Scale Up Ethernet Framework

Specification

Table of Contents

Introduction 5
1.1 Deployment Model 5
1.2 SUE Overview 6

SUE Requirements 8
2.1 Protocol Requirements - Background 8

SUE Interfaces 10
3.1 XPU Management Interface 10
3.2 XPU Command Interface 10
3.3 Ethernet Interface 13

SUE Operation 16
4.1 Memory Model and Semantics 16
4.2 SUE Processing Overview 16
4.3 Load Balancing 18

Appendix A: Latency Budget 19
A.1 SUE E2E One Way Latency Budget 19

Appendix B: Glossary 20

Revision History 21
Scale-Ethernet-RM101; May 1, 2025 21
Scale-Ethernet-RM100; April 28, 2025 21

List of Tables

Table 1 – XPU to SUE Signal Interface 11
Table 2 – SUE to XPU Signal Interface 12
Table 3 – SUE RH Fields 12
Table 4 – Ethernet Ports with 2 SUE per Port Macro 13

List of Figures

Figure 1 – Example Single hop Switched Deployment 5
Figure 2 – Example Mesh Deployment 6
Figure 3 – SUE Stack 7
Figure 4 – SUE Interfaces 10
Figure 5 – SUE Reliability Header 12
Figure 6 – Standard Ethernet Format 13
Figure 7 – AI Forwarding Header 13
Figure 8 AI Fabric Header Compression 14
Figure 9 – AIFH Field Mapping 14
Figure 10 – AIFH 6B Header 14
Figure 11 – SUE Processing Flow 17
Figure 12 – Example XPU Load Balancer 18
Figure 13 – SUE End-to-End One-Way Latency Budget 19

Introduction

As workloads continue to expand in complexity in areas such as machine learning and AI inference, the need for parallel processing power increases significantly. Scaling up a GPU (or more generically an XPU, to include various custom ML accelerators) cluster to a rack or multi-rack level is needed to enhance its ability to process larger datasets, train deeper neural networks, handle more simultaneous tasks, and reduce execution time and improve overall system efficiency.

Ethernet offers many advantages for designing such a fabric, from industry leading high-speed links and high-capacity switches to a well-developed ecosystem and well-understood operational methods. Multiple industry groups are currently involved in developing networking technologies for AI based networks that extend Ethernet or use some of its components as a building block.

SUE provides a framework to provide low latency, high bandwidth connectivity for XPU scale up networks based on Ethernet. The intent of Scale Up Ethernet (SUE) is to provide the transport and ethernet datalink to move memory transactions between XPUs.

1.1 Deployment Model

SUE provides communication between XPUs in a multi-XPU system. Each SUE provides network connectivity which can be configured as one, two or four ports. For example, an 800G SUE instance supports 1x 800G, 2x 400Gbps or 4x 200Gbps. The primary network configuration is expected to be a single switch hop. The port configuration is chosen based on the required switch radix and serdes count, as well as redundancy and failover considerations.

SUE is designed to enable multiple instances of SUE per XPU. For example, a single XPU may include 8 or 16 SUE instances. Figure 1 shows an example configuration of 64 XPUs with twelve 800G SUE instances per XPU. Twelve Ethernet switches are used with 64 ports each. In this example, with fully switched connectivity, any pair of XPUs has up to 9.6Tbps (12x 800G) bandwidth between them.

Alternatively, SUE can be used in a mesh with direct connection between a pair of XPUs, as illustrated in Figure 2.

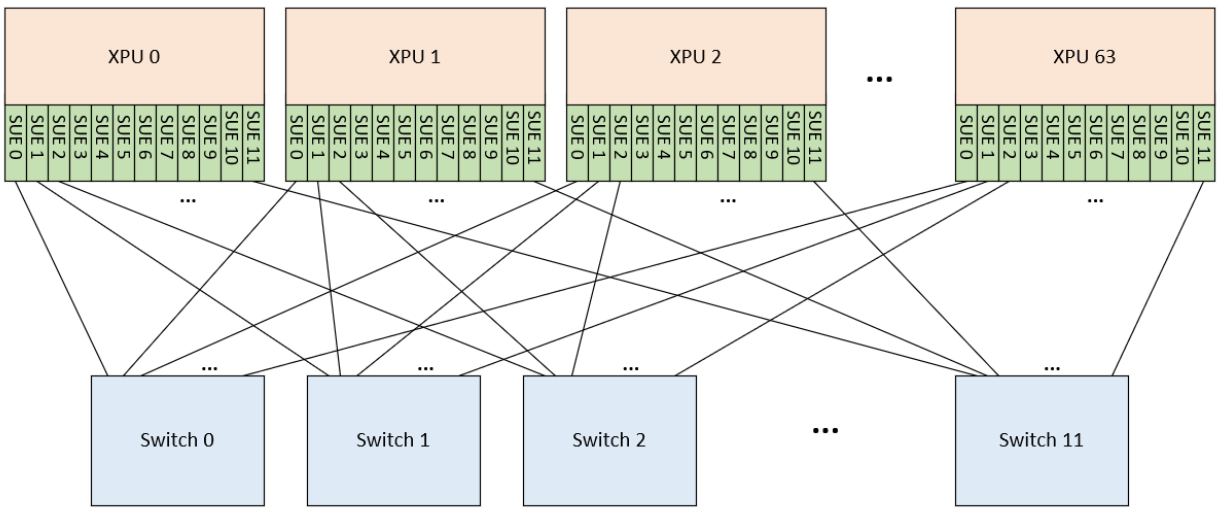


Figure 1 – Example Single hop Switched Deployment

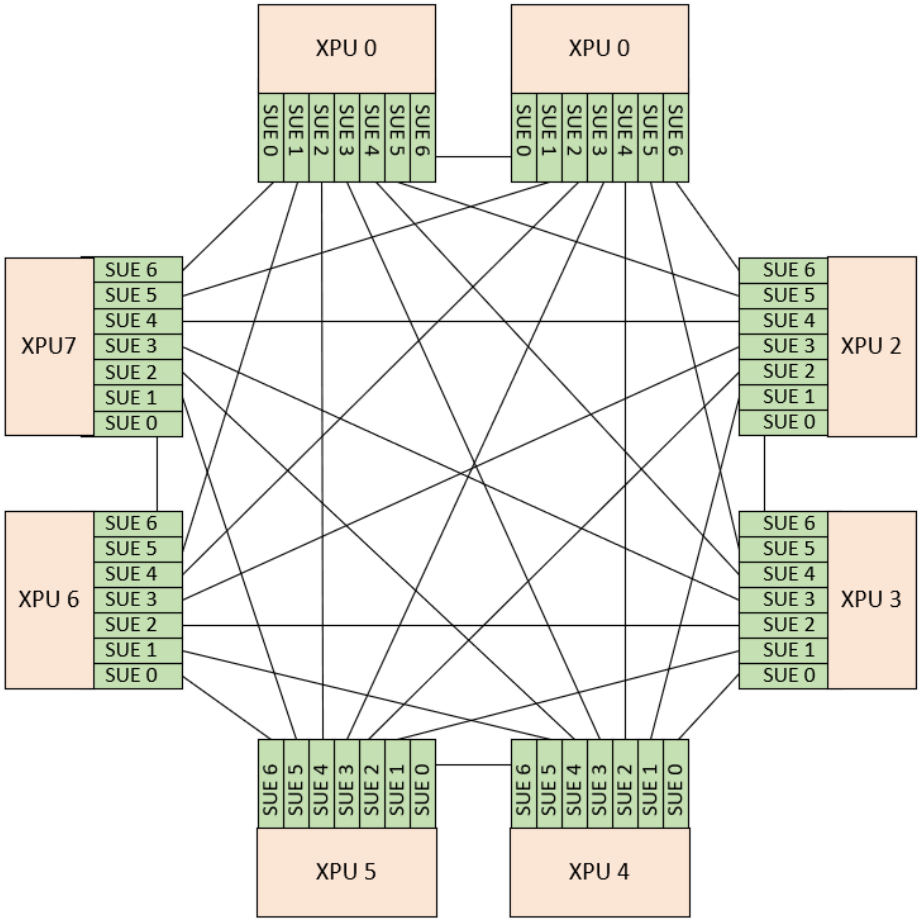


Figure 2 – Example Mesh Deployment

1.2 SUE Overview

SUE provides a mechanism for generic command / response transactions. The set operations and structure of the commands are XPU specific and transparent to SUE. This command channel can be used to implement a variety of services such as put, get, and atomic operations. Additional services such as cache coherency may be defined – full implementation of these services is beyond the scope of SUE.

SUE views all transactions as commands. The XPU supports command/response by defining appropriate opcodes (e.g. read and read response) and by mapping these to different traffic classes.

Figure 3 provides an overview of the SUE stack. The XPU NOC sends a command to SUE. SUE provides a duplex data interface similar to AXI. This interface includes control and data. The control content is largely opaque. It includes a command, the remote XPU identifier and fields used to determine if data is present as well as length information.

The mapping and packing layer organizes the command by destination and opportunistically packs multiple into a single SUE protocol data unit (PDU). Each SUE PDU is destined to a single XPU on a single traffic class. Virtual channel (VC) in the control is used to map the packet to the appropriate traffic class.

SUE provides two ordering modes: strict ordering and unordered. In strict mode, all transactions between a source and destination are passed from the SUE transport to the destination XPU in order. When configured for more than one port, SUE supports an unordered mode where the transactions are load balanced across the ports and order is not maintained.

Each SUE PDU is mapped to a single Ethernet packet. Reliability, data integrity, and encryption are provided at the packet level.

The network layer builds the header based on the destination XPU. This may be a standard Ethernet header or use an optimized format, as described in section 3.3.1.

Link level retry (LLR) and priority flow control (PFC) or credit based flow control (CBFC) are used to provide lossless service. Above this, the SUE transport provides retransmission should an unrecoverable error occur.

At the destination, the SUE PDU is unpacked and each command is passed to the XPU.

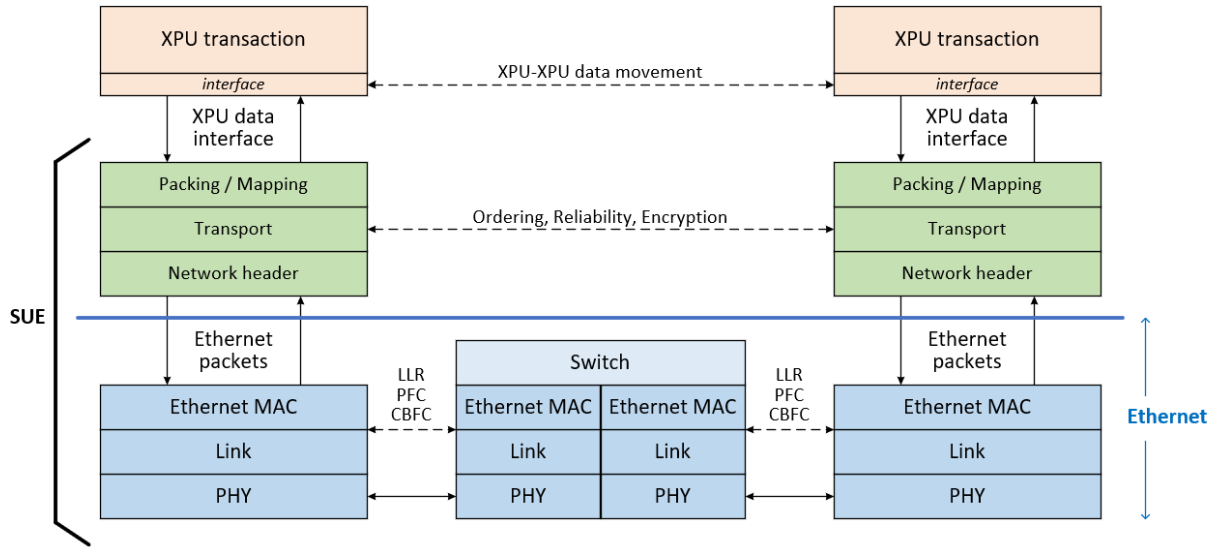


Figure 3 – SUE Stack

SUE Requirements

The design philosophy of SUE is to provide low latency and high bandwidth in a manner that allows highly efficient implementations, enabling multiple instances to be instantiated with efficient area and power. Based on this philosophy, the following requirements were used to guide the development of SUE and are not strictly required. SUE can be adapted to specific use cases.

	Requirement	Notes
Number of XPUs	Up to 1024	Single hop, low latency
Transaction types	1-sided operations Memory load-store-atomics Smaller transfers	
Memory architecture	Shared memory	E.g., PGAS, registration outside scope of SUE
E2E latency	< 2us RTT	Round trip time
Cable length	Up to 10m from SUE to switch	For latency
Bandwidth per SUE	800Gbps	Scale to 1.6Tbs
Serdes	200Gbps	Supports 100G & 50G
Port configuration	1, 2 or 4	Enable higher radix switches, redundancy and failover
Virtual channels	Up to 4	Independent traffic classes selected via QoS field, avoid deadlock scenarios
ETH compliance	Support standard Ethernet	Provide option to compressed header format, LLR, CBFC

SUE is not a general purpose network interface with a full suite of services available in modern NICs. It also does not define an XPU level protocol to enable interoperation between different XPU devices.

2.1 Protocol Requirements - Background

There is a large amount of parallelism in XPUs and the scale-up transport protocol must enable XPU to XPU connectivity concurrently across many planes. To achieve power and area efficiency, and because the deployment topologies for XPU to XPU are constrained, a relatively simple transport is utilized. The transport protocol needs to provide for the reliable delivery of higher-layer data from the source to destination XPU across a single switch hop.

The scale-up topology may be both multi-plane and multi-rail, the same as the scale-out network. The key part of building an efficient transport protocol for such networks is effectively load balancing the data across the available planes and handling many-to-one traffic (incast) traffic patterns without causing head-of-line blocking in the network.

In the scale-out topology this is achieved by having the transport layer explicitly include multipath operation, which is done today using QP scaling in collective libraries such as NCCL for RoCEv2 and natively by packet spraying transports such as Ultra Ethernet Transport. To deal with incast traffic patterns, PFC is used by RoCEv2 but deprecated in next generation transport to avoid the performance anomalies due to head-of-line blocking; instead, packet trimming is used where the networks avoid silent packet drops by forwarding the header of the dropped packet with priority to the destination.

Adopting the same solution for scale-up networks is not practical. Scale-up networks have different requirements – much higher bandwidth, lower RTT, smaller scale, etc. – which leads to different trade-offs.

Scale-out networks leveraging packet spraying are complex because they must support network reordering which requires per PSN and per-path tracking at both the sender and the receiver. This complexity is useful for the scale-out network to avoid collisions but is simply not needed for the scale-up network: collisions within a plane cannot appear in a single-tier topology. The scale-out approach to use trimming and selective retransmissions is very useful for larger networks, but it is also complex as it requires packet tracking bitmaps and selective acknowledgements, each of which increase area and cost.

Scale-up transport speeds are expected to be an order of magnitude larger than scale-out speeds. The current bandwidth supported by scale-out NICs is 800Gbps, implying that many such NICs would be required for the scale-up network. This is challenging from an area and power perspective.

Finally, the endpoints that the scale-up transport must serve are all known in advance (all active XPUs). Connections can be set up statically at boot time and kept alive as long as both the plane and target XPU are alive. This further simplifies connection setup and scaling decisions in the transport protocol.

A much simpler solution is to use a single-path transport per plane. Relying on lossless operation results in a much simpler transport, where simple go-back-N is used to recover from infrequent losses, resulting in significantly smaller transport state and simpler send / receive processing. To further reduce packet loss due to corruption, link-layer retransmit (LLR is a switch feature that retransmits packets when the FCS checksum fails at the receiver) will be enabled when available.

To efficiently support a go-back-N transport, the network will be configured to enable lossless operation between XPUs either via Priority Flow Control (PFC) or the newer Credit-Based Flow Control (CBFC). To avoid triggering flow control and the resulting head-of-line blocking, state-of-the-art congestion control should be used at the endpoints.

Using multiple network interfaces per XPU implies that traffic must be load balanced across those interfaces by an external module (e.g. the XPU). When a single SUE instance provides multiple port, that network instance can load balance only across its ports/planes when in unordered mode.

In other words, the load balancing layer must spread traffic across all the planes of all its scale-up interfaces. This is done above SUE.

Given the need for near-optimal performance, the per-plane transport must start at line rate. The only performance problem that can appear in a single-tier network is incast when multiple senders start sending at the same time towards the same receiver.

SUE Interfaces

Each SUE instance has three interfaces, as illustrated in Figure 4:

- XPU command interface
- XPU management interface
- Ethernet interface

A description of each of these is provided in the following three subsections.

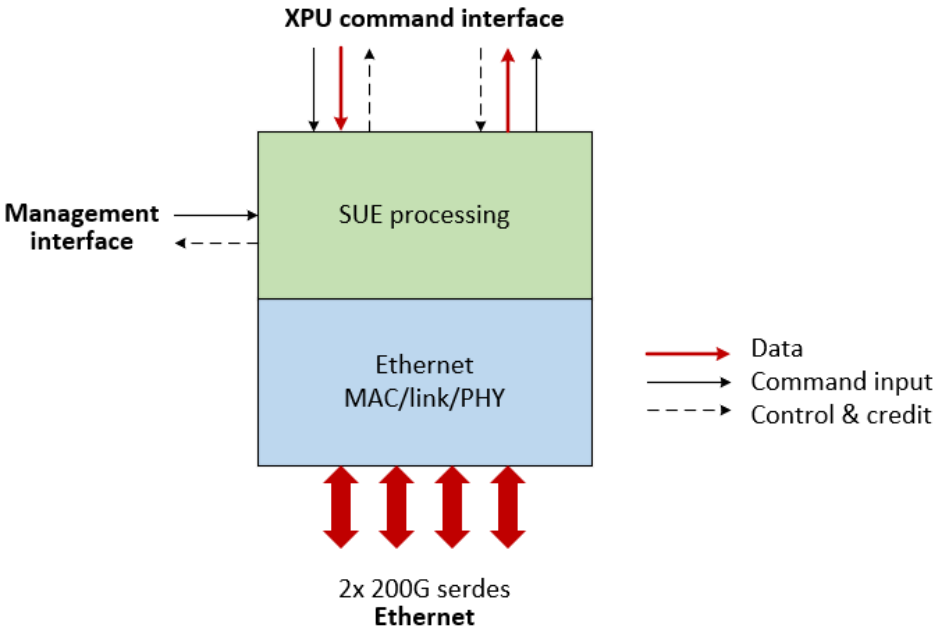


Figure 4 – SUE Interfaces

3.1 XPU Management Interface

SUE uses an AXI target interface for control. Register access is provided.

In addition to configuration and status, the control interface can be used to send and receive packets. The XPU builds the full packet with SUE adding the Ethernet FCS. This transport is unreliable. It supports rates up to 10K packets per second.

3.2 XPU Command Interface

The XPU-SUE interface is essentially a FIFO interface using credits. The XPU writes a command to the SUE as well as any associated data. SUE generates flow control on a per {destination, VC} basis and on an aggregate rate basis using a credit mechanism.

In order to initiate transfer of data, the XPU sends the information listed in Table 1 to SUE over the command interface. Input refers to signals from XPU to SUE.

Field Name	I/O	Size (bits)	Description
tx_cycle	Input	4	Data cycle input number, transfer up to 256B of data – number of cycles is implementation specific depending on data bus width
tx_valid	Input	2	Type of operation: none, command only, command with data
tx_vc	Input	2	Virtual channel
xpuid	Input	10	Remote XPU identifier
tx_control	Input	144	Control fields, processed and packed into transaction to be sent to remote XPU – this carries command and length information. First bytes) include command and length.
tx_data	Input	N	Data, serialized over multiple cycles depending on N; for example, if N = 256 bits (32B) then 8 cycles are used to transfer 256B.
tx_err	Input	1	Uncorrectable ECC error on interface
tx_credit	Output	6	Credit & valid for cmd and cmd+data
tx_reset_in	Input	1	Reset
tx_reset_out	Output	1	Indicates SUE is in reset

Table 1 – XPU to SUE Signal Interface

As packets arrive, SUE verifies the network header and SUE PDU. It then unpacks and sends the transactions over the command interface. Table 2 list the SUE to XPU signals on this interface.

Field Name	I/O	Size (bits)	Description
rx_cycle	Output	4	Data cycle input number, transfer up to 256B of data – number of cycles is implementation specific depending on data bus width
rx_valid	Output	2	Type of operation: none, command only, command with data
rx_vc	Output	2	Virtual channel
xpuid	Output	10	Remote XPU identifier
rx_control	Output	144	Control fields, processed and packed into transaction to be sent to remote XPU – this carries command and length information. First bytes) include command and length.
rx_data	Output	N	Data, serialized over multiple cycles depending on N; for example, if N = 256 bits (32B) then 8 cycles are used to transfer 256B.
rx_credit	Input	6	Credit & valid for cmd and cmd+data
rx_reset_in	Input	1	Reset

Field Name	I/O	Size (bits)	Description
rx_reset_out	Output	1	Indicates SUE is in reset

Table 2 – SUE to XPU Signal Interface

3.2.1 SUE Encapsulation

SUE opportunistically packs the received transactions to the same {destination, VC}, creating up to 4096 bytes of packed SUE PDU. When scheduled for transmission, a reliability header (RH) is added to the header of the packed PDU and a 32-bit CRC (R-CRC) is added to the tail for data integrity.

The RH carries the fields listed in Table 3 and illustrated in Figure 5.

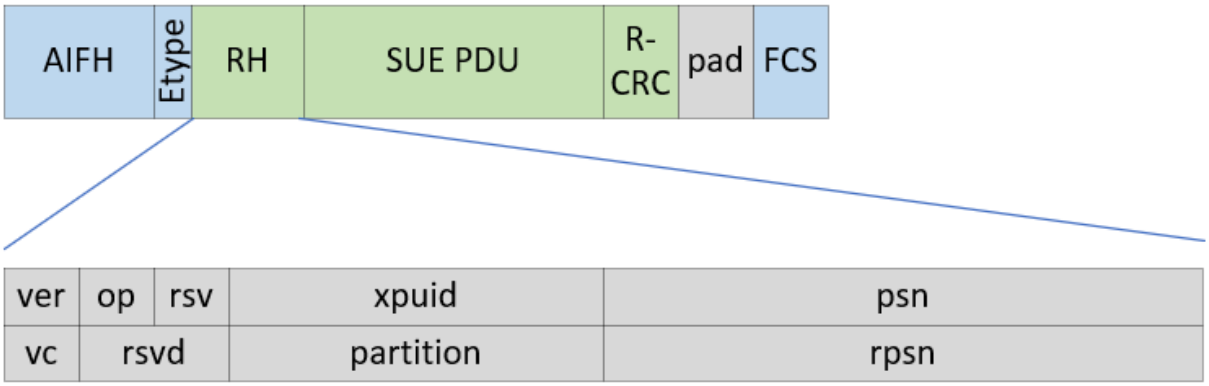


Figure 5 – SUE Reliability Header

Field Name	Size (bits)	Description
ver	2	Version
op	2	Opcode indicating if rpsn is an ACK, NACK or invalid
rsv	2	Reserved
xpuid	10	XPU identifier
psn	16	Packet sequence number
vc	2	Virtual channel, maps to traffic class
rsvd	4	Reserved
partition	10	Used to provide multi-tenant isolation
rpsn	16	ACK or NACK packet sequence number

Table 3 – SUE RH Fields

3.3 Ethernet Interface

SUE provides standard Ethernet interface(s) using 200G or 100G serdes rates. Table 4 describes the supported configurations.

Ethernet Port Speed	# Ports per SUE Instance	Serdes Lane Speed	# Serdes Lane per Ethernet Port
800G	1	200G	4
400G	2	200G	2
200G	4	200G	1
400G	1	100G	4
200G	2	100G	2
100G	4	100G	1

Table 4 – Ethernet Ports with 2 SUE per Port Macro

3.3.1 Network Encapsulation

SUE uses the destination XPU identifier, xpuid, and virtual channel, vc, to build the network header. A standard Ethernet header over IPv4 or IPv6 can be used or a more optimized AI Forwarding Header (AIFH) can be selected. AIFH reduces overhead on the wire.

When using UDP/IP, SUE is identified using the UDP destination port. This format is illustrated in Figure 6. When using AIFH, SUE is identified using the Ethertype. This format is illustrated in Figure 7.

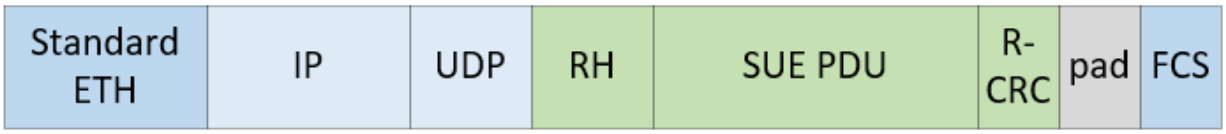


Figure 6 – Standard Ethernet Format

Figure 7 – AI Forwarding Header

3.3.2 AIFH

AI Forwarding Header is an optimized header which is aligned with existing Ethernet standards to minimize change while providing a much smaller network header. Based on the Structured Local Address Plan in IEEE 802.c-2017, two options are defined - 12B and 6B headers.

Using Administratively Assigned Identifier with IEEE compliant encoding, as illustrated below, the XPU identifiers are mapped to 16b values and populated in the DIP and SIP areas.

- M = 0/1 (multicast)
- X = 1 (locally assigned)
- Y = Z = 0 (AAI encoding per SLAP)

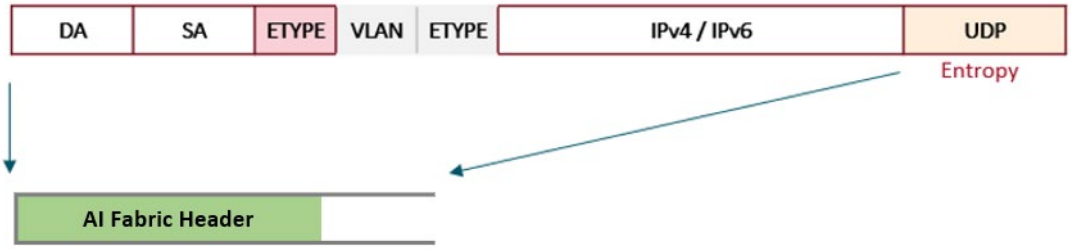


Figure 8 AI Fabric Header Compression

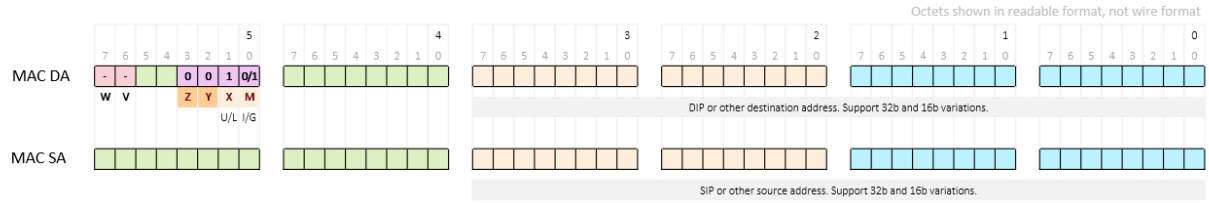


Figure 9 – AIFH Field Mapping

The resulting format for the AI Forwarding Header is illustrated in Figure 10.

Figure 10 – AIFH 6B Header

3.3.3 FEC

FEC options with smaller FEC block size enable lower latency. Options such as RS-272 can be selected over the standard IEEE RS-544. RS-272 corrects less errors with the tradeoff of improved latency.

At 400G and 800G speeds, the IEEE requires the interleaving of FEC blocks, the latency can be reduced by supporting non-interleaved or less interleaved modes of FEC blocks across lanes.

3.3.4 Link Level Retry (LLR)

LLR is used to improve the reliability beyond FEC, such that if an individual packet is corrupted it can be retransmitted between peer devices rather than waiting for the end points of the transaction to determine if the packet has been lost or corrupted. Link level retry regardless of FEC will allow for a packet to be retransmitted if it was dropped due to link level errors between two switches if FEC could not correct the packet.

3.3.5 Priority Flow Control

Support 8 classes of PFC flow control. Different PFC classes can be used to avoid deadlock between the XPUs and the switch, as well as to provide differentiated service..

3.3.6 Credit Based Flow Control (CBFC)

CBFC can provide a credit-based mechanism at the link level to control traffic between SUE and the switch instead of PFC. Different CBFC classes are used to avoid deadlock between the XPUs and the switch. The VC is used to map to the CBFC class.

3.3.7 Link Failures

The SUE provides link status for all the ethernet ports attached to it.

When operating in a two or four port configuration with unordered mode, SUE determines when a remote XPU is not reachable over a specific port / plane. When detected, SUE moves the associated work to another port.

SUE Operation

4.1 Memory Model and Semantics

SUE uses a shared memory model. Address translation, when required, is handled by the XPU outside of the SUE instance.

SUE is defined to support services for load/store – put, get, atomic. The source XPU issues a command and SUE delivers it, along with associated data, to the destination XPU and provides a completion indication to the source XPU.

SUE uses 1-sided memory semantics. Each command is done unilaterally from the source with no indication provided to the destination. Memory registration and management, such as protection domains, are handled outside of SUE. SUE provides a partition field which can be used for access control. Examples of similar technologies include PCIe and UALink.

This is in contrast to network semantics used by RDMA over Converged Ethernet (RoCE), InfiniBand and TCP/IP. These require establishment of a connection to the remote XPU.

4.2 SUE Processing Overview

Figure 11 illustrates the processing flow for SUE.

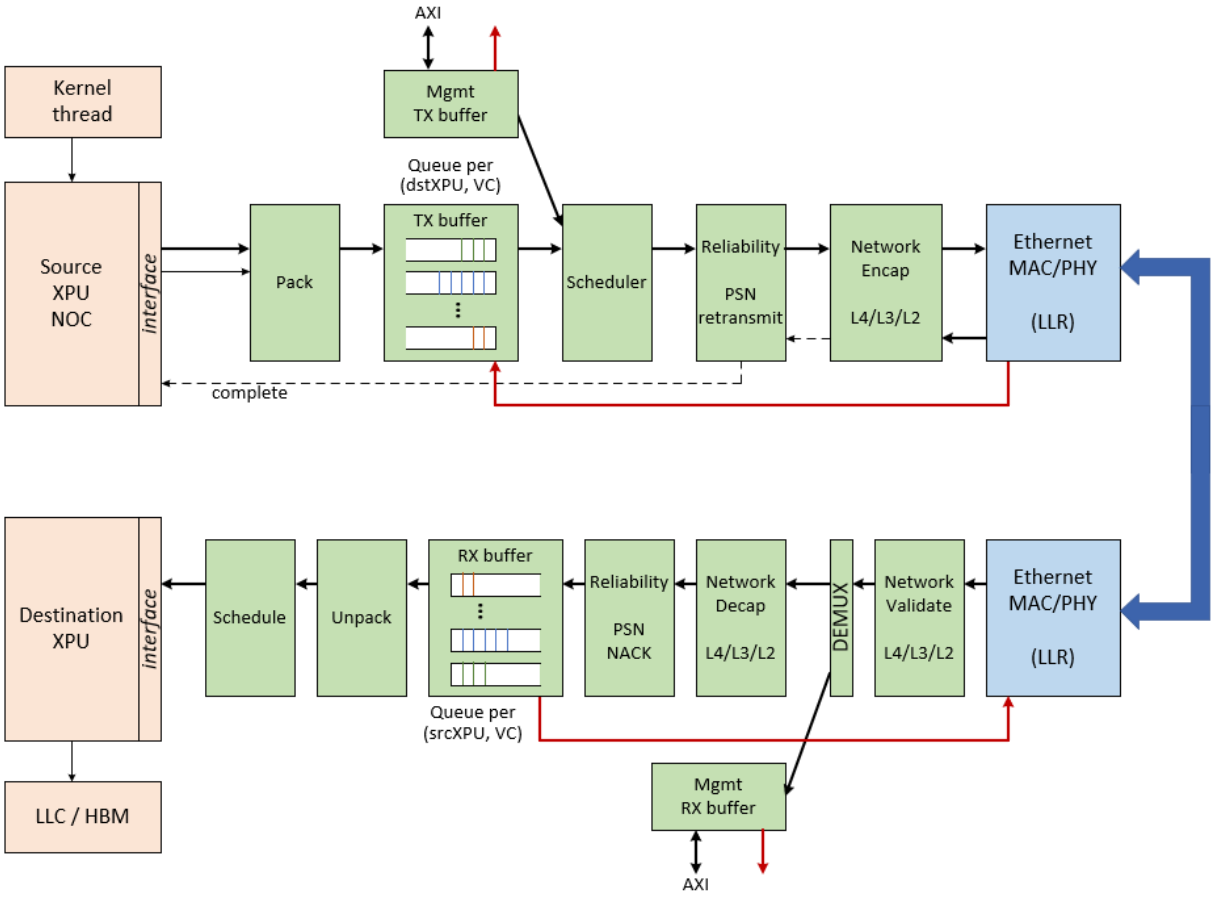


Figure 11 – SUE Processing Flow

The SUE processing flow follows the following steps:

1. XPU kernel issues a command to SUE over the XPU command interface
2. SUE accepts the operation and packs the control and data, if present, into the per destination transmit buffer. XPU can indicate the operation should be transmitted without waiting for additional operations. The buffer function determines when a queue should be serviced based on packing and flow control state (e.g. CBFC input) and indicates to the scheduler when a queue is eligible.
    a. First bytes of the control are the command type and length. The command type defines the length of the control information to be carried to the destination (in 2B units) and whether data is present. If data is present, the data length is defined.
    b. Based on these two lengths, the command (control and data) is packed.
3. Scheduler provides round robin across VCs and round robin across queues within the VC
    a. When a packet is transmitted via the control interface, it is merged into the data path.
4. Each packed group of commands is encapsulated with a reliability header which assigns a PSN based on {output port, destination XPU}. PSNs are assigned as monotonically increasing numbers which increase by 1 with each packet. If there is an ACK or NACK to be sent, this is added to the RH. The R-CRC is added over the RH and packed operations to create the SUE PDU.
    a. There is a connection between XPUs on a per physical port basis, the connection state is compact, consisting of the next expected PSN and a few control fields to manage ACKs & NACKs.
    b. SUE provides in-order delivery on each plane. Use of lossless traffic classes and LLR reduce the likelihood of packet loss dramatically, however it is still possible. If SUE determines a packet loss event occurred, it uses GoBackN to recover.
5. A network header is added to the SUE PDU. The destination XPU and VC are used to look up the required address fields.
6. The Ethernet packet is transmitted and arrives at the destination.
7. Ethernet validity checks are performed and any packets destined to the control interface are demultiplexed.

8. The Ethernet header is verified and then the RH header checked. If the expected PSN for that source XPU is received, the PDU is put in the RX buffer and scheduled to be sent to the XPU NOC.
    a. If an unexpected PSN is received, a NACK is sent to the source and any arriving packets from the source are dropped until the expected PSN arrives.
    b. When the expected PSN is received, the packet is ACK'd.
9. RX buffer received commands which are then scheduled, unpacked and sent to the XPU NOC.

4.3 Load Balancing

It is presumed that there will be multiple instances of SUE connected to an XPU. Load balancing across the SUE instances can be handled in software or by including a hardware block in the XPU which distributes the operations across the SUE modules. Each SUE instance provides congestion state to enable this option. Refer to Figure 12 below.

Within the SUE, load balancing may be enabled when using two or more ports. In this case, SUE will dynamically assign each packed group of operations to a port based on available bandwidth.

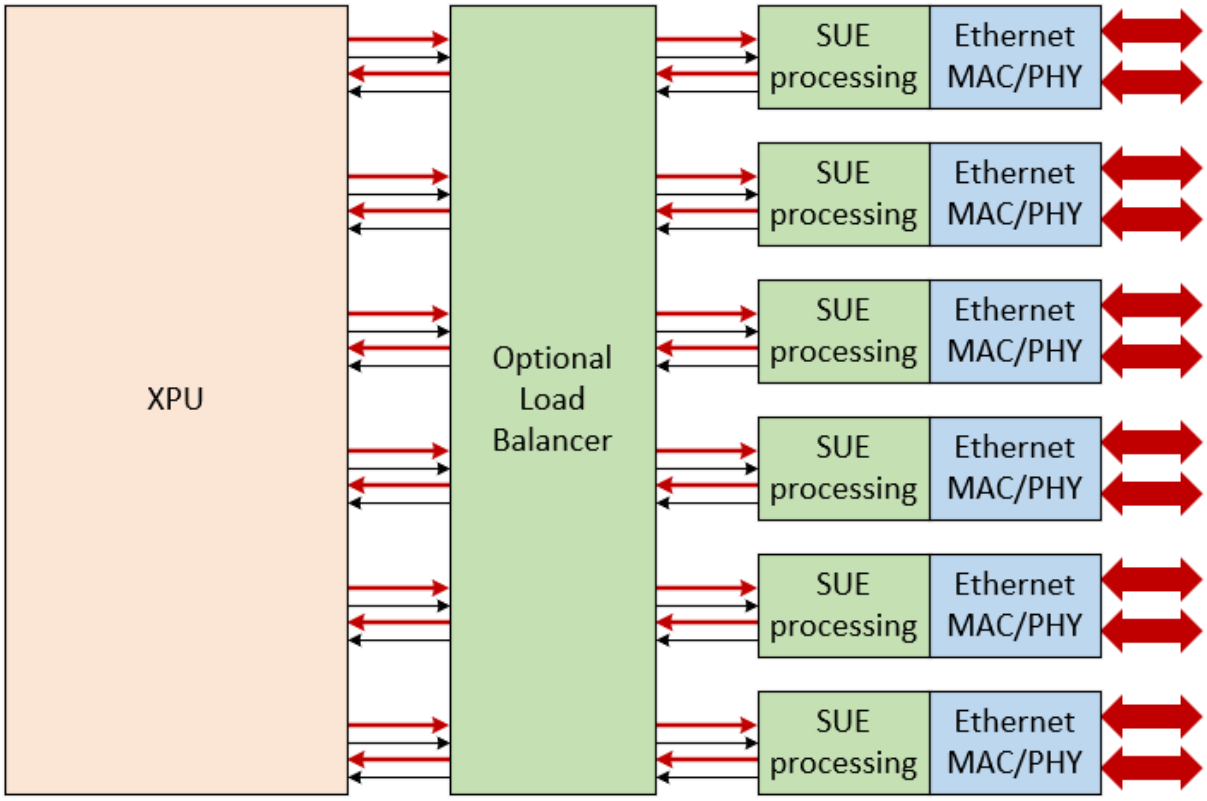
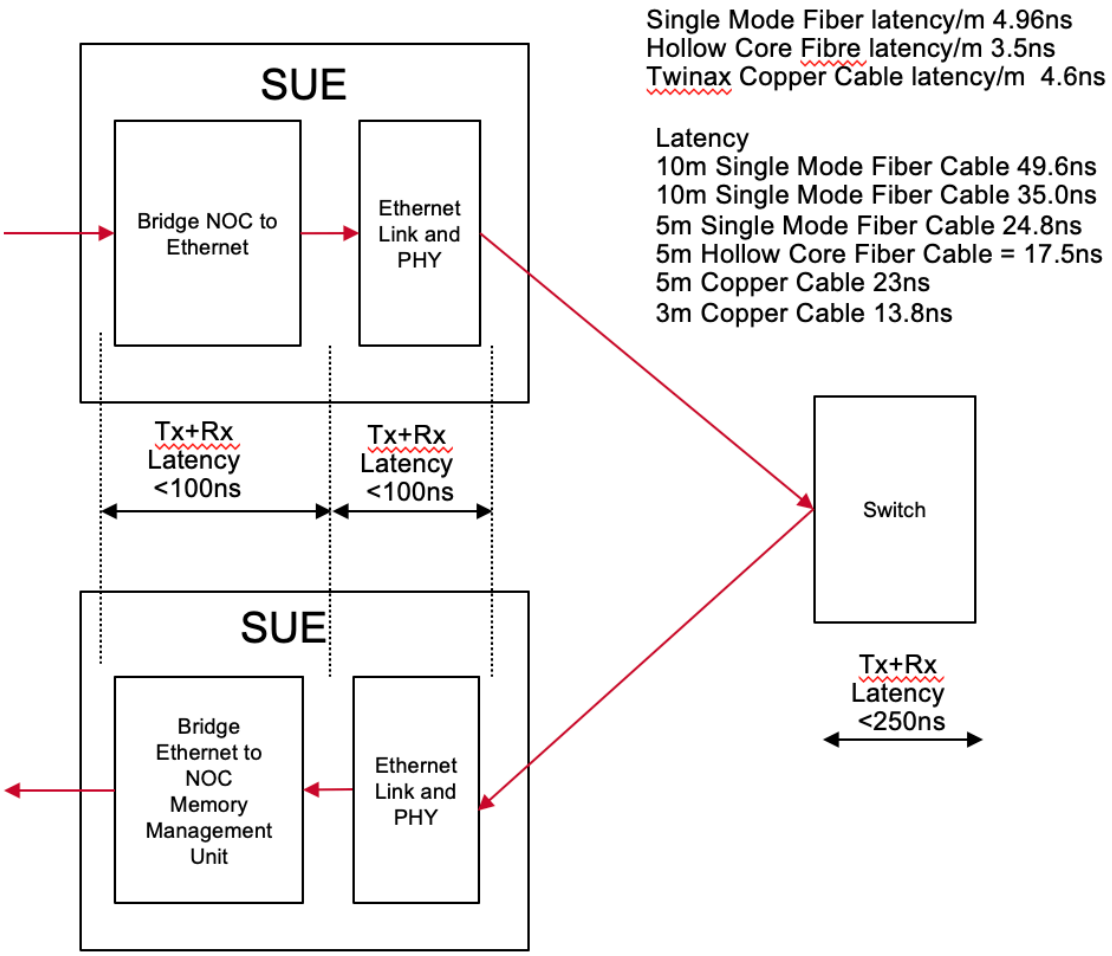


Figure 12 – Example XPU Load Balancer

Appendix A: Latency Budget

A.1 SUE E2E One Way Latency Budget

The throughput of a multi-XPU shared memory system is determined by the latency of the XPU communication. Therefore, minimizing the latency of the interconnect is of great importance. The length and type of cabling, switch latency are all critical components. Figure 13 illustrates the latency of the various components in the SUE system.



Total Latency Budget
10m Single Mode Fiber - 100 + 100 + 49.6 + 49.6 + 250 = 549.2ns
10m Hollow Core Fiber - 100 + 100 + 35.0 + 35.0 + 250 = 520ns

5m Single Mode Fiber - 100 + 100 + 24.8 + 24.8 + 250 = 499.6ns
5m Hollow Core Fiber - 100 + 100 + 17.5 + 17.5 + 250 = 496ns
5m Twinax Copper - 100 + 100 + 23 + 23 + 250 =

3m Twinax Copper - 100 + 100 + 13.8 + 13.8 + 250 =477.6 ns

Figure 13 – SUE End-to-End One-Way Latency Budget

Appendix B: Glossary

Acronym / Term	Definition
SUE	Scale Up Ethernet
CBFC	Credit base flow control
LLR	Link Level Retry
NIC	Network Interface Card
NOC	Network on chip, referring to interconnect within XPU
QoS	Quality of service
XPU	Generic term for ML/AI/HPC accelerator including GPUs, CPUs, etc.

Revision History

Scale-Ethernet-RM101; May 1, 2025

Deleted duplicate image on page 20.

Scale-Ethernet-RM100; April 28, 2025

Initial document version.


BROADCOM®