CSIX-L1: Common Switch Interface Specification-L1

Summary:

This specification defines CSIX-L1 a Common Switch Interface (CSIX) between a Traffic Manager and a switch fabric for ATM, IP, MPLS, Ethernet, and similar data communications applications.

TABLE OF CONTENTS

1 Introduction

1.1 CSIX-L1 Overview

CSIX-L1 is the Common Switch Interface. It defines a physical interface for transferring information between a traffic manager (Network Processor) and a switching fabric, as shown in figure 1.

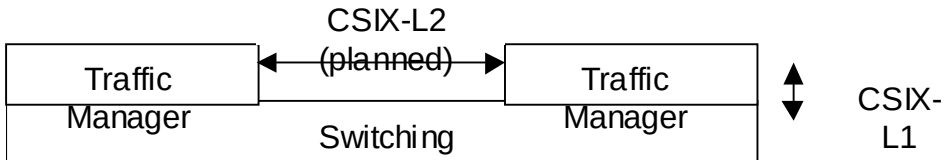

Figure 1--CSIX-L1 Logical Overview

A CFrame is the base information unit transferred between Traffic Managers and a CSIX Fabric. A CFrame consists of a header, payload, and a vertical parity trailer. These three major CFrame sections can be seen in figure 2: The CFrame Header contains the information fields needed to control the behavior of the Traffic Manger to CSIX Fabric interface. The CFrame Header can be further divided into the Base header and Extension header. The format and values of the CFrame Header is what is referred to as CSIX-L1 and a major focus of this specification. The Payload is variable in length and is passed by the CSIX Fabric from ingress Traffic Manager to egress Traffic Manager. The format of the payload is currently not defined by CSIX but is the focus of CSIX-L2. While the format of the payload is currently undefined by CSIX it's contents is where the Traffic Manager to Traffic Manager (TM-TM) Interface will reside. The TM-TM interface will also use a layered approach and contain sections such as a TM Header and TM Payload. The vertical parity trailer is used for error detection at the CSIX-L1 layer.

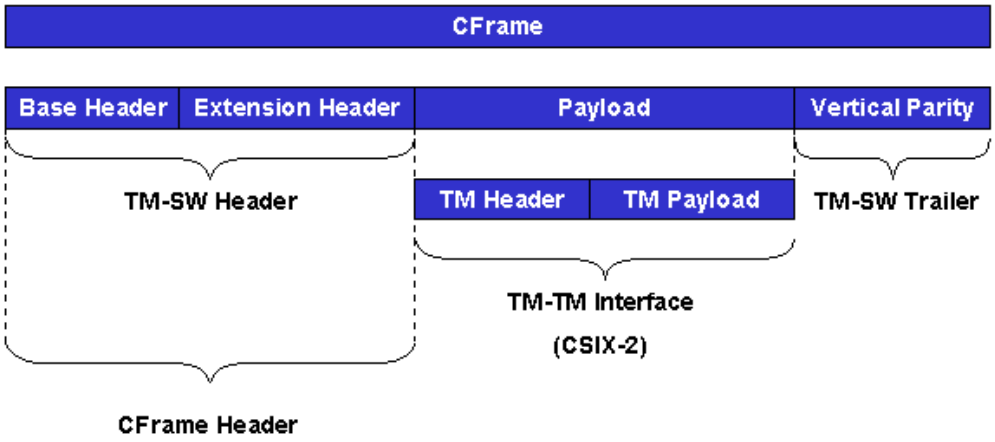

Figure 2-- CFrame Format

When the TM-TM Interface message size is less than or equal to the maximum CFrame Payload, a single CFrame can be used to transfer the information from ingress to egress TM.

When the TM-TM Interface message size is greater than the maximum CFrame Payload size, multiple CFrames must be transferred from ingress to egress TM in order. An ordered collection of CFrames is defined as a CFrame Sequence. An example of a CFrame Sequence can be seen in figure 3. A frame received at the ingress TM will be classified and forwarded to the egress TM across the CSIX Fabric. It is assumed that some amount of classification information will need to be associated with the ingress Frame and used by the egress TM. Examples of classification information could be a Next Hop IP address, or Egress connection identifier. The classification information associated with the ingress frame is defined here as the Frame Header for example purposes and beyond the scope of this specification. The Frame header and ingress frame is defined here as the TM-TM Frame and is assumed to be 4X larger than the maximum CFrame Payload size. In order to send the TM-TM Frame from ingress to egress TM, four CFrames must be used forming a CFrame Sequence. In each of the CFrames the TM-TM Interface will contain the necessary TM Header and TM payload information needed to reassemble the TM-TM Frame at the egress TM.

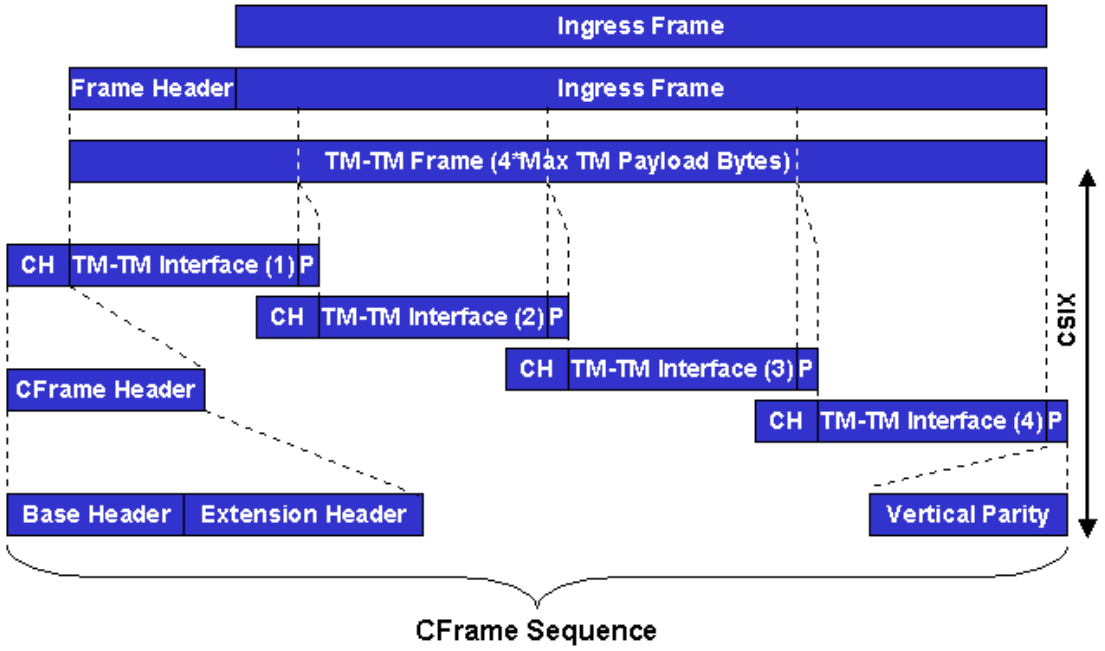

Figure 3-- CFrame Sequence Example

Traffic Manager-to-Traffic Manager information goes in the payload. The details of Traffic Manager-to-Traffic Manager information transfer will be addressed in CSIX-L2.

CSIX-L1 is intended to support board-level connectivity with trace length of up to 6-8 inches between the traffic manager and the switching fabric. It does not define a connector interface and it does not define operations of either the traffic manager or the fabric. Off-board connectors may be provided in the fabric.

CSIX-L1 is designed to support operations up to 32 Gb/s and is optimized to support OC48 and above.

1.2 Objectives and non-objectives for this specification

The objectives for this specification are:

- Provide a standard interface for connecting traffic managers to switching fabric.
- Support the communication of like Traffic Managers through the switch fabric.
- Support board level connections of 6-8 inches
- Support both cell-based and packet-based protocols

The following are “non-objectives” of this specification:

- Support a connector between traffic managers and fabric
- Provide interoperability between different traffic manager products

- Provide interoperability between different switch fabric products
- Provide end-to-end flow control

Nothing in this specification precludes TM-TM interoperability, however this specification does not address the TM-TM interoperability problem.

1.3 Possible implementations using CSIX

CSIX-L1 can be used to connect multiple traffic managers to a switching fabric to create the core of a switch design as shown in Figure 4.

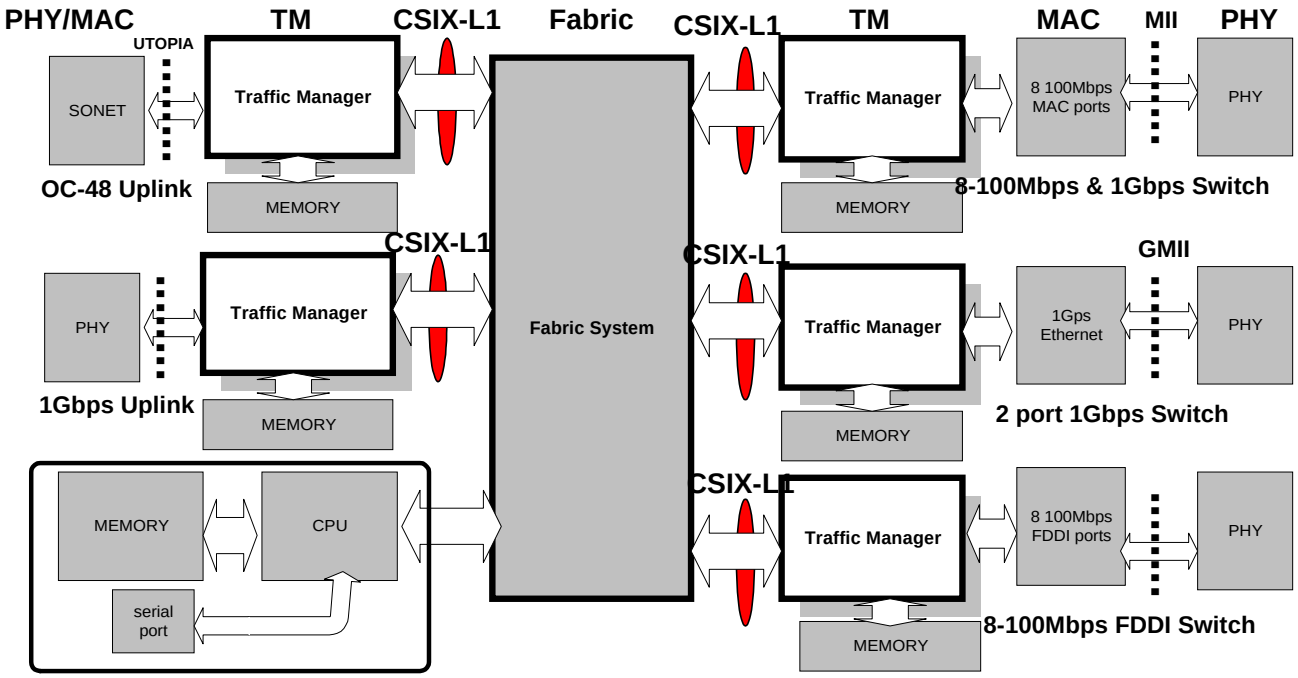

Figure 4--Prototype CSIX-based integrated fabric switch design

Some switching fabric vendors may offer modular fabric systems that support connectors within the fabric. When used with such a fabric system, CSIX-based modular designs can be created by combining one or more traffic managers on a line card with a portion of the switching fabric as shown in Figure 5.

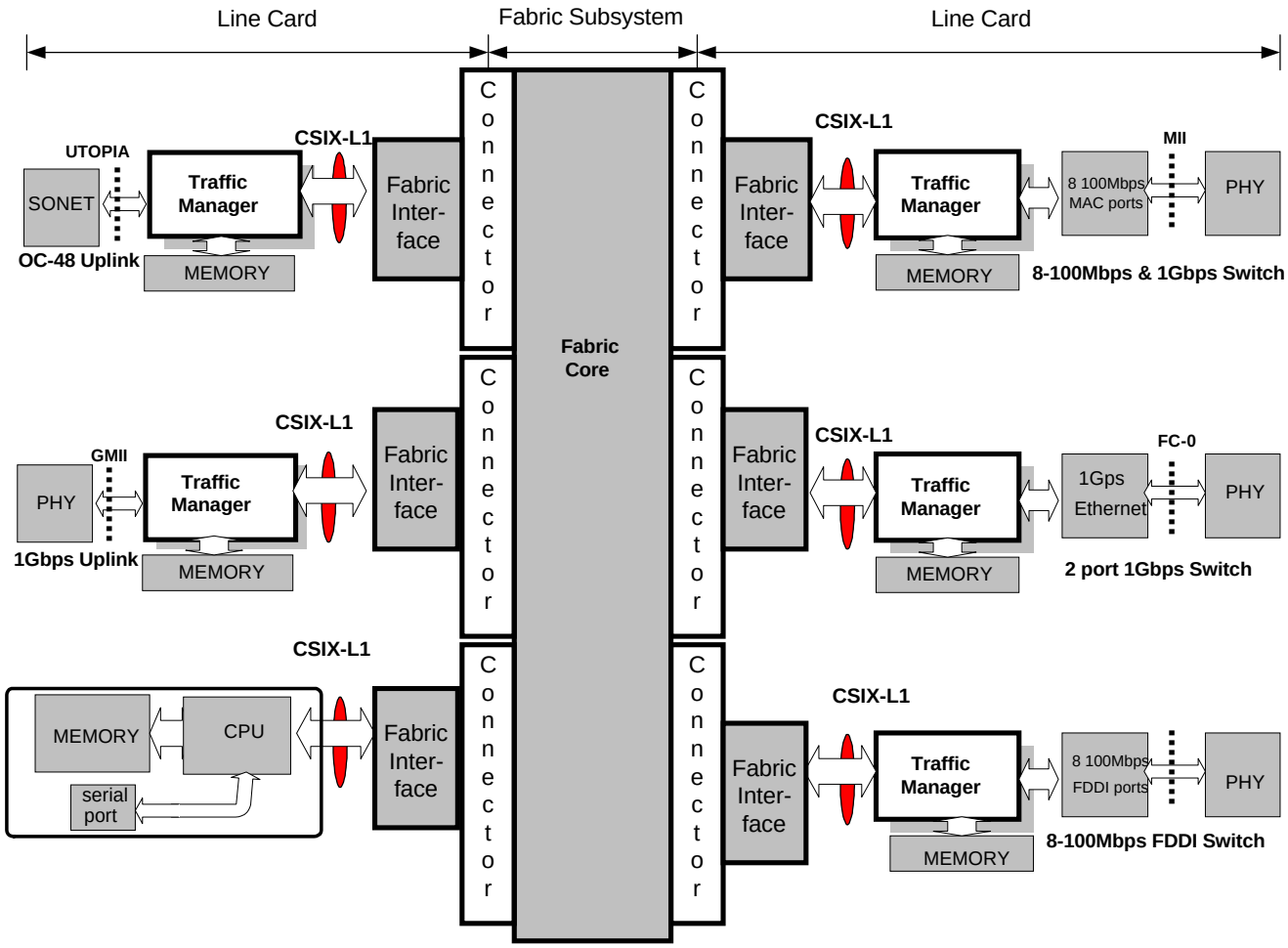

Figure 5--Prototype CSIX-based modular line card design with fabric connector

1.4 Conventions in this specification

The following conventions are followed in this specification:

1.4.1 Byte and bit ordering conventions

Byte ordering is big-endian; bit ordering follows Utopia conventions (e.g.—highest bit number of lowest byte number comes first.) This is shown in figure 6 below.

Byte0	byte1	byte2	byte3
7-0	7-0	7-0	7-0

31----------0

Figure 6-- CSIX-L1 Byte and bit ordering conventions

1.4.2 Interface conventions

The interface where data flows from traffic manager to switch fabric is labeled the receive interface and the interface where data flows from switch fabric to traffic manager is labeled the transmit interface.

1.4.3 State machine conventions

If there is a conflict between the text of this document and a state machine, the state machine takes precedence over the text.

Table 1 State machine operators

Character	Meaning
*	Boolean AND
+	Boolean OR
!	Boolean NOT

1.4.4 Mandatory features and PICs Pro Forma

This specification follows the IEEE802 style of using the verb “shall” to indicate mandatory features and provision of a PICs Pro Forma. All features which must be provided to conform to this specification are listed in Section 9: Conformance Requirements.

1.5 Definitions

The following terms are used throughout this specification:

CFrame: The information unit transferred between Traffic Managers and CSIX-L1 Fabric. A CFrame consists of a header of 2-6 bytes, Payload, and a 16-bit vertical parity field. The CFrame payload size is up to a maximum of 256 bytes, making the maximum CFrame size 264 bytes (up to 6 bytes of header, up to 256 bytes of payload and 2 bytes of vertical parity. The size of CFrame (including padding) shall be divisible by the size of a CWord.

CFrame sequence: A connected series of CFrames.

Class: An 8-bit variable used to discriminate and manage traffic flows between Traffic Managers through the CSIX-L1 interface. Class differentiates CFrames going to the same Fabric Port so that the Fabric can handle the CFrames differently. Class services are defined by the switching fabric and may be vendor-specific; traffic manager vendors should provide sufficient class operation flexibility to accommodate a variety of class implementations.

Typically, it is expected that vendors will not support all of the 256 possible class values offered by the 8-bit Class field. A vendor that implements a smaller number of classes shall encode their classes beginning with the most significant Class bit. For example, a fabric with 16 class values would take its 4-bits from Class [7:4].

Different CSIX-L1 addressing systems (e.g., unicast, multicast) make different use of the class variable. Unicast, binary copy multicast, and TM-based multicast all use the class variable to map the traffic flow to the class structure provided by the fabric vendor. Multicast ID and

multicast Mask addresses use the class variable to define multicast queue numbers that are used by multicast flow control operations.

CSIX Fabric: An intelligent switch fabric that schedules, buffers and switches data between its Inputs and Outputs. The Traffic Managers provide the CSIX fabric with information needed to perform scheduling and switching by means of a small CSIX header, which is prepended to the data payload.

CSIX Port: Access point to the fabric.

CSIX Reserved: With regards to fields and encodings, CSIX Reserved means that a field or an encoding is not being defined at this time, and is being reserved for future use and therefore shall not be used by CSIX compliant devices for proprietary applications. CSIX Reserved bits are always set to 0 on transmission, and ignored on reception. It is not implied that CSIX Reserved bits are carried through the CSIX fabric.

CWord: The data transferred in a single bus cycle. A CWord can be 32, 64, 96 or 128 bits.

Dead cycle: A one-clock-tick period during which there is no information transmitted across the CSIX-L1 interface.

Destination Address: A 12-bit Traffic Manager number representing up to 4k traffic-manager ports of a fabric.

Error Detecting Code: a 16-bit vertical parity field appearing in the last two bytes of the CFrame.

Ingress: Used to designate information/data flow to the fabric.

Egress: Used to designate Information/data flow from the fabric.

Padding: Zero (0) characters added between a CFrame payload and the vertical parity field to fill in required space when a vendor chooses to support less than the maximum allowed capacity of the field. Padding characters shall be ignored when processing the field but shall be included in vertical parity calculations (if vertical parity is implemented.)

Payload: The portion of CFrame that does not include any headers or Error Detection Code. The maximum payload size is vendor-specific, but shall not exceed 256 bytes.

Private: With regards to fields and encodings, Private means that a field or an encoding is not and shall not be defined by CSIX. CSIX compliant devices may use private fields for proprietary applications. It is not implied that CSIX Private bits are carried through the CSIX fabric.

Receive: Direction defined as from the TM to the Fabric.

Tick: One clock cycle.

Traffic Manager (TM): A logical entity performing collection of functions (e.g., fragmentation and reassembly, traffic shaping and security processing) implemented by a switch fabric user to manage the flow of data between a number of ports (line ends) and the switch fabric.

Traffic Manager (TM) Port: The physical or logical connections to a Traffic Manager other than the connection to the CSIX Fabric. Examples of physical TM Ports (line ends) would be OC-192, OC-48, Gigabit Ethernet, and T1/E1. ATM virtual circuits (VCs) would be an example of logical TM Ports.

Traffic Types: Traffic flows supported by CSIX including: Unicast, Multicast Mask, Multicast ID, Multicast Binary Copy, and Broadcast.

Transmit: Direction defined as from Fabric to TM.

1.6 Abbreviations

EDC: Error Detecting Code

Rx: Receive Direction

TM: Traffic Manager

Tx: Transmit Direction

1.7 Related Documents

EIA/JESD8-5: “2.5V±0.2V (Normal Range), and 1.8V to 2.7 V (Wide Range) Power Supply Voltage and Interface Standard for Nonterminated Digital Integrated Circuit.” EIA/JEFEC, October 1995.

EIA/JESD8-6: “High Speed Transceivers Logic (HSTL): A 1.5V Output Buffer Supply Voltage Based Interface Standard for Digital Integrated Circuits.” EIA/JEFEC, August 1995.

2 Overview

This document defines the system level requirements for the Common Switch Interface (CSIX). It specifies the interface between a traffic manager and switch fabric for ATM, Ethernet and similar data communications applications.

2.1 Architectural overview

Figure 7 illustrates the application of CSIX; additional detail is provided in Section 3.

A traffic manager is defined as a device that performs the functions defined as layer 2 or higher within the Open Systems Interconnect (OSI) 7-layer model. The switch fabric performs the physical layer transportation of user data elements from an ingress traffic manager to an egress traffic manager. A traffic manager may support any number of physical line ends, although this layer of functionality falls outside the scope of this specification.

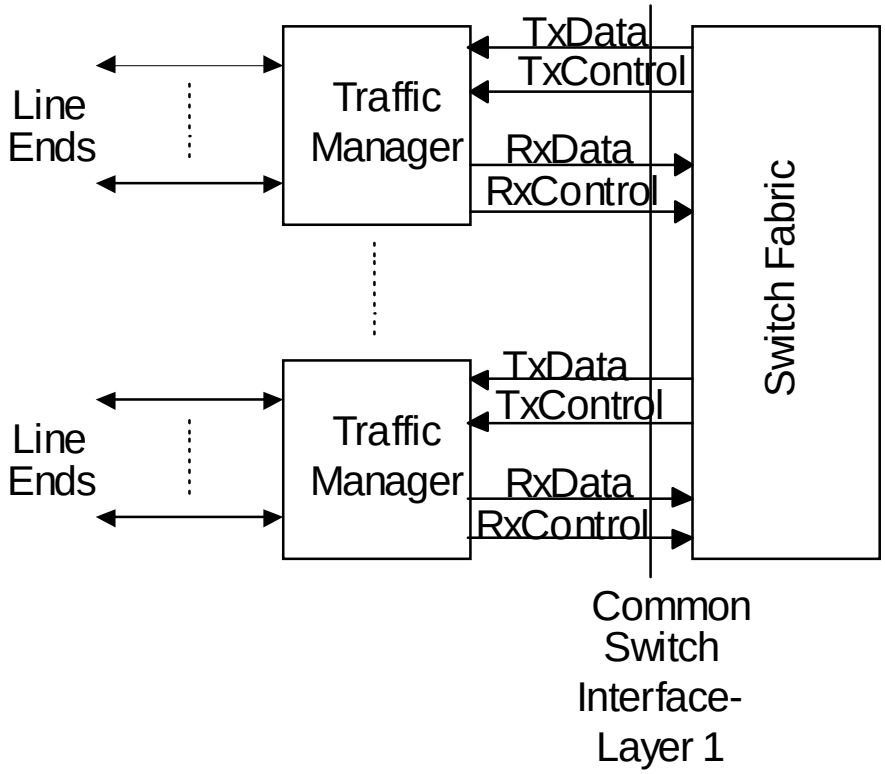

Figure 7-- The Common Switch Interface-Layer 1 (CSIX-L1)

2.2 Functional overview

CSIX-L1 supports up to 4096 CSIX ports for connecting traffic managers to the fabric. The number of ports is determined by the fabric and is vendor-specific. The CSIX 8-bit class variable provides a mechanism to manage traffic flows between a traffic manager and a fabric

port. Classes can be used to manage priority and QOS services provided by the fabric. The number of classes supported is also determined by the fabric and is vendor-specific.

Information is transported across CSIX-L1 via CFrames. CFrames can be used to transfer data and control messages across CSIX. The CSIX-L1 data streams can be managed via traffic manager-based traffic shaping or via fabric-based flow control. CSIX-L1 provides a flexible array of control frame mechanisms to help manage these traffic management functions.

2.3 Fabric Assumptions

CSIX-L1 defines a physical interface between traffic managers and switching fabric silicon. CSIX-L1 is intended to support point-to-point PC board-level connections of six to eight inches; it is not intended to support a physical connector. It is assumed that connectors (e.g., for line card connections) will be provided within the switching fabric architecture. While this standard does not preclude a wide range of fabric architectures, it has taken into account a trend towards cell based fabric architectures with buffering within the switching fabric.

2.3.1 Guarantee of in-order CFrame delivery

The CSIX fabric shall ensure that all CFrames of a specific class from a specific ingress TM are delivered to the egress TM in exactly the same sequence that they were presented to the fabric by the ingress TM. In addition, the fabric shall guarantee this in-order delivery even if the fabric internally has multiple routes between ingress TM and egress TM. There are no such guarantees with regard to CFrames of different classes presented to the fabric for delivery to the same egress TM.

2.4 Traffic Manager Assumptions

Like the fabric, there is a wide range of architectural choices that can be made in the design of a TM. Some architectures may employ extreme software programmability while others may be more hardwired. It is not the intent to limit the architectural choices made by TMs in this standard. However, it is generally assumed that where flexibility is needed in the CSIX-L1 standard to support varying Fabric architectures, it is the TM that will absorb the flexibility and allow maximum probability of interoperation between that TM and fabrics.

Each TM needs to support significant numbers of segmentation and reassembly contexts for this model. If a fabric has 64 ports each with 8 classes, the desired number of segmentation/reassembly contexts would be 512. This number can be reduced if each TM limits the number of simultaneous classes that can have outstanding packets to a given destination.

2.4.1 Line Ends Connected to the TM

CSIX-L1 can be incorporated into a wide range of potential product configurations. CSIX-L1 supports switch architectures with up to 4096 traffic managers. The definition of CSIX-L1 places no limits on the number of TM Ports that can be connected to a traffic manager.

3 Functional description

CSIX-L1 provides for the transfer of data and control information between the user's traffic manager and a switch fabric. Each function is described in more detail in section 3.1.

3.1 Transmit data and receive data

A CSIX Frame (CFrame) is the information unit transferred between Traffic Managers and CSIX Fabric. A CFrame consists of a base header and an optional extension header, a variable length payload, and a 16-bit Vertical Parity field.

CFrames are transmitted across the transmit and receive data paths which can be nx32-bit, where n= 1,2,3 or 4. At 100 MHz, a 32-bit wide data path can support 2488.32 Mb/s (OC-48); at 200 MHz a 64-bit wide transmit data path can support up to 9953.28Mb/s (OC-192.) The content of the payload of these frames falls outside the scope of this specification except when the frame is of type flow control, for which the contents are specified in this document.

The interface definition supports several different frame types. The use of different frame types provides flexibility to handle many different features.

The length of the CFrame transferred across CSIX-L1 may be different from the size of the data element handled by the switch fabric. Variable length payloads minimize the waste of CSIX-L1 bandwidth on those occasions when it is necessary to carry the small remainder of a segmented packet. For example, ignoring any header requirements, if a switch fabric transports only 64-byte data elements and a 65-byte data element arrives then this will be segmented into two units, one of 64 bytes and one of 1 byte. While this inefficiency must be absorbed by the switch fabric, it is not necessary for CSIX-L1 to carry this overhead. In this case the payload field of the CSIX frame can be defined to be 1 byte long.

Different fabric implementers may optimize their designs for different payload sizes. Each fabric will have at least one value supported for the parameter MAX_FRAME_PAYLOAD_SIZE that can be anywhere between 1 and 256 bytes. A fabric may support a range of values or multiple discrete points; if a fabric supports a range of MAX_FRAME_PAYLOAD_SIZE values, the fabric shall provide a mechanism for programming this value. The fabric shall support CFrame sizes of any size equal to or less than the maximum value specified for MAX_FRAME_PAYLOAD_SIZE.

Each TM should be able to support a range of values for MAX_FRAME_PAYLOAD_SIZE from 1 byte up to the maximum supported size. The TM shall provide a means of programming the exact value of MAX_FRAME_PAYLOAD_SIZE for a given system.

System integrators should use the smaller of the maximum values for MAX_FRAME_PAYLOAD_SIZE specified by the TM and fabric being integrated.

3.2 Unicast Operations

Unicast operations carry messages from the source TM to a designated destination TM. Unicast messages are subject to flow control operations. All fabrics and all TMs shall support unicast operations.

3.2.1 Unicast Destination Address

Each user payload that is destined for a single traffic manager shall be accompanied by a unicast Destination Address, which consists of a 12-bit Traffic Manager number and indicates the final destination of the payload. The switch fabric shall translate the Destination Address into a route through to the destination traffic manager. To accomplish this, each traffic manager must know how many traffic managers are connected to the switch fabric and must have a means of translating a network address to a physical address.

3.2.2 Unicast Class

The Class number is an 8-bit variable that specifies traffic isolation within the fabric and possibly different quality of service handling.

Class number should not be used to specify individual TM Ports on a Traffic Manager. Traffic Managers are expected to utilize the payload portion of CFrames to specify the TM Port to which traffic is directed, flow control for a specific TM Port, etc. If a vendor's fabric does not utilize all 8-bits of the Class field, the fabric vendor MAY choose to not pass the unused class bits from TM to TM.

3.3 Multicast Operations

Multicast operation carries messages from the source TM to a designated group of other TMs. Multicast messages are subject to flow control operations. Support of multicast messages is optional.

3.3.1 Multicast Destinations

Switch fabric support of multicast operation requires a mechanism for indicating multiple destinations. CSIX-L1 defines four options for providing multicast service across CSIX: bitmask based, multicast ID based, binary copy and "no-support" (where the multicast service is defined by the TM and passed across the interface as unicast messages.) These options are not exhaustive. Implementers are cautioned that for each multicast option there may be many fabric architectures and approaches that are not addressed in this document.

Multicast ID works by using a tag or ID to identify a multicast group. This requires that the fabric be configured. When the fabric receives a multicast ID frame, it uses the ID to determine which CSIX ports should receive copies and then delivers copies of the frame to each of these CSIX ports.

Note: There are several architectural choices regarding this mechanism that are not defined here (e.g. should the ID's be unique on a fabric port basis or are the ID's shared).

Multicast bitmask is intended for use with small (16-32 CSIX ports) systems It requires no configuration of the fabric with regards to multicast. In multicast bitmask mode, multicast frame has attached to it a bitmask representing every output. If a representative bit is '1' then the output receives a copy; else the output does not receive a copy. CSIX-L1 uses a modified version of bitmask that uses a 16-bit bitmask in combination with a partial address to provide multicast support for up to 4096 TMs. The partial address can be used to select a group of contiguous fabric ports that can receive copies of a frame. (In the worst case a 4k, port fabric would require 256 separate frames to be generated by the TM and sent into the fabric.)

Binary Copy is a simple extension of unicast addressing to a pair of outputs. Each binary copy multicast frame indicates two 12-bit destination ports. The fabric is responsible for copying the frame to both of the indicated ports. In the worst case (multicast to 4k ports), 2k frames would need to be generated by the TM. Binary copy multicast does not require any state setup in the fabric.

Unicast-only transport by the fabric offers a fourth form of multicast. In this case, the TM is completely responsible for all replication. It is the responsibility of the receiving traffic managers to provide copies on a per TM Port (line-end or VC) basis, using a secondary look-up mechanism. The data supporting this look-up mechanism are considered as 'user-to-user' information and form part of the payload; as such, these data are not visible at the CSIX-L1 level. Obviously, no state setup is required in the fabric for this form of multicast.

3.3.2 Multicast use of the class variable

CSIX-L1 offers several methods of multicast operation: binary copy multicast, TM-based multicast, multicast ID and multicast mask. Binary copy multicast, and TM-based multicast operation both use the class variable in the same way that unicast operation does: to map the traffic flow to a class structure provided by the fabric vendor. Multicast ID and multicast Mask operation use the class variable to define multicast queue numbers that are used by multicast flow control operations.

3.4 Broadcast Operations

Broadcast operation carries messages from the source TM to all other TMs. Broadcast messages are subject to flow control operations. Support of broadcast messages is optional.

3.5 Flow Control

CSIX-L1 provides multiple levels of flow (or congestion) control and supports flow control in both (TM-to-fabric and fabric-to-TM) directions.

At the lowest level there is link level flow control. Link level flow control is symmetric across the CSIX-L1 interface and provides independent control for data and control queues. For each queue there is a dedicated bit (the ready bit) in every CFrame base header indicating the congestion status for the receive queue of the respective traffic type.

At Ingress, TM to fabric flow control function is intended to resolve short periodic lack of buffering resources at the egress TM by temporarily taking advantage of buffering resources inside the fabric. When this causes the fabric buffering resources to become depleted, appropriate action should be taken by the systems to re-configure traffic distributions and resource allocations. At Egress, fabric-to-TM flow control can be used to resolve short-term lack of buffer resources at the ingress fabric port by taking advantage of buffer resources within the associated TM.

At the next level, the fabric exerts fabric flow control in response to congestion of fabric buffers. This flow control typically indicates a fabric port and class. There are semantics to "wild card" fabric port numbers and class numbers. For example, it might be useful to wildcard the class number in order to quickly shut off all traffic to a congested output for certain conditions. On the other hand, wildcarding of port number might be useful in a fabric where there is a mix of dedicated and shared buffering between the ports. (In this architecture it may occasionally be

necessary to throttle a line card of all traffic for low priority classes while the shared buffering drains.)

While most flow control mechanisms have traditionally used simple Xon/Xoff mechanisms, larger systems may require more complicated mechanisms to yield good fabric performance. CSIX-L1 flow control CFrames have a 4-bit speed variable which fabric vendors may use to provide finer flow control granularity. (See 5.9.1.1.6.)

3.5.1 Link-level Flow Control Model Assumptions

Link level flow control assumes a simple XON/XOFF model and uses the ready bits in the CFrame base header to perform flow control of data and control traffic.

3.5.2 Fabric Flow Control Model Assumptions

CSIX-L1 flow control from the fabric to the TM is based on an event-driven model whereby information is only passed across the CSIX interface when there is a change of status in a fabric queue. Each CSIX (Traffic Manager and fabric) component shall continuously maintain all flow control state information.

3.5.2.1 Flow Control state recovery

There is a (low probability) possibility that a CSIX component could lose flow control state information. Implementers may wish to periodically resend the state across the interface. Two possible approaches to flow control refresh are described below.

Each component cycles through all classes at very low frequency and transmits the flow control status for each class across the CSIX-L1 interface. In the case of flow control message loss, if the message was changing the speed variable, change is delayed until the flow control status is refreshed. This approach could cause some delay between the loss of the message and the flow control state refresh. However, since it is highly probable that only one port would lose a flow control message at any point in time, at the very worst the fabric will receive data for the congested output at the same rate that the fabric is draining. The period of the refresh would be fabric vendor defined.

Future CSIX work will define a Command and Status frame type that could be used to augment Flow Control state recovery operations.

3.5.3 Unicast Fabric Flow Control

Flow control CFrames can be used to manage transmissions for any specific TM/class combination. Broadcast data CFrames can be used to send global flow control messages.

3.5.4 Multicast Fabric Flow Control

While CSIX-L1 supports several forms of multicast, only two forms of multicast flow control are defined. One uses multicast mechanisms; the other uses unicast mechanisms.

The first type of multicast flow control is based on explicit multicast queues. The multicast ID and Multicast Mask mechanisms use explicit multicast queues. Multicast queue assignment shall be orthogonal to the Multicast ID or the Bitmask. The class field shall be used to indicate the multicast queue number for Multicast ID and Multicast Mask operations—both for multicast frames and for multicast flow control frames. Vendors of fabrics and Traffic Managers may

choose not to support 256 multicast queues. The second group of multicast mechanism includes the binary copy multicast and the completely TM-based multicast approaches. These mechanisms perform group flow control on a unicast basis, where multicast frames are queued with the unicast traffic.

3.5.5 Broadcast Fabric Flow Control

Broadcast messages can be flow controlled at the link level and by flow control messages.

3.6 Command and Status

Command and status frames will be defined in future CSIX work.

4 Physical Implementation

CSIX-L1 shall utilize a nx32-bit data path, where n= 1, 2, 3 or 4. The frequency of operation for CSIX-L1 is specified for up to 250MHz.

4.1 Interface signals

The following signals are used to transfer data, status indications and control information across the CSIX-L1 interface:

TXData

TxPar

TxClk

TxSOF

RxData

RxPar

RxClk

RxSOF

Transmit signals carry information from the switch fabric to the traffic manager; receive signals carry information from the traffic manager to the switch fabric. Each of these signals is defined below.

4.1.1 TxData[n..0]

The switch fabric shall present data to the traffic manager via the vector TxData. The size of the TxData vector is determined by the application and may be 32, 64, 96 or 128 bits. TxData[n] is the most significant bit, where n = 31, 63, 95, or 127.

4.1.2 TxPar[m..0]

The switch fabric shall present data parity to the traffic manager via the vector TxPar[m..0] (horizontal parity). The size of the TxPar vector is determined by the application and may be 1, 2, 3 or 4 bits, for m = 0, 1, 2 or 3, respectively.

4.1.2.1 TxPar definition

Each bit of the TxPar[m..0] vector is a horizontal odd parity across 32-bits of TxData as described below.

TxPar[j] = !(TxData[32*j+31] ^ TxData[32*j+30]^ ……. TxData[32*j])

Where:

j = 0 for TxData size = 32

j = 0..1 for TxData size = 64

j = 0..2 for TxData size = 96

j = 0..3 for TxData size = 128

4.1.3 TxClk[k..0]

The switch fabric shall provide data transfer/synchronization clock(s) to the traffic manager for synchronizing transfers on TxData. The number of TxClk signals required is determined by the application and may be 1, 2, 3 or 4, for k = 0, 1, 2 or 3, respectively. There is one TxClk signal required for a group of at-least 32 bits of TxData.

4.1.4 TxSOF[k..0]

The switch fabric asserts TxSOF to indicate start of CFrame. The number of TxSOF signals required is determined by the application and may be 1, 2, 3 or 4, for k = 0, 1, 2 or 3, respectively. One TxSOF signal is required for each TxCLK clock group. When more than one TxSOF signal is required, all TxSOF signals shall have identical values during each TxClk cycle.

TxSOF can take on one of two values of the form:

TRUE	TxSOF is asserted
FALSE	TxSOF is deasserted

4.1.4.1 When generated

TxSOF shall be asserted during the first transfer cycle of each CFrame. Sequential TxSOFs shall be separated by at least one tick.

4.1.5 RxData[n..0]

The traffic manager shall present data to the switch fabric via the vector RxData. The size of the RxData vector is determined by the applications and may be 32, 64, 96 or 128 bits. RxData[n] is the most significant bit, where n = 31, 63, 95 or 127.

4.1.6 RxPar[m..0]

The traffic manager shall present data parity to the switch fabric via the vector RxPar[m..0] (horizontal parity). The size of RxPar vector is determined by the application and may be 1, 2, 3 or 4 bits, for m = 0, 1, 2 or 3, respectively.

4.1.6.1 RxPar definition

Each bit of the RxPar[m..0] vector is a horizontal odd parity across 32-bits of RxData as described below.

RxPar[j] = !(RxData[32*j+31] ^ RxData[32*j+30]^ ……. RxData[32*j])

Where:

j = 0 for RxData size = 32

j = 0..1 for RxData size = 64

j = 0..2 for RxData size = 96

j = 0..3 for RxData size = 128

4.1.7 RxClk[k..0]

The traffic manager shall provide data transfer/synchronization clock(s) to the switch fabric for synchronizing transfers on RxData. The number of RxClk signals required is determined by the application and may be 1, 2, 3 or 4, for k = 0, 1, 2 or 3, respectively. There is one RxClk signal required for a group of at-least 32 bits of RxData.

4.1.8 RxSOF[k..0]

The traffic manager asserts RxSOF to indicate start of CFrame. The number of RxSOF signals required is determined by the application and may be 1, 2, 3 or 4, for k = 0, 1, 2 or 3, respectively. There is one RxSOF signal required for each RxCLK clock group. When multiple RxSOF signals are required, they shall have identical values during each RxClk cycle.

RxSOF can take on one of two values of the form:

TRUE RxSOF is asserted

FALSE RxSOF is deasserted

4.1.8.1 When generated

RxSOF shall be asserted during the first transfer cycle of each CFrame. Sequential RxSOFs shall be separated by at least one tick.

4.2 32-bit Interface

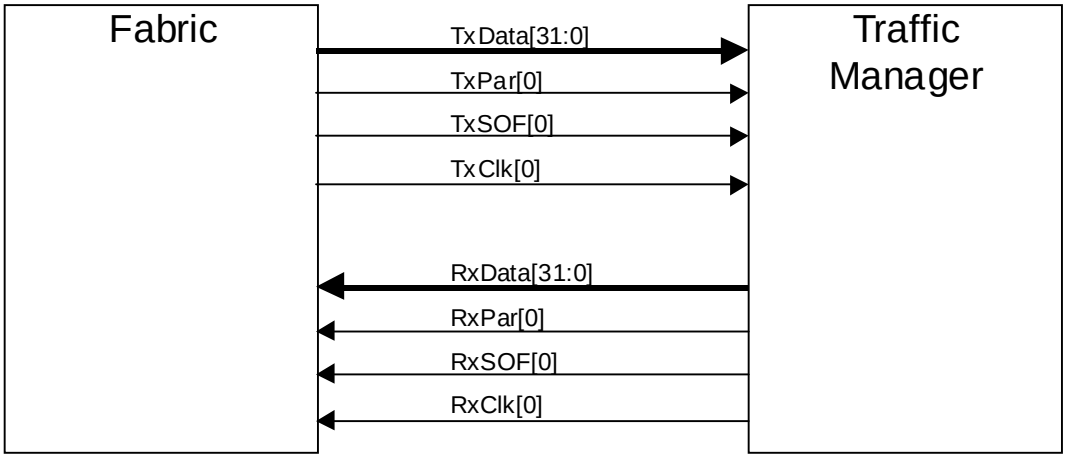

Figure 8-- The 32-bit interface

Table 2: 32-bit signals

Signal	Direction	Function
RxData[31:0]	TM to Fabric	Receive Data
RxPar[0]	TM to Fabric	Receive Data Odd Parity RxPar[0] -> RxData[31:0]
RxSOF[0]	TM to Fabric	Receive Start of Frame
RxClk[0]	TM to Fabric	Receive Clock
TxData[31:0]	Fabric to TM	Transmit Data
TxPar[0]	Fabric to TM	Transmit Data Odd Parity TxPar[0] -> TxData[31:0]
TxSOF[0]	Fabric to TM	Transmit Start of Frame
TxClk[0]	Fabric to TM	Transmit Clock

4.3 64-bit Interface

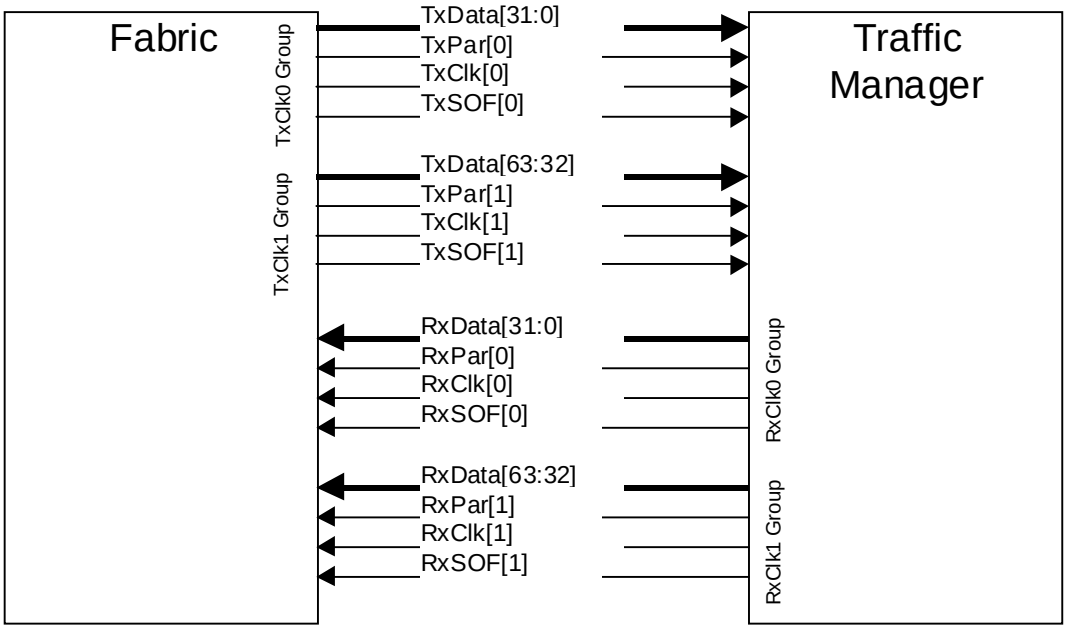

Figure 9-- The 64-bit Interface

Table 3: **64-bit signals**

Signal	**Direction**	**Function**
RxData[63:0]	TM to Fabric	Receive Data
RxPar[1:0]	TM to Fabric	Receive Data Odd Parity RxPar[0] -> RxData[31:0] RxPar[1] -> RxData[63:32]
RxSOF[1:0]	TM to Fabric	Receive Start of Frame
RxClk[1:0]	TM to Fabric	Receive Clock RxClk[0] Group: RxData[31:0], RxPar[0] and RxSOF[0] RxClk[1] Group: RxData[63:32], RxPar[1] and RxSOF[1]
TxData[63:0]	Fabric to TM	Transmit Data
TxPar[1:0]	Fabric to TM	Transmit Data Odd Parity TxPar[0] -> TxData[31:0] TxPar[1] -> TxData[63:32]
TxSOF[1:0]	Fabric to TM	Transmit Start of Frame
TxClk[1:0]	Fabric to TM	Transmit Clock TxClk[0] Group: TxData[31:0], TxPar[0] and TxSOF[0] TxClk[1] Group: TxData[63:32], TxPar[1] and TxSOF[1]

4.4 96-bit Interface

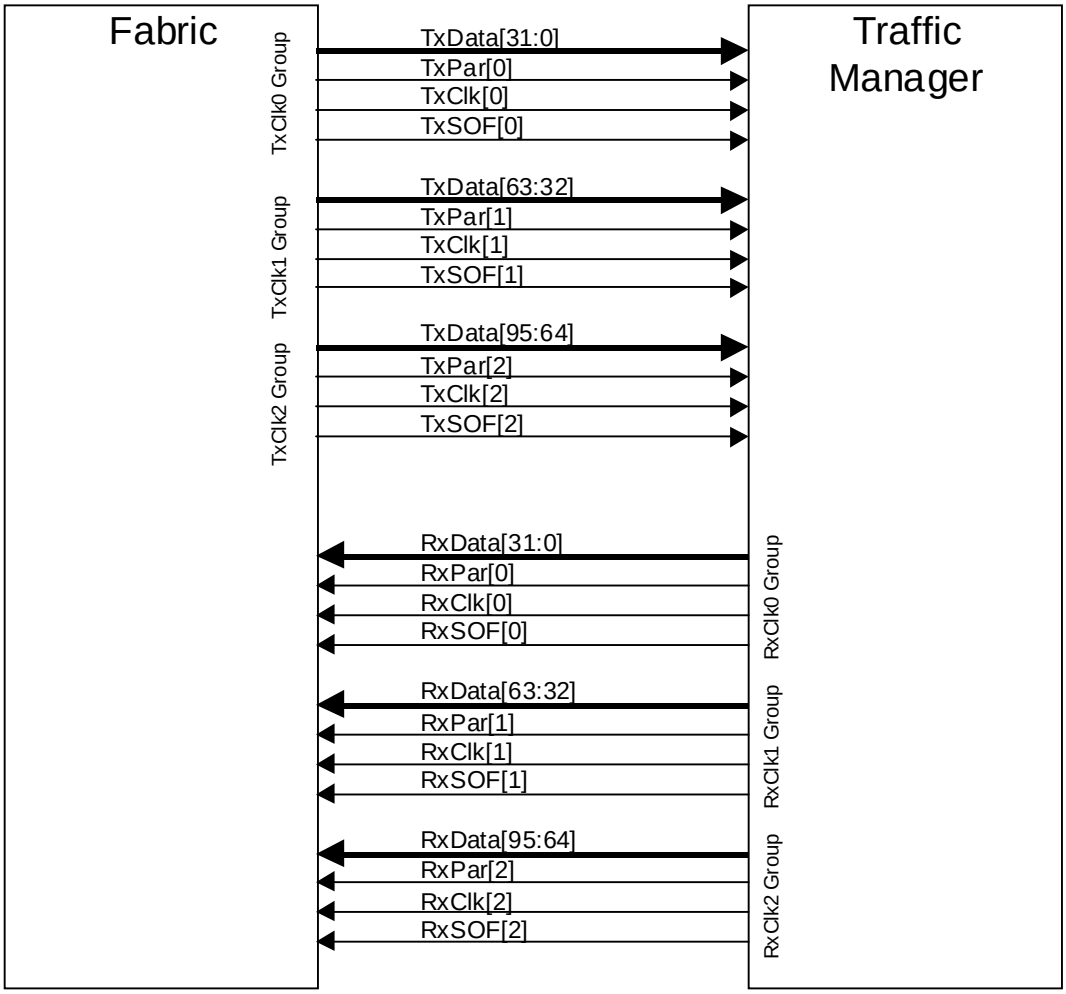

Figure 10-- The 96-bit Interface

Table 4: 96-bit signals

Signal	Direction	Function
RxData[95:0]	TM to Fabric	Receive Data
RxPar[2:0]	TM to Fabric	Receive Data Odd Parity RxPar[0] -> RxData[31:0] RxPar[1] -> RxData[63:32] RxPar[2] -> RxData[95:64]
RxSOF[2:0]	TM to Fabric	Receive Start of Frame
RxClk[2:0]	TM to Fabric	Receive Clock RxClk[0] Group: RxData[31:0], RxPar[0] and RxSOF[0] RxClk[1] Group: RxData[63:32], RxPar[1] and RxSOF[1] RxClk[2] Group: RxData[95:64], RxPar[2] and RxSOF[2]
TxData[95:0]	Fabric to TM	Transmit Data
TxPar[2:0]	Fabric to TM	Transmit Data Odd Parity TxPar[0] -> TxData[31:0] TxPar[1] -> TxData[63:32] TxPar[2] -> TxData[95:64]
TxSOF[2:0]	Fabric to TM	Transmit Start of Frame
TxClk[2:0]	Fabric to TM	Transmit Clock TxClk[0] Group: TxData[31:0], TxPar[0] and TxSOF[0] TxClk[1] Group: TxData[63:32], TxPar[1] and TxSOF[1] TxClk[2] Group: TxData[95:64], TxPar[2] and TxSOF[2]

4.5 128-bit Interface

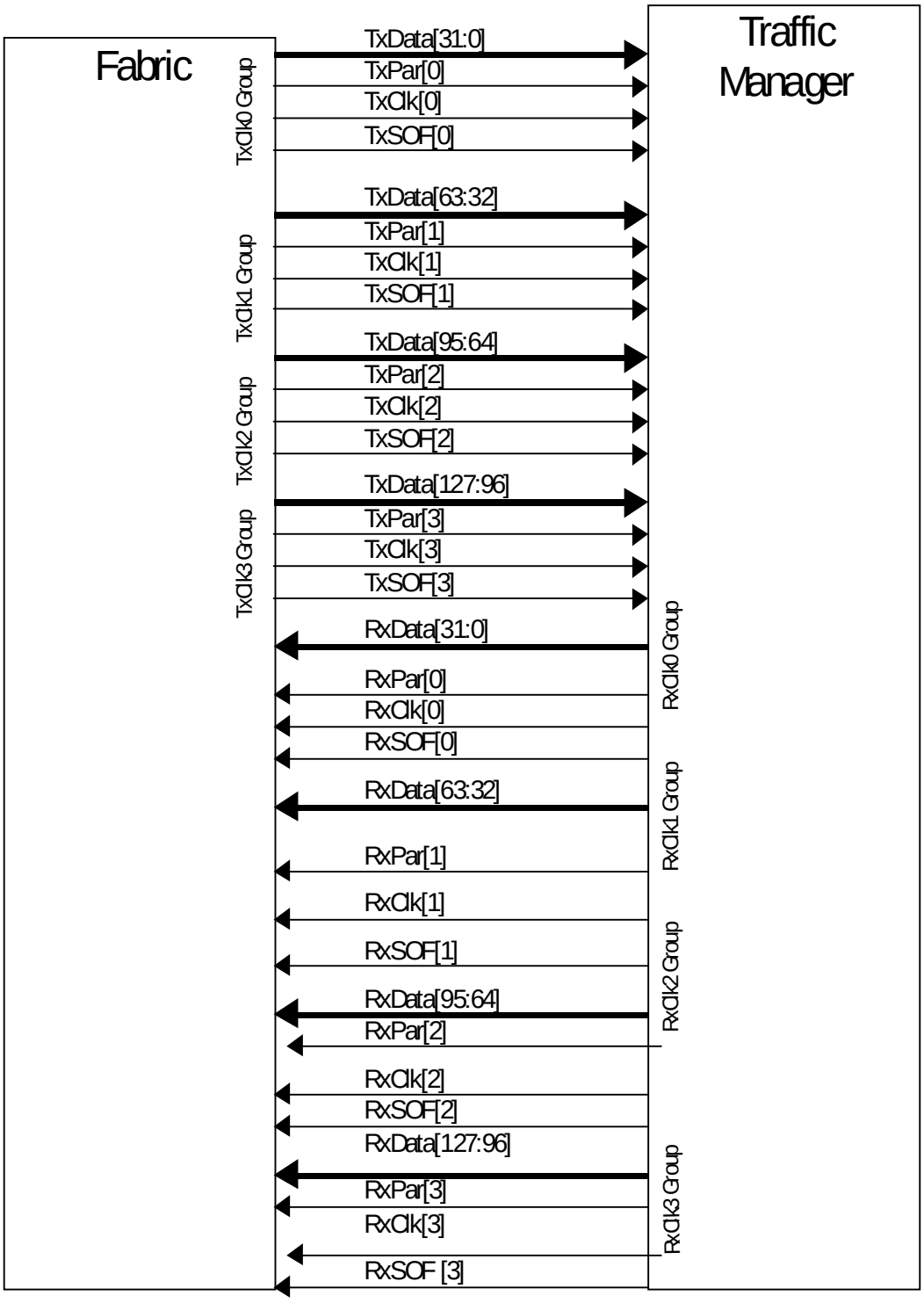

Figure 11-- The 128-bit interface

1 **Table 5: 128-bit signals**

Signal	Direction	Function
RxData[127:0]	TM to Fabric	Receive Data
RxPar[3:0]	TM to Fabric	Receive Data Odd Parity RxPar[0] -> RxData[31:0] RxPar[1] -> RxData[63:32] RxPar[2] -> RxData[95:64] RxPar[3] -> RxData[127:96]
RxSOF[3:0]	TM to Fabric	Receive Start of Frame
RxClk[3:0]	TM to Fabric	Receive Clock RxClk[0] Group: RxData[31:0], RxPar[0] and RxSOF[0] RxClk[1] Group: RxData[63:32], RxPar[1] and RxSOF[1] RxClk[2] Group: RxData[95:64], RxPar[2] and RxSOF[2] RxClk[3] Group: RxData[127:96], RxPar[3] and RxSOF[3]
TxData[127:0]	Fabric to TM	Transmit Data
TxPar[3:0]	Fabric to TM	Transmit Data Odd Parity TxPar[0] -> TxData[31:0] TxPar[1] -> TxData[63:32] TxPar[2] -> TxData[95:64] TxPar[3] -> TxData[127:96]
TxSOF[3:0]	Fabric to TM	Transmit Start of Frame
TxClk[3:0]	Fabric to TM	Transmit Clock TxClk[0] Group: TxData[31:0], TxPar[0] and TxSOF[0] TxClk[1] Group: TxData[63:32], TxPar[1] and TxSOF[1] TxClk[2] Group: TxData[95:64], TxPar[2] and TxSOF[2] TxClk[3] Group: TxData[127:96], TxPar[3] and TxSOF[3]

2

5 CFrame Formats

A CFrame consists of a base header, an optional (determined by type) extension header, an optional payload, optional padding bits and a 16-bit vertical parity field.

CFrame headers are based on a layered approach to minimize total overhead for varying types of frames. Every CFrame begins with the base header, which is 2 bytes in length and contains the payload length, frame type, and ready bits (for link level flow control). The base header is followed by type-specific extension headers. The extension header format is determined by the frame type in the base header. Extension headers contain additional information needed to handle the frame, such as the destination fabric port for unicast frames. The Header formats and field definitions are described in subsequent sections.

If needed, padding bits (zero (0) character bits that may need to be added to ensure that the CFrame is an appropriate length) are inserted between the payload and the vertical parity field. (See 1.5.) In the case of an Idle CFrame, which has no payload, padding bits are inserted between the base header and the vertical parity field.

A 2-byte vertical parity field follows the payload and any padding bits that are added. The parity bytes are always highest numbered bytes of the last word (recall that the most significant bytes will be the least significant bits of the last word). If the payload does not provide sufficient space for the vertical parity field in the last CWord, a new CWord shall be added and the vertical parity bytes will appear in the least significant bits of the added word.

Table 6: CFrame structure

CFRAME COMPONENT	Base Header	Extension Header	Payload	Vertical Parity
LENGTH	2 bytes	0-4 bytes	Maximum allowable length is 256 bytes.	2 bytes
COMMENTS		Number of bytes determined by CFrame and address type	Features of the traffic manager and fabric determine actual maximum length.	Field is required; use of field is optional.

5.1 Summary of frame overhead

Table 7 summarizes the number of overhead bytes for each frame type.

Table 7: Overhead by frame type

Type Encoding	Frame Type	Total CSIX-L1 Overhead (Base Hdr + Extension Header + EDC)
0	Idle	2+0+2=4 bytes
1	Unicast	2+4+2=8 bytes
2	Multicast Mask	2+4+2=8 bytes
3	Multicast ID	2+4+2=8 bytes
4	Multicast Binary Copy	2+4+2=8 bytes
5	Broadcast	2+4+2=6 bytes
6	Flow Control Frame	2+4*+2=8 bytes
7	Command and Status	TBD
8-f	CSIX Reserved	N/A

* The Flow Control CFrame does not have an extension header, the bytes between the Base Header and EDC contain flow control information.

5.2 Base Header

Table 8 shows the layout of the base header. Bits 0 and 1 of Byte 0 are CSIX Reserved (CR) and Private (P) bits respectively.

Table 8: Base header

Byte Number	Bit Position															
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
0/1	Ready		Type				CR	P	Payload Length							

5.2.1 Type Field

The type field (4 bits) encodes the type of CFrame being transferred. The payloads of all the frame types except flow control are unspecified by the CSIX-L1 specification. The flow control frame payload is fully defined in sub-section 5.8.2.1. The command and status frame payload is reserved and will be defined in future CSIX work. For the unicast, broadcast and multicast frames there is an extension header defined to follow the base header. The formats for these extension headers are defined in the next sub-sections.

Table 9: Type field values

Type	Encoding	Mandatory/Optional
Idle	0x0	M
Unicast	0x1	M
Multicast Mask	0x2	O
Multicast ID	0x3	O
Multicast Binary Copy	0x4	O
Broadcast	0x5	O
Flow Control Frame	0x6	M
Command and Status	0x7	Reserved
CSIX Reserved	0x8-0xb	
Private	0xc-0xf	

5.2.2 Ready Field

The ready field (2 bits) is used to indicate when the transmitting entity is ready to receive data. A low (0) ready bit indicates that the entity is not ready to receive the specified traffic type; a high (1) ready bit indicates that the entity is ready to receive the specified traffic type.

There are 2 ready bits representing 2 link level queues. The two groups and the CFrame types assigned to these groups are given below.

Ready[0] (bit 6 of byte 0) is for Control traffic

-- Command & Status CFrames(TBD in future CSIX work)

-- Flow control CFrames

Ready[1] (bit 7 of byte 0) is for Data Traffic

-- Unicast

-- Multicast

-- Broadcast

When no data is being transmitted, the ready field is kept alive by regular transmission of idle CFrames as specified in 6.2.

For purposes of link-level management, any vendor-specific frame types will be grouped with either the control or data group by the vendor. Idle frames are neither control or data frames and can be sent at any time.

Both ready bits are deasserted if an error (such as a parity error) is detected on a received CFrame.

Table 10 shows which ready bit controls the transmission of each specific CFrame type.

Table 10: Ready bit assignment by CFrame type

Type	**Ready Bit** CRdy=Ready[0] DRdy=Ready[1]
Idle	None
Unicast	Drdy
Multicast Mask	Drdy
Multicast ID	Drdy
Multicast Binary Copy	Drdy
Broadcast	Drdy
Flow Control	Crdy
Command and Status	Reserved
CSIX Reserved	
Private	

5.2.3 Payload Length Field

Payload length (8 bits) is the number of bytes in the payload of the message, excluding any padding (which is inserted between the payload and the vertical parity field.). Fabric vendors may specify the maximum payload length for their product. TMs can enforce their own maximum payload size for unicast and multicast CFrames by never injecting into the fabric anything over their desired maximum reception size.

For data CFrames, “0” indicates a payload of 256 bytes; for idle CFrames, the payload length field shall be set to “0.” (A “0” payload length for idle frames does not indicate a payload of 256 bytes.)

5.3 Idle CFrames

Idle frames are transmitted during startup and in periods of no activity to maintain both Ready bits and synchronization of data clocks.

5.3.1 Idle CFrame Format

An Idle CFrame is four bytes and consists of a Base Header and a 2-byte Vertical Parity field.

Table 11: Idle CFrame format

Byte Number	Contents
0-1	Base Header

2-3	Vertical Parity

5.4 Unicast CFrames

5.4.1 Unicast CFrame Format

Table 12: Unicast CFrame format

Byte Number	Contents
0-1	Base Header [15:0]
2-3	Unicast Extension Header [15:0]
4-5	Unicast Extension Header [15:0]
6-7	Payload
	-----
n	End of Payload (pad to interface width)
	Vertical Parity

5.4.2 Unicast Extension Header

The unicast extension header is shown in Table 13.

Table 13: Unicast extension header

Byte Number	Bit Position															
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
2/3	Class								P		CSIX Reserved					
4/5	CSIX Reserved								Destination Address							

5.4.2.1 Destination Address

The 12-bit destination address byte 4 (3:0) and byte 5 (7:0) provides addressing for up to 4096 Traffic Managers. The Destination Address field is valid in the ingress direction; once the fabric has delivered the CFrame to the appropriate egress TM(s), the destination address is no longer needed and is ‘field not defined.”

5.4.2.2 Class

Class is an 8-bit field and in unicast frames represents up to 256 isolated classes for every destination address. CFrames are delivered in the same order as received within a class, but not across classes. Not all 256 classes need to be supported by a fabric or TM. If less than 256

classes are supported, the existing class encodings follow the bit ordering conventions of 1.4. The Class field is valid in the ingress and egress directions.

It is not required that the fabric carry the full 8-bits of class from TM to TM if it is not using all 8-bits for its own traffic prioritization. If sub-class information is needed that the fabric does not use, that information should be embedded within the CFrame payload.

5.5 Multicast Mask CFrames

The properties of this method are that for small systems few frames will need to be generated by the TM, but as the system grows it is likely that larger number of frames will need to be generated to cover the possible multicast groups. The Destination Address field is not defined on the egress of the fabric.

5.5.1 Multicast Mask CFrame Format

Table 14: Multicast Mask CFrame format

Byte Number	Contents
0-1	Base Header [15:0]
2-3	Multicast Bitmask Ext. Header [15:0]
4-5	Multicast Bitmask Ext. Header [15:0]
6-78-9	Payload
	-----
n	End of Payload (pad to interface width)
	Vertical Parity

5.5.2 Multicast Bitmask Extension Header

The multicast bitmask header is shown in Table 15.

Table 15: Multicast bitmask extension header

Byte Number	Bit Position															
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
2/3	Class[7:0]								Bitmask header[7:0]							
4/5	Bitmap[15:0] (definable bitmap size)															

5.5.2.1 Bitmask header [7:0]

Multicast bitmask addresses use a combination of a 16-bit bitmap and an 8-bit mask header to represent all possible CSIX port addresses (see byte 3 in Table 15.) The bitmap can represent 16 TMs but not the 4096 possible TMs in a CSIX-L1 system. The multicast mask header contains the upper 8 bits of the destination address. In this way it is possible to use this

multicast frame type to reach all possible TMs in a CSIX-L1 system. The bitmask header field is undefined on the egress of the fabric.

5.5.2.2 Bitmap

The bitmap is simply 16-bits - ‘1’ indicates the representative output TM should receive a copy of the multicast frame. The upper 8 bitmask header bits choose which group of 16 TMs are addressed by the bitmap. The bitmap field is undefined on the egress of the fabric.

5.5.2.3 Class

The class field in the multicast bitmask header actually represents a multicast queue. A more in-depth discussion regarding multicast queues is in 3.3. The class field is valid in both the ingress and egress directions. It is not required that the fabric carry the full 8-bits of class from TM to TM if it is not using all 8-bits for its own traffic prioritization. If sub-class information is needed that the fabric does not use, that information should be embedded within the CFrame payload.

5.6 Multicast ID CFrames

5.6.1 Multicast ID CFrame format

Table 16: Multicast ID CFrame format

Byte Number	Contents
0-1	Base Header [15:0]
2-3	Multicast ID Extension Header [15:0]
4-5	Multicast ID Extension. Header [15:0]
6-7	Payload
	-----
n	End of Payload (pad to interface width)
	Vertical Parity

5.6.2 Multicast ID Extension Header

The multicast ID extension header is shown in Table 17. Bit 7 of byte 3 is Private; bit 6 of byte 3 is CSIX Reserved (CR).

Table 17: Multicast ID extension header

Byte Number	Bit Position															
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
2/3	Class								. P	C R	Multicast ID[21:16]					
4/5	Multicast ID[15:0]															

5.6.2.1 Multicast ID

The 22-bit Multicast ID communicates a lookup tag to the fabric. The fabric uses this lookup tag to determine which set of TMs should receive copies of the frame. The mechanism for co-ordinating the state set-up between the fabric and TM is outside the scope of this specification. Multicast ID is valid in both the egress and ingress directions.

5.6.2.2 Class

The multicast ID header class represents a multicast queue (see 3.3.1). The class field is valid in both the ingress and egress directions.

It is not required that the fabric carry the full 8-bits of class from TM to TM if it is not using all 8-bits for its own traffic prioritization. If sub-class information is needed that the fabric does not use, that information should be embedded within the CFrame payload.

5.7 Multicast Binary Copy CFrames

5.7.1 Multicast Binary Copy CFrame Format

Table 18: Multicast Binary Copy CFrame format

Byte Number	Contents
0-1	Base Header
2-3	Multicast Binary Copy Extension Header
4-5	Multicast Binary Copy Extension. Header
6-7	Payload
	-----
n	End of Payload (pad to interface width)
	Vertical Parity

5.7.2 Binary Copy Multicast Extension Header

The binary copy multicast extension header from the traffic manager to the fabric is shown in Table 19.

Table 19: Binary Copy Multicast Extension Header

(Traffic Manager to Fabric)

Byte Number	Bit Position															
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
2/3	Class								Left Destination Address [11:4]							
4/5	LDA[3:0]				Right Destination Address[11:0]											

5.7.2.1 Left and Right Destination Addresses

The binary copy form of multicast includes two TM addresses, each a complete unicast Destination Address. These two addresses are designated as left and right, such that an indicator bit can be flagged from the fabric to the output TM indicating which copy of the frame that TM is receiving. (This could be used so that if dual data structures were placed in the payload header of the frame, the indicator bit could indicate which data structure to extract at each of the 2 outputs so that output specific information could be passed.) The right and left destination addresses are valid in the ingress direction and undefined in the egress direction.

5.7.2.2 Class

Class is an 8-bit field and in unicast frames represents up to 256 isolated classes for every destination address. CFrames are delivered in the same order as received within a class, but not across classes. Not all 256 classes need to be supported by a fabric or TM. If less than 256 classes are supported, the existing class encodings follow the bit ordering conventions of 1.4. The Class field is valid in the ingress and egress directions.

It is not required that the fabric carry the full 8-bits of class from TM to TM if it is not using all 8-bits for its own traffic prioritization. If sub-class information is needed that the fabric does not use, that information should be embedded within the CFrame payload.

Unlike the other 2 forms of multicast discussed in preceding sub-sections, binary copy uses the class field like a unicast extension header. Both copies of the multicast are required to be of the same class. Flow control for binary copy multicast is performed in a unicast context only. Therefore, congestion of a unicast destination address and class will also affect all binary multicast frames with matching class, and one of the two destination addresses matching the congested destination address.

5.8 Broadcast CFrames

5.8.1 Broadcast CFrame Format

Table 20: Broadcast CFrame format

Byte Number	Contents
0-1	Base Header
2-5	Broadcast Extension Header
6-7	Payload
	-----
n	End of Payload (pad to interface width)
	Vertical Parity

5.8.2 Broadcast Extension Header

The class field in the broadcast extension header indicates one of up to 256 broadcast queues.

Table 21: Broadcast control extension header

Byte Number	Bit Position															
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
2/3	Class								Private				CSIX Reserved			
4/5	CSIX Reserved															

5.8.2.1 Class

Class is an 8-bit field. The class field in the broadcast extension header indicates one of up to 256 broadcast queues. Class is valid for both Ingress and Egress.

5.9 Flow Control CFrames

Flow Control CFrames provide a finer level of flow control than link level flow control by specifying the specific TM and Class that are oversubscribed. Flow control on a TM Port (line-end or VC) basis is the responsibility of cooperating TMs, and is outside the scope of this specification. Flow Control frames can go in both the ingress and egress directions.

5.9.1 Flow Control CFrame Format

A flow control frame consists of a base header with type set to flow control frame, a variable number of 4-byte flow control entries up to the maximum allowed payload size set by the combination of the fabric and TM vendors, any padding necessary, and the 2-byte vertical parity.

The flow control frame format is shown in Table 22.

Table 22: Flow Control CFrame format

Byte Number	Contents
0,1	Base Header
2,3	Flow Control Entry 1a
4,5	Flow Control Entry 1b
6,7	Flow Control Entry 2a
8,9	Flow Control Entry 2b

	Last Flow Control Entry in frame, bytes 1&2
	Last Flow Control Entry in frame, bytes 3&4
n	(pad to interface width)
	Vertical Parity

5.9.1.1 Flow Control Entry Format

The Flow Control entry format is shown below.

Table 23: flow control entry format

Flow Control Entry Byte Number	Bit Position															
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
2/3	Class								FC Entry Type		C *	P *	Speed			
4/5	P		CSIX		Destination Address											

Bit 5 of Byte 3 is the class wildcard; bit 4 of Byte 2 is the port wildcard.

Each of the fields is discussed in the following sub-sections.

5.9.1.1.1 Class

For unicast the class field is the same 8-bit class field specified in unicast extension headers.
For Multicast the class field specifies one of the 256 multicast queues

5.9.1.1.2 Entry Type

The entry type (see Table 24 below) indicates what resource is being reported. The entry type determines what the meaning is and which of the other fields are processed.

Note that multicast flow control entries will only be generated when the multicast mechanism is multicast mask or multicast ID.

Table 24: Flow control entry type values

Encoding	Name
00	Unicast
01	Multicast
10	Broadcast
11	All

5.9.1.1.3 Class Wildcard

The class wildcard is used to extend the scope of the flow control message. The impact of class wildcard is determined by the message type.

For unicast messages, setting the class wildcard bit to '1' causes the associated flow control message to be applied to all classes associated with the indicated destination address

For multicast messages, the class value represents one of up to 256 multicast queues that can be used to transfer payloads to multiple ports. Setting the class wildcard bit to "1" means that the associated flow control message is applied to all extant multicast queues.

5.9.1.1.4 Port Wildcard

The port wild card is used to extend the impact of Fabric to TM flow control messages for unicast traffic. When the fabric sets the port wild card to "1" for unicast traffic, the flow control message is applied to all destinations. The port wild card is not used for Traffic Manager to Fabric flow control frames.

5.9.1.1.5 Destination Address

The destination address for unicast queues has the same field definition as in the unicast extension header. For multicast and broadcast, the destination address field is ignored.

5.9.1.1.6 Speed

The speed field specifies how rapidly the TM is allowed to transmit CFrames to the specified destination. Although the Speed field is defined to be 4 bits, specific fabric implementations may choose to ignore the low order 1, 2, or 3 bits. The 16 possible values range from 1111 (Send full speed) to 0000 (Stop – send no CFrames to this queue), with smaller numbers indicating proportionally more flow control – fewer CFrames per second allowed to enter the queue.

The speed field (at least the most significant bit) shall be generated and processed for all entry types. All implementations shall be capable of distinguishing and responding to speed field values “1111” and “0000”

5.9.1.1.7 TM to fabric flow control ignore option

During periods where the TM asserts flow control to the fabric, the fabric may experience a lack of available buffering resources to support the flow control request. In this situation, the fabric may decide to discard traffic internally. Alternatively, the fabric may decide to time out the asserted TM flow control request by increasing transmission rate sufficiently (i.e., ignore TM-to-fabric flow control) to avoid activation of any such fabric discard mechanism.

Note that the fabric is not allowed to perform an equivalent time out function for the link level flow control function.

5.9.2 TM to Fabric flow control summary

The following table summarizes the functions that can be performed with the TM to Fabric flow control frames.

Table 25: TM to Fabric flow control summary

Flow Control Function	Flow Control CFrame Fields generated by TM			
	CLASS	C*	P*	DA
Traffic TYPE = Unicast				
Adjust speed for unicast traffic for all classes	N/A	1	N/A	N/A
Adjust speed for unicast traffic for a specific class only	Class data	0	N/A	N/A
Traffic TYPE = Multicast (does not affect broadcast traffic)				
Adjust speed for multicast traffic for all classes	N/A	1	N/A	N/A
Adjust speed for multicast traffic for a specific class only	Class data	0	N/A	N/A
Traffic TYPE = Broadcast (does not affect multicast traffic)				
Adjust speed for broadcast traffic for all classes	N/A	1	N/A	N/A
Adjust speed for broadcast traffic for a specific class only	Class data	0	N/A	N/A
Traffic TYPE = All (Unicast/Multicast/Broadcast)				
Adjust speed for all traffic types and all classes	N/A	1	N/A	N/A
Adjust speed for all traffic types for a specific class only	Class data	0	N/A	N/A

1) The SPEED field is valid for all of the above function
2) All fields being N/A should be set to all “0” by the TM

Any flow control function performed with a flow control entry carrying one or more enabled wildcard fields (class wildcard C*, port wildcard P* or traffic type TYPE = All) can always alternatively be performed using multiple flow control entries where none of the wildcard fields are enabled. This means that regardless whether a specific flow control function is performed with a single wildcarded flow control entry or multiple non-wildcarded flow control entries, the resulting flow control function is 100% identical for the two cases. The advantage of applying the wildcard fields whenever possible, is that the bandwidth used for flow control CFrames on the physical CSIX interface is minimized

5.9.3 Fabric to TM flow control summary

The following table summarizes the functions that can be performed with the Fabric to TM flow control frames.

Table 26: Fabric to TM flow control summary

Flow Control Function	Flow Control CFrame Fields generated by Fabric			
	CLASS	C*	P*	DA
Traffic TYPE = Unicast				
Adjust speed for unicast traffic for a specific destination (for all classes)	N/A	1	0	DA data
Adjust speed for unicast traffic for a specific destination and specific class only	Class data	0	0	DA data
Adjust speed for all unicast traffic for all destinations and all classes	N/A	1	1	N/A
Adjust speed for all unicast traffic for all destinations and specific class only	Class data	0	1	N/A
Traffic TYPE = Multicast (does not affect broadcast traffic)				
Adjust speed for all multicast traffic for all classes	N/A	1	N/A	N/A
Adjust speed for all multicast traffic for a specific class only	Class data	0	N/A	N/A
Traffic TYPE = Broadcast (does not affect multicast traffic)				
Adjust speed for all broadcast traffic for all classes	N/A	1	N/A	N/A
Adjust speed for all broadcast traffic for a specific class only	Class data	0	N/A	N/A
Traffic TYPE = All (Unicast/Multicast/Broadcast)				
Adjust speed for all traffic types and all classes	N/A	1	N/A	N/A
Adjust speed for all traffic types for a specific class only	Class data	0	N/A	N/A

1) The SPEED field is valid for all of the above function
2) All fields being N/A should be set to all “0” by the Fabric

Any flow control function performed with a flow control entry carrying one or more enabled wildcard fields (class wildcard C*, port wildcard P* or traffic type TYPE = All) can alternatively be performed using multiple flow control entries where none of the wildcard fields are enabled. This means that regardless whether a specific flow control function is performed with a single wildcarded flow control entry or multiple non-wildcarded flow control entries, the effective flow control mechanism is 100% identical for the two cases. The advantage of applying the wildcard fields whenever possible, is that the bandwidth used for flow control CFrames on the physical CSIX interface is minimized.

5.10 Command and Status CFrames

Command and Status CFrames are not defined in this (CSIX-L!) specification. Command and Status CFrames are reserved and will be defined in future CSIX work.

5.11 Parity

CSIX-L1 provides both horizontal and vertical parity to validate information transferred across the interface. Horizontal parity is mandatory. Support of the vertical parity field is mandatory, but use of vertical parity is optional.

5.11.1 Horizontal Parity

Horizontal parity is mandatory and is described in 4.1.2 and 4.1.6.

5.11.2 Vertical Parity

Vertical parity provides an optional second set of parity bits that provide a substantial improvement (over just horizontal parity) in the probability of error detection. To calculate vertical parity bits, the CFrame is treated as a series of 16-bit words, organized in a two-dimensional block as shown in Table 27: Vertical parity. A Vertical parity bit is generated for each of the 16 bit positions (columns) in the block across all rows required to contain the CFrame payload. The resulting 16-bit Error Detecting Code (VPar) is appended to the end of the payload.

Table 27: Vertical parity

Word 0	B(15,0)	b(14,0)		B(0,0)
Word 1	B(15,1)	b(14,1)		B(0,1)
Word m	B(15,m)	b(14,m)		B(0,m)
VPar	***VPar15***	***VPar14***		***VPar0***

VPar[i] = !(b(i,0) ^ b(i,1) ^ ….. ^ b(i,m));

Where,

Vpar[i] = *i*th bit of Vertical Parity word.

b(i,m) = *i*th bit of data word m

m+1 = number of 16-bit words (Excluding Vpar word) in CFrame

6 Operation and timing

CSIX-L1 operation and timing is defined in the same way for both the receive path and the transmit path. The width of the receive and transmit paths shall be the same, however other parameters (such as the minimum frame time) may be different in the two directions.

6.1 Start-up

CSIX-L1 uses a simple, robust startup mechanism based on use of the ready bits (see 5.2.2) and the idle CFrame as defined in 6.4.1.

At Power-Up or RESET, each CSIX port entity (traffic manager and fabric) begins transmitting idle CFrames while holding both ready bits low. When an entity detects receipt of idle CFrames, it raises both ready bits to high and continues to send idle CFrames. When an entity detects receipt of idle CFrames and high ready bits from its connected entity, it assumes regular operation. This is depicted in the state machine diagram shown in figure 12.

When in normal operation, if RESET occurs or if data clock synchronization fails (i.e. Receiver_Synchronization=FALSE), then the system resets and restarts.

6.2 Transmission

At any point in time the CSIX-L1 interface may contain a data, control or idle CFrame or a Dead Cycle. During times when no data is being transmitted, a pattern of alternating idle CFrames and Dead Cycles is transmitted to maintain synchronization and keep the ready bits alive. The state machine to govern CFrame and Dead Cycle transmission is shown in figure 13. Transition variables for the Transmission state machine are shown in 6.3.1.

6.3 State Machine Variables:

6.3.1 State Machine Variables

Transmitting_Frame (TF)

Indicates that a CFRAME is being transmitted

VALUES: TRUE: Transmission of the current CFRAME is not complete.

FALSE: A CFRAME is not currently being transmitted

New_Frame_Ready (NFR)

Indicates that either a new data CFrame is queued and ready to transmit and that ready bits [1] of the receiving entity is high or that a new control CFrame is queued and ready to transmit and that ready bit [0] of the receiving entity is high

VALUES: TRUE: New Frame Ready conditions are satisfied

FALSE: New Frame Ready conditions are not satisfied

Ready_Bit_High

Indicates the value of the ready bits in CFrames received from the entity on the other side of the CSIX-L1 interface.

Values:	TRUE:	One or both of the two received ready bit is high
	FALSE:	Both received ready bits are low or not set.

Receive_Idle_Cell

Indicates that the entity is receiving idle cells across the CSIX-L1 interface.

Values:	TRUE:	Entity is receiving idle cells
	FALSE:	Entity is not receiving idle cells

Receiver_Synchronization

The receiver of the CSIX-L1 interface (when data width > 32) must perform data alignment across all clock groups.

Values:	TRUE:	receiver is synchronized across all clock groups
	FALSE:	receiver is not synchronized

RESET:

Resets all functions. Reset can be performed hot (when the power is already on) or cold (a power-on reset.)

Values: ON or OFF

6.4 State Machines

6.4.1 Startup State Machine

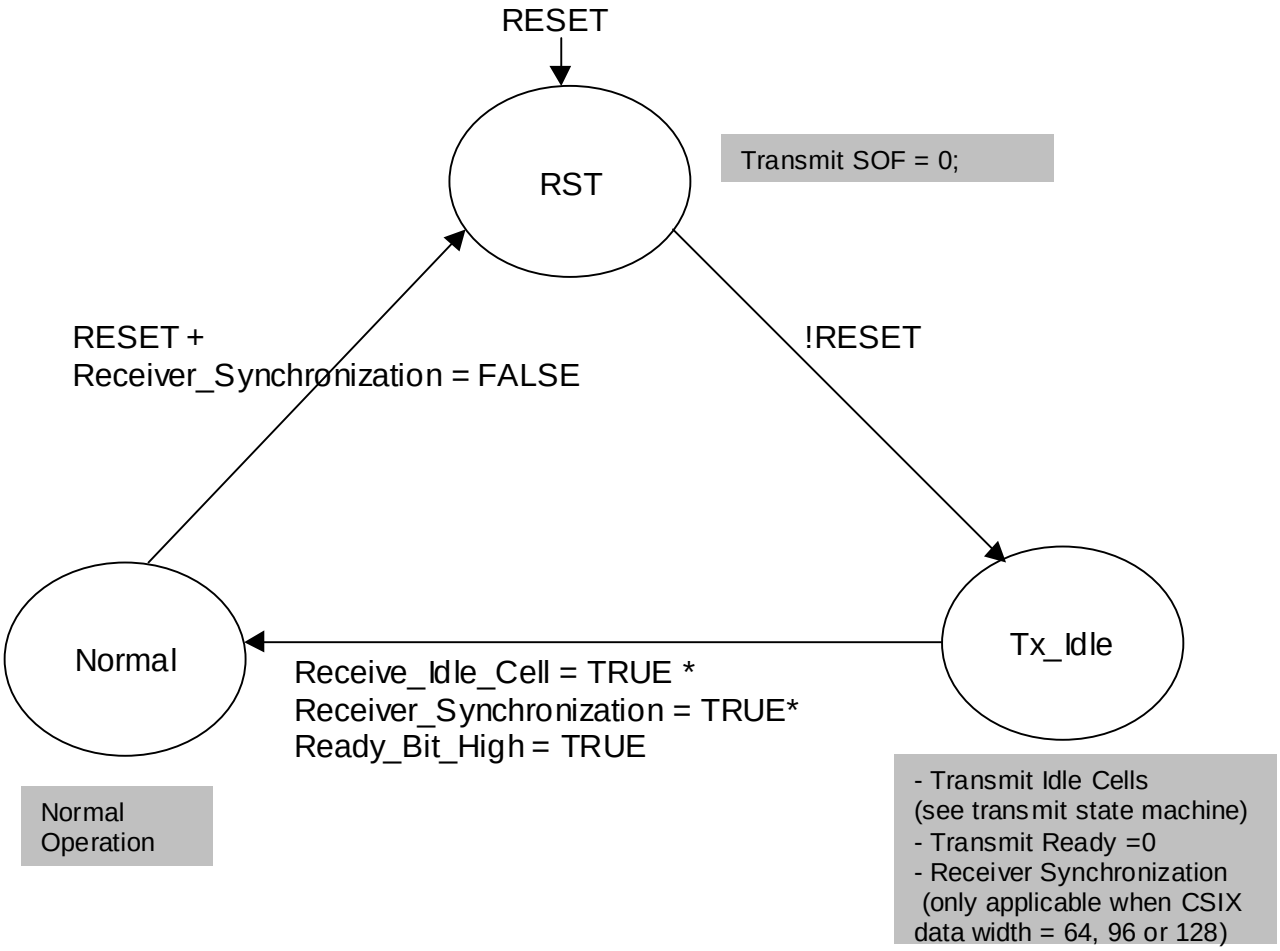

Figure 12--Start-up state machine diagram

6.4.2 Transmission State Machine

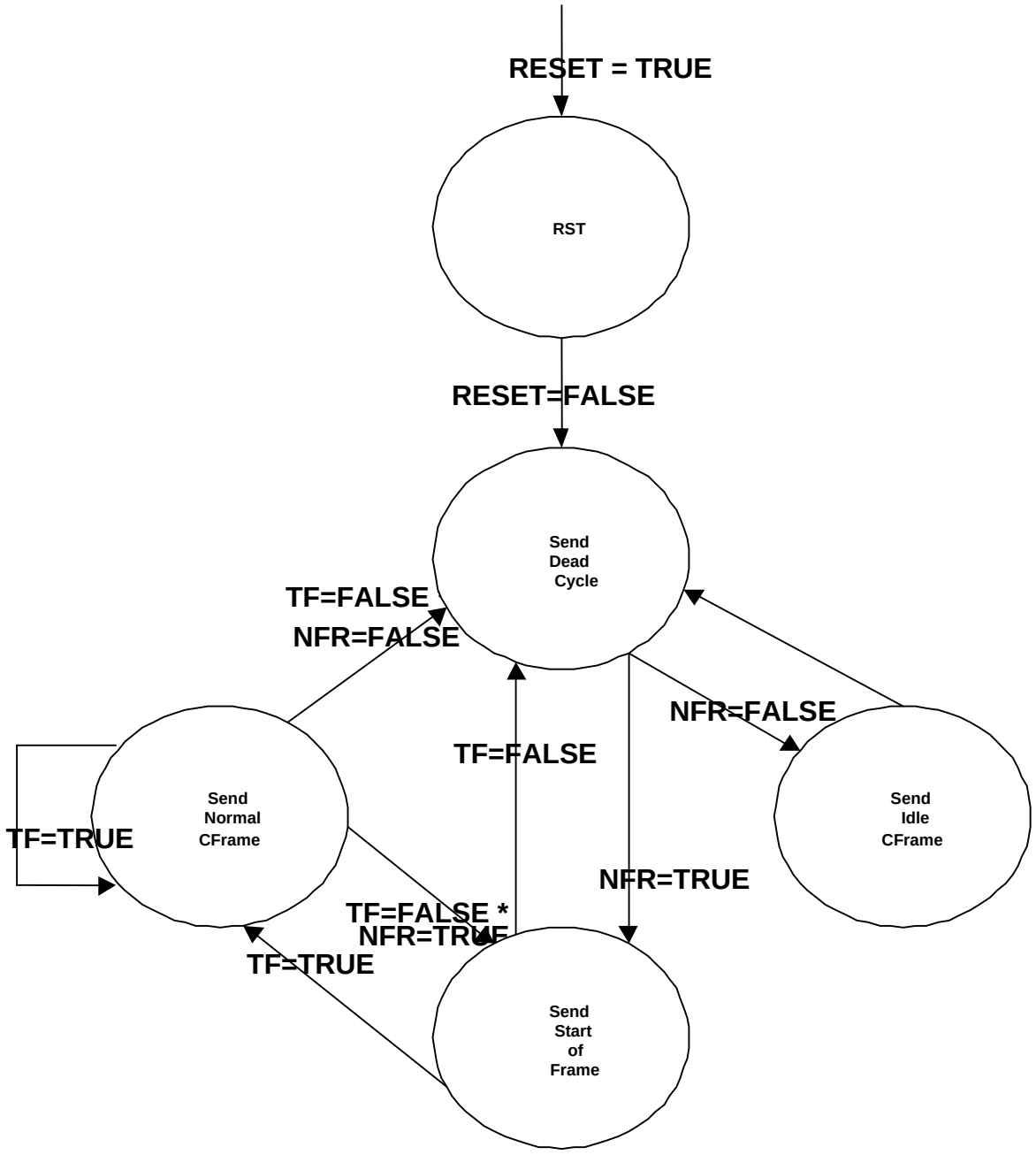

Figure 13: Transmission state machine

6.5 Pause and Resume operation

The pause and resume information (carried with Ready bits in the base header of every CFrame) provides a way to control the flow of data across CSIX. Deassertion of a Ready bit

indicates that the asserting device can not accept another frame after the one currently being transferred for that link level queue. The device receiving the pause shall complete the current frame and shall not resume transmission of CFrames until an asserted ready bit is received indicating a resume.

There are 2 link level queues and 2 corresponding Ready bits in every header. One queue is for control traffic and the other queue is for data traffic.

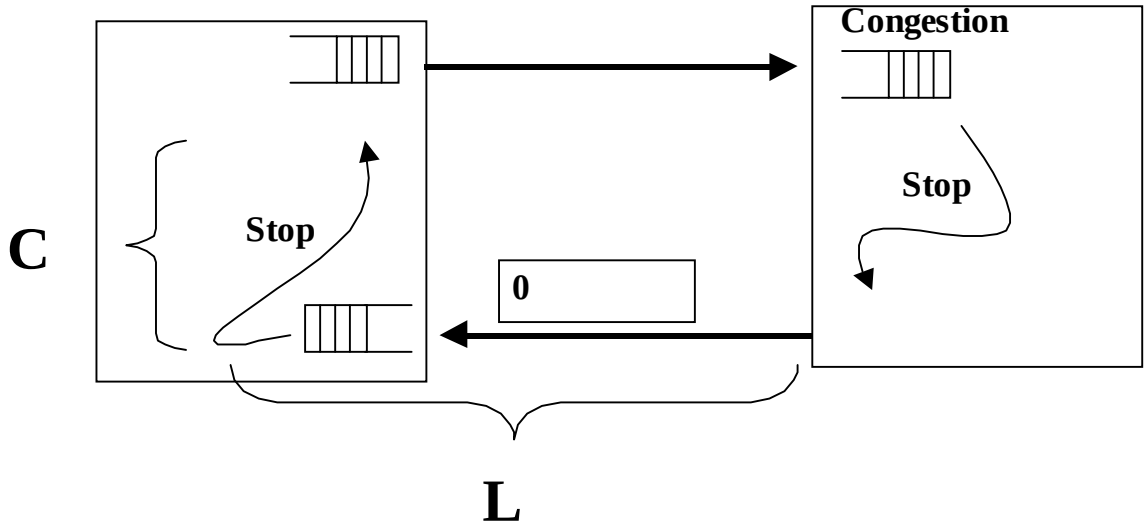

Figure 14-- CSIX-L1 Pause and Resume response requirement

Response Requirement: From the tick that a frame containing a Pause status leaves one component, after maximum response (n*T) time has elapsed, no new frame can begin transmission between the components for the congested link level queue.

Details of the requirement are as follows. From the tick that the ready field leaves a component the MAXIMUM response time for a Pause operation is defined as: n*T, n=C+L

-- T is the clock period of the interface

-- n is the maximum number of ticks for the response

-- C is a constant for propagating the field within the "other" component (or chipset as the case may be) to the interface logic controlling the reverse direction data flow**.** C is defined to be 32 ticks.

-- L is the maximum number of ticks to transport the maximum fabric CFrame size.

6.6 Fabric Flow Control Response Time

The response time for fabric flow control is not specified by the CSIX-L1 standard in the same detail as link level flow control.

6.7 Frame Transfer Timing

The following figures show timing operations under different transmission scenarios. In these figures, CFa.n indicates the portion (n) of a specific CFrame (a) sent across CSIX-L1 at each

tick, DC indicates a dead cycle, ICF indicates an idle CFrame and SOF indicates Start of Frame.

Data	CF1.1	CF1.2	…..	CF1.n	CF2.1	CF2.2	….	CF2.n
SOF	HI	LO			HI	LO		

Figure 15-- Transmission of two consecutive CFrames with no break

Data	CF1.1	CF 1.2	…	CF1.n	DC	CF2.1	CF2.2	….	CF2.n
SOF	HI	LO				HI	LO		

Figure 16-- Transmission of two CFrames with a one-tick separation

Data	CF1.1	CF 1.n	DC	ICF	DC	ICF	DC	CF2.1	CF2.n
SOF	HI	LO		HI	LO	HI	LO	HI	LO

Figure 17-- Transmission of two CFrames with a multi-tick separation

6.8 Dealing with a parity error

When a parity error (horizontal or vertical) is detected on the CFrame, ready bits from the CFrame shall be ignored and the device shall interpret them as not ready. This means that the device which detects the parity error(s) stops sending further data or control CFrames until it receives the next CFrame without any errors and corrects the state of the ready bits.

6.9 Dealing with an unexpected SOF

If a device encounters an unexpected SOF, it shall ignore the previous data and commence processing the CFrame signaled by the new SOF.

7 A.C. characteristics

7.1 AC Timing Classes

CSIX-L1 AC parameters are specified in three different classes as shown in Table 28.

Different timing classes provide flexibility for devices designed for different max performance.

Devices with LVCMOS signaling shall be in a separate class due to different IO characteristics from HSTL.

Table 28: Timing classes

AC Class	Interface Signaling	Operating Frequency
AC_Class0	LVCMOS	100MHz – 166MHz
AC_Class1	HSTL	100MHz – 166MHz
AC_Class2	HSTL	100MHz – 250MHz

7.2 Timing Paradigm

The following figures describe the definitions of AC parameters for CSIX-L1 source synchronous interface.

7.2.1 Source Interface Timing Definitions

From a source device, source data is driven along with source clock with the timing parameters shown in the figure below. The on-chip delay on the source data and source clock should be balanced, hence minimising the total skew. The timing parameters are restricted within one clock group, which has 32-bits of data, 1 bit of parity and 1-bit SOF.

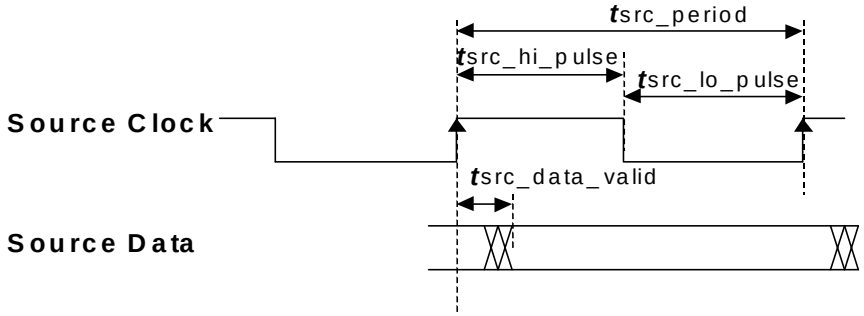

Figure 18-- Source interface timing

7.2.2 Destination Interface Timing Definitions

Similar to source device interface, the destination device receives the destination data along with destination clock with the timing parameters shown in the figure below. The on-chip delay

on the destination data and destination clock should be balanced, so that the set-up and hold timing requirements on the receiving flip-flops are met. The timing parameters are restricted within one clock group, which has 32-bits of data, 1 bit of parity and a 1-bit SOF.

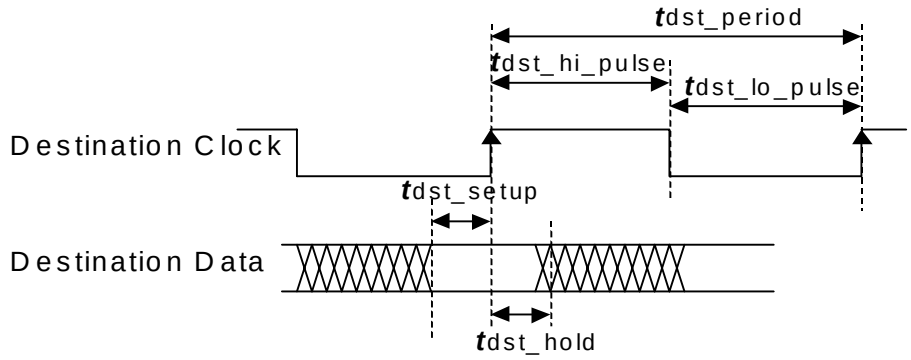

Figure 19-- Destination interface timing

7.2.3 Source Clock / Destination Clock Skew

This timing parameter applies to interfaces where the bus is 64-bits, 96-bits or 128-bits wide and split into groups of 32-bit source synchronous busses. It is important to keep the source synchronous clocks from each group within certain skew window.

Note: *There is no such requirement for clocks in opposite direction since they are asynchronous to each other.*

7.2.3.1

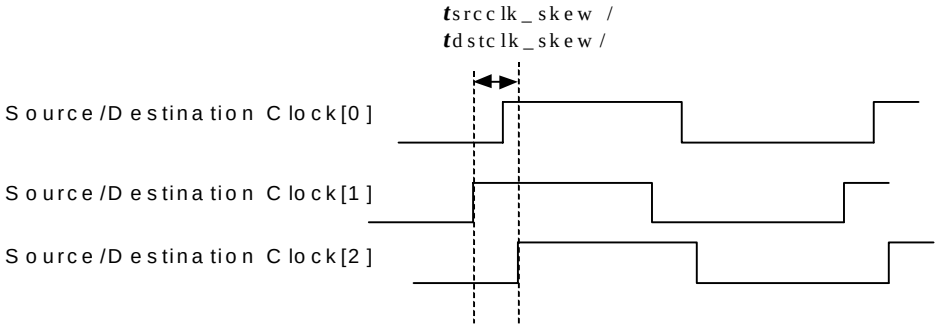

Figure 20-- Source/Destination clock skew

7.3 AC_Class0 Timings (LVCMOS, 100MHz – 166MHz)

AC timing data for Class0 devices is shown in Table 29.

Table 29: Class 0 AC timing data

Name	Characteristics	Min	Max	Note
tsrc_freq	Source Clock Frequency		166 MHz	
tsrc_period	Source Clock Period	1/ ***t***src_freq	1/ ***t***src_freq	1
tsrc_hi_pulse	Source Clock High Pulse Width	.4 * ***t***src_period	.6 * ***t***src_period	1
tsrc_lo_pulse	Source Clock Low Pulse Width	.4 * ***t***src_period	.6 * ***t***src_period	2
tsrc_rise_slew	Source Clock Rise Slew		.5 ns	
tsrc_fall_slew	Source Clock Fall Slew		.5 ns	
tsrc_data_valid	Source Data Valid from Source Clock Rising Edge	.5 ns	2.0 ns	3
*t*srcclk_skew	Skew among Source Clocks from different clock groups		1.2 ns	
tdst_freq	Source Clock Frequency		166 MHz	
tdst_period	Destination Clock Period	1/ ***t***dst_freq	1/ ***t***dst_freq	1
tdst_hi_pulse	Dest Clock High Pulse Width	.35 * ***t***dst_period	.65 * ***t***dst_period	1
tdst_lo_pulse	Dest Clock Low Pulse Width	.35 * ***t***dst_period	.65 * ***t***dst_period	2
tdst_rise_slew	Dest Clock Rise Slew		.7 ns	
tdst_fall_slew	Dest Clock Fall Slew		.7 ns	
tdst_setup	Dest Data set-up to Dest Clock Rising Edge	1.5 ns		3
tdst_hold	Dest Data hold from Dest Clock Rising Edge	0		3
tdstclk_skew	Skew among Dest Clocks from different clock groups		1.5 ns	

Notes:

1. This clock timing parameter is measured from VIH to VIH.
2. This clock timing parameter is measured from VIL to VIL.
3. The data valid, set-up and hold are measured from the midpoint of the clock (VDD/2) to either:

    2.0 V for transitions from Hi-Z or Low logic values to High logic values, or

    0.8 V for transitions from Hi-Z or High logic values to Low logic values.

The load model for drivers meeting the Class 0 AC timings is assumed to be the following.

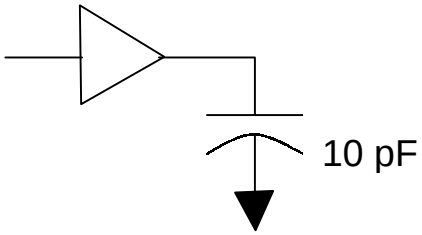

Figure 21—Load model for drivers meeting the Class 0 AC timings

AC_Class1 (HSTL, 100MHz – 166MHz)

Class 1 timing data is shown in Table 30.

Table 30: Class 1 AC timing data

Name	Characteristics	Min	Max	Note
tsrc_freq	Source Clock Frequency		166 MHz	
tsrc_period	Source Clock Period	1/ ***t***src_freq	1/ ***t***src_freq	2
tsrc_hi_pulse	Source Clock High Pulse Width	0.45 * ***t***src_period	0.55 * ***t***src_period	2
tsrc_lo_pulse	Source Clock Low Pulse Width	0.45 * ***t***src_period	0.55 * ***t***src_period	2
tsrc_rise_slew	Source Clock Rise Slew		0.3 ns	4
tsrc_fall_slew	Source Clock Fall Slew		0.3 ns	4
tsrc_data_valid	Source Data Valid from Source Clock Rising Edge	0.3 ns	3.0 ns	1,3
tsrcclk_skew	Skew among Source Clocks from different clock groups		1.2 ns	2
tdst_freq	Source Clock Frequency		166 MHz	
tdst_period	Destination Clock Period	1/ ***t***dst_freq	1/ ***t***dst_freq	2
tdst_hi_pulse	Dest Clock High Pulse Width	0.40 * ***t***dst_period	0.60 * ***t***dst_period	2
tdst_lo_pulse	Dest Clock Low Pulse Width	0.40 * ***t***dst_period	0.60 * ***t***dst_period	2
tdst_rise_slew	Dest Clock Rise Slew		0.4 ns	4
tdst_fall_slew	Dest Clock Fall Slew		0.4 ns	4
tdst_setup	Dest Data set-up to Dest Clock Rising Edge	2.0 ns		1,3
tdst_hold	Dest Data hold from Dest Clock Rising Edge	0 ns		1,3
tdstclk_skew	Skew among Dest Clocks from different clock groups		1.5 ns	2

Notes:

1. Assuming a perfect board distribution, this spec implies 1ns of set-up margin and 0.3ns of hold margin at 166MHz. (Refer to Appendix-A)
2. The clock timing parameters are measured from VDDQ/2 (VREF cross over point)
3. The data valid, set-up and hold are measured from VDDQ/2 (VREF cross over point)
4. Slew rates are measured between VIH and VIL levels

7.4 AC_Class2 (HSTL, 100MHz – 250MHz)

Class 2 timing data is shown in Table 31.

Table 31: Class 2 AC timing data

Name	Characteristics	Min	Max	Note
tsrc_freq	Source Clock Frequency		250 MHz	
tsrc_period	Source Clock Period	1/ ***t***src_freq	1/ ***t***src_freq	2
tsrc_hi_pulse	Source Clock High Pulse Width	0.45 * ***t***src_period	0.55 * ***t***src_period	2
tsrc_lo_pulse	Source Clock Low Pulse Width	0.45 * ***t***src_period	0.55 * ***t***src_period	2
tsrc_rise_slew	Source Clock Rise Slew		0.3 ns	4
tsrc_fall_slew	Source Clock Fall Slew		0.3 ns	4
tsrc_data_valid	Source Data Valid from Source Clock Rising Edge	0.3 ns	1.8 ns	1,3
tsrcclk_skew	Skew among Source Clocks from different clock groups		0.8 ns	2
tdst_freq	Source Clock Frequency		250 MHz	
tdst_period	Destination Clock Period	1/ ***t***dst_freq	1/ ***t***dst_freq	2
tdst_hi_pulse	Dest Clock High Pulse Width	0.40 * ***t***dst_period	0.60 * ***t***dst_period	2
tdst_lo_pulse	Dest Clock Low Pulse Width	0.40 * ***t***dst_period	0.60 * ***t***dst_period	2
tdst_rise_slew	Dest Clock Rise Slew		0.4 ns	4
tdst_fall_slew	Dest Clock Fall Slew		0.4 ns	4
tdst_setup	Dest Data set-up to Dest Clock Rising Edge	1.5 ns		1,3
tdst_hold	Dest Data hold from Dest Clock Rising Edge	0 ns		1,3
tdstclk_skew	Skew among Dest Clocks from different clock groups		1.0 ns	2

Notes:

1. Note that assuming a perfect board distribution, this spec implies 0.7 ns of set-up margin and 0.3ns of hold margin at 250 MHz. (Refer to Appendix-A)
2. The clock timing parameters are measured from VDDQ/2 (VREF cross over point)
3. The data valid, set-up and hold are measured from VDDQ/2 (VREF cross over point)
4. Slew rates are measured between VIH and VIL levels

8 D.C. characteristics

8.1 LVCMOS Interface

The following LVCMOS interface specifications are consistent with normal range values shown in EIA/JEDC standard EIA/JESD8-5: 2.5 0.2V± (Normal Range), and 1.8V to 2.7 V (Wide Range) Power Supply Voltage and Interface Standard for Nonterminated Digital Integrated Circuit (October 1995.) Refer to that specification for more information.

Table 32: Class 0 DC characteristics

Symbol	Parameter		Min	Max	Unit	Notes
V_{DD}	Supply voltage		2.3	2.7	V	
V_{IH}	Hi-level input voltage		1.7	V_{DD} + 0.3	V	1
V_{IL}	Low-level input voltage		-0.3	0.7	V	2
V_{OH}	Output high voltage	I_{OH} = -100µA	2.1	3.5	V	3
		I_{OH} = -1mA	2		V	3
		I_{OH} = -2mA	1.7		V	3
V_{OL}	Output low voltage	I_{OL} = 100µA		0.2	V	3
		I_{OL} = 1mA		0.4	V	3
		I_{OL} = 2mA		0.7	V	3
I_I	Input current	Except I/O parts		±5	µA	4
		I/O parts		±15	µA	4

Notes

1. $V_{OUT} \geq V_{OH}$(min)
2. $V_{OUT} \leq V_{OL}$(max)
3. $V_{DD} = VDD_{MIN}$
4. $V_{DD} = VDD_{MAX}$

8.2 HSTL, Class-1 Interface

Following HSTL single-ended Interface specifications are consistent with EIA/JEDEC standard, EIA/JESD8-6 “High Speed Transceivers Logic (HSTL): A 1.5V Output Buffer Supply Voltage Based Interface Standard for Digital Integrated Circuits”, Aug 1995. Refer to the specification for more information. This applies to AC Class 1 and AC Class 2.

Table 33: Class 1 DC characteristics

Symbol	Parameter	Min	Nom	Max	Unit	Notes
VDDQ	Output Supply Voltage	1.4	1.5	1.6	V	1
VREF	Differential Input Reference Voltage	0.68	0.75	0.9	V	2
Voh	Output High Voltage	VDDQ – 0.4			V	3,5
Vol	Output Low Voltage			0.4	V	4,5
Vih	Input High Voltage	VREF+ 0.1		VDDQ+ 0.3	V	6
Vil	Input Low Voltage	-0.3		VREF – 0.1	V	6

Notes:

1. There is no specific device core supply voltage requirement for HSTL compliance.
2. VREF = VDDQ/2
3. Ioh $\geq$ 8mA
4. Iol $\leq$ -8mA
5. Push-pull output buffer for unterminated loads.
6. Inputs should not be within VREF –0.1 and VREF +0.1 V.

9 Conformance Requirements

To be conformant with the CSIX-L1 specification, a device shall provide the following functions.

Item	Feature	§	Support	Value/Comment
1.	A CFrame shall	1.5	Yes ☐	be divisible by the size of a CWord.
2.	A vendor that implements less than 256 classes shall	1.5	Yes ☐	encode their classes beginning with the most significant Class bit.
3.	CSIX Reserved bits	1.5	Yes ☐	shall not be used for proprietary messages
4.	Padding characters shall	1.5	Yes ☐	be ignored when processing fields but shall be included in vertical parity calculations (if vertical parity is implemented.)
5.	Maximum payload size shall not exceed	1.5	Yes ☐	256 bytes.
6.	The CSIX fabric	2.3.1	Yes ☐	shall ensure that all CFrames from a specific ingress TM to a specific Class queue of a specific egress TM are delivered to the egress TM in exactly the same sequence that they were presented to the fabric by the ingress TM.
7.	The fabric	2.3.1	Yes ☐	shall guarantee this (see item 6 above) in-order delivery even if the fabric internally has multiple routes between ingress TM and egress TM.
8.	If a fabric supports a range of MAX_FRAME_PAYLOAD_SIZE values, the fabric	3.1	Yes ☐	shall provide a mechanism for programming this value.
9.	The fabric	3.1	Yes ☐	shall support CFrame sizes of any size equal to or less than the maximum value specified for MAX_FRAME_PAYLOAD_SIZE.
10.	The TM	3.1	Yes ☐	shall provide a means of programming the exact value of MAX_FRAME_PAYLOAD_SIZE for a given system.
11.	Each user payload that	3.2.1	Yes ☐	shall be accompanied by a unicast

	is destined for a single traffic manager			Destination Address, which consists of a 12-bit Traffic Manager number indicates the final destination of the payload.
12.	The switch fabric	3.2.1	Yes ☐	shall translate the Destination Address into a route through to the destination traffic manager.
13.	All fabrics and all TMs	3.2.1	Yes ☐	Shall support the unicast frame type.
14.	Each CSIX-L1 component	3.5.2	Yes ☐	shall continuously maintain all flow control state information.
15.	The class field	3.5.4	Yes ☐	shall be used to indicate multicast queue number for Multicast ID and Multicast Mask.
16.	The class field	3.5.4	Yes ☐	shall be orthogonal to bitmask and Ids.
17.	CSIX-L1 shall utilize a nx32-bit data path,	4	Yes ☐	where n=1, 2, 3 or 4.
18.	The switch fabric	4.1.1	Yes ☐	shall present data to the traffic manager via the vector TxData.
19.	The switch fabric	4.1.2	Yes ☐	shall present data parity to the traffic manager via the vector TxPar (horizontal parity).
20.	The switch fabric	4.1.3	Yes ☐	shall provide data transfer/synchronization clock(s) to the traffic manager for synchronizing transfers on TxData.
21.	When more than one TxSOF signal is required, all TxSOF signals	4.1.4	Yes ☐	shall have identical values during each TxClk cycle.
22.	TxSOF	4.1.4.1	Yes ☐	shall be asserted during the first transfer cycle of each CFrame
23.	Sequential TxSOFs shall be separated by at least	4.1.4.1	Yes ☐	One tick.
24.	The traffic manager	4.1.5	Yes ☐	shall present data to the switch fabric via the vector RxData.
25.	The traffic manager	4.1.6	Yes ☐	shall present data parity to the switch fabric via the vector RxPar

				(horizontal Parity).
26.	The traffic manager	4.1.7	Yes ☐	shall provide data transfer/synchronization clock(s) to the switch fabric for synchronizing transfers on RxData.
27.	When multiple RxSOF signals are required, they	4.1.8.1	Yes ☐	shall have identical values during each RxClk cycle.
28.	RxSOF	4.1.8.1	Yes ☐	shall be asserted during the first transfer cycle of each CFrame.
29.	Sequential RxSOFs shall be separated by at least	4.1.8.1	Yes ☐	One tick.
30.	If the payload does not provide sufficient space for the vertical parity field in the last CWord,	5	Yes ☐	a new CWord shall be added and the vertical parity bytes will appear in the least significant bits of the added word
31.	For idle CFrames, the payload length field shall be set to	5.2.3	Yes ☐	“0.”
32.	The speed field (at least the most significant bit)	5.9.1.1.6	Yes ☐	shall be generated and processed for all entry types.
33.	All implementations shall be capable of	5.9.1.1.6		distinguishing and responding to speed field values “1111” and “0000”
34.	The width of the receive and transmit paths	5	Yes ☐	shall be the same.
35.	A device receiving the pause	6.5	Yes ☐	shall complete the current frame and shall not resume transmission of CFrames until a ready bit is received asserted indicating a resume.
36.	When a parity error (horizontal or vertical) is detected on the CFrame,	6.8	Yes ☐	ready bits from the CFrame shall be ignored and the device shall interpret them as not ready. This means that the device which detects the parity error(s) stops sending further data or control CFrames until it receives the next CFrame without any errors and corrects the state of

				the ready bits.
37.	If a device encounters an unexpected SOF	6.9	Yes ☐	it shall ignore the previous data and commence processing the CFrame signaled by the new SOF.
38.	Devices with LVCMOS signaling	7.1	Yes ☐	shall be in a separate class due to different IO characteristics from HTSL.

1

Appendix A: Set-up/hold timing margin calculations

A.1 AC_Class0

tsrc_period = ***t***dst_period = 6ns (166 MHz)

tsrc_data_valid (source data valid from source clock rising edge)
0.3ns Min, 2.0ns Max

tdst_setup (dest data setup to dest rising edge)
1.5ns Min

tdst_hold (dest data hold to dest rising edge)
0.0ns Min

board_skew (data and clock on source synchronous path on board)

Setup Margin = ***t***dst_period - ***t***src_data_valid(Max) - ***t***dst_setup - board_skew
= 6ns – 2ns – 1.5ns - board_skew
= 1.5ns - board_skew

i.e. Assuming the perfect board distribution, the spec implies 1.5ns of setup margin

Hold Margin = ***t***src_data_valid(Min) - ***t***dst_hold - board_skew
= 0.5ns – 0ns – board_skew
= 0.5ns - board_skew

i.e. Assuming the perfect board distribution, the spec implies 0.5ns of hold margin

A.2 AC_Class1

tsrc_period = ***t***dst_period = 6ns (166 MHz)

tsrc_data_valid (source data valid from source clock rising edge)
0.3ns Min, 3.0ns Max

tdst_setup (dest data setup to dest rising edge)
2.0ns Min

tdst_hold (dest data hold to dest rising edge)
0.0ns Min

board_skew (data and clock on source synchronous path on board)

Setup Margin = ***t***dst_period - ***t***src_data_valid(Max) - ***t***dst_setup - board_skew
= 6ns – 3ns – 2ns - board_skew
= 1ns - board_skew

i.e. Assuming the perfect board distribution, the spec implies 1ns of setup margin

Hold Margin = ***t***src_data_valid(Min) - ***t***dst_hold - board_skew
= 0.3ns – 0ns – board_skew
= 0.3ns - board_skew

i.e. Assuming the perfect board distribution, the spec implies 0.3ns of hold margin

A.3 AC_Class2

tsrc_period = ***t***dst_period = 4ns (250 MHz)

tsrc_data_valid (source data valid from source clock rising edge)
0.3ns Min, 1.8ns Max

tdst_setup (dest data setup to dest rising edge)
1.5ns Min

tdst_hold (dest data hold to dest rising edge)
0.0ns Min

board_skew (data and clock on source synchronous path on board)

Setup Margin = ***t***dst_period - ***t***src_data_valid(Max) - ***t***dst_setup -board_skew
= 4ns – 1.8ns – 1.5ns - board_skew
= 0.7ns - board_skew

i.e. Assuming the perfect board distribution, the spec implies 0.7ns of setup margin

Hold Margin = ***t***src_data_valid(Min) - ***t***dst_hold - board_skew
= 0.3ns – 0ns - board_skew
= 0.3ns - board_skew

i.e. Assuming the perfect board distribution, the spec implies 0.3ns of hold margin

Appendix B: Example System Requirement for Pause/Resume

Consider a system integration where the fabric has a maximum CFrame size of 74 bytes, the CSIX-L1 interface is 32-bits wide, and the interface clock is 150 MHz.

The CSIX-L1 response specification is calculated as follows (Assuming the C factor is 32 ticks):

--L= ceiling(74/4) = 19 ticks

--n=32+19=51 ticks: Response time specification

Therefore the maximum response time = 6.66 ns * 51 ticks = 340 ns.

To get an approximate sizing of the link level Receive buffers, the following analysis can be done for our example system.

The RxBuffer needs a minimum delta between congestion threshold and the buffer size of (3L + C) ticks * bus width

-- L within congested component to get ready field into header

-- L within other component to error check frame with ready field

-- C to get to Tx Logic of other component

-- L to transmit current frame before the next frame can be stopped

The delta between congestion threshold and per link level queue for our example system can be calculated to be 3*19+32=89 ticks * 32 bits = 2.8 Kbits.

With 2-ready bits, and 2 separate link level queues needed, the summed delta is 5.6 Kbits. If the queues were dimensioned to be say twice the size of the delta between congestion threshold and max buffer size the link level buffers would need to be 11.2 Kbits in size.