JUNIPER NETWORKS | Engineering Simplicity

DAY ONE GREEN: CONNECTING MULTI-TERABIT PACKET PROCESSING ASICS USING HIGH THROUGHPUT MULTI-TERABIT FABRIC ASICS

Juniper ASIC architecture ensures that fabric architecture with an optimized link count can consume less power.

By Harshad Agashe

Day One Green

Connecting Multi-Terabit Packet Processing ASICs Using High Throughput Multi-Terabit Fabric ASICs

To build multi-terabit routers, Juniper builds systems using multiple packet forwarding engine (PFE) ASICs. A PFE ASIC supports a few tera-bits of packet processing capabilities. They are interconnected using cell-based fabric interconnect. Figure 1 illustrates a typical Juniper system. The PFE in Figure 1 could either be Juniper Express or TRIO silicon.

A cell-based interconnect approach provides simultaneous connections to all PFEs without any restriction on flow rates. Given the scale of the bandwidth (BW) requirements, a PFE internetwork is composed of multiple fabric ASICs with multi-gigabit links.

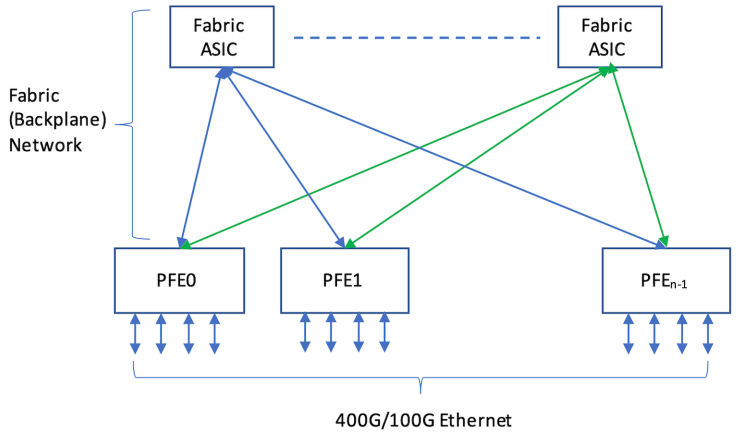

Figure 1 *Fabric Network Example*

A typical chassis-based system is composed of line cards and fabric cards. Line cards (LC) host PFE ASICs while fabric cards host fabric ASICs. Line cards and fabric cards are connected using high speed serial links. A link segment consists of SerDes on the ASIC, board routing, and connectors and repeaters (for managing signal integrity).

The goal of a fabric network is to provide non-blocking interconnect with almost 90% utilization. Fabric networks should also provide consistent latency behavior with a predictable performance for all packet sizes.

Fabric networks consists of data and protocol traffic along with congestion control mechanisms. Protocol engines are implemented in hardware engines which maintain states per flow.

High-speed links used for fabric networks consume power in PFE ASICs and in fabric ASICs, so a fabric architecture with optimized link count would consume less power over an architecture requiring more links. This implies that protocol traffic needs to ensure performance is achieved along with the power of the system within specified limits. Juniper ASIC architecture optimizes protocols to achieve these goals.

Connecting Multi-terabit Packet Processing ASICs Using High Throughput Multi-terabit Fabric ASICs

A multi-terabit system would have multiple PFE and fabric devices. As PFEs are interconnected using multiple fabric devices, PFEs would have to utilize multiple paths simultaneously to achieve performance.

There are two fundamental ways of building fabric-based systems: using cells across the fabric or using packets. A packet-based system may seem obvious since every Ethernet interface of a PFE would be packet-based. However, packets pose challenges in achieving high fabric link utilization.

The Problem with Packets

There are two fundamental approaches to forwarding packets across a Clos fabric: *flow-based* and *spray*.

For flow-based forwarding across a fabric, the flow-identifying header fields of each packet are hashed, arriving at a flow-identifying value. Many actual flows may hash to the same value. This value is then used to select one of the many equivalent paths across the fabric. Thus, all the packets belonging to a flow are assured of following the same path across the fabric and remaining in order. See Figure 2.

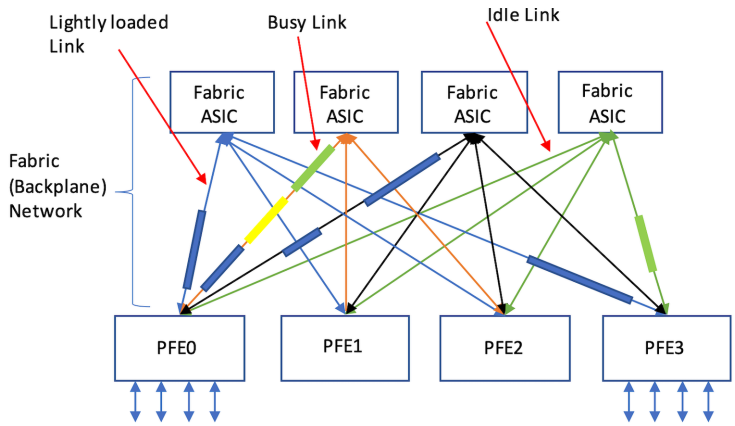

Figure 2 *Flow-based Packet Distribution*

The problem with this approach is twofold. First, hashing is imperfect, making the distribution of flows to flow-identifying hash values non-uniform. Second, not all flows are created equal: some flows are busy while others are quiet. The hash function has no way of determining how busy a particular flow might be. The upshot of these weaknesses is that some fabric links may reach saturation while others are much less busy. Once one of the links becomes busy, the system has effectively reached its total BW limit.

If Ethernet packets are distributed across fabric using whichever link currently has the shortest queue, the packets within a particular flow can use separate paths across the fabric. This can re-order packets within a flow and that means that packets need to include a sequence number of some kind so that their proper order can be restored. Fabric devices do the store-and-forward and packet size can vary from 64 bytes to 16K bytes. This takes time during which the packet may become mis-ordered and can be quite large, making the re-ordering context within the egress PFE large, complex, and expensive in terms of silicon real estate.

Behavior of the network would be dependent on how flows arrive and could become unpredictable. Fabric utilization can go down to even 60%.

Cell-based Fabric

A cell-based fabric splits packets into almost-fixed cells and *sprays* them over all available links. PFE logic will make sure all links are equally utilized for all egress PFEs. This nullifies the problem of saturating a few links over others as in a packet fabric. Cells are stamped with sequence numbers and a protocol engine makes sure oversubscription is handled by granting line rate worth traffic. This ensures fabric devices do not get oversubscribed. Fabric ASICs handle almost-same cell sizes so

its design is simple compared to any Ethernet packet switch. The look up engine is simple since its only switching cells from source to destination. This helps in making fabric ASIC latency behavior predictable and allows a reorder engine design with reasonable complexity and storage.

Overall, a fabric ASIC will consume much less power than a similar capacity Ethernet switch. Cell fabrics can achieve a utilization of almost 90% and they will behave very predictably with consistent latency characteristics.

Cell-based Fabric Components

Let's look more closely at the cell-based PFE internetwork.

Figure 3 shows a system using N PFEs. These PFEs are connected using a NXN crossbar built using fabric ASICs. PFEs communicate over the fabric using protocol engines. Fabric protocol engines enforce non-blocking behavior by exchanging *Request-Grant* messages. This allows them to compute load on egress PFEs. Engines maintain states for multiple PFEs, or even flows within the PFEs, allowing them to avoid head of line blocking. For example, if PFEs were treated as S streams, then engines maintain state for N*S streams.

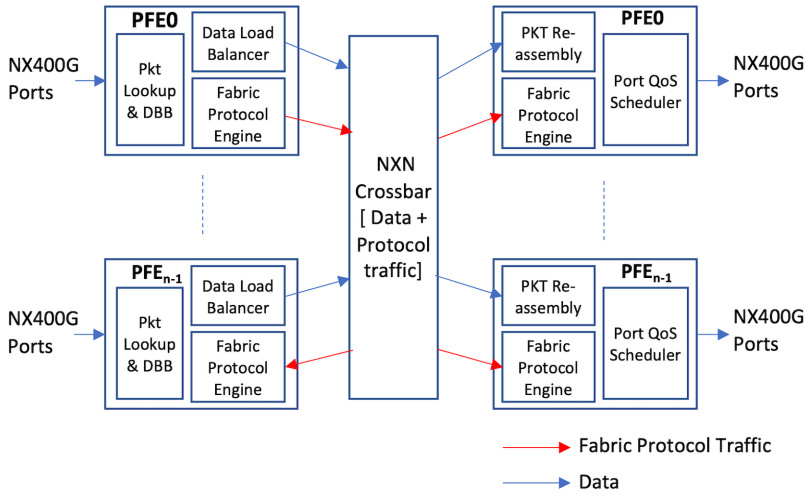

Figure 3 *Cell-based PFE Internetwork*

As stated previously the data load balancer will convert packets to cells and spray them over available links by making sure that BW is equally utilized for every egress PFE. Cell reorder blocks put cells coming from various paths back in order and then stitch the packets.

Figure 3's NXN crossbar is implemented using fabric ASICs. Depending on the size of the system (therefore, how many PFEs need to connect), multiple fabric ASICs are used. Each fabric ASIC works internally as a switch which handles cells of almost-fixed size. Look-up engines in fabric ASICs are simple compared to any Ethernet switch and can be statically programmed. Links coming from PFEs are treated as single links allowing large systems to be built.

Fabric Link Utilization and System Power

To achieve full non-blocking behavior, protocol engines need to communicate with each other and that will require BW over and above actual port traffic. BW consumed by protocol traffic will be decided by cell size, cell headers, and request/grant message sizes and their frequency.

A typical fabric network protocol makes sure none of the paths are oversubscribed at any given time. For this to happen ingress-egress communicates using request, grant messages. A request-grant data protocol will have ingress PFEs sending requests to egress. Egress will arbitrate across requests and send back grants and data will flow after grants are received. Egress will grant as per actual flow rate of stream. This makes sure data cells are not oversubscribing any paths.

Data cells would have headers that carry output queue, sequence number, source address, and other control information. Request and grant headers would also carry similar information.

So how much BW to be allocated for request and grant traffic and data cell header will be decided by how large a system is built. System power is decided by link count, since SerDes, repeater, and fabric switching power depends on this.

Let's assume data cell size is fixed at 128B and the request and grant header are 8Bytes while each data cell header is also 8Bytes. This implies each cell size would be 128B + 24B = 152B and that would give each cell an efficiency of 128/152 = 84.211%.

Data payload in data cells can be in terms of 4B/8B/16B blocks which can cause wastage when packet sizes don't fall on exact boundaries. In some cases, assuming an 8B alignment, data cell efficiency could be as low as (128-7)/152 = 79.6%. This implies total fabric BW utilization goes down to ~80%.

If cell size increased to 256B and keeping the rest of factors the same, utilization would grow to (256-7)/(256+24) = 88.9%. This implies such system would need ~11% extra BW compared to 20% extra BW.

A typical 50Gbps SerDes consumes approximately 0.5W in 7 nm technology. A system having 4000 fabric links would take 4000 * 0.5 * 2 (SerDes at each end of link) = 4000 W, so a reduction of 10% in links would provide 400W in savings.

A 4000-fabric link system would need more fabric chips over 3600 links. This can roughly save additional ~200W power. A 10% link reduction will result in ~600W power reduction. Power reduction helps system design (less resources for cooling, power delivery, etc.) and reduces running costs for customers.

The following blog covers the cooling system used by the PTX10008 system. This is an 8-slot chassis with each line card supporting 14.4 Tbps WAN BW. As mentioned in the blog, the number of fans needed to cool down to avoid any component shutting down is significant. Any increase in power further makes cooling costly: https://www.juniper.net/documentation/us/en/hardware/ptx10008/topics/topic-map/ptx10008-cooling-system.html.

How to Optimize Fabric Protocol BW

There are two major components of protocol traffic: data cell overheads and requests and grant messages.

Data cell overhead can be reduced by choosing the data cell size. The size needs to be optimally sized so that the date header overheads are low. A very large data cell size will have data header overhead low but slow flows may not be able to use it optimally. An approach of slightly variable cell size helps in optimizing such cases.

For request and grant headers reducing its size helps (for example a 12B request and grant header versus 16B) and major improvement can be gained by reducing request and grant traffic. Request BW can be reduced by having a request for group of cells (1 to N) and protocol engines can dynamically control request size. This is a very powerful tool which allows a fixed allocation for request and grant BW but needs intelligence in the protocol engine to use that as needed.

Summary

Juniper Networks fabric architecture can use all of these design elements to optimize the fabric BW required. Systems built with this approach consume less power, reduce total cost of system, and reduce operating costs. Speak to your Juniper Networks' account manager or Professional Services rep about fabric architecture optimization.

3100045-001-EN-Nov.2022