VC Control and Status Registers Bit[0:7] -VC Refresh Interval The number of 1024 code group intervals over which the VC status must be refreshed. Refresh Interval:? 0x0 - 1K code groups, 0xF - 16K code groups, 0xFF - 256K code groups Implementers are required to support a maximum VC refreshing period of at least 1024 x 16 = 16K code groups in size. The maximum possible VC refreshing period that can be supported is 1024 x 256 = 256K code groups. Writing to this field with a value greater than the maximum supported value by the port will set the field to the maximum value supported by the port	Part6, Sec 7.8.2.2	

9	**Embedded Control Symbols**	Part6, Sec 6.5.3.3	srio_pl_embedded_control_symbol_test.sv
	Any control symbol that does not contain a packet delimiter may be embedded in a packet.		
	An embedded control symbol may contain any defined encoding of stype0 and a stype1 encoding of “Timing” or “NOP”.		
	Control symbols with stype1 encodings of start-of-packet, end-of-packet, stomp, restart-from-retry, or link-request cannot be embedded as they would terminate the packet		
	When a Control Symbol 24 or Control Symbol 48 is embedded in a packet, the delimited control symbol shall begin on a 4-character boundary of the packet		
	When a Control Symbol 64 is embedded in a packet, the control symbol shall begin on a 8-byte boundary of the packet		
	Embedding packet acknowledgment control symbols reduces the delay in freeing packet buffers in the transmitting port which can increase packet throughput and reduce packet propagation delay in some situations, which can be desirable.		
10	**Timing Control Symbols**	Part6, Sec 6.5.3.4	
	Timing control symbols can trigger activity on other links of a device.		
10a	**Multicast-Event Control Symbols**	Part6, Sec 3.5.6.1	srio_pl_multicast_event_cs_test.sv
	The multicast-event control symbol allows the occurrence of a user-defined system event to be multicast throughout a system.		
	The multicast-event control symbol differs from other control symbols in that it carries information not related to the link carrying the control symbol		
	The Multicast-Event control symbol provides a mechanism through which end points are notified that some system defined event has occurred	Part6, Sec 6.5.3.4.1	
	embedding multicast-event control symbols allows their propagation delay and delay variation through switch processing elements to be minimized and is highly desirable for some multicast-event applications.	Part6, Sec 6.5.3.3	
	Port n Control CSRs Bit[12] - Multicast-event Participant Send incoming Multicast-event control symbols to this port (multiple port devices only)	Part6, Sec 7.6.11	
	Port *n* Timestamp Synchronization Command CSR Bit[29-31] - Command Contents of the “Cmd” field of a Timing control symbol to send to the link partner. Legal values are: 0b000 - Send Multicast Event Control Symbol	Part6, Sec 7.9.12	

7.4 Idle Sequence Compliance

Table 19: Idle Sequence Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	**IDLE1**	Part6, Sec 4.7	
	IDLE1 was designed for LP-Serial Baud Rate Class 1 links and transmitters and receivers that do not use adaptive equalization.		
	Idle Sequence 1 is a sequence of the special characters A, K and R.	Part6, Sec 4.7.2	
	Each instance of an IDLE1 sequence shall begin with the K special character.		
	The second, third and fourth characters of each IDLE1 sequence may be the R special character. This allows the first four characters of an IDLE1 sequence to be K,R,R,R, the “clock compensation sequence”.		
	The number of non-A special characters between A special characters within an IDLE1 sequence shall be no less than 16 and no more than 31.		
	The requirement on the number of characters between successive A special characters should be maintained between successive IDLE1 sequences to ensure that two successive A special characters are always separated by at least 16 non-A characters.		
	Each instance of IDLE1 shall be a new IDLE1 sequence that is unrelated to any previous IDLE1 sequence.		

Item No	Compliance Item	Spec Reference	Testcase Name
1a	**Code-Group Corruption Caused by Single Bit Errors**	Part6, Sec 4.5.8	
	Corruption /SC/ => /INVALID/ Detection on links using idle sequence 1 and Control Symbol 24 Detectable as an error when decoding the code-group. When this error occurs within a packet, it is indistinguishable from a /Dx.y/ => /INVALID/. When this error occurs outside of a packet, the type of error can be inferred from whether the /INVALID/ is followed by the three /Dx.y/ that comprise the control symbol data		
	Corruption /PD/ => /INVALID/ Detection on links using idle sequence 1 and Control Symbol 24 Detectable as an error when decoding the code-group. When this error occurs within a packet, it is indistinguishable from a /Dx.y/ => /INVALID/. When this error occurs outside of a packet, the type of error can be inferred from whether the /INVALID/ is followed by the three /Dx.y/ that comprise the control symbol data.		
	Corruption /A/, /K/ or /R/ => /Dx.y/ Detection on links using idle sequence 1 and Control Symbol 24 Detectable as an error as /Dx.y/ is illegal outside of a packet or control symbol and /A/, /K/ and /R/ are illegal within a packet or control symbol.		
	Corruption /A/, /K/ or /R/ => /INVALID/ Detection on links using idle sequence 1 and Control Symbol 24 Detectable as an error when decoding the code-group.		
	Corruption /Dx.y/ => /A/, /K/ or /R/ Detection on links using idle sequence 1 and Control Symbol 24 Detectable as an error as /A/, /K/ and /R/ are illegal within a packet or control symbol and /Dx.y/ is illegal outside of a packet or control symbol.		
	Corruption /Dx.y/ => /INVALID/ Detection on links using idle sequence 1 and Control Symbol 24 Detectable as an error when decoding the code-group.		
	Corruption /Dx.y/ => /Du.v/ Detection on links using idle sequence 1 and Control Symbol 24 Detectable as an error by the packet or control symbol CRC. The error will also result in a subsequent unerrored code-group being decoded as INVALID, but that resulting INVALID code-group may occur an arbitrary number of code-groups after the errored code-group.		

Item No	Compliance Item	Spec Reference	Testcase Name
2	**IDLE2**	Part6, Sec 4.7	
	IDLE2 was designed for LP-Serial Baud Rate Class 2 links and transmitters and receivers using adaptive equalization.		
	In addition to the minimum idle sequence functionality, IDLE2 provides link width, lane identification and lane polarity information, randomized data for equalizer training and a command and status channel for receiver control of the transmit equalizer.		
	IDLE 2 is a sequence of data characters and the special characters A, K, M and R.		
	The IDLE sequence 2 shall be comprised of a continuous sequence of idle frames and clock compensation sequences.		
	Each clock compensation sequence shall be followed by an idle frame		
	Each idle frame shall be followed by either a clock compensation sequence or another idle frame.		
	Each instance of IDLE2 shall be a new IDLE2 sequence that is unrelated to any previous IDLE2 sequence		
	Port n Error and Status CSRs Bit[0] - Idle Sequence 2 Support Bit[1] - Idle Sequence 2 Enable Bit[2-3] -Idle Sequence Indicates which IDLE sequence is active. 0b00 - idle sequence 1 is active. 0b01 - reserved. 0b10 - idle sequence 2 is active. 0b11 - idle sequence 3 is active	Part6, Sec 7.6.10	
2a	**IDLE2 termination**	Part6, Sec 4.7	
	Each M special character transmitted that is part of the idle frame random data field shall be followed by a minimum of four (4) random data field random data characters.		
	The sequence of four (4) M special characters at the beginning of a CS field marker shall not be truncated.		
	A port operating in Nx mode shall terminate an IDLE2 sequence at exactly the same character position in the sequence for each of the N lanes.		
2b	**IDLE Frame**	Part6, Sec 4.7.4.1	
	Each idle frame shall be composed of three parts, a random data field, a command and status (CS) field marker and an encoded CS field		
2c	**IDLE Sequence 2 Random Data Field**	Part6, Sec 4.7.4.1.1	
	The total length of the random data field shall be no less than 509 and no more than 515 characters		
	The random data field of an idle frame that immediately follows a clock compensation sequence shall begin with a M special character		

Item No	Compliance Item	Spec Reference	Testcase Name
	The lengths of the contiguous sequences should be uniformly distributed across the range of 16 to 31 characters.		
	Adjacent contiguous sequences shall be separated by a single A or M special character.		
	The length of the first contiguous sequence of pseudo-random characters in the random data field shall be no less than 16 and no more than 35 characters		
	The length of the last contiguous sequence of pseudo-random characters in the random data field shall be no less than 4 and no more than 35 characters.		
	When a port is operating in Nx mode, the location A or M special characters in a random data field shall be identical for all N lanes		
2d	**IDLE Sequence 2 CS Field Marker**	Part6, Sec 4.7.4.1.2	
	The CS field marker shall be the 8 character sequence M, M, M, M, D21.5, Dx.y, D21.5, Dx.y		
	The “M, M, M, M” sequence that begins the CS field marker is unique and is used to locate the start of the CS data field.		
	The sequence occurs only between the Idle Sequence 2 idle frame random data and CS fields.		
	It never occurs in control symbols or packet data and can not be created by an isolated burst error of 11 bits or less at the code-group level.		
	The character D21.5 provides lane polarity indication		
	The lane polarity determination for any of a port’s lanes shall not be changed when the state machine variable port_initialized is asserted		
2e	**IDLE2 Command and Status Field (CS field)**	Part6, Sec 4.7.4.1.3	srio_pl_aet_test.sv
	The CS field allows a port to provide certain status information about itself to the connected port and to control the transmit emphasis settings of the connected port if the connected port supports adaptive transmit emphasis.		
	The CS field shall have 32 information bits, cs_field[0-31], and 32 check bits, cs_field[32-63]. The check bits cs_field[32-63] shall be the bit wise complement of the information bits cs_field[0-31] respectively.		
2f	**IDLE2 CS Field Use**	Part6, Sec 4.7.4.1.4	
	A receiver may issue the following commands. Only one of these commands may be issued at a time. reset emphasis preset emphasis modify the emphasis provided by tap(-1), if tap(-1) is implemented modify the emphasis of tap(+1), if tap(+1) is implemented		
	Specific command bits may be changed only when the ACK and NACK bits are both de-asserted and the CMD bit is either de-asserted or transitioning from de-asserted to asserted		

Item No	Compliance Item	Spec Reference	Testcase Name
	Once the CMD bit is asserted, the connected port will either assert ACK after accepting and executing the command or assert NACK if the command cannot be executed.		
	The assertion of ACK or NACK shall occur no more than 250usec after the assertion of CMD		
	ACK and NACK shall never be asserted at the same time.		
	Once ACK or NACK is asserted in a CS field received by the port issuing the command, the CMD bit is de-asserted.		
	ACK or NACK, whichever is asserted, shall be de-asserted within 250usec of receipt of a CS field with the CMD bit deasserted.		
	If, for any reason, the connected port fails to assert ACK or NACK within 250usec of the assertion of CMD, CMD may be deasserted.		
	Once deasserted, CMD shall remain deasserted for at least 250usec before being reasserted.		
2g	**Idle Sequence Selection**	Part6, Sec 4.7.5	srio_pl_baud_rate1_idle2_test.sv srio_pl_baud_rate1_idle1_test.sv
	LP-Serial Baud Rate Class 2 links shall always use the IDLE2 sequence		
	LP-Serial Baud Rate Class 1 links shall support use of the IDLE1 sequence and may support use of the IDLE2 sequence.		
	If a LP-Serial port is operating at Baud Rate Class 1 and both ports on the link support both IDLE1 and IDLE2, the port shall determine which idle sequence to use on the link by using the following algorithm during the port initialization process.		
	If a LP-Serial port is operating at Baud Rate Class 1, supports the IDLE2 sequence and its configuration allows it to use the IDLE2 sequence, the port shall transmit the IDLE2 sequence when it enters the SEEK state of the port initialization process.		
	A LP-Serial port transmitting the IDLE2 sequence shall monitor the idle sequence it is receiving from the connected port		
	If the LP-Serial port that is transmitting the IDLE2 sequence receives IDLE2 from the connected port, IDLE2 shall be the idle sequence used on the link until the port reenters the SEEK state.		
	If the port receives IDLE1 from the connected port, the port shall switch to transmitting IDLE1 and IDLE1 shall be the idle sequence used on the link until the port reenters the SEEK state.		

7.5 Scrambling Compliance

Table 20: Scrambling Compliance

1	**Scrambling**	Part6, Sec 4.8	
	Scrambling of packet and control symbol data characters is used only on links operating with idle sequence 2 (IDLE2).		
	It is not used on links operating with idle sequence 1 (IDLE1)		
1a	**Scrambling Rules**	Part6, Sec 4.8.1	
	idle sequence 1 is selected (IDLE1) for use on the link, no characters shall be scrambled before transmission on the link.		
	If idle sequence 2 is selected (IDLE2) for use on the link, control symbol and packet data characters shall be scrambled by the transmitter before transmission on the link and descrambled in the receiver upon reception		
	Special characters, CS field marker data characters, and CS field data characters shall not be scrambled before transmission.		
	Scrambling and descrambling of control symbol and packet data characters can be disabled for test purposes by setting the Data scrambling disable bit in the Port *n* Control 2 CSR.		
1b	**Descrambler Synchronization**	Part6, Sec 4.8.2	
	Each lane descrambler shall synchronize itself to the scrambled data stream it is receiving by using the scrambling sequence extracted from the pseudo-random data characters received by the lane to re-initialize the state of the descrambler		
	After a lane descrambler has been re-initialized, the next two descrambler sync tests, shall be used to verify descrambler synchronization.		
	If the result of both lane descrambler sync tests is “pass”, the descrambler shall be determined to be “in sync”		
	Otherwise, the lane descrambler shall be determined to be “out of sync” and the resynchronization process shall be repeated.		
2	**Sync Sequence**	Part6, Sec 4.8.2	
	To ensure that a port that may have lost descrambler sync is able to recover descrambler sync before it is sent a link maintenance protocol link-request control symbol, a LP-Serial port that is operating with IDLE2 shall transmit a SYNC sequence before transmitting any link-request control symbol		
	The SYNC sequence shall be transmitted in parallel on each of the N active lanes of a link operating in Nx mode and shall immediately precede the link-request control symbol.		
	If the link is operating in 1x mode, the last character of the SYNC sequence is immediately followed by the first character of the link-request.		
	If the link is operating in Nx mode, the last column of the SYNC sequence is immediately followed by the column containing the first characters of the link-request.		
	The SYNC sequence will appear as four repetitions of ||M||D||D||D||D|| on a link operating in Nx mode.		

	The SYNC sequence shall be comprised of four contiguous repetitions of a five character sequence that begins with a M special character immediately followed by 4 pseudo-random data characters, i.e. the SYNC sequence is MDDDD MDDDD MDDDD MDDDD.		
3	**Descrambler Synchronization Verification**	Part6, Sec 4.8.3	srio_pl_descr_sync_break_test.sv
	A descrambler sync check trigger event is defined as the occurrence of one of the following character sequences in the received character stream of an active lane A single K, M or R special character that is not part of a contiguous sequence of K, M and/or R special characters A contiguous sequence of K and/or R special characters possibly followed by a M special character.		
	The descrambler sync check shall consist of inspecting the descrambled values of the four contiguous characters following the trigger sequence.		
	These four characters are designated the descrambler sync “check field”		
	The check field for the first type of trigger event shall be the four characters immediately following the K, M or R special character.		
	The check field for the second type of trigger event that does not end with a M special character shall be the four characters immediately following the contiguous sequence of K and/or R special characters.		
	The check field for the second type of trigger event that ends with a M special character shall be the four characters immediately following the M special character		
	If the descrambled value of each of the four characters in a check field is D0.0, the result of the descrambler sync test shall be “pass”. Otherwise, the result of the descrambler sync test shall be “fail” and the descrambler shall be determined to be “out of sync”.		
	A sync test can fail because of either a loss of descrambler sync or a data transmission error(s) in either the sync trigger sequence or the check field.		
	If a descrambler sync test fails, the port shall immediately enter the Input Error-stopped state if it is not already in that state and resynchronize the descramble		
	All control symbols and packet received while a lane descrambler is out of sync shall be ignored and discarded.		
	The cause field in the packet-not-accepted control symbol issued by the port on entering the Input Error-stopped state due to a sync check failure shall indicate “loss of descrambler sync”.		

7.6 Baud rate discovery Compliance

Table 21: Baud rate discovery compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	**Baud Rate Discovery** Baud rate discovery occurs during the SEEK state of the 1x_Initialization and 1x/Nx_Initialization state machines. The port shall look for an inbound signal on lane 0 or lane R of the link from a connected port. The method of detecting the presence of an inbound signal from a connected port is implementation specific and outside the scope of this specification.	Part6, Sec 4.12.3	
1a	If the baud rate of the inbound signal is the same as the baud rate at which the port is transmitting, the link shall operate at that per lane baud rate until the port reenters the SEEK state and the baud rate discovery process is complete.	Part6, Sec 4.12.3	srio_pl_tx_baud_rate_rx_baud_rate_same_test.sv
1b	If the baud rate on the inbound signal is less than the baud rate of the idle sequence being transmitted by the port, the port shall reduce the baud rate at which it is transmitting to the next lowest baud rate that it supports and that is enabled for use and again the baud rate discovery process continue.		srio_pl_tx_baud_rate_less_than_rx_baud_rate_test.sv
1c	If the baud rate on the inbound signal is greater than the baud rate of the idle sequence being transmitted by the port, the port shall continue transmitting at the current baud rate and again the baud rate discovery process continue.		srio_pl_tx_baud_rate_greater_than_rx_baud_rate_test.sv
1d	Port n Control 2 CSRs These registers are accessed when a local processor or an external device wishes to examine the port baud rate information.	Part6, Sec 7.6.9	

7.7 Initialization State machine Compliance

Table 22: Initialization State machine Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	**1x Mode Initialization State Machine**	Part6, Sec 4.12.4.5	
	The 1x_Initialization state machine specified in this section shall be used by ports that support only 1x mode (1x ports).		
	Silent-> Seek-> 1x_mode		
	Silent-> Seek-> 1x_mode -> Silent		srio_pl_1xmode_sl_sm_test.sv
	Silent-> Seek-> 1x_mode -> Seek		srio_pl_1xmode_seek_sm_test.sv
2	**1x/Nx Mode Initialization State Machine for N = 4, 8, 16**	Part6, Sec 4.12.4.6	
2a	Silent-> Seek-> Discovery->Nx_Mode		
2b	Silent-> Seek-> Discovery->Nx_Mode ->Silent-> Seek-> Discovery 1x_Mode_Lane0		srio_pl_nxmode_sl_seek_dis_1xmode_ln0_sm_test.sv
2c	Silent-> Seek-> Discovery->Nx_Mode ->Silent-> Seek->Discovery->1x_Mode_Lane2		srio_pl_nxmode_sl_seek_dis_1xmode_ln2_sm_test.sv
2d	Silent-> Seek-> Discovery->Nx_Mode ->Discovery->Nx_Mode		srio_pl_nxmode_dis_nxmode_sm_test.sv
2e	Silent-> Seek-> Discovery->Nx_Mode ->Discovery->1x_Mode_Lane2->1x_Recovery->1x_Mode_Lane0		srio_pl_nxmode_dis_1xmode_ln2_1xrec_1xmode_ln0_test.sv
2f	Silent-> Seek-> Discovery->Nx_Mode ->Discovery->1x_Mode_Lane0->1x_Recovery->1x_Mode_Lane0		srio_pl_nxmode_dis_1xmode_ln0_1xrec_1xmode_ln0_test.sv

Item No	Compliance Item	Spec Reference	Testcase Name
2g	Silent-> Seek-> Discovery->1x_Mode_Lane2->1x_Recovery->1x_Mode_Lane0		srio_pl_1xmode_ln2_1xrec_1xmode_ln0_test.sv
2h	Silent-> Seek-> Discovery->1x_Mode_Lane2->1x_Recovery->Silent ->Seek_> DISCOVERY-> Nx_MODE		srio_pl_1xmode_ln2_1xrec_sl_seek_dis_nxmode_test.sv
2i	Silent-> Seek-> Discovery->1x_Mode_Lane0->1x_Recovery->1x_Mode_Lane0		srio_pl_1xmode_ln0_1xrec_1xmode_ln0.sv
2j	Silent-> Seek-> Discovery->1x_Mode_Lane0->1x_Recovery->Silent-> Seek->DISCOVERY->Nx_MODE		srio_pl_1xmode_ln0_1xrec_sl_seek_dis_nxmode_test.sv
2k	Silent-> Seek-> Discovery->1x_Mode_Lane0->1x_Recovery->Silent-> Seek->DISCOVERY->2x_MODE		srio_pl_1xmode_ln0_1xrec_sl_seek_dis_2xmode_test.sv
2l	Silent-> Seek-> Discovery->1x_Mode_Lane2->Silent->Seek->Discovery->1x_Mode_Lane0		srio_pl_1xmode_ln2_sl_seek_dis_1xmode_ln0_test.sv
2m	Silent-> Seek-> Discovery->1x_Mode_Lane2->Silent->Seek->1x_Mode_Lane0		srio_pl_1xmode_ln2_sl_seek_1xmode_ln0_test.sv
2n	Silent-> Seek-> Discovery->1x_Mode_Lane0->Silent ->Seek->Discovery->1x_Mode_Lane0		srio_pl_1xmode_ln0_sl_seek_dis_1xmode_ln0_test.sv
2o	Silent-> Seek-> Discovery->1x_Mode_Lane0->Silent ->Seek->1x_Mode_Lane0		srio_pl_1xmode_ln0_sl_seek_1xmode_ln0_test.sv

Item No	Compliance Item	Spec Reference	Testcase Name
3	**Alternate 1x/4x_Initialization State Machine**	Part6, Sec 4.12.4.6	
3a	Silent->Seek->Discovery->4x_Mode->Discovery->1x_Mode_Lane2->Silent->Seek->Discovery after that cover any valid state transition		srio_pl_4xmode_dis_1xmode_ln2_sl_seek_dis_test.sv
3b	Silent->Seek->Discovery->4x_Mode->Discovery->1x_Mode_Lane0->Silent->Seek->Discovery after that cover any valid state transition.		srio_pl_4xmode_dis_1xmode_ln0_sl_seek_dis_test.sv
3c	Silent->Seek->Discovery->1x_Mode_Lane0->Silent->Seek->Discovery->1x_Mode_Lane0		srio_pl_1xmode_ln0_sl_seek_dis_1xmode_ln0_test.sv
3d	Silent->Seek->Discovery->1x_Mode_Lane0->Silent->Seek->Discovery->1x_Mode_Lane2		srio_pl_1xmode_ln0_sl_seek_dis_1xmode_ln2_test.sv
3e	Silent->Seek->Discovery->1x_Mode_Lane0->Silent->Seek->Discovery->4x_Mode		srio_pl_1xmode_ln0_sl_seek_dis_4xmode-test.sv
3f	Silent->Seek->Discovery->1x_Mode_Lane2->Silent->Seek->Discovery->1x_Mode_Lane0		srio_pl_1xmode_ln2_sl_seek_dis_1xmode_ln0_test.sv
3g	Silent->Seek->Discovery->1x_Mode_Lane2->Silent->Seek->Discovery->4x_Mode		srio_pl_1xmode_ln2_sl_seek_dis_4xmode_test.sv

Item No	Compliance Item	Spec Reference	Testcase Name
4	**1x/2x_Initialization State Machine**	Part6, Sec 4.12.4.7	
	Silent->Seek->Discovery->2x_Mode		
	Silent->Seek->Discovery->2x_Mode ->Discovery->1x_Mode_Lane0		srio_pl_2xmode_dis_1xmode_ln0_test.sv
	Silent->Seek->Discovery->2x_Mode ->Discovery->1x_Mode_Lane1		srio_pl_2xmode_dis_1xmode_ln1_test.sv
	Silent->Seek->Discovery->1x_Mode_Lane0->Silent->Seek->Discovery->2xMode		srio_pl_1xmode_ln0_sl_seek_dis_2xmode_test.sv
	Silent->Seek->Discovery->1x_Mode_Lane0->1x_Recovery->1x_Mode_Lane0		srio_pl_1xmode_ln0_1xrec_1xmode_ln0_test.sv
	Silent->Seek->Discovery->1x_Mode_Lane1->Silent>Seek->Discovery->2xMode		srio_pl_1xmode_ln1_sl_seek_dis_2xmode_test.sv
	Silent->Seek->Discovery->1x_Mode_Lane1->1x_Recovery->1x_Mode_Lane0		srio_pl_1xmode_ln1_1xrec_1xmode_ln0_test.sv
5	force_1x_mode Asserted when all Nx (multi-lane) modes are disabled. When asserted, forces the 1x/Nx Initialization state machine to use 1x mode.	Part6, Sec 4.12.4.1.3	
6	The input signal force_reinit allows the port to force link initialization at any time.	Part6, Sec 4.12.4.5	
7	Port n Initialization Status CSRs Bit[11-15] -Port Initialization State Machine State of the 1x/2x/4x/8x/16x port initialization state machine. This field shall be 0 if the port can only operate in 1x mode.	Part6, Sec 7.6.8	

7.8 Lane_Synchronization State Machine Compliance

Table 23: Lane_Synchronization State Machine Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	Icounter Counter used in the Lane_Synchronization state machine to count INVALID received code-groups. There is one Icounter for each lane in a Nx mode receiver.	Part6, Sec 4.12.4.2	srio_pl_lane_sync_sm_test.sv
2	Kcounter Counter used in the Lane_Synchronization state machine to count received code-groups that contain a comma pattern. There is one Kcounter for each lane in a Nx mode receiver.		
3	lane_sync Asserted by the Lane_Synchronization state machine when it determines that the lane it is monitoring is in bit synchronization and code-group boundary alignment. Otherwise de-asserted.		
4	signal_detect is may be continuously asserted or it may be used to require that some implementation defined additional condition be met before the Lane_Synchronization state machine is allowed to exit the NO_SYNC state.		
5	Vcounter Vcounter is used in the Lane_Synchronization state machine to count VALID received code-groups. There is one Vcounter for each lane in a Nx mode receiver.		
6	NO_SYNC -> NO_SYNC1-> NO_SYNC2->NO_SYNC1-> SYNC after that cover any valid state transition		
7	NO_SYNC -> NO_SYNC1->NO_SYNC2->NO_SYNC3->NO_SYNC2->NO_SYNC1->SYNC after that cover any valid state transition		
8	NO_SYNC -> NO_SYNC1->NO_SYNC2->NO_SYNC3->NO_SYNC2->NO_SYNC after that cover any valid state transition		
9	NO_SYNC->NO_SYNC1->NO_SYNC2->NO_SYNC after that cover any valid state transition		
10	SYNC-> SYNC1->SYNC2->SYNC1 after that cover any valid state transition		
11	SYNC-> SYNC1->SYNC2->SYNC3->SYNC2->SYNC1 after that cover any valid state transition		
12	SYNC-> SYNC1->SYNC2->SYNC3->SYNC2->SYNC3->SYNC4 after that cover any valid state transition		
13	SYNC-> SYNC1->SYNC2->SYNC3->SYNC2->SYNC3->SYNC4 after that cover any valid state transition		
14	SYNC-> SYNC1->SYNC2->SYNC1->NO_SYNC after that cover any valid state transition		

7.9 Lane Alignment State Machine Compliance

Table 24: Lane Alignment State Machine Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	Acounter A counter used in the Lane Alignment state machine to count received alignment columns (||A||s).	Part6, Sec 4.12.4.2	srio_pl_lane_alignment_sm_test.sv
2	Mcounter Mcounter is used in the Lane_Alignment state machine to count columns received that contain at least one /A/, but not all /A/s.		
3	N_lanes_aligned Asserted by the Lane_Alignment state machine when it determines that lanes 0 through N-1 are in sync and aligned.		
4	NOT_ALIGNED->NOT_ALIGNED_1->NOT_ALIGNED_2->NOT_ALIGNED after that cover any valid state transition		
5	NOT_ALIGNED->NOT_ALIGNED_1->NOT_ALIGNED_2->NOT_ALIGNED2->NOT_ALIGNED after that cover any valid state transition		
6	ALIGNED->ALIGNED_1->ALINGED_2->ALIGNED_1 after that cover any valid state transition		
7	ALIGNED->ALIGNED_1->ALINGED_2->ALIGNED_2->ALIGNED1-> after that cover any valid state transition		
8	ALIGNED->ALIGNED_1->ALINGED_2->ALIGNED_3->ALIGNED2->ALIGNED1 after that cover any valid state transition		
9	ALIGNED->ALIGNED_1->ALINGED_2->ALIGNED_3->ALIGNED2->ALIGNED1 after that cover any valid state transition		
10	Port n Initialization Status CSRs Bit[0-4] - Lane Alignment State of the lane alignment state machine. This field shall be 0 if the port can only operate in 1x mode	Part6, Sec 7.6.8	

7.10 Virtual Channel Compliance

Table 25: Virtual Channel Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	The protocol supports up to nine (9) virtual channels (VC0-VC8).	Part6, Sec 6.1	srio_pl_multiple_vc_support_test.sv

Item No	Compliance Item	Spec Reference	Testcase Name
	Virtual Channel 0 (VC0) is always active and always operates in reliable traffic mode.		
	When only VC0 is active, a link is said to be operating in single VC mode.		
	Virtual channels provides a mechanism that allows the bandwidth of a link to be allocated amongst various unrelated “streams” and types of traffic in a manner that ensures that each stream, or group of streams, receives a guaranteed minimum fraction of the link bandwidth.	Part6, Sec 6.4	
	Any of VC1 through VC8 that are implemented shall support operation in RT mode and may optionally support and be configured for operation in CT mode		
	CT VCs operate independent of each other.		
	RT VCs operate as a “RT Group”.		
	when the error recovery protocol is used to recover a damaged packet, the unacknowledged packets for all VCs in RT mode are retransmitted.		
	The number of VCs implemented is up to the implementer		
	The number of channels for VCs 1-8 may be 0, 1, 2, 4, or 8. The binary configuration allows traffic to be combined (by ignoring bits in the VC field) in a predictable manner		
2	**Virtual Channel Utilization**	Part6, Sec 6.4.3	
	Packets are transmitted from one or more virtual channels according to the weighted distribution of bandwidth for each channel		
	The weighting is such that under demand for full utilization of the link’s bandwidth, each active VC is guaranteed a certain portion of that bandwidth.		
	When the demand for bandwidth is less than the allocation for any VC, the extra bandwidth may be distributed among the other VCs giving them more than their allotment.		
	Processing elements shall not assume any packet ordering guarantees between VCs		
	Packets within a VC in VCs 1 - 8 are equally weighted and must be kept in order.		
	For VC0, flows identified as A - F (or higher) are synonymous with 0A - 0F, etc.	Part6, Sec 6.6.3	
	Flows for VCs 1-8 (A and higher) are identified as 1A, 2A,...8A.		
	All traffic in flows 1A-8A are transaction requests which do not require a response.		
	Transaction requests that require responses, and their corresponding responses, must use VC0 with the appropriate priority.		
	Devices supporting Virtual Channels contain an additional register block for configuring VC support for each port	Part6, Sec 7.1	

Item No	Compliance Item	Spec Reference	Testcase Name
	Virtual Channel Extended Features Block This Extended Features register block is assigned Extended Features block EF_ID=0x000A.	Part6, Sec 7.8	
3	**VC Register Block Header** The LP-Serial VC register block header register contains the EF_PTR to the next extended features block and the EF_ID that identifies this as the Virtual Channel Extended Features Block.	Part6, Sec 7.8.2.1	
4	**Port *n* VC Control and Status Registers** Bit[0-7] - VC Refresh Interval Bit[8-15] CT Mode Enables VCs to operate in CT mode beginning with VC8 Bit[16-23] VCs Support Bit[24-31] - VCs Enable	Part6, Sec 7.8.2.2	
5	**Port *n* VC0 BW Allocation Registers** This register is used to enable and configure VC0's participation in the band-width reservation scheduling. Bit[0] - VC0 Bandwidth Reservation Capable Bit[1] - VC0 BW Res Enable Bit[8-15] - Bandwidth Reservation Precision Bit[16-31] - Bandwidth Allocation	Part6, Sec 7.8.2.3	
6	**Port *n* VC*x* BW Allocation Registers** This register is used to enable and program VCs 1-8 participation in the bandwidth reservation scheduling. Bit[0-15] - Bandwidth Allocation Bit[16-31] - Bandwidth Allocation	Part6, Sec 7.8.2.4	

7.11 Time Synchronization Protocol Compliance

Table 26: Time Synchronization Protocol Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	Time synchronization is the method of synchronizing the “sense of time” between RapidIO processing elements	Part6, Sec 6.5.3.5	srio_pl_time_synchronization_baud_rate1_test.sv srio_pl_time_synchronization_baud_rate2_test.sv srio_pl_time_synchronization_baud_rate3_test.sv
	The “sense of time” is embodied in a Time Stamp Generator (TSG) for each node.		
	The Timestamp Generator LSW CSR register contains the least significant 32 bits of the TSG.		
	The Timestamp Generator MSW CSR register contains the most significant 32 bits of the TSG.		
	A timestamp is a 64-bit value, consisting of the MSW register in the most significant bits and the LSW register in the least significant bits.		
	Synchronization of TSGs with microseconds of accuracy is required. TSGs are synchronized between link partners using specific control symbols. TSGs advance at the same frequency, +/- 100 PPM.		
	Synchronization of TSGs within less than a microsecond is required. TSGs are synchronized between link partners using specific control symbols. The difference in frequency between link partners is calibrated and compensated for. Timestamp Generator master devices regularly update Timestamp Generator Slave devices.		
	Synchronization of TSGs within 100 nanoseconds or less is required. All TSGs use the same clock frequency to control the rate at which time advances. The delay between link partners is calibrated using control symbols to ensure maximum accuracy. TSGs are synchronized between link partners regularly, adjusting for the delay between link partners.		
	For devices that can be both a TSG Master and Slave, Slave control symbol support is required in TSG Slave mode, and Master control symbol support is required when in TSG Master mode, where the TSG mode is determined by the Port Operating Mode field of the Port n Timestamp Generator Synchronization CSR.		
2	**Setting and Reading a Timestamp Generator**	Part6, Sec 6.5.3.5.1	
	To set the TSG registers using Maintenance Writes, first write the TSG LSW register and then write the TSG MSW register.		
	To read the TSG registers using Maintenance Reads, first read the TSG MSW register, then the TSG LSW register, and then the TSG MSW register again. If the value of the TSG MSW register has not changed, then the timestamp value has been read successfully. If the value of the MSW register has changed, the TSG LSW register shall be read again to compose an accurate timestamp		
	The TSG of a link partner can also be set using Timestamp control symbols.		
	A sequence of timestamp control symbols shall be sent to set the link partner timestamp generator value,		

Item No	Compliance Item	Spec Reference	Testcase Name
	When either control symbols or maintenance packets are used to change a TSG to a value that is less than the current TSG value, the TSG value shall be held constant for the difference in time between the current TSG value and the time that was programmed.		
3	**Calibrating Transmission Delay**	Part6, Sec 6.5.3.5.2	
	Send Loop-Timing Request: The TSG Master sends a Loop-Timing Request control symbol to the TSG Slave by writing 0x00000003 to the Port n Timestamp Synchronization Command CSR. This latches the current TSG Master TSG value in the Port n Timestamp 0 MSW CSR and Port n Timestamp 0 LSW CSR		
	Process Loop-Response: The TSG Master receives a Loop-response for the Loop-Timing Request control symbol which contains the Delay amount in the TSG Slave. This also causes the current TSG Master TSG value to be latched in the Port n Timestamp 1 MSW CSR and Port n Timestamp 1 LSW CSR.		
	Compute Loop Delay: The TSG Master computes the loop timing delay as described at the end of this section		
	Set Timestamp Offset: The TSG Master programs its Port n Timestamp Offset register value to the Loop Delay computed in Compute Loop Delay step		
	Set Slave TSG: The TSG Master sets the TSG Slaves Timestamp Generator value by writing 0x00000010 to the Port n Timestamp Synchronization Command CSR.		
	A loop-response for a loop-timing request must be received within the link response timeout period.		
	If the loop-response is not received within the timeout period, then the loop-timing request shall be treated as completed.		
	A loop-timing request shall not be retransmitted in the event of a timeout.		
4	**Regular Timestamp Generator Re-synchronization**	Part6, Sec 6.5.3.5.3	
	Two methods of automatic re-synchronization are possible		
	If a device supports both a TSG Slave port that receives timestamp updates, and TSG Master ports that transmit timestamp updates, it is easiest to automatically update the TSG Master ports link partners whenever the TSG Slave port is updated.		
	If a device is the TSG Master for the entire system, the device can be configured to regularly update its link partner's sense of time		
	A TSG Master port can be configured to update its link partner whenever the TSG is updated by setting the "Auto-update Link Partner Timestamp Generators" field to 1 in the Port n Timestamp Synchronization CSR.		
	A TSG Master port can be configured to periodically update its link partner. The period is programmed by setting the Update Period field of the Port n Auto Update Counter CSR		

Item No	Compliance Item	Spec Reference	Testcase Name
	Periodic updates are enabled by setting the “Periodically Update Link Partner Timestamp Generators” bit to 1 in the Port n Timestamp Synchronization CSR.		
5	**Timestamp Control Symbol**	Part6, Sec 3.4.4	
	Timestamp control symbols are used to set the timestamp generator value of the link partner with a high degree of accuracy, and for links operating at Baud Rate Class 1 or Baud Rate Class 2 as a response to a loop-timing request (loop-response).		
	When links are operating at Baud Rate Class 1 or 2, a sequence of 8 timestamp control symbols is sent to set the link partner’s timestamp generator value		
	The Control Symbol 64 timestamp sequence consists of four control symbols		
	When links are operating at Baud Rate Class 1 or 2, a loop-response shall consist of a single Timestamp control symbol transmitted in response to a loop-timing request		
	For Control Symbol 24 and Control Symbol 48 formats, the Timestamp carries a single Delay value that represents the number of nanoseconds between the time the loop-timing request was received by the link partner, and the time the loop-response was generated.		
	When a loop-response is received while a loop-timing request is outstanding, the current value of the Timestamp Generator shall be captured in the Port n Timestamp 1 MSW CSR and Port n Timestamp 1 LSW CSR, and the delay value of the loop-response control symbol shall be captured in the Port n Timestamp Synchronization Status CSR		
	A processing element shall support receiving a loop-response when the Timestamp Master Supported bit of the Timestamp CAR is 1.		
	A processing element shall support transmitting a loop-response when the Timestamp Slave Supported bit of the Timestamp CAR is 1.		
6	**Loop Timing Control Symbol**	Part6, Sec 3.5.6.2	
	The loop-timing control symbol requests the receiver to send a loop-response control symbol in order to determine the transmission delay from the transmitting link partner to the receiving link partner.		
	A processing element shall support transmitting a loop-timing request when the Timestamp Master Supported bit of the Timestamp CAR		
	A processing element shall support receiving a loop-timing request when the Timestamp Slave Supported bit of the Timestamp CAR is 1.		
7	**Control Symbol 64 Loop-Response Control Symbol**	Part6, Sec 3.4.8	
	The loop-response control symbol is used by ports operating at Baud Rate Class 3 to respond to a Loop-timing Request control symbol.		
	The loop-response control symbol carries a single 12-bit value, Delay, which represents the number of nanoseconds between the time the loop-timing request was received by the link partner, and the time the loop-response was generated		

7.12 Packet Compliance

Table 27: Packet Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	**Packet Delimiting**	Part6, Sec 6.6.1	
	LP-Serial packets are delimited for transmission by control symbols.		
	Since packet length is variable, both start-of-packet and end-of-packet delimiters are required.		
	The start-of-packet delimiter immediately precedes the first character of the packet or an embedded delimited control symbol.		
	the control symbol marking the end of a packet (packet termination) immediately follows the last character of the packet or the end of an embedded delimited control symbol		
	The following control symbols are used to delimit packets Start-of-packet End-of-packet Stomp Restart-from-retry Any link-request		
2	**Packet Start** The beginning of a packet (packet start) shall be marked by a start-of-packet control symbol.	Part6, Sec 6.6.1.1	
	Start-of-Packet Control Symbol	Part6, Sec 3.5.1	
	The start-of-packet control symbol is used to delimit the beginning of a packet		
	The Control Symbol 64 start-of-packet control symbol has two variants – start-of-packet-unpadded and start-of-packet-padded – to indicate if a previous packet has padding appended to achieve a total length that is a multiple of 8 bytes.		
	The start-of-packet-unpadded control symbol shall be used for cases where the start-of-packet does not terminate a previous packet or where the start-of-packet terminates a packet that was not padded.		
	The start-of-packet-padded control symbol shall be used when the start-of-packet terminates a packet that was padded to multiple of 8 bytes.		
	The end of a packet is marked with a start-of-packet control symbol that also marks the beginning of a new packet.	Part6, Sec 6.6.1.2	
3	**Packet Termination** A packet shall be terminated in one of the following ways: The end of a packet is marked with an end-of-packet control symbol. The end of a packet is marked with a start-of-packet control symbol that also marks the beginning of a new packet The packet is canceled by a restart-from-retry or stomp control symbol The packet is canceled by any link-request control symbol	Part6, Sec 6.6.1.2	

Item No	Compliance Item	Spec Reference	Testcase Name
4	**End-of-Packet Control Symbol**	Part6, Sec 3.5.3	
	The end-of-packet control symbol is used to delimit the end of a packet.		
	The Control Symbol 64 end-of-packet control symbol has two variants – end-of-packet-unpadded and end-of-packet-padded – to indicate if a previous packet has padding appended to achieve a total length that is a multiple of 8 bytes.		
	The end-of-packet-unpadded control symbol shall be used when the end-of-packet terminates a packet that was not padded		
	The end-of-packet-padded control symbol shall be used when the end-of-packet terminates a packet that was padded to multiple of 8 bytes		
	The end of a packet is marked with an end-of-packet control symbol.	Part6, Sec 6.6.1.2	
4	Acknowledgment Identifier	Part6, Sec 6.6.2	
	Each packet requires an identifier to uniquely identify its acknowledgment control symbol.		
5	**Packet-Accepted Control Symbol**	Part6, Sec 3.4.1	
	The packet-accepted control symbol indicates that the port sending the control symbol has taken responsibility for sending the packet or packets to its final destination and that resources allocated to the packet or packets by the port receiving the control symbol can be released		
	This control symbol shall be generated only after the entire packet or packets has been received and found to be free of detectable errors		
	Devices that support Baud Rate Class 3 operation shall support configuration values whereby Packet Accepted controls symbols sent and/or received acknowledge multiple packets	Part6, Sec 6.6.2	
	Devices operating at Baud Rate Class 3 may transmit one Packet Accepted control symbol for multiple received packets		
	It shall be possible to configure devices operating at Baud Rate Class 3 to transmit a Packet Accepted control symbol for each received packet.		
	Devices operating at Baud Rate Class 1 and 2 may optionally support reception of Packet Accepted control symbols which acknowledge all outstanding packets up to and including the packet ackID.		
	Bit[0] - Multiple Acknowledges Supported. (Port n Latency Optimization CSR) Indicates whether the port supports reception of Packet Accepted, Packet Not Accepted, and Retry control symbols which acknowledge multiple out-standing ackIDs. 0b0 - A control symbol shall always acknowledge one ackID 0b1 - A control symbol shall acknowledge multiple outstanding ackIDs. This bit shall be read-only.	Part6, Sec 7.6.15	

Item No	Compliance Item	Spec Reference	Testcase Name
	The port optionally acknowledges multiple packets with a single packet-accepted control symbol		
	• Received one packet accepted control symbol for multiple request/ response packets		
	• Received one packet accepted control symbol for one request/response packets		
6	**Packet Priority and Transaction Request Flows**	Part6, Sec 6.6.3	
	Within VC0 each packet has a priority, and optionally a critical request flow, that is assigned by the end point processing element that is the source of (initiates) the packet.		
	The priority is carried in the prio field of the packet and has four possible values: 0, 1, 2, or 3.		
	Packet priority increases with the priority value with 0 being the lowest priority and 3 being the highest.		
	Packet priority is used in RapidIO for several purposes which include transaction ordering and deadlock prevention		
	The critical request flow is carried in the CRF bit. It allows a flow to be designated as a critical or preferred flow with respect to other flows of the same priority		
	When a transaction is encapsulated in a packet for transmission, the transaction request flow indicator (flowID) of the transaction is mapped into the prio field (and optionally the CRF bit) of the packet		

7.13 Link Maintenance Protocol Compliance

Table 28: Link Maintenance Protocol Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	The link maintenance protocol involves a request and response pair between ports connected by a LP-Serial link.	Part6, Sec 6.7	
1a	**Software triggered Link-Request Control Symbol**		srio_pl_software_triggered_link_req_cs_test.sv
	For software management, the request is generated through ports in the configuration space of the sending device		
	An external host write of a command to the link-request register with an I/O logical specification maintenance write transaction causes a link-request control symbol to be issued onto the output port of the device, but only one link-request can be outstanding on a link at a time.		
	The device that is linked to the sending device shall respond with an link-response control symbol if the link-request command required it to do so		
	The external host retrieves the link-response by polling the link-response register with I/O logical maintenance read transactions.		
1b	**Port *n* Link Maintenance Request CSRs**	Part6, Sec 7.6.5	
	The port link maintenance request registers are accessible both by a local processor and an external device.		
	A write to one of these registers generates a link-request control symbol on the corresponding RapidIO port interface.		
	Bit[29-31] - Command Command to be sent in the link-request control symbol. If read, this field returns the last written value.		
1c	**Port *n* Link Maintenance Response CSRs**	Part6, Sec 7.6.6	
	The port link maintenance response registers are accessible both by a local processor and an external device		
	A read to this register returns the status received in a link-response control symbol.		
	Bit[0] - response_valid		
	Bit[3-14] -port_status_cs64		
	Bit[15-26] - ackID_status		
	Bit[27-31] -port_status		
1d	**Hardware triggered Link-Request Control Symbol**	Part6, Sec 6.7	srio_pl_error_stopped_state_test.sv
	The automatic error recovery mechanism relies on the hardware generating packet-not-accepted and link-request/port-status control symbols under the transmission error conditions and using the corresponding link-response information to attempt recovery		

Item No	Compliance Item	Spec Reference	Testcase Name
1e	**Link-Request Control Symbol**	Part6, Sec 3.5.5	
	A link-request control symbol is used by a port to either issue a command to the connected port or request its input port status.		
	A link-request control symbol always cancels a packet whose transmission is in progress and can also be sent between packets		
1f	**Reset-Device Command**	Part6, Sec 3.5.5.1	srio_pl_link_req_rst_dev_cs_test.sv
	The reset-device command causes the receiving device to go through its reset or power-up sequence		
	All state machines and the configuration registers reset to the original power on states.		
	The reset-device command does not generate a link-response control symbol.		
	A port receiving a reset-device command in a link-request control symbol shall not perform the reset function unless it has received four reset-device commands in a row without any other intervening packets or control symbols, except status control symbols		
1g	**Port-status Command**	Part6, Sec 3.5.5.2	srio_pl_error_stopped_state_test.sv
	The port-status command requests the receiving port to return a link-response containing the ackID value it expects to next receive on its input port and the current input port operational status for informational purposes		
	The receiver then responds with a link-response control symbol.		
	The port-status command of the link-request/port-status control symbol is used by the hardware to recover from transmission errors.	Part6, Sec 6.7	
	If the input port had stopped due to a transmission error that generated a packet-not-accepted control symbol back to the sender, the link-request/port-status control symbol acts as a link-request/restart-from-error control symbol, and the receiver is re-enabled to receive new packets after generating the link-response control symbol		
	The link-request/port-status control symbol can also be used to restart the receiving device if it is waiting for a restart-from-retry control symbol after retrying a packet		
	port receiving a link-request/port-status control symbol returns a link-response control symbol containing two pieces of information port_status, ackID_status		

Item No	Compliance Item	Spec Reference	Testcase Name
1h	**Reset-port Command**	Part6, Sec 3.5.5.3	srio_pl_link_req_rst_port_cs_test.sv
	A reset-port command is intended to allow packet exchange to resume after an unrecoverable link error condition has been detected and system software has handled this condition.		
	The reset-port command shall not generate a link-response control symbol.		
	After a port transmits a reset-port request, if the port's initialization state machine transitions to the SILENT state within one link-response timeout period, the port shall behave as if it has received a reset-port request		
	A port receiving a reset-port command in a link-request control symbol shall not perform the reset-port function unless it has received four reset-port commands in a row without any other intervening packets or control symbols, except status control symbols		
1i	LP-Serial port that is operating with IDLE2 shall transmit a SYNC sequence before transmitting any link-request control symbol.	Part6, Sec 4.8.2	
	LP-Serial port that is operating with IDLE3 shall transmit a Seed ordered sequence before every transmitted link-request control symbol.	Part6, Sec 5.5.4.2	
	For reset-device or reset-port where four link-request control symbols are transmitted each of the four link-request control symbols shall be preceeded by a Seed ordered sequence.	Part6, Sec 5.5.4.2	
	The cancellation of a packet by a link-request control symbol is subject to the requirement that every link-request control symbol transmitted on a link operating with IDLE2 be immediately preceded by a SYNC sequence, or subject to the requirements of that every link-request control symbol transmitted on a link operating with IDLE3 be immediately preceded by a Seed ordered sequence	Part6, Sec 6.6.1.2	
	If a link-request control symbol terminates a packet on a link that is operating with IDLE2, the SYNC sequence is required to precede the link-request control symbol.		
	If a link-request control symbol terminates a packet on a link that is operating with IDLE3, the Seed ordered sequence is required to precede the link-request control symbol.		
1j	**Port Link Timeout Control CSR**	Part6, Sec 7.6.2	
	The port link timeout control register contains the timeout timer value for all ports on a device.		
	This timeout is for link events such as sending a packet to receiving the corresponding acknowledge and sending a link-request to receiving the corresponding link-response.		
	Bit[0–23] - timeout value		

Item No	Compliance Item	Spec Reference	Testcase Name
	Link-Response Control Symbol	Part6, Sec 3.4.7	srio_pl_error_stopped_state_test.sv
	The link-response control symbol is used by a port to respond to a link-request control symbol		
	The status reported in the port_status field shall be the status of the port at the time the associated port-status link-request control symbol was received.		
	For backwards compatibility, when operating with lane speeds of less than 3.5 Gbaud, the port_status field shall only use one of the following values: 0b00010, 0b00100, 0b00101 or 0b10000 even if other values are defined in the specification.		
	0b00010 - Error. The port has encountered an unrecoverable error and is unable to accept packets.		
	0b00100 - Retry-stopped The port has retried a packet and is waiting in the input retry-stopped state to be restarted.		
	0b00101 - Error-stopped The port has encountered a transmission error and is waiting in the input error-stopped state to be restarted.		
	0b10000 -OK The port is accepting packets.		
	Port-status Field Definitions for Baud Rate Class 3		

Item No	Compliance Item	Spec Reference	Testcase Name
	Bit[1:2] - input port status 0b00 - No input error condition exists 0b01 - Port n Error and Status CSR “Input retry-stopped” status bit is asserted 0b10 - Port n Error and Status CSR “Input Error-Stopped” status bit is asserted 0b11 - Implementation specific Input Port Fatal Error condition The values are encoded in increasing order of priority. 0b00 is the lowest priority. When multiple conditions exist simultaneously the highest priority condition shall be encoded.		
	Bit[3] - Input Port Enabled This bit shall be set if all of the following conditions are true, otherwise this bit shall be cleared: - The Port n Control CSR Input Port Enabled bit is set. - All implementation specific bits allow physical layer packet acceptance.		
	Bit[5:6] Output Port Status 0b00 - No output error condition exists 0b01 - Port n Error and Status CSR “Output retry-stopped” bit is asserted 0b10 - Port n Error and Status CSR “Output Error-Stopped” status bit is asserted 0b11 - Output Port Fatal Error condition. The values are encoded in increasing order of priority. 0b00 is the lowest priority. When multiple conditions exist simultaneously the highest priority condition shall be encoded.		
	Bit[7] Output Port Enabled This bit shall be set if both of the following conditions are true, otherwise this bit shall be cleared: - The Port n Control CSR Output Port Enabled bit is set. - All implementation specific bits allow physical layer packet acceptance.		
	Bit[8] - Port-Write Pending The port has encountered a condition which required it to initiate a Maintenance Port-write operation.		

7.14 Flow control Compliance

Table 29: Flow control Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	The purpose of link level flow control is to prevent the loss of packets due to a lack of buffer space in a link receiver.	Part6, Sec 6.9	

Item No	Compliance Item	Spec Reference	Testcase Name
2	**Flow Control Mode Negotiation**	Part6, Sec 6.9.3	
	If the port and its link partner both support transmitter-controlled flow control, then both ports shall use transmitter-controlled flow control.		
	Otherwise, both ports shall use receiver-controlled flow control.		
	If multiple VCs are used, then a port shall have either all channels in receiver based flow control or all channels in transmitter based flow control.		
3	**Transmitter-Controlled Flow Control**	Part6, Sec 6.9.2	srio_pl_transmitter_controlled_flow_ctrl_test.sv
	In transmitter-controlled flow control, the receiving port provides information to its link partner about the amount of buffer space it has available for packet reception.		
	A port signals its link partner that it is operating in transmitter-controlled flow control mode by setting the buf_status field to a value different from all 1's in every control symbol containing the field that the port transmits.		
	The value conveyed by the buf_status field is the number of maximum length packet buffers currently available for packet reception up to the limit that can be reported in the field.		
	If a port has more buffers available than the maximum value that can be reported in the buf_status field, the port sets the field to that maximum value.		
	A port may report a smaller number of buffers than it actually has available, but it shall not report a greater number		
	A port whose link partner is operating in transmitter-control flow control mode should never receive a packet-not-accepted (or packet-retry control symbol if operating in single VC mode) from its link partner unless the port has transmitted more packets than its link partner has receive buffers, has violated the rules that all input buffers may not be filled with low priority packets or there is some fault condition.		
4	**Speculative packet transmission**	Part6, Sec 6.9.2.3	srio_pl_speculative_pkt_trans.sv
	A port whose link partner is operating in transmitter-controlled flow control mode may send more packets on a given VC than the number of free buffers indicated by the link partner as being available for that VC		
	Packets transmitted in excess of the free_buffer_count are transmitted on a speculative basis and are subject to retry by the link partner.		
	The link partner accepts or rejects these packets on a packet by packet basis in exactly the same way it would if operating in receiver-controlled flow control mode.		
	speculative transmission that results in a significant number of retries and discarded packets can reduce the effective bandwidth of the link.		

Item No	Compliance Item	Spec Reference	Testcase Name
5	**Receiver -Controlled Flow Control** Receiver-controlled flow control is the simplest and basic method of flow control A port signals its link partner that it is operating in receiver-controlled flow control mode by setting the buf_status field to all 1's in every control symbol containing the field that the port transmits.	Part6, Sec 6.9.1	
6	**Reliable Traffic VC Receivers** If buffer space is not available, the port rejects the packet. If multiple VCs are active, and the VC is in reliable traffic mode, the rejected packet shall be acknowledged with the packet-not-accepted control symbol. The cause field of the control symbol should be set to "packet not accepted due to lack of resources"	Part6, Sec 6.9.1.1	
7	**Continuous Traffic VC Receivers** If buffer space is not available, and the VC is in CT mode, the packet is acknowledged as accepted, and the packet is discarded This preserves the order of the normal link response and does not impact performance.	Part6, Sec 6.9.1.2	
8	**Port *n* Error and Status CSRs** Bit[4] - Flow Control Mode Indicates which flow control mode is active (read only). 0b0 - receiver-controlled flow control is active. 0b1 - transmitter-controlled flow control is active.		

7.15 Cancelling Packet Compliance

Table 30: Cancelling Packet Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	When a port becomes aware of some condition that will require the packet it is currently transmitting to be retransmitted, the port may cancel the packet	Part6, Sec 6.10	
	A port may cancel a packet if the port detects a problem with the packet as it is being transmitted or if the port receives a packet-retry or packet-not-accepted control symbol for a packet that is still being transmitted or that was previously transmitted.		
	The sending device shall use the stomp control symbol, the restart-from-retry control symbol (in response to a packet-retry control symbol), or any link request control symbol to cancel a packet.		
	A port receiving a canceled packet shall drop the packet.		
	The cancellation of a packet shall not result in the generation or report of any errors.		
2	The stomp control symbol is used to cancel a partially transmitted packet	Part6, Sec 3.5.2	srio_pl_stomp_pkt_cancel_test.sv
	If the packet was canceled by other than a restart-from-retry or link-request/ port-status control symbol and the port is operating in single VC mode (only VC0 is active), the port shall immediately enter the Input Retry-Stopped state and follow the Input Retry-Stopped recovery process	Part6, Sec 6.10	
	If the packet was canceled by other than a restart-from-retry or link-request/ port-status control symbol and the port is operating in multiple VC mode (at least one of VC1-8 is active), the port shall immediately enter the Input Error-Stopped state and follow the Input Error-Stopped recovery process		
3	The restart-from-retry control symbol cancels a current packet	Part6, Sec 3.5.4	srio_pl_retry_stopped_state_test.sv
	If the packet is canceled by a restart-from-retry control symbol a protocol error has occurred and the port shall immediately enter the Input Error-stopped state and follows the Input Error-stopped recovery process	Part6, Sec 6.10	
4	A link-request control symbol always cancels a packet whose transmission is in progress and can also be sent between packets.	Part6, Sec 3.5.5	srio_pl_error_stopped_state_test.sv
	If the packet is canceled by a link-request/port-status control symbol, the port shall drop the packet without reporting a packet error.	Part6, Sec 6.10	

7.16 Retry protocol compliance

Table 31: Retry Protocol Compliance

1	**Packet-Retry Control Symbol**	Part6, Sec 3.4.2	srio_pl_retry_stopped_state_test.sv
	A packet-retry control symbol indicates that the port sending the control symbol was not able to accept the packet due to some temporary resource conflict such as insufficient buffering and the packet must be retransmitted		
	The packet-retry control symbol shall be used in single VC mode		
	Packet retry is replaced with error recovery when multiple VCs are active.		
2	**Restart-From-Retry Control Symbol**	Part6, Sec 3.5.4	
	This control symbol is used to mark the beginning of packet retransmission, so that the receiver knows when to start accepting packets after the receiver has requested a packet to be retried		
	The restart-from-retry control symbol cancels a current packet and may also be transmitted on an idle link.		
3	**Single VC Retry protocol**	Part6, Sec 6.9.1.3	
	When operating with a single VC (VC0), the receiver may use the retry protocol for handling receiver overruns		
	As part of the Input Retry-stopped recovery process, the port sends a packet-retry control symbol to its link partner indicating that the packet whose ackID is in the packet_ackID field of the control symbol and all packets subsequently transmitted by the port have been discarded by the link partner and must all be retransmitted.		
4	**Input Retry-Stopped Recovery Process**		
	The retry-stopped state indicates that the port has retried a packet and is waiting to be restarted. This state is cleared when a restart-from-retry (or a link-request/port-status) control symbol is received.	Part6, Sec 6.7	
	Discards the rejected or canceled packet without reporting a packet error and ignores all subsequently received packets while the port is in the Input Retry-stopped state	Part6, Sec 6.9.1.4	
	Causes the output side of the port to issue a packet-retry control symbol containing the ackID value of the retried packet in the packet_ackID field of the control symbol. (The packet-retry control symbol causes the output side of the link partner to enter the Output Retry-stopped state and send a restart-from-retry control symbol.)		
	When a restart-from-retry control symbol is received, exit the Input Retry-stopped state and resume packet reception.		
5	**Output Retry-Stopped Recovery Process**		

	Immediately stops transmitting new packets	Part6, Sec 6.9.1.5	
	Resets the link packet acknowledgment timers for all transmitted but unacknowledged packets.		
	Transmits a restart-from-retry control symbol.		
	Backs up to the first unaccepted packet (the retried packet) which is the packet whose ackID value is specified by the packet_ackID value contained in the packet-retry control symbol. (The packet_ackID value is also the value of ackID field the port retrying the packet expects in the first packet it receives after receiving the restart-from-retry control symbol.		
	Exits the Output Retry-stopped state and resumes transmission with either the retried packet or a higher priority packet which is assigned the ackID value contained in the packet_ackID field of the packet-retry control symbol.		
	If a port, operating in single VC mode, for whose link partner is operating in transmitter-control flow control mode, receives a packet-retry control symbol, the output side of the port immediately enters the Output Retry-stopped state and follows the Output Retry-stopped recovery process	Part6, Sec 6.9.2	
6	**Port *n* Error and Status CSRs** Bit[11] - Output Retry-encountered Bit[12] - Output Retried Bit[13] - Output Retry-stopped Bit[21] - Input Retry-stopped	Part6, Sec 7.6.10	
7	**Port *n* Latency Optimization CSRs** Bit[0] -Multiple Acknowledges Supported Indicates whether the port supports reception of Packet Accepted, Packet Not Accepted, and Retry control symbols which acknowledge multiple outstanding ackIDs. 0b0 - A control symbol shall always acknowledge one ackID 0b1 - A control symbol shall acknowledge multiple outstanding ackIDs. This bit shall be read-only.	Part6, Sec 7.6.15	
	When the ability to acknowledge multiple packets with a single control symbol is enabled, the packet-retry control symbol shall acknowledge all packets up to, but not including, the ackID in the retry control symbol.	Part6, Sec 6.9.1.3	
	Devices may optionally support configuration values whereby Retry control symbols sent and/or received acknowledge multiple packets		
	When transmitting control symbols, devices operating at Baud Rate Class 1 and 2 shall support a default configuration in which the Retry control symbol does not acknowledge packets		
	Devices operating at Baud Rate Class 1 and 2 may optionally support a configuration in which Retry control symbols acknowledge all packets up to, but not including, the ackID_status in the Retry control symbol.		
	Devices operating at Baud Rate Class 3 shall support a default configuration whereby a transmitted Retry control symbol acknowledges all packets up to, but not including, the ackID_status in the Retry control symbol.		
	Devices operating at Baud Rate Class 1 and 2 may optionally support reception of Retry control symbols that acknowledge all outstanding packets up to, but not including, the ackID_status.		

	Devices operating at Baud Rate Class 3 shall support reception of Retry control symbols that acknowledge all outstanding packets up to, but not including, the ackID_status		
	It shall be possible to configure devices operating at Baud Rate Class 3 to transmit a Retry control symbol only when all packets up to the Retried packet have been acknowledged.		

7.17 Error recovery Compliance

Table 32: Error recovery Compliance

1	**Input Error-Stopped State Recovery Process**	Part6, Sec 6.13.2.6	srio_pl_error_stopped_state_test.sv
	When the input side of a port detects a transmission error, it immediately enters the Input Error-stopped state.		
	Record the condition(s) that caused the port to enter the Input Error-stopped state.		
	If an error(s) was detected in a control symbol or packet, ignore and discard the corrupted control symbol or packet.		
	Cause the output side of the port to issue a packet-not-accepted control symbol.		
	When a link-request/port-status control symbol is received from the connected port, cause the output side of the port to transmit a link-response control symbol and if the transmitter-controlled flow control is in use on the link, to also transmit a VC_Status control symbol for each of VC1-8 that is active.		
	The transmission of a VC_Status control symbol for each of VC1-8 that is active is optional if receiver-controlled flow control in use on the link.		
	The input side of the port should also cause the output side of the port to transmit a status control symbol (for VC0).		
	The input side of the port then exits the Input Error-stopped state and resumes normal packet reception.		
	The actual transmission of the link-response, VC-status, and status control symbols may occur after the input side of the port exits the Input Error-stopped state and resumes normal packet reception		
	The link-response control symbol shall be transmitted either before any of the status and VC-status control symbols are transmitted or after all of the status and VC-status control symbols are transmitted.		
	If a status control symbol is transmitted it shall be transmitted first before any of the VC-status control symbols. Any VC-status control symbols that are transmitted shall be transmitted after the status control symbol and in order of increasing VCID.		
	The link-response control symbol shall not be transmitted until the input side of the port is ready to resume packet reception and either the buffer consumption of all packets received by the port before the link-request/port-status control symbol has been determined or the port can maintain the distinction after packet reception resumes between packets received before the reception of the link-request/port-status control symbol and packets received after the reception of the link-request/port-status control symbol		
	The status or VC-status control symbol for a VC operating in RT mode shall indicate the number of receive buffers available for that VC inclusive of the buffer consumption of all packets received and accepted by the port for that VC before the event that caused the port to enter the Input Error-stopped state.		

	The VC-status control symbol for a VC operating in CT mode shall indicate the number of receive buffers available for that VC inclusive of the buffer consumption of all packets received and accepted by the port for that VC before the link-request/port-status control symbol was received		
	The status and VC-status control symbols shall be transmitted before any packet acknowledgment control symbols are transmitted for packets received after the link-request/port-status control symbol was received.		
2	**Output Error-Stopped Recovery Process**		
	Immediately stops transmitting new packets		
	Resets the link packet acknowledgment timers for all transmitted but unacknowledged packets		
	Transmits a link-request/port-status (restart-from-error) control symbol		
	If the optional ability to perform error recovery with the ackID in the packet-not-accepted control symbol is enabled, and receipt of a Packet Not Accepted control symbol was the cause of entering the Output Error-Stopped state, then the port exits the output error-stopped state.		
	If the optional ability to perform error recovery with the ackID in the packet-not-accepted control symbol is disabled, or the port entered the Output Error-Stopped state for a reason other than receipt of a packet not accepted control symbol, the port waits until the link-response is received		
	If the ability to perform error recovery using the ackID in the packet-not-accepted control symbol is enabled, and receipt of a Packet Not Accepted control symbol was the cause of previously entering the Output Error-Stopped state, then receipt of a link-response shall complete the outstanding link-request/port-status control symbol, allowing another link-request/port-status control symbol to be transmitted. The contents of the link-response control symbol shall be treated as informational in this case.		
3	**Port *n* Latency Optimization CSRs**	Part6, Sec 7.6.15	
	Bit[1] - Error Recovery with ackID in PNA Supported Indicates whether the port can use the ackID value optionally found in a Packet Not Accepted control symbol to start transmitting packets before receipt of a link-response control symbol. 0b0 - The port cannot use the ackID value in a Packet Not Accepted control symbol 0b1 - The port can use the ackID value in a Packet Not Accepted control symbol This bit shall be read-only.		
	Bit[2] TX AckID_Status in PNA Supported Indicates whether the port places the ackID of the next expected packet in the “arbitrary, or ackID_status” field of a Packet Not Accepted control symbol, and transmits Status and VC_Status control symbols when a Packet Not Accepted control symbol is sent. 0b0 - The Packet Not Accepted “arbitrary, or ackID_status” field contains arbitrary values. 0b1 - The Packet Not Accepted “arbitrary, or ackID_status” field contains the ackID of the next expected packet. This bit shall be set if the “Error Recovery with ackID in PNA Supported” field is set. This bit shall be read-only.		

	Bit[9] - Error Recovery with ackID in PNA Enabled Controls when the port shall use the ackID value found in a Packet Not Accepted control symbol to start transmitting packets before receipt of a link-response control symbol. 0b0 - The port shall not use the ackID value in a Packet Not Accepted control symbol 0b1 - The port shall use the ackID value in received Packet Not Accepted control symbols. The port shall transmit the ackID value in a Packet Not Accepted control symbol. If the Error Recovery with ackID in PNA Supported field is clear, this field shall be reserved.		
4	**Packet Not Accepted Control Symbol**	Part6, Sec 3.4.3	
	The packet-not-accepted control symbol indicates that the port sending the control symbol has either detected an error in the received character stream or, when operating in multiple VC mode, has insufficient buffer resources and as a result may have rejected a packet or control symbol		
	The “cause” field is used to provide information about the type of error that was detected for diagnostics and debug use.		
	0b00001 - Received a packet with an unexpected ackID		
	• Received packet Not accepted control symbol with 00001 cause field value		
	0b00010 Received a control symbol with bad CRC		
	• Received packet Not accepted control symbol with 00010 cause field value		
	0b00011 Non-maintenance packet reception is stopped		
	• Received packet Not accepted control symbol with 00011 cause field value		
	0b00100 Received a packet with bad CRC		
	• Received packet Not accepted control symbol with 00100 cause field value		
	0b00101 Received an invalid character or codeword, or valid but illegal character		
	• Received packet Not accepted control symbol with 00101 cause field value		
	0b00110 Packet not accepted due to lack of resources		
	• Received packet Not accepted control symbol with 00110 cause field value		
	0b00111 Loss of descrambler sync		
	• Received packet Not accepted control symbol with 00111 cause field value		
	0b11111 General error		
	• Received packet Not accepted control symbol with 11111 cause field value		
	The packet-not-accepted control symbol causes the output side of the receiving port to enter the Output Error-stopped state		

7.18 Idle Sequence Errors Compliance

Table 33: Idle Sequence Errors Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	**IDLE1**		srio_pl_idle1_corruption_test.sv
	The IDLE1 sequence is comprised of A, K, and R (8B/10B special) characters. If an input port detects an invalid character or any valid character other then A, K, or R in an IDLE1 sequence and the port is not in the Input Error-stopped state, the port shall immediately enter the Input Error-stopped state and follow the Input Error-stopped recovery process	Part6, Sec 6.13.2.2.1	
	• A single bit transmission error can change an /A/, /K/, or /R/ code-group into a /Dx.y/ (data) code-group which is illegal in an idle sequence		
	• single bit transmission error can change an /A/, /K/, or /R/ code-group into an invalid code-group.		
	• A single bit transmission error can change an /SP/ or /PD/ (control symbol delimiters) into an invalid code-group.		
1a	Each instance of an IDLE1 sequence shall begin with the K special character.	Part6, Sec 4.7.2	
	• Received IDLE1 sequence not begin with the K special character		
1b	First four characters of an IDLE1 sequence to be K,R,R,R, the “clock compensation sequence		
	• Received IDLE1 sequence first four characters not a clock compensation sequence		
1c	The number of non-A special characters between A special characters within an IDLE1 sequence shall be no less than 16 and no more than 31.		
	• Received IDLE1 non-A special characters between A special characters within an IDLE1 sequence shall be less than 16		
	• Received IDLE1 non-A special characters between A special characters within an IDLE1 sequence shall be more than 31		
1d	Two successive A special characters are always separated by at least 16 non-A characters.		
	• Received IDLE1 sequence, two successive A special characters are always separated by less than 16 non-A characters.		
1e	Each instance of IDLE1 shall be a new IDLE1 sequence that is unrelated to any previous IDLE1 sequence.		
	• Received IDLE1 sequence is same as previous IDLE1 sequence		

Item No	Compliance Item	Spec Reference	Testcase Name
2	**IDLE2**		srio_pl_idle2_corruption_test.sv
	The IDLE2 sequence is comprised of A, K, M and R special characters and data characters. If an input port detects any of the following errors in an IDLE2 sequence and the port is not in the Input Error-stopped state, the port shall immediately enter the Input Error-stopped state and follow the Input Error-stopped recovery process	Part6, Sec 6.13.2.2.2	
	• An invalid character or any special character other than A, K, M or R		
2a	After lane alignment is achieved,		
	• a column that contains an A, but is not all As,		
	• a column that contains a K, but is not all Ks,		
	• a column that contains a M, but is not all Ms,		
	• a column that contains a R, but is not all Rs or		
	• a column that contains a data character, but is not all data characters.		
2b	Each clock compensation sequence shall be followed by an idle frame	Part6, Sec 4.7.4	
	• Received idle 2 clock compensation not followed by an idle frame		
2c	The sequence of four (4) M special characters at the beginning of a CS field marker shall not be truncated		
	• Received idle 2 with idle 2 truncation at CS field marker		
2d	Each instance of IDLE2 shall be a new IDLE2 sequence that is unrelated to any previous IDLE2 sequence		
	• Received idle2 sequence same as previous idle2 sequence		

Item No	Compliance Item	Spec Reference	Testcase Name
2e	**IDLE Sequence 2 Random Data Field**	Part6, Sec 4.7.4.1.1	srio_pl_ilde2_psr_corruption_test.sv
	The total length of the random data field shall be no less than 509 and no more than 515 characters		
	• Received idle2 random data length less than 509		
	• Received idle2 random data length more than 515		
	The pseudo-random data characters in the random data field shall occur in contiguous sequences of not less than 16 and no more than 31 pseudo-random characters		
	• Received pseudo-random data characters in the random data field shall occur in contiguous sequences of less than 16 random characters		
	• Received pseudo-random data characters in the random data field shall occur in contiguous sequences of more than 31 random characters		
	The length of the first contiguous sequence of pseudo-random characters in the random data field shall be no less than 16 and no more than 35 characters.		
	• Received first contiguous sequence of pseudo-random characters in the random data field less than 16		
	• Received first contiguous sequence of pseudo-random characters in the random data field greater than 35		
	The length of the last contiguous sequence of pseudo-random characters in the random data field shall be no less than 4 and no more than 35 characters		
	• Received last contiguous sequence of pseudo-random characters in the random data field less than 16		
	• Received last contiguous sequence of pseudo-random characters in the random data field greater than 35		
2f	**IDLE Sequence 2 CS Field Marker**	Part6, Sec 4.7.4.1.2	srio_pl_idle2_csmarker_corruption_test.sv
	A CS field marker whose first four characters are not all M special characters, fifth and seventh characters are not both D21.5 or D10.2 or sixth and eight character are not the bit wise complements of each other shall be determined to be corrupted		
	• Received CS field marker whose first four characters are not all M special characters		
	• Received CS field fifth and seventh characters are not both D21.5 or D10.2		
	• Received CS field sixth and eight character are not the bit wise complements of each other		
	Any error detected in a truncated and/or corrupted CS field marker that is determined to be the result of a transmission error and not the result of truncation, such as an “invalid” or “illegal” character, shall be reported as an input error.		

Item No	Compliance Item	Spec Reference	Testcase Name
2g	**IDLE2 Command and Status Field (CS field)**	Part6, Sec 4.7.4.1.3	srio_pl_idle2_csfield_corruption_test.sv
	A CS field whose bits [32-63] are not the bit wise complement of bits [0-31] respectively shall be determined to be corrupted. A received CS field that is determined to be truncated and/or corrupted shall be		
	• Received idle 2 CS field whose bits [32-63] are not the bit wise complement of bits [0-31]		
	Any error detected in a truncated and/or corrupted CS field that is determined to be the result of a transmission error and not the result of truncation, such as an “invalid” or “illegal” character, shall be reported as an input error		
2f	**IDLE2 CS Field Use**	Part6, Sec 4.7.4.1.4	
	Only one of the command may be issued at a time		
	• Received idle 2 with reset emphasis, preset emphasis issued at same time		
	• Received idle 2 with reset emphasis, Tap(+1) Command issued at same time		
	• Received idle 2 with reset emphasis, Tap(-1) Command issued at same time		
	• Received idle 2 with Tap(-1), Tap(+) Command issued at same time		
	• Received idle 2 with preset emphasis, Tap(+1) Command issued at same time		
	• Received idle 2 with preset emphasis, Tap(-1) Command issued at same time		
	Specific command bits may be changed only when the ACK and NACK bits are both de-asserted and the CMD bit is either de-asserted or transitioning from de-asserted to asserted		
	• Received IDLE 2 with cmd bit set during ack bit set		
	• Received IDLE 2 with cmd bit set during nack bit set		
	• Received IDLE2 with command bits changed during CMD bit assertion		
	ACK and NACK shall never be asserted at the same time.		
	• Received IDLE2 with ACK, NACK bit asserted at the same time		
	The assertion of ACK or NACK shall occur no more than 250usec after the assertion of CMD.		
	• Received IDLE2 with ACK assertion after 250usec of the CMD assertion		
	• Received IDLE2 with NACK assertion after 250usec of the CMD assertion		
	If, for any reason, the connected port fails to assert ACK or NACK within 250usec of the assertion of CMD, CMD may be deasserted. Once deasserted, CMD shall remain deasserted for at least 250usec before being reasserted.		
	• Received IDLE2 with CMD not remain deasserted for at least 250usec before being reasserted.		

Item No	Compliance Item	Spec Reference	Testcase Name
	ACK or NACK, whichever is asserted, shall be de-asserted within 250usec of receipt of a CS field with the CMD bit deasserted.		
	• Received ACK whichever is asserted not deasserted within 250usec of receipt of a CS field with the CMD bit deasserted.		
	• Received NACK whichever is asserted not deasserted within 250usec of receipt of a CS field with the CMD bit deasserted.		

7.19 Control symbol Error Compliance

Table 34: Control symbol Error Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	**Control Symbol Errors**	Part6, Sec 6.13.2.3	
	There are two types of detectable control symbol errors An uncorrupted control symbol that violates the link protocol A corrupted control symbol		
2	**Link Protocol Violations**	Part6, Sec 6.13.2.3 .1	srio_pl_link_protocol_violation_test.sv
	The reception of a control symbol with no detected corruption that violates the link protocol shall cause the receiving port to immediately enter the appropriate Error-stopped state.		
	Stype1 control symbol protocol errors shall cause the receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery process		
	Stype0 control symbol protocol errors shall cause the receiving port to immediately enter the Output Error-stopped state if not already in the Output Error-stopped state and follow the Output Error-stopped recovery process		
	If both stype0 and stype1 control symbols contain protocol errors, then the receiving port shall enter both Error-stopped states and follow both error recovery processes		
	• Received Unexpected packet-accepted, packet-retry, or packet-not-accepted control symbol		
	• Received Packet acknowledgment control symbol with an unexpected packet_ackID value		
	• Link timeout while waiting for an acknowledgment or link-response control symbol		
	• Receipt of a packet-retry symbol when operating in multi-VC mode		
	• Receipt of an unsolicited Timestamp Control Symbol when the device is a timestamp Master		

Item No	Compliance Item	Spec Reference	Testcase Name
3	**Corrupted Control symbols**	Part6, Sec 6.13.2.3.2	srio_pl_corrupted_control_symbol_test.sv
	The reception of a control symbol with detected corruption shall cause the receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery process		
	• A Control Symbol 24 or Control Symbol 48 containing invalid characters or valid but non-data characters		
	• A Control Symbol 64 containing invalid codewords		
	• A control symbol with an incorrect CRC value		
	• A Control Symbol 48 or Control Symbol 48 whose start delimiter (SC or PD) occurs in a lane whose lane_number mod4!= 0		
	• A Control Symbol 48 that does not have a end delimiter in the seventh character position after its start delimiter and with the same value as the start delimiter		
	• Control Symbol 64 control codeword out of sequence or incomplete sequence of Control Symbol 64 control codewords		
4	**Link-request corruption**		srio_pl_link_request_corruption_test.sv
4a	For reset-device or reset-port where four link-request control symbols are transmitted each of the four link-request control symbols shall be preceeded by a Seed ordered sequence.	Part6, Sec 5.5.4.2	
	• Received reset-device link request command shall not preceeded by a seed ordered sequence.		
	• Received reset-port link request command shall not preceeded by a seed ordered sequence		
4b	If a packet is cancelled with a link-request control symbol, a Seed ordered sequence shall be transmitted between the end of the packet and the link-request control symbol.	Part6, Sec 5.6.2.2	
	• Received a link-request packet termination delimiter not preceeded by Seed ordered sequence.		
4c	The Seed ordered sequence shall be transmitted before each link-request control symbol.	Part6, Sec 5.8.1	
	• Received link request control symbol not preceeded by Seed ordered sequence		
4d	If a link-request control symbol terminates a packet on a link that is operating with IDLE2, the SYNC sequence is required to precede the link-request control symbol	Part6, Sec 6.6.1.2	
	• Received link-request control symbol terminates a packet on a link that is operating with IDLE2, not preceded by SYNC sequence		
4e	If a link-request control symbol terminates a packet on a link that is operating with IDLE3, the Seed ordered sequence is required to precede the link-request control symbol.	Part6, Sec 6.6.1.2	
	• Received link-request control symbol terminates a packet on a link that is operating with IDLE3, not preceded by Seed ordered sequence		

Item No	Compliance Item	Spec Reference	Testcase Name
4f	The reset-device command does not generate a link-response control symbol.	Part6, Sec 3.5.5.2	
	• Received link-response control symbol for reset-device command		
4g	A port receiving a reset-device command in a link-request control symbol shall not perform the reset function unless it has received four reset-device commands in a row without any other intervening packets or control symbols, except status control symbols.	Part6, Sec 3.5.5.1	
	• Received sop/eop in between reset-device command		
4h	The reset-port command shall not generate a link-response control symbol.	Part6, Sec 3.5.5.3	
	• Received link-response control symbol for reset-port command		
4i	The link-request/port-status control symbol requires a response.	Part6, Sec 6.7	
	Port Link Timeout Control CSR Port Link Timeout is for link events such as sending a packet to receiving the corresponding acknowledge and sending a link-request to receiving the corresponding link-response	Part6, Sec 7.6.2	
	• Allow link timeout		

7.20 Detectable Error Compliance

Table 35: Detectable Error Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	The start-of-packet control symbol is used to delimit the beginning of a packet	Part6, Sec 3.5.1	srio_pl_err_test.sv
	• Received io packet without start of packet control symbol		
	• Received msg packet without start of packet control symbol		
	• Received GSM packet without start of packet control symbol		
	• Received DS packet without start of packet control symbol		
2	The stomp control symbol is used to cancel a partially transmitted packet	Part6, Sec 3.5.2	
	• Received stomp control symbol when there is no data progress		
3	The link-response control symbol is used by a port to respond to a link-request control symbol	Part6, Sec 3.4.7	
	• Received link-response control symbol without link request control symbol		

Item No	Compliance Item	Spec Reference	Testcase Name
4	The control symbol containing the start of packet delimiter shall immediately precede the first byte of the packet or the first byte of an embedded control symbol.	Part6, Sec 5.6.2	
	• Received start of packet delimiter shall immediately precede the first byte of the idle sequence		
5	When a restart-from-retry control symbol is received, exit the Input Retry-stopped state and resume packet reception.	Part6, Sec 6.9.1.4	
	• Received start of packet control symbol in Input Retry-stopped state		
	• Received io packets in Input Retry-stopped state		
	• Received msg packets in Input Retry-stopped state		
	• Received GSM packets in Input Retry-stopped state		
	• Received DS packets in Input Retry-stopped state		
	• Received stomp control symbol in Input Retry-stopped state		
	• Received packet not accepted control symbol in Input Retry-stopped state		
	• Received packet retry control symbol in Input Retry-stopped state		
	• Received status control symbol in Input Retry-stopped state		
	• Received Link response control symbol in Input Retry-stopped state		
	• Received Link request/reset device control symbol in Input Retry-stopped state • Received Link request/reset port control symbol in Input Retry-stopped state		
	• Received restart-from-retry control symbol with crc error in Input Retry-stopped state		
6	When a link-request/port-status control symbol is received from the connected port, cause the output side of the port to transmit a link-response control symbol. The input side of the port should also cause the output side of the port to transmit a status control symbol (for VC0). The input side of the port then exits the Input Error-stopped state and resumes normal packet reception	Part 6, Sec 6.13.2.6	
	• Received start of packet control symbol in Input Error-stopped state		
	• Received io packets in Input Error-stopped state		
	• Received msg packets in Input Error-stopped state		
	• Received GSM packets in Input Error-stopped state		
	• Received DS packets in Input Error-stopped state		
	• Received stomp control symbol in Input Error-stopped state		
	• Received packet not accepted control symbol in Input Error-stopped state		
	• Received packet retry control symbol in Input Error-stopped state		
	• Received status control symbol in Input Error-stopped state		
	• Received Link response control symbol in Input Error-stopped state		

Item No	Compliance Item	Spec Reference	Testcase Name
	• Received Link request/reset device control symbol in Input Error-stopped state • Received Link request/reset port control symbol in Input Error-stopped state		
	• Received Link request/port status control symbol with crc error in Input Error-stopped state		
7	The port exits the Output Error-stopped state and resumes transmission with the next RT or CT packet according to the bandwidth allocation algorithm using the ackID value contained in the link-response control symbol.	Part6, Sec 6.13.2.7	
	• Received start of packet control symbol in Output Error-stopped state		
	• Received io packets in Output Error-stopped state		
	• Received msg packets in Output Error-stopped state		
	• Received GSM packets in Output Error-stopped state		
	• Received DS packets in Output Error-stopped state		
	• Received stomp control symbol in Output Error-stopped state		
	• Received packet not accepted control symbol in Output Error-stopped state		
	• Received packet retry control symbol in Output Error-stopped state		
	• Received status control symbol in Output Error-stopped state		
	• Received Link response control symbol with crc error in Output Error-stopped state		
	• Received Link request/reset device control symbol in Output Error-stopped state • Received Link request/reset port control symbol in Output Error-stopped state • Received Link request/port status control symbol in Output Error-stopped state		
8	LP-Serial packets shall have a length that is an integer multiple of 32 bits. If the length of a packet defined by the above logical and transport specifications is an odd multiple of 16 bits, a 16-bit pad whose value is 0 (0x0000) shall be appended at the end of the packet such that the resulting padded packet is an integer multiple of 32 bits in length	Part6, Sec 2.3	
	• Received LP serial packet length not an integer multiple of 32 bits		

8 Part 9 - Flow Control Logical Layer Extensions Specification

8.1 Congestion Management Compliance

Table 36: Congestion Management Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	Congestion management may be implemented by endpoints or switches independent of the flow arbitration protocol.	Part9, Sec 1.1	srio_ll_lfc_xoff_xon_test.sv
	The XOFF CCPS are sent to shut off select flows at their source end points	Part9, Sec 2.1	
	The XON CCPS are sent to the same source end points to restore those flows	Part9, Sec 2.1	
	Flow control packets do not contain response packets.	Part9, Sec 2.3	
	The congestion management commands affect flows on a packet boundary basis.	Part9, Sec 1.2.2	
2	**Orphaned XOFF Mechanism**	Part9, Sec 2.4.2.1	srio_ll_lfc_orphaned_xoff_test.sv
	Due to the possibility of XON flow control packets being lost in the fabric, there shall be an orphaned XOFF mechanism for the purpose of restarting orphaned flows which were XOFF'd but never XON'd in end points.		
	The Orphaned XOFF Mechanism is intended to work with the rest of the XON/XOFF CCPs to handle the short term congestion problem as previously described, and so shall operate such that software intervention is not required or inadvertently invoked.		
3	**Controlled Flow List**	Part9, Sec 2.4.2.2	
	It is required that elements which send XOFFs keep a list of flows they have stopped, along with whatever flow-specific information is needed to select flows for restart, such as per-flow XON watermark level, or relative shut off order.		
	This information shall be stored along with flow identification information in a "controlled flow list", a memory structure associated with the controlling element. It shall be permissible in the time following the sending of a XOFF CCP for the flow control -initiating element to re-evaluate system resources and modify the flow restart ordering or expected XON watermark level within the controlled flow list to better reflect current system state		
	Should the source fail to utilize the resource in an expected interval, the destination may take action to recover the resource		
4	**XOFF/XON Counters**		

Item No	Compliance Item	Spec Reference	Testcase Name
	XOFF/XON counters shall be instantiated for some number of output flows at the end point.		
	all flows must be associated with a counter		
	The counter is initialized to zero at start up or when a new DestinationID and given Priority is initialized.		
	The counter increments by one for each associated XOFF CCP and decrements by one for each associated XON CCP, stopping at zero		
	Only when this counter is equal to zero is the flow enabled		
	In no event shall the counter wrap upon terminal count.		
	If the orphaned XOFF mechanism activates, the counter is reset to zero and the flow is restarted.		
5	**End Point Congestion Management Rules**	Part9, Sec 2.4.3	
	A XOFF flow control transaction stops all transaction request flows of the specified priority and lower targeted to the specified destination and increments the XON/XOFF counter associated with the specified flowID		
	A XON flow control transaction decrements the XON/XOFF counter associated with the specified flowID. If the resulting value is zero, the transaction request flows for that flowID and flowIDs of higher priority are restarted.		
	An end point must be able to identify an orphaned XOFF'd flow and restart it.		
	A destination end point issuing a XOFF Flow Control transaction must maintain the information necessary to restart the flow with a XON flow control transaction when congestion abates		
	Upon detection of congestion within one of its ports, the destination end point shall send required CCP(s) as quickly as possible to reduce latency back to the source end point.		

8.2 Flow Arbitration Compliance

Table 37: Flow Arbitration Compliance

1	The flow arbitration protocol only applies to endpoints	Part9, Sec 1.1	
	The arbitration protocol does consist of multiple transactions between the source of a data flow and the destination.	Part9, Sec 2.3	
2	The flow arbitration protocol extends the Congestion Control Packet (CCP) protocol REQUEST XOFF to indicate un-availability of resources. XON to allow and grant use of resources RELEASE	Part9, Sec 2.2	
3	Fixed / Static Resource Allocation	Part9, Sec 1.2.1	
	Fixed allocation of resources occurs by system design.		
	Systems with smaller topologies, or with endpoint resources sufficient for all anticipated flows, do not require any specific management		
	Resources may also be statically allocated on an individual connection basis		
4	Dynamic Resource Allocation Protocol	Part9, Sec 1.2.2	
	The dynamic arbitration protocol is designed to arbitrate and allocate resources to flows for short durations of time.		
	It allows a fewer number of resources to be dynamically shared among a larger number of flows		
	The dynamic arbitration of resources will prevent data loss caused by over-running the receiver		
5	**Arbitration Protocol**	Part9,Sec 2.2.1	srio_ll_lfc_flow_arb_spdu_test.sv
	Single PDU Transfer Scenario		
	The transmitting endpoint sends a single PDU request		
	The receiving endpoint will respond with either a XON(ARB) or XOFF(ARB) message depending on whether it has buffer and context resources available.		
	if the receiving endpoint responds with a XOFF(ARB), the transmitting endpoint can send a new REQUEST message to ask for resources		
	If the receiving endpoint responds with a XON(ARB), the transmitting endpoint can start transmitting the PDU segments once it receives the XON(ARB) message		
	The receiver will automatically de-allocate resources once it receives the last packet for the PDU.		
	Multi-PDU Transfer Scenario		srio_ll_lfc_flow_arb_mpdu_test.sv
	When the transmitting endpoint sends a request pertaining to the transfer of multiple PDUs the receiving endpoint, similarly to the single PDU case, shall respond with either the XON(ARB) or the XOFF(ARB) protocol depending on the availability of buffering resources		

	If the receiving endpoint responds with a XON(ARB) message, the transmitting endpoint can start sending the PDU segments once it receives the XON(ARB) message		
	The transmitter can send multiple PDUs without having to renegotiate the resources.		
	The receiver will hold the allocated resources until it receives a RELEASE message from the transmitting endpoint.		
	The receiver can also inform the transmitting endpoint of its desire to de-allocate the resources, by sending a XOFF(ARB) message.		
	The receiver shall de-allocate the resources only when it has received the RELEASE message.		
6	**Number of Outstanding Requests**	Part9, Sec 2.2.2	srio_ll_lfc_flow_arb_spdu_pipeline_req_test.sv srio_ll_lfc_flow_arb_mpdu_pipeline_req_test.sv
	In the arbitration protocol the transmitting endpoint has to wait at least a round trip time after it has sent the request message before it can start transmitting		
	In order to overlap the request phase with the data transmission phase, the transmitter is allowed to have a maximum of one outstanding request in the system, that is, it can pipeline requests to increase the efficiency of the system.		
	The requests and the corresponding responses are identified by a 1 bit sequence number		
	This pipelining of requests is allowed for both single PDU and multi-PDU requests.		
	The transmitting endpoint issues a request.		
	The request is processed by the receiving endpoint and a XON issued.		
	Once the transmitter receives the XON message, it can start transmitting the data and it may also pipeline another request		
	The pipelined request shall not be honored until the current transaction has been completely received and resources are available for the next transaction.		
	The pipelining of requests is managed by the source. It only issues the next request when the current request has been acknowledged.		
	The destination only acknowledges the next request when the current request has completed		
	So, the destination only has to queue up one outstanding request per flow.		
	This pipelining also offers the destination an opportunity to use the pending requests to get a look at the incoming traffic and make better allocation decisions should there be limited resources.		
7	**Endpoint Rules for the Arbitration Protocol**	Part9, Sec 2.4.7	
	The transmitter shall issue a REQUEST when it wishes the receiver to allocate a resource to a particular flow. Note that this does not imply a PDU is immediately ready for transmission. The algorithms for resource allocation are up to the implementation		

	The receiver shall respond to all REQUEST messages with a XOFF(ARB) or XON(ARB) depending on the availability of resources and the arbitration policy at the receiver		
	The transmitter may send a new REQUEST message if: 1) it did not receive a response to the previous REQUEST message and timed out, or 2) it received a XOFF(ARB) message from the receiving endpoint		
	Sequence numbers shall remain coherent for each individual flow. The sequence number shall remain the same for REQUESTs reissued for a given flow, without having received a response. The sequence number shall advance for any new REQUESTs on a given flow.		
	If a single PDU request is granted, the receiver may deallocate the resources at any time after: 1) it receives the last segment for the PDU, or 2) it does not receive a packet and an idle counter for the session times out (see rule I). The transmitter shall assume the context is no longer available upon sending the last segment for the PDU.		
	If the resources were granted in response to a multi-PDU request the transmitter may transmit PDUs continuously on that flow until the resource is de-allocated		
	The transmitter may relinquish a multi-PDU context by sending a RELEASE message after completion of the current PDU.		
	The receiver may send a XOFF(ARB) message during the multi-PDU transfer to indicate its desire to deallocate resources. The transmitter, upon receiving a XOFF(ARB) message during the multi-PDU transfer, shall complete transmission of the current PDU and send a RELEASE message to allow the receiver to de-allocate the resources. The receiver may not deallocate the resources until the RELEASE message is received.		
	The receiver may delay sending responses to the REQUEST commands to consider which REQUESTs to grant or reject. There is no ordering requirement for processing requests from different flows		
	Only a single instance of resources shall be allocated to a flow at any point in time		
	The transmitter may issue a single additional REQUEST in advance of completion of the current PDU. However, the transmitter shall not have more than one outstanding REQUEST at any point in time.		
	REQUEST, XON(ARB), and XOFF(ARB) messages may be sent in any flow (such as a high priority channel). RELEASE messages shall be sent in the same flow that the context is allocated for.		
8	**Abnormal De-allocation of Resources**	Part6, Sec 2.4.8	
	Single PDU request		
	The PDU may be aborted. An aborted PDU results in the de-allocation of the resources		srio_ll_lfc_flow_arb_spdu_abort_test.sv
	If the receiver does not receive a packet from the transmitter before the idle counter for the session times out, the resources would be de-allocated. Note that the use of a timer is implementation specific. Incorrect use of a timer may result in packet loss		srio_ll_lfc_flow_arb_spdu_timeout_test.sv

	Multi-PDU request		
	A PDU may be aborted. The resources will still not be de-allocated until a RELEASE message is received.		srio_ll_lfc_flow_arb_mpdu_abort_test.sv
	If the receiver does not receive a packet from the transmitter before the idle counter for the session times out, the receiver shall first attempt to use the XOFF(ARB) / RELEASE handshake to deallocate the context. If a subsequent timeout is encountered, the SAR resources are asynchronously deallocated.		srio_ll_lfc_flow_arb_mpdu_timeout_test.sv

8.3 Packet Format Compliance

Table 38: Packet Format Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	**Logical Layer Packet Format**	Part6, Sec 3.2	
	The type 7 FLOW CONTROL packet formats (Flow Control Class) are used by a RapidIO switch or end point processing element to stop (XOFF) and start (XON) the flow of traffic to it from a targeted RapidIO end point processing element.		
	Type 7 packets do not have a data payload and do not generate response packets		
	The origin of a flow control packet shall set the SOC (Source of Congestion) bit to (SOC=0) if it is a switch or (SOC=1) if it is an end point.		
	The Flow Arbitration Message (FAM) field is used to modify the XON or XOFF message for the purposes of flow arbitration.		
2	**Specific Field Definitions and Encodings for Type 7 Packets**		
	XON/XOFF - 0 XOFF For devices not supporting flow arbitration. Stop issuing requests for the specified and lower priority transaction request flows		
	XON/XOFF - 1 XON For devices not supporting flow arbitration. Start issuing requests for the specified and higher priority transaction request flows		

Item No	Compliance Item	Spec Reference	Testcase Name
	flowID Highest priority affected transaction request flow for VC0 0000000 (Flow0A) 0000001 (Flow 0B) 0000010 (Flow 0C) 0000011 (Flow 0D) 0000100 (Flow 0E) 0000101 (Flow 0F)		srio_ll_lfc_multi_xoff_xon_same_flowid_test.sv srio_ll_lfc_multi_xoff_xon_diff_flowid_test.sv
	For VC1-VC8 the following flow IDs will result in the VC1-VC8 flow control 1000001 (Flow 1A) 1000010 (Flow 2A) 1000011 (Flow 3A) 1000100 (Flow 4A) 1000101 (Flow 5A) 1000110 (Flow 6A) 1000111 (Flow 7A) 1001000 (Flow 8A)		
	destinationID Indicates which end point the CCP is destined for (sourceID of the packet which caused the generation of the CCP)		
	tgtdestinationID Combined with the flowID field, indicates which transaction request flows need to be acted upon (destinationID field of the packet which caused the generation of the CCP).		
	SOC 0b0 - Source Of Congestion is a Switch 0b1 - Source Of Congestion is an End Point		
3	**Flow Arbitration Message Fields (FAM)**	Part6, Sec 3.3	
	The flow arbitration protocol uses the 3 FAM bits along with the XON/XOFF bit to identify the messages		
	XON/XOFF FAM **0** 0b000 - XOFF: Transmit off 0 0b010 XOFF(ARB): Flow Request Rejected. Message with sequence number 0 (LSB) in response to REQUEST with sequence number 0. 0 0b011 XOFF(ARB) XOFF(ARB):Flow Request Rejected. Message with sequence number 1(LSB) in response to REQUEST with sequence number 1 0 0b10Y RELEASE 1 0b000 XON: Transmit on 1 0b01YXON(ARB): 1 0b10YREQUEST: Request Flow Single PDU. 1 0b11Y REQUEST: Request Flow Multi-PDU		

Item No	Compliance Item	Spec Reference	Testcase Name
	Transport and Physical Layer Packet Format	Part9, Sec 3.4	
	The destinationID field of the CCP packet is the sourceID field from packets associated with the congestion event, and is the target of the flow control transaction.		
	The tgtdestinationID field is the destinationID field from packets associated with the congestion event, and was the target of those packets.		
	The tgtdestinationID field is used by the target of the flow control packet to identify the transaction request flow that needs to be acted upon.		
	when tt=0b01 there will be a pad after the CRC.		

8.4 Flow Arbitration Error Compliance

Table 39: Flow Arbitration Error Compliance

1	The transmitter may send a new REQUEST message if: 1) it did not receive a response to the previous REQUEST message and timed out.	Part9, Sec 2.4.7	srio_ll_lfc_flow_arb_spdu_err_test.sv srio_ll_lfc_flow_arb_mpdu_err_test.sv
	• Received a new Request, before the previous request not acknowledged. (Not received XON message)		
2	The receiving endpoint responds with a XON(ARB), the transmitting end-point can start transmitting the PDU segments once it receives the XON(ARB) message	Part9, Sec 2.2.1	
	• Received pdus before the previous request not acknowledged. (Not received XON message)		
3	The sequence number shall remain the same for REQUESTs reissued for a given flow, without having received a response.	Part9, Sec 2.4.7	
	• Received same requests reissued for a given flow with different sequence number without having received a response		
4	The sequence number shall advance for any new REQUESTs on a given flow.	Part9, Sec 2.4.7	
	• Received different requests with same sequence number		
5	REQUEST, XON(ARB), and XOFF(ARB) messages may be sent in any flow (such as a high priority channel). RELEASE messages shall be sent in the same flow that the context is allocated for.	Part9, Sec 2.4.7	
	• Received Release message with different flow that the context is allocated for		

6	The transmitter may issue a single additional REQUEST in advance of completion of the current PDU. However, the transmitter shall not have more than one outstanding REQUEST at any point in time	Part9, Sec 2.4.7	
	• Received more than 1 outstanding request		
7	The transmitter, upon receiving a XOFF(ARB) message during the multi-PDU transfer, shall complete transmission of the current PDU and send a RELEASE message to allow the receiver to de-allocate the resources	Part9, Sec 2.4.7	
	• Received continuous pdus, even though XOFF message is sent. (i.e Transmitter not send Release message, even though it receives XOFF message. It started transmitting next pdu without send release message		

8.5 Detectable Error Compliance

Table 40: Detectable Error Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	Type 7 packets do not have a data payload and do not generate response packets.	Part9, Sec 3.2	srio_ll_lfc_err_test.sv
	• Received response for Type 7 packets		
	• Received Type7 packets with data payload		
2	**Processing Elements Features CAR** Bit[24] - Flow Control Support	Part9, Sec 4.2.1	
	• Change the Processing Elements Features CAR Bit[24] to zero. • Received Type 7 packet		
3	**Processing Elements Features CAR** Bit[20] - Flow Arbitration Support		
	• Change the Processing Elements Features CAR Bit[24] to zero. • Received Type7 packet for Flow arbitration		
4	Cyclic Redundancy Code used to detect transmission errors in the packet	Part6, Sec 2.2	
	Packet with an incorrect CRC value The receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	Bit[13] (Received packet with bad CRC) in Port n Error Detect CSR	Part8, Sec 2.5.15	
	• Received Type7 request packet with incorrect CRC		

Item No	Compliance Item	Spec Reference	Testcase Name
5	Bit[12] (Received packet with unexpected ackID) Port n Error Detect CSR	Part8, Sec 2.5.15	
	Packet with an unexpected ackID value Reception of packet (with an unexpected ackID value) causes the receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	ackID values on each link are assigned sequentially for each subsequent transmitted packet, an error in the ackID field is easily detected.	Part6, Sec 2.4	
	• Received Type7 request packet with incorrect ackid		

8.6 Random Compliance

Table 41: Random Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	Randomly generate flow control packets with all io packets		srio_ll_lfc_xoff_xon_random_io_test.sv
2	Randomly generate flow control packets with all message packets		srio_ll_lfc_xoff_xon_random_msg_test.sv
3	Randomly generate flow control packets with all GSM packets		srio_ll_lfc_xoff_xon_random_gsm_test.sv

9 Part 10 - Data Streaming Logical Specification

9.1 Functional Features Compliance

Table 42: Functional Features Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	Support for Protocol Data Units (PDUs) of up to 64k bytes through Segmentation and Reassembly (SAR).	Part10, Sec 1.3.1	srio_ll_ds_max_test.sv
2	Support for hundreds of traffic classes.		srio_ll_ds_traffic_mgmt_test.sv
3	Support for thousands of data streams between end points		srio_ll_ds_multi_stream_test.sv
4	Support for concurrent interleaved PDUs between end points		srio_ll_ds_interleaved_test.sv

9.2 Performance Features Compliance

Table 43: Performance Features Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	Multiple transactions are allowed concurrently in the system, otherwise a majority of the potential system throughput is wasted.	Part10, Sec 1.3.3	srio_ll_ds_concurrent_test.sv
2	Multiple end point to end point concurrent data streams are supported for high fabric utilization.		

9.3 Operation Ordering Compliance

Table 44: Operation Ordering Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	A transaction request flow is defined as an ordered sequence of request transactions comprising a specific PDU from a given source ID to a given destination ID.	Part10, Sec 2.4	srio_ll_ds_operation_ordering_test.sv
	RapidIO allows multiple transaction request flows between any source ID and destination ID pair.		
	single virtual channel application which must follow the same prioritization of flows and labeling as the other logical layers (flowID A, flowID B, flowID C, etc.). The channel label (0) is dropped. This channel may include traffic from the other logical layers.		
	A traffic stream being transmitted between a source and a destination ID pair must utilize the same flowID value so that the ordering of the traffic stream is maintained.		
2	It is expected that in a mixed control and data plane application that both I/O logical and data streaming transaction request flows will exist in a RapidIO system simultaneously, possibly between the same ID pairs.		srio_ll_io_ds_test.sv

9.4 Basic feature Compliance

Table 45: Basic feature Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	Data stream transactions are not acknowledged with a response packet.	Part10, Sec 2.2	
2	From the source's viewpoint: destination ID+cos+streamID represents a unique stream.	Part10, Sec 3.2.2	
3	From the destination's viewpoint: source ID+cos+streamID represents a unique stream.		
4	The maximum size of a PDU that a particular destination can accept is specified in a CAR	Part10, Sec 3.2.4	
	The block size used for the segmentation process is specified by the Maximum Transmission Unit, or MTU, parameter.		

Item No	Compliance Item	Spec Reference	Testcase Name
	Since there may be a large number of PDU sources and concurrent contexts per source, the amount of context state that a destination may have to handle can potentially get very large. The number of contexts that can be supported by a particular destination end point is specified in a CAR		
5	**Multi Segment**	Part10, Sec 3.2.4	srio_ll_ds_sseg_test.sv
	A typical sequence is made up of three types of transactions, a start segment, some number of continuation segments, and an end segment.		
	Start segments and continuation segments are always filled to the MTU size.		
	End segments are variable in size containing the remainder of the PDU.		
	Since flowIDs and the cos are assigned on a PDU basis, all segments of a PDU must also have that same flowID and cos assignments		
	An end segment may be received immediately after a start segment.		
5a	O bit - 0 P bit - 0 Even number of half-words and no pad byte	Part10, Sec 4.2	
	• Received multi segment pdu with Even number of half-words and no pad byte		
5b	O bit - 0 P bit - 1 Even number of half-words and a pad byte		
	• Received multi segment pdu with Even number of half-words and a pad byte		
5c	O bit - 1 P bit - 0 Odd number of half-words and no pad byte		
	• Received multi segment pdu with Odd number of half-words and no pad byte		
5d	O bit - 1 P bit - 1 Odd number of half-words and a pad byte		
	• Received multi segment pdu with Odd number of half-words and a pad byte		
5e	PDU length - This is the length in bytes of the segmented PDU		
	• Received multi segment pdu to cover all the possible PDU length combination		
5f	For a packet whose length, exclusive of CRC, is 80 bytes or less, a single CRC is appended at the end of the logical fields. Unpadded Packet of Length 80 Bytes or Less Padded Packet of Length 80 Bytes or Less	Part6, Sec 2.4.1	
	• Received multi segment with MTU size value less than 80 bytes		

Item No	Compliance Item	Spec Reference	Testcase Name
	For packets whose length, exclusive of CRC, is greater than 80 bytes, a CRC is added after the first 80 bytes and a second CRC is appended at the end of the logical layer fields. Unpadded Packet of Length Greater than 80 Bytes Padded Packet of Length Greater than 80 Bytes		
	• Received multi segment with MTU size value greater than 80 bytes		
	If the CRC appended to the end of the logical layer fields does not cause the end of the resulting packet to align to a 32-bit boundary, a two byte pad of all logic 0s is postpended to the packet		
	The pad of logic 0s allows the CRC check to always be done at the 32-bit boundary. A corrupt pad may or may not cause a CRC error to be detected, depending upon the implementation.		
6	**Single Segment**	Part10, Sec 3.2.4	srio_ll_ds_meg_test.sv
	If a PDU is equal to or less than the MTU size, it is carried in a single segment		
	A single segment may also be variable in size, matching the PDU payload		
	If the end of the PDU data is encountered, the start segment then re-marked as a single segment.	Part10, Spec 3.2.5	
6a	O bit - 0 P bit - 0 Even number of half-words and no pad byte	Part10, Sec 4.2	
	• Received single segment pdu with Even number of half-words and no pad byte		
6b	O bit - 0 P bit - 1 Even number of half-words and a pad byte		
	• Received single segment pdu with Even number of half-words and a pad byte		
6c	O bit - 1 P bit - 0 Odd number of half-words and no pad byte		
	• Received single segment pdu with Odd number of half-words and no pad byte		
6d	O bit - 1 P bit - 1 Odd number of half-words and a pad byte		
	• Received single segment pdu with Odd number of half-words and a pad byte		
6e	PDU length - This is the length in bytes of the segmented PDU		
	• Received single segment pdu to cover all the possible PDU length combination		

Item No	Compliance Item	Spec Reference	Testcase Name
6f	For a packet whose length, exclusive of CRC, is 80 bytes or less, a single CRC is appended at the end of the logical fields. Unpadded Packet of Length 80 Bytes or Less Padded Packet of Length 80 Bytes or Less	Part6, Sec 2.4.1	
	• Received single segment with pdu size 80 bytes or less		
	For packets whose length, exclusive of CRC, is greater than 80 bytes, a CRC is added after the first 80 bytes and a second CRC is appended at the end of the logical layer fields. Unpadded Packet of Length Greater than 80 Bytes Padded Packet of Length Greater than 80 Bytes		
	• Received single segment with pdu size greater than 80 bytes		
	If the CRC appended to the end of the logical layer fields does not cause the end of the resulting packet to align to a 32-bit boundary, a two byte pad of all logic 0s is postpended to the packet		
	The pad of logic 0s allows the CRC check to always be done at the 32-bit boundary. A corrupt pad may or may not cause a CRC error to be detected, depending upon the implementation		

Item No	Compliance Item	Spec Reference	Testcase Name
7	**Rules for Segmentation and Reassembly**	Part10, Sec 3.2.5	
	Segmentation (source)		
	No more than one PDU from a given stream shall be segmented at a time		
	No more than one PDU from a given RapidIO flow shall be segmented at a time		
	PDUs associated with different RapidIO flows may be segmented concurrently		
	The first segment is marked as start segment		
	The start segment is filled to the end of the PDU data or to the MTU size.		
	If the start segment reaches MTU size (and there is remaining PDU data), the start segment is encapsulated, and a continuation segment is opened		
	Continuation segments are filled to MTU size from the PDU data, in order.		
	When the end of PDU data is encountered, the segment is marked as the end segment. The end segment data payload size may be less than or equal to the MTU size		
	Reassembly (destination)		
	Upon receiving a segment with a start bit, the reassembly unit opens a “context” containing the virtual stream ID and associates it with the segmentation context consisting of the source ID, the destination ID, and the physical channel ID. (The destination ID is only required for devices supporting multicast and/or multiple destination IDs.)		
	The reassembly process transfers the entire payload into the reassembly buffer in order. The amount of data transferred is counted for comparison to the length field.		
	If the packet is a single segment, the amount of payload data must be equal to or less than the MTU size or the PDU is defective.		
	If the packet is a start segment and the payload data does not match the MTU size the PDU is defective.		
	Reassembly continues with continuation packets. All continuation packets must match the MTU size or the PDU is defective. All data transferred to the reassembly buffer is counted.		
	An end segment terminates the reassembly process. An end segment may be received immediately after a start segment. The data payload size must be less than or equal to the MTU size or the PDU is defective. The data from the end segment is transferred according to the data payload size and counted		

9.5 Traffic Streams Compliance

Table 46: Traffic Streams Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	A stream identifier identifies independent streams of traffic between the end producer and end consumer of the encapsulated data	Part10, Sec 2.3	
	A traffic stream is a series of PDUs that have an ordering relationship between each other.		
	A PDU may or may not have an ordering relationship with another PDU being transmitted between that same producer and consumer, depending upon the higher layer protocol being carried.		
2	All traffic streams are mapped onto transaction request flows	Part10, Sec 2.4	
3	The data streaming logical layer uses a virtual stream identifier (VSID) to allow multiple end to end traffic streams of PDUs to be uniquely identified and managed concurrently within the RapidIO system	Part10, Sec 2.3	
	The VSID allows the traffic to be reassociated with an appropriate application at the egress without having to perform a second protocol-specific classification.		

9.6 Traffic Management Compliance

Table 47: Traffic Management Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	Data Streaming Traffic Management (TM) supports end-to-end flow control through multiple mechanisms	Part10, Sec 3.4	
	Traffic Management uses the extended packet format for Type 9		
	The traffic management format makes up a message that contains: <VSID><Wildcard+Mask> <Message XType><Parameter 1><Parameter 2>		
	The VSID plus the wildcard and mask fields in the extended packet header forms the operand (the queue or queues) for the traffic management message.		
	The message type and type specific parameters form the operation.		
2	Traffic Management Operand	Part10, Sec 3.4.1	
	All Data Streaming traffic management protocols use a common mechanism for queue designation.		
	The message overloads the data streaming VSID fields to define the stream to which to apply the message		
	In addition, the extended header format includes modifiers, in the form of a wild card and a mask to expand the scope of the message beyond a single stream.		
	The message can apply to: • A single stream, identified by <Dest><cos><streamID> • A group of streams in a class identified by <Dest><cos> • A group of classes, identified by <Dest><cos><mask> • All classes for a given port, identified by <Dest>		
	The operand is always based on a VSID that is the target (intended destination) of the data.		
3	**On/Off Traffic Management**	Part10, Sec 3.4.2	srio_ll_ds_on_off_traffic_mgmt_test.sv
	Egress to Ingress: <Q> XOFF <Q> XON		
	Ingress to Egress <Q> Q_STATUS <Level>		
	Basic On/Off traffic management consists of egress to ingress messages that direct the operand <Q> be stopped (XOFF) or started (XON).		
	The ingress may signal the need for service with the Q_STATUS message, with *level* indicating the relative fullness of the queue.		

Item No	Compliance Item	Spec Reference	Testcase Name
4	**Rate Base Traffic Management**	Part10, Sec 3.4.3	srio_ll_ds_rate_based_traffic_mgmt_test.sv
	From Egress to Ingress <Q> XON <Q> XOFF? <Q> INCREASE <Amount> <Q> DECREASE <Amount>		
	From Ingress to Egress? <Q> Q_STATUS <Level>		
	Rate based flow control is a *relative* control protocol.		
	<Amount> is a ratio relative to the current rate of traffic flow		
	The ingress is pre-configured with a specific traffic scheduling algorithm.		
	The egress uses the INCREASE / DECREASE mechanism to modify the ingress' scheduling process, usually based on the egress' ability to move the traffic to its next destination.		
	DECREASE<0> is a special message meaning MAINTAIN current rate, and INCREASE<max> is a special message to DOUBLE the current rate		
	The ingress may use the Q_STATUS message to indicate the status of its queues, with level indicating the relative fullness of the queue, allowing a closed loop decision process.		
5	**Credit Based Traffic Management**	Part10, Sec 3.4.4	srio_ll_ds_credit_based_traffic_mgmt_test.sv
	Credit based traffic management permits the egress to control traffic on a PDU by PDU basis.		
	In this mode PDUs are only transmitted when an ingress is given an allocation of credits for a specific queue (identified by the operand <Q>)		
	Traffic flow stops when the allocation of credits reaches zero.		
	Egress to Ingress		
	<Q> XON <Q> XOFF <Q> ALLOCATE <AU> <Credits>		
	Ingress to Egress		
	<Q> CREDIT STATUS <AU> <Credits>		
	<Q> Q_STATUS <Level>		
	Allocate is used by an egress endpoint to tell an ingress endpoint that it has <some number of> credits available for use.		
	The credits are assigned to an allocation unit <AU>.		
	The allocation unit allows blocks of resources to be grouped, permitting coarser management, and requiring less precision in the synchronization between an ingress and the egress		
	The protocol allows or pipelining up to 16 allocation units		
	The egress may issue any number of allocation units (starting with 0) and rolling over at whatever limit it supports		

Item No	Compliance Item	Spec Reference	Testcase Name
	It is up to the egress to be sure an AU is unused before reusing that number		
	<Q> ALLOCATE <AU> <0> is used to retire an allocation.		
	It may be used by an egress endpoint to free a block of buffering for a number of reasons.		
	It can be used to pause a transmission by forcing the only credits to zero.		
	It can be used to clean up memory allocation by forcing an ingress onto the next allocation unit.		
	Credit Status: is sent by the ingress to delimit the use of allocation units, and to indicate status to trigger new allocations		
	<Q> CREDIT STATUS <AU> <nn>, where nn is the number of initial credits for an AU, is sent ahead of the first packet to delimit the beginning of the use of that AU.		
	<Q> CREDIT STATUS <AU> <00> is always sent after the last packet resulting in the number of credits going to 0 for that AU		
	The egress can close out that chunk of buffering, even if its notion of the number of PDUs received does not agree with the ingress (a PDU has been lost somewhere)		
	<Q> CREDIT STATUS <AU> <00> is also sent in response to an <Q> ALLOCATE <AU> <00> acknowledging that the ingress will send no more PDUs for that allocation, delimiting any PDUs in the pipeline.		
	<Q> CREDIT STATUS <AU> <xx>, where xx is some number of remaining credits, may be sent by an ingress to indicate a low level of credit allocation as a trigger to request more credits.		
	Queue Status: provides a means to indicate the overall status of a specific queue. The source can use this message to get attention for a queue that has become active, needing credits (transitioning from empty).		
	It can be used to indicate a queue that has gone empty, allowing the destination to deallocate the remaining credits and retire the AU.		
6	**Rules for Traffic Management**	Part10, Sec 3.4.5	
	valid combinations for supporting traffic management are: Basic Only Basic + Rate Basic + Credit Basic + Rate + Credit		
	All TM transactions use the VSID in the type 9 header, so by definition, all TM transactions occur in the same flows as the data. When wild cards are used, the don't care fields shall be set to values that put the message in the same flow as the affected data		
	Any rules used by the ingress to associate traffic with a specific VSID shall be used in reverse to associate the VSID in the message from the egress with the at least one queue.		

Item No	Compliance Item	Spec Reference	Testcase Name
	Endpoints may implement <Amount>, or <Level> with less precision than described		
	An ingress shall not overrule a TM directive from the egress. The ingress may discard traffic should the egress not adequately permit that traffic onto the fabric.		
	There are no requirements for timers. An ingress may “insist” by sending repeated Q_STATUS messages		
	Lost messages are recovered by sending duplicates		
	An XOFF to a <Q> already in the off state is ignored		
	An XON to a <Q> already in the on state is ignored		
	A lost Rate DECREASE message results in the egress sending a second DECREASE with a larger requested decrease amount.		
	A lost Rate INCREASE message results in the ingress issuing more urgent Q_STATUS messages and an the egress issuing additional INCREASE messages.		
	If the <Q> CREDIT STATUS <AU> <0> message is lost, the allocation unit will be assumed closed when the <Q> CREDIT STATUS <AU+1> <N> is received opening operation on the next allocation unit.		
	If the <Q> CREDIT STATUS <AU+1> <N> message is lost, the next allocation unit is assumed to be opened when the next PDU for that stream is received.		
	ALLOCATE messages are never duplicated. If an allocation unit message is lost, the egress may recover it with a <Q> ALLOCATE <AU><0> message, insuring it is de-allocated before reusing it.		

9.7 Packet format Compliance

Table 48: Packet Format Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	**Type 9 Packet Format (Data-Streaming Class)**		
	The type 9 packet format is the DATA STREAMING transaction format	Part10, Sec 4.2	
	Type 9 packets always have a data payload, unless terminating the PDU.		
	the data payload length is defined as a multiple of half-words rather than double-words		
	pad bit allows a sender to transmit an odd number of bytes in a packet.		
	An odd bit indicates that the data payload has an odd number of half-words.		
	An extended header bit allows future expansion of the functionality of the type 9 packet format.		
2	**Type 9 Extended Packet Format**	Part10, Sec 4.3	
	The type 9 extended packet format for traffic management between two data streaming endpoints is shown below.		
	The extended type 9 packet is identified first by the XH bit equal to 1.		
	The Class of Service field and the StreamID are included from the data packet format		
	The segmentation bits, Start and End are not used, or are the Odd and Pad fields.		

Item No	Compliance Item	Spec Reference	Testcase Name
	TM OP 0b0000 Basic Stream Management Message 0b0001 Rate Control Stream Management Message: 0b0010 Credit Control Stream Management Message 0b0011 Application Defined Stream Management Message		
	wildcard 0bnxx VSID dest wildcard If this bit is set, the message applies to all destinations 0bxnxVSID class wildcard:If this bit is set the message applies to all classes 0bxxn VSID stream wildcardI f this bit is set the message applies to all streams		
	mask - Class Mask Used to mask portions of the class of service (COS) field to allow groups of classes to be included in the message.		
	Parameter1: Argument specific to the TM message operation		
	Parameter2: Argument specific to the TM message operation		
3	**TM Operand**	Part10, Sec 4.3.1	
	The operand for the specific TM operation is defined by the following fields: <Source or Destination ID> <CoS> <StreamID> + <wild cards> + <Mask>		
	Specific Stream: <DestID><CoS><StreamID> + <wc=000> + <m = 0x00>		
	All Streams in a specific class: <DestID><CoS>< xx > + <wc = 001> + <m = 0x00>? where StreamID is a don't care		
	All Streams in a group of classes: <DestID><CoS>< xx > + <wc = 001> + <m = 0xnn>		
	All Streams and Classes (all traffic) to a specific destination <DestID>< xx >< xx > + <wc = 011> + <m = xx>? where CoS, StreamID and mask are don't cares		
	All traffic from this source < xx >< xx >< xx > + <wc = 111> + <m = xx>		
4	**Basic Traffic Management**	Part6, Sec 4.3.2	srio_ll_ds_on_off_traffic_mgmt_test.sv
	TM OP - 0b0000 Parameter 1- 0b0000 0000 Parameter 2 - 0b0000 0000 XOFF: Transmit off		
	Parameter 1- 0b0000 0000 Parameter 2 - 0b11111111 XON: Transmit on		
	Parameter 1- 0b0000 0011 Parameter 2 - 0b00000000 -0b11111111 Q_Status: Source queue is n/255 full (where n is parameter 2). 0 = empty, 0xFF = full		

Item No	Compliance Item	Spec Reference	Testcase Name
5	**Rate Based Traffic Management**	Part10, Sec 4.3.3	srio_ll_ds_rate_based_traffic_mgmt_test.sv
	TM OP - 0b0001 Parameter 1- 0b0000 0000 Parameter 2 - 0b0000 0000 XOFF: Transmit off		
	TM OP - 0b0001 Parameter 1- 0b0000 0000 Parameter 2 - 0b11111111 XON: Transmit On.		
	TM OP - 0b0001 Parameter 1- 0b0000 0y01 Parameter 2 - 0b0000 0000 Maintain_Rate: Maintain the current rate		
	TM OP - 0b0001 Parameter 1- 0b0000 0y01 Parameter 2 - 0b0000 0001 REDUCE: Reduce the current rate to = CurrentRate x (1-1/256)		
	TM OP - 0b0001 Parameter 1- 0b0000 0y01 Parameter 2 - 0b0000 0010 REDUCE: Reduce the current rate to = CurrentRate x (1-2/256)		
	TM OP - 0b0001 Parameter 1- 0b0000 0y01 Parameter 2 - 0b0000 0011- 0b11111111 REDUCE: Reduce the current rate to = CurrentRate x (1-n/256)		
	TM OP - 0b0001 Parameter 1- 0b0000 0y10 Parameter 2 - 0b0000 0001- 0b11111110 INCREASE: Increase the current rate to = CurrentRate x (1+ n/256)		
	TM OP - 0b0001 Parameter 1- 0b0000 0y10 Parameter 2 - 0b11111111 DOUBLE: Double the current rate		
	TM OP - 0b0001 Parameter 1- 0b0000 0011 Parameter 2 - 0b00000001-0b11111111 Q_Status: Source queue is n/255 full 0x00 = Empty, 0xFF = Full		

Item No	Compliance Item	Spec Reference	Testcase Name
6	**Credit Based Traffic Management**	Part10, Sec 4.3.4	srio_ll_ds_credit_based_traffic_mgmt_test.sv
	TM OP - 0b0010 Parameter 1 - 0b00000000 Parameter 2 - 0b00000000 XOFF: Transmit off		
	TM OP - 0b0010 Parameter 1 - 0b00000000 Parameter 2 - 0b11111111 XON: Transmit On.		
	TM OP - 0b0010 Parameter 1 - 0b0001nnnn Parameter 2 - 0b00000000 - 0b11111111 Allocate: nnnn - Allocation Unit Parameter 2 - number of credits		
	TM OP - 0b0010 Parameter 1 - 0b0010nnnn Parameter 2 - 0b00000000 - 0b11111111 Credit status: nnnn - Allocation Unit Parameter 2 - number of credits		
	TM OP - 0b0010 Parameter 1 - 0b00110000 Parameter 2 - 0b00000000 - 0b11111111 Queue Status: Source Queue is n/255 full (where n is parameter 2) 0 = Empty; 0xFF = Full		

9.8 Detectable Error Compliance

Table 49: Detectable Error Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	**Single Segment**		srio_ll_ds_sseg_err_test.sv
1a	If the packet is a single segment, the amount of payload data must be equal to or less than the MTU size or the PDU is defective.	Part10, Sec 3.2.5	
	Received Single segment data payload greater than MTU size		
1b	Type 9 packets always have a data payload, unless terminating the PDU	Part10, Sec 4.2	
	• Received Single segment message without data payload		
1c	Bit[14] (Received packet exceeds 276 Bytes) in Port n Error Detect CSR. Received packet that exceeds the maximum allowed size.	Part8, Sec 2.5.15	

Item No	Compliance Item	Spec Reference	Testcase Name
	Packet that exceeds the maximum packet size. Reception of packet (that exceeds the maximum packet size) causes the receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	• Received single segment data payload greater than 256 bytes		
1d	Bit[9] (Unsupported Transaction) Logical/Transport Layer Error Detect CSR-A transaction is received that is not supported in the Destination Operations	Part8, Sec 2.5.3	
	Change the Destination Operation CAR bit[13] to zero. • Received single segment message		
1e	Cyclic Redundancy Code used to detect transmission errors in the packet.	Part6, Sec 2.2	
	Packet with an incorrect CRC value The receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	Bit[13] (Received packet with bad CRC) in Port n Error Detect CSR	Part8, Sec 2.5.15	
	• Received single segment message packet with crc error		
1f	Bit[12] (Received packet with unexpected ackID) Port n Error Detect CSR	Part8, Sec 2.5.15	
	Packet with an unexpected ackID value Reception of packet (with an unexpected ackID value) causes the receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	ackID values on each link are assigned sequentially for each subsequent transmitted packet, an error in the ackID field is easily detected.	Part6, Sec 2.4	
	• Received single segment message packet with incorrect ackid		
1g	A packet carrying a response shall have a priority at least one priority level higher than the priority of the associated request.	Part6, Sec 6.11	
	• Received single segment message packet with priority value 3		
2	**Multi segment**		srio_ll_ds_mseg_err_test.sv
	If the packet is a start segment and the payload data does not match the MTU size the PDU is defective	Part 10, Sec 3.2.5	
	• Received Multi segment message packet with start segment size greater than MTU size		
2a	All continuation packets must match the MTU size or the PDU is defective.		
	• Received Multi segment message packet with continuation segment size greater than MTU size		
2b	End segment data payload size must be less than or equal to the MTU size or the PDU is defective		
	• Received Multi segment message packet with end segment size greater than MTU size		

Item No	Compliance Item	Spec Reference	Testcase Name
2c	Once all the data has been reassembled, the length (provided by the end segment packet header) is checked against the received data count. A mis-match indicates a lost continuation segment and the PDU is defective		
	• Received Multi segment message packet end segment with incorrect pdu length field		
2d	Receiving a continuation or end segment on a closed context indicates a lost start segment and the PDU is defective		
	• Received multi segment continuation segment without start segment		
	• Received multi segment end segment without start segment		
2e	Receiving a start or single segment on an open context indicates a lost end segment and the PDU is defective. The existing context is closed, and the new context is opened.		
	• Received multi segment start segment for already open context		
	• Received single segment packet for already open context		
2f	If the source wishes to abort a PDU transmission, it sends an end segment with no data payload and with the length field set to zero		
	• Received multi segment start segment with no data payload		
	• Received multi segment continuous segment with no data payload		
	• Received multi segment end segment with no data payload, but length field value as non zero		
2g	Bit[14] (Received packet exceeds 276 Bytes) in Port n Error Detect CSR. Received packet that exceeds the maximum allowed size.	Part8, Sec 2.5.15	
	Packet that exceeds the maximum packet size. Reception of packet (that exceeds the maximum packet size) causes the receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	• Received multi segment start segment data payload greater than 256 bytes		
	• Received multi segment continuous segment data payload greater than 256 bytes		
	• Received multi segment end segment data payload greater than 256 bytes		
2h	Bit[9] (Unsupported Transaction) Logical/Transport Layer Error Detect CSR- A transaction is received that is not supported in the Destination Operations	Part8, Sec 2.5.3	
	Change the Destination Operation CAR bit[13] to zero. • Received multi segment start segment. • Received multi segment continuous segment. • Received multi segment end segment.		
2i	Cyclic Redundancy Code used to detect transmission errors in the packet.	Part6, Sec 2.2	

Item No	Compliance Item	Spec Reference	Testcase Name
	Packet with an incorrect CRC value The receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	Bit[13] (Received packet with bad CRC) in Port n Error Detect CSR	Part8, Sec 2.5.15	
	• Received multi segment start segment with incorrect CRC value		
	• Received multi segment continuous segment with incorrect CRC value		
	• Received multi segment end segment with incorrect CRC value		
2j	Bit[12] (Received packet with unexpected ackID) Port n Error Detect CSR	Part8, Sec 2.5.15	
	Packet with an unexpected ackID value Reception of packet (with an unexpected ackID value) causes the receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Input Error-stopped recovery.	Part6, Sec 6.13.2.4	
	ackID values on each link are assigned sequentially for each subsequent transmitted packet, an error in the ackID field is easily detected.	Part6, Sec 2.4	
	• Received multi segment start segment with incorrect ack id		
	• Received multi segment continuous segment with incorrect ack id		
	• Received multi segment end segment with incorrect ack id		
2k	A packet carrying a response shall have a priority at least one priority level higher than the priority of the associated request.	Part6, Sec 6.11	
	• Received multi segment start segment with request priority value as 3		
	• Received multi segment continuous segment with request priority value as 3		
	• Received multi segment end segment with request priority value as 3		

9.9 Random Compliance

Table 50: Random Compliance

Item No	Compliance Item	Spec Reference	Testcase Name
1	Randomly initiate multi segment, single segment packets.		srio_ll_ds_pdu_random_test.sv
2	Randomly initiate multi segment packet up to 64 k pdu size		srio_ll_ds_sseg_max_random_test.sv
3	Randomly initiate single segment packet up to 64k pdu size		srio_ll_ds_mseg_max_random_test.sv
4	Randomly generate data streaming packets with IO packets		srio_ll_ds_io_random_test.sv
5	Randomly generate data streaming packets with Message packets		srio_ll_ds_msg_random_test.sv
6	Randomly generate data streaming packets with GSM packets		srio_ll_ds_gsm_random_test.sv
7	Randomly generate IO/MSG/GSM/DS transaction		srio_ll_io_msg_gsm_ds_random_test.sv
8	Randomly generate data streaming normal transaction followed by error transaction		srio_ll_ds_normal_error_test.sv
9	Randomly generate back to back transaction		srio_ll_ds_random_back_to_back_test.sv
10	Generate packet accepted control symbol in random interval for single segment/multi segment transaction		srio_ll_ds_delay_ack_random_test.sv