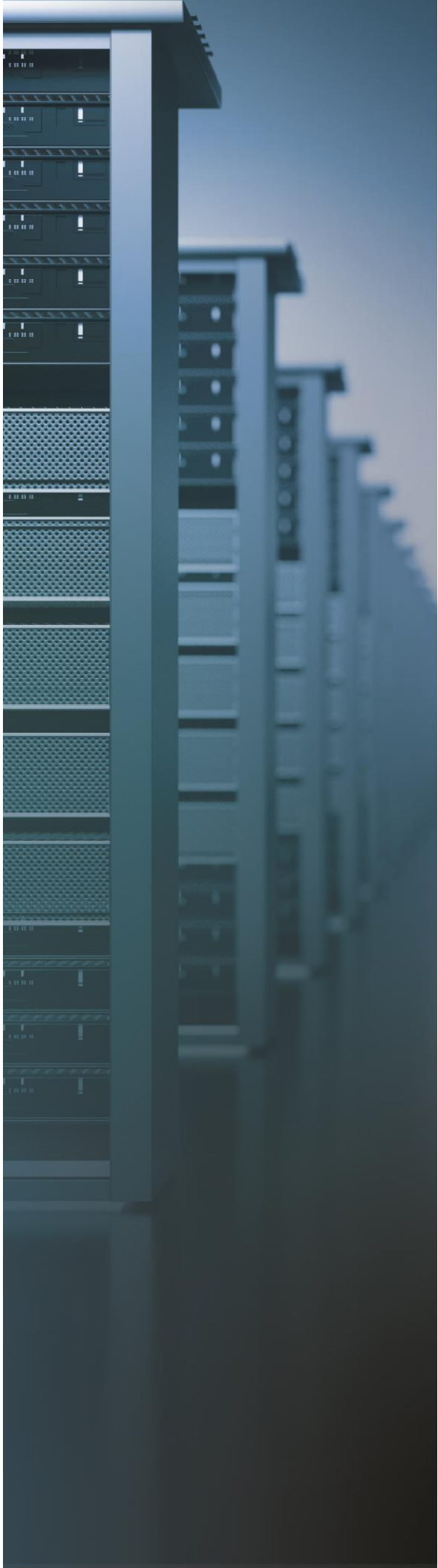
WHITE PAPER

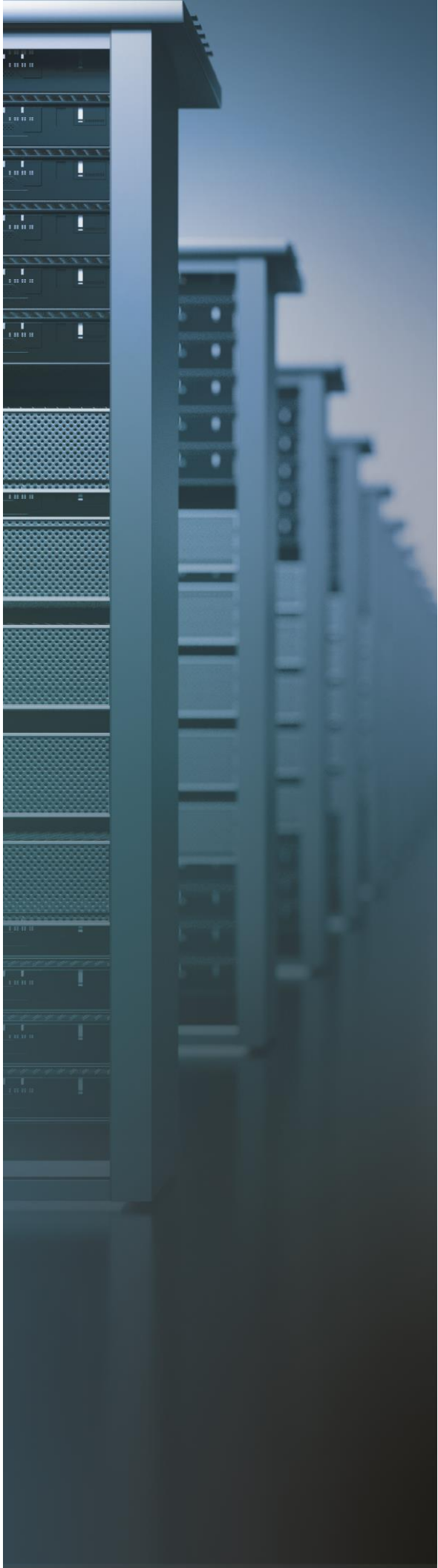
Stepping into a New Epoch

X1 for HPC and AI

DECEMBER, 2020

Table of Contents





Abstract

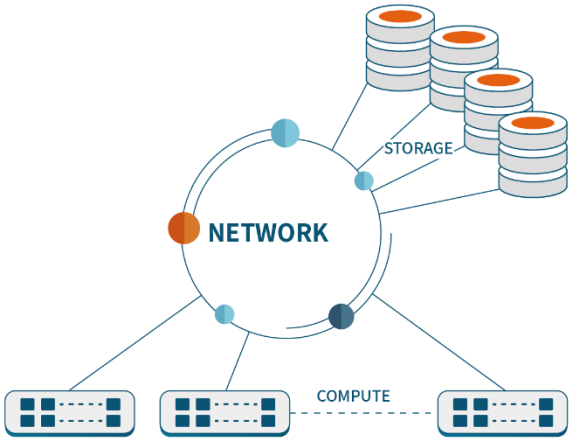
This paper discusses parallel distributed computing in high performance computing (HPC) and artificial intelligence (AI) clusters. It focuses on the network portion of a cluster. The paper lists the challenges that parallel computing places on the network. Network infrastructure plays a critical role in overall high-performance computing as well as AI cluster performance and system scalability.

Xsight Labs introduces the X1 family of fully programmable switching ASIC devices, optimized for AI and HPC cluster interconnect. It delivers best-in-class 25.6 Tbps full-duplex throughput, (with a robust 12.8 Tbps variation supported), ultra-low power, low latency, and revolutionary traffic management. It incorporates highly sophisticated mechanisms to reduce flow completion time (FCT) and avoid parallel slowdown. Xsight's Intelligent Queueing (X-IQ™) and Application Optimized Switching enable distributed parallel computing systems to scale.

High Performance Computing

High performance computing (HPC) evolved over multiple stages, beginning from very expensive, very large, and powerful supercomputers based on powerful processors. To overcome the limits of a single processor's capabilities, multiple monolithic processors interconnected by purposely designed interfaces were introduced into supercomputers, followed by multi-core interconnected processors. Economics and technological progress enabled the usage of commodity processors and components to create a supercomputer. The evolution of networking and storage technologies, economics, and exponential growth in compute and storage needs has created a need for scalability that supercomputers cannot address. Distributed computing lead to widely deployed distributed compute and storage models. Typically, the compute chassis' (each containing a number of commute nodes) and distributed storage coupled with remote direct memory access (RDMA) are connected over an Ethernet network to build compute clusters that are capable of simultaneously processing huge amounts of data over a large number of compute nodes.

Distributed compute infrastructure alone is not enough to address HPC needs. Compute nodes must work on a task and access storage in parallel. Parallelism in HPC clusters is enabled by compute nodes communicating over a message passing interface (MPI) and accessing distributed storage over the Ethernet using RDMA or RoCE (RDMA over converged Ethernet). For these reasons, the network is a critical resource in parallel distributed computing.



Distributed Computing System

The Rise of Artificial Intelligence

The world has stepped into a data driven era where data has become ubiquitous. Data creation, processing, and sharing is easier than ever. Enormous amounts of data are being created and stored in modern data centers. Data is extremely valuable for making decisions as well as for advancing business and technology. However, data sets still need to be processed and prepared for use. Moreover, the exploding amount of data makes its processing and

analysis by human or traditionally focused programs an impossible task. Artificial intelligence (AI) can be used to solve this problem. AI techniques allow systems to learn, plan, reason, and solve problems based on a given data set without being specifically programmed to perform well-defined tasks. Since its inception in the 1950's by Minsky and McCarthy, AI evolved from being an esoteric academic exercise to being vastly deployed everywhere. AI concepts and services are extensively applied on many aspects of data-driven life. For example, medical decisions, business logic, targeted ad content language processing, and autonomous vehicles are just part of a long list of AI use cases.

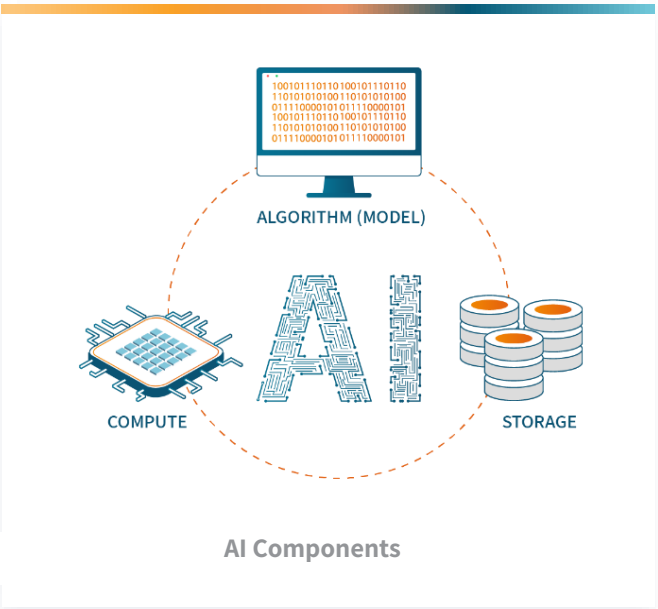
AI Components

Recent dramatic technological breakthroughs have made AI a reality and have allowed AI to become a major part of modern data center compute and storage algorithms.

Mathematical and statistical algorithms that have evolved over last two centuries are being used to enable AI for things such as linear regression, logistic regression, LDA, support vector machines, KNN, learning vector quantization, and more.

Compute nodes and data storage have been able to grow tremendously due to powerful AI-oriented accelerators, such as Intel/Habana's Gaudi, enabling large matrix computation, ultra-fast IOs and built-in storage and network virtualization.

Compute and storage nodes deliver exponential growth in their respective performance and capacity. Efficient and accurate AI requires huge amounts of specific computations (think: matrix math) within a small timeframe. Traditional CPU architectures were not oriented to perform such computations. Distributed services and storage models applied via AI intrinsically address these issues.



AI Components

Distributed AI Computing

Data sets for AI models are extremely large, for example, MRI scans are many Terabytes large, and learning processes may use tens or hundreds of thousands of images. AI models used to process such data sets require amounts of compute that cannot be achieved by a single AI accelerator or by a single AI accelerator chassis. The solution is distributed compute and storage using distributed parallel compute HPC principals applied on AI as well. AI compute often runs on HPC infrastructure.

AI Training Stages

AI model training is a multi-stage and iterative process. First, the data must be collected. The data needs to be as diverse as possible, unbiased and abundant in order to ensure model accuracy. The next step is to prepare and clean the data set. This stage usually includes data reformatting, cleaning and normalization based on a model's input requirements.

Parallelism

There are two main types of computing parallelism: Data Parallelism and Task Parallelism (aka Model Parallelism for AI). In the Data Parallel approach, data is broken down into non-overlapping batches. Each data batch is fed into a compute node. All compute nodes are loaded with identical models (AI) or tasks (HPC).

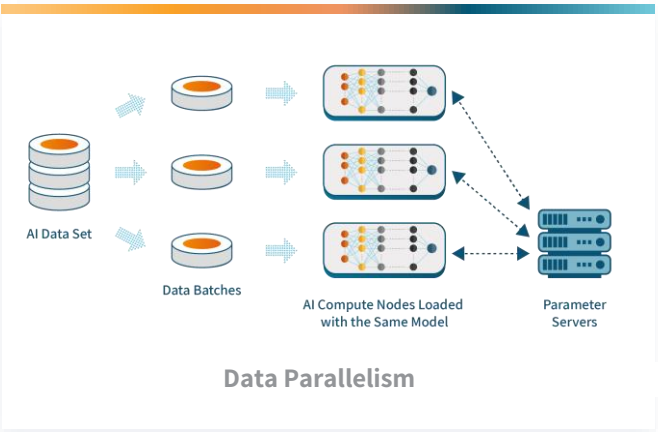
AI Stage-by-Stage

The primary stage is training. Prepared data is modeled to learn until it reaches a pre-defined accuracy. Next, the model is evaluated using test data sets that differ from training data. If evaluation results produce expected model accuracy, the model's hyperparameters are tuned and the training and evaluation process is repeated. The goal at this stage is to improve model performance. Once performance and accuracy are satisfactory, the model is deployed for inference, in other words, testing the model with real-world data. Models are constantly trained in order to achieve higher accuracy and to absorb new data.

Let's take an AI model training as an example. Parallelism is often applied on AI algorithms using stochastic gradient descent. The algorithm's goal here is to estimate parameters based on a specified data set.

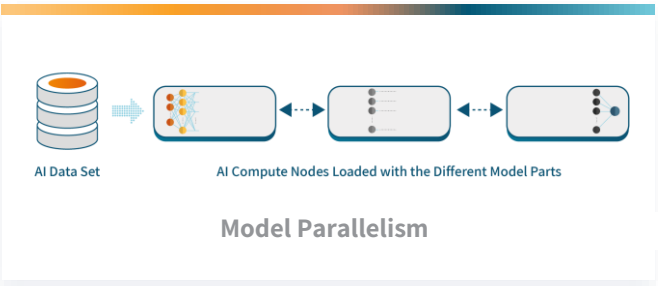
Each node computes parameters based on its local data set. Output parameters from all compute nodes are synchronized to define model-wide parameters set. This iterative process of parameters update continues until the desired model accuracy is achieved. The main advantage of using the Data Parallelism method is that separating data into batches is relatively simple and it works for any AI cluster/model architecture. However,

synchronization creates a bottleneck when a large number of parameters are used.



Data Parallelism

In the Model Parallel approach, the AI model is split between compute nodes. The model is broken into layers. Each model layer is further broken down and loaded into a set of parallel compute nodes, thus creating a number of AI model layers, each consisting of a set of compute nodes. Nodes that are loaded with input AI model layers are fed with full data sets. Once all nodes within same layer finish their compute iteration and are synchronized, their output is passed on to next AI model layer that is represented by a set of computation nodes.



Model Parallelism

The Model Parallelism works well with a large number of parameters. It significantly reduces memory footprint in AI accelerators. However, splitting the model, in particular within a given layer, is not trivial and may affect model behavior and accuracy. Worker compute nodes communicate between themselves, and require synchronization to complete. Communication and synchronization delays may, however, affect overall training speed.

There are additional approaches to parallelism that combine data and model parallelism: pipeline and hybrid parallelism.

Network Challenges in AI and HPC Clusters

Networks play a critical role in any distributed compute and storage architecture. Network infrastructure bandwidth, latency, congestion management, reliability, as well as many other factors greatly affect distributed system performance. A malfunctioning network can create a cascading effect causing prolonged severe performance degradations. Distributed AI and HPC systems will amplify network infrastructure criticality by their iterative, heavy, frequent, and synchronous communication. A number of acute network infrastructure attributes must be defined, that are essential for the enhanced performance of distributed AI and HPC computing performance.

Network Latency

Breakthroughs in both compute (AI accelerators and CPUs) and storage technologies pushed the limits of network infrastructure. Storage Seek time and Sustained Throughput parameters show significant evolution as follows: Seek time dove from 2-5 ms for HDD, through 0.2 ms for SSD, all the way to 0.02 ms for NVMe. Sustained throughput grew from 200 MBps for HDD to 3 GBps for NVMe storage technologies[1].

At the same time, compute, with rise of specialized AI accelerators, delivers extremely low latency and throughput capabilities.

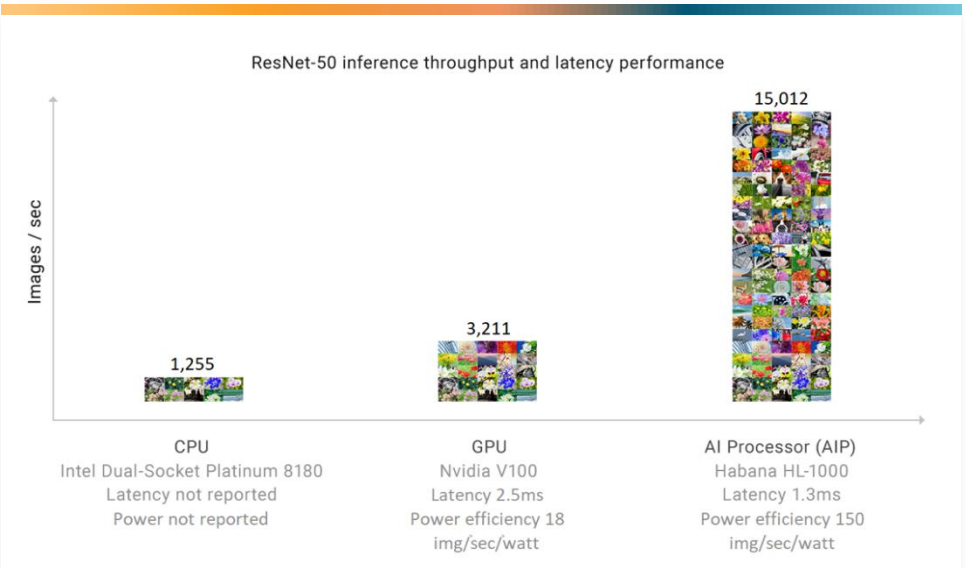
The network's portion in overall distributed compute system latency is small (about 10%) for HDD storage and traditional CPUs. Migration to AI distributed computing with ultra-fast distributed storage and specialized AI accelerators or HPC pods make network latency a major system-wide latency factor. The network must evolve and deliver fast, lower latency infrastructure. Network latency can be broken down to 2 types: static and dynamic latency.

STATIC LATENCY

Static latency is latency associated with switch forwarding latency and optical/electrical transmission latency. This latency is derived from switch ability and transmission distance. Switch latency in a distributed parallel computing system should be as small as possible to enable fast communication and compute progress. Static latency is, however, only a part of network latency.

DYNAMIC LATENCY

Dynamic latency is associated with queuing delay, flow completion time, tail latency, packet drop, burst



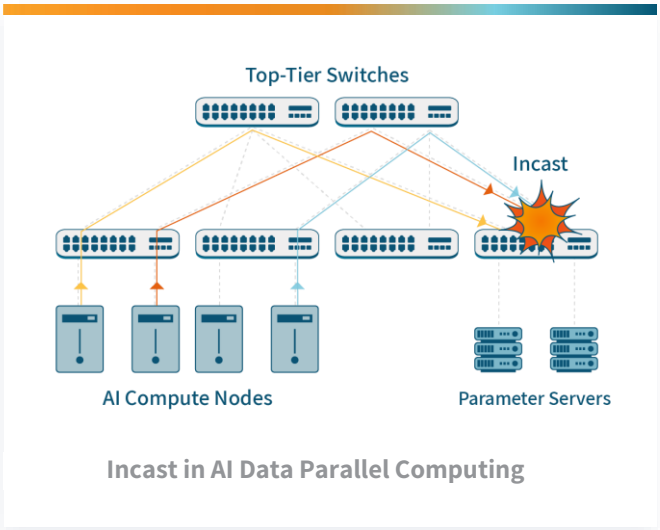
AI Computing Technology Performance Improvement – Taken from [2]

and microbursts. Dynamic latency is the main contributor to overall network latency and a key factor for distributed computing.

AI and HPC systems rely on parallelism and concurrency. Parallelism for those services requires frequent spikes in data transferred for synchronization and parameter exchange. Any delay in this process causes extremely valuable compute resources to stall and thus degrades overall system performance greatly. Network packet loss is responsible for the largest network latency spikes since it requires TCP retransmission which stalls compute resources and places a huge load on the CPUs managing the system. Furthermore, remote storage access using TCP based protocol, iWARP, is not scalable for large systems due to its large number of connections. Therefore, large distributed systems rely on UDP based RoCEv2 (RDMA over Converged Ethernet). While UDP is scalable and doesn't carry TCP management overhead, it doesn't provide built-in reliability and therefore requires a lossless network. Lossless switching for RoCEv2 is implemented by Priority Flow Control (PFC) and Explicit Congestion Notification (ECN).

Incast and Bursts

AI and HPC distributed systems put pressure on network infrastructure by creating frequent incast (many to one) traffic patterns causing congestion that incurs significant latency spikes. Incast may lead switch queues to fill up and trigger a PFC response which in its turn slows down the network and causes other system performance degradation events. The highly concurrent nature of AI and HPC systems along with their need for synchronization are the main factors contributing to incast traffic. AI data's parallel computing has a built-in incast problem as parameters from different data batches must be propagated into the parameter servers, almost concurrently, for each compute iteration



Incast in AI Data Parallel Computing

AI model parallelism and HPC cluster parallel compute rely on large numbers of messages communicating between the compute nodes. Sub-jobs running on different compute nodes communicate with each other in order to complete the task. Similarly, compute nodes that represent an AI model layer (in AI model parallelism) use communications in order to achieve task completion. Large numbers of compute nodes communicate concurrently in distributed systems. Therefore, micro-bursts that fill network node queues are created, and this leads to increased queueing time and thereby increases communication latency.

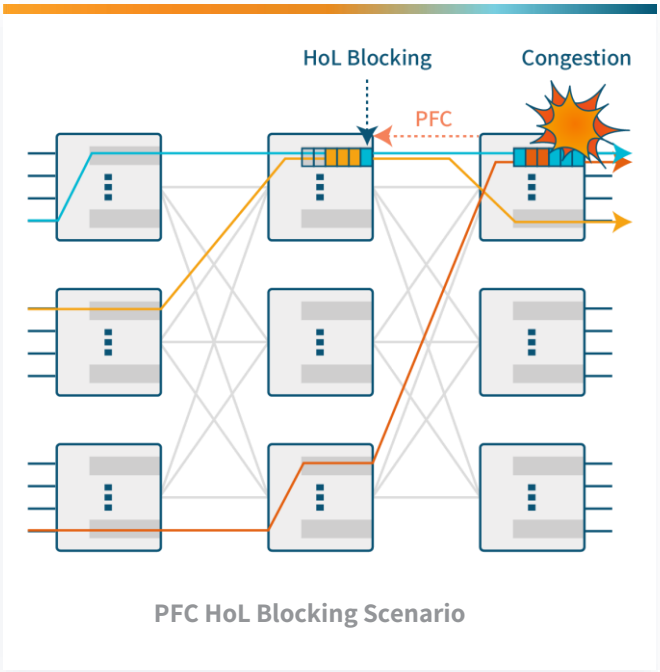
Flow Completion Time (FCT) and Tail Latency

While bursts cause queueing delays and increase dynamic latency, they also lead to an increase in Flow Completion Time (FCT) causing some flows to lag behind others. AI model parallel computing requires synchronization within the AI model layer represented by a set of AI processors. Synchronization is a frequent, repetitive, and concurrent event often occurring during parallel HPC computing as some jobs are dependent on result of others. Therefore, the increased FCT creates a system wide phenomena known as tail latency. Tail latency is the latency of the fraction of the flows that take the longest to complete. Tail latency is critical to distributed parallel computing since it has a cascading effect and degrades the performance of the entire system. HPC parallel compute stalls while jobs dependent on output from other jobs idle, or when AI training stalls and doesn't progress to the next layer until all nodes are synchronized. Network infrastructure must mitigate FCT increase in order to decrease tail latency.

PFC and Head of Line (HoL) Blocking

RoCEv2 requires a lossless network. PFC is the traditional hop-to-hop protocol that allows one network node (a receiver) to pause its peer (a transmitter) once a certain threshold is reached on the receiver's queues. It pauses a single priority and not the entire interface. However, there are a number of problems in this mechanism that are amplified in parallel computing systems such as HPC and AI.

In a multi-tier network PFC has a built-in Head of Line Blocking (HoL) problem. PFC supports only eight priorities, and in many switches deployed today, only up to four are practically supported due to buffer management, size, and architecture limitations. The granularity of eight priorities is far from ideal. Many



PFC HoL Blocking Scenario

flows are mapped to a single PFC priority because there are a small number of queues compared to the number of flows. HoL blocking may also occur when congestion created by parallel computing bursts, has a flow destined to a non-congested path is blocked by another flow scheduled first and is destined to a congested path. This phenomenon is more severe in AI and HPC clusters due to their burstiness and large number of flows. In particular, in multi-tenant deployments this problem affects business operations since one tenant's traffic flow may cause

delays in the traffic flow of the other tenants due to an insufficient amount of queues.

In high scale deployments, PFC may cause the network to stall because of PFC deadlock [3].

Parallel Slowdown

The highly concurrent nature of AI and HPC compute, PFC's lack of granularity, HoL blocking, and possible deadlocks can all lead to a significant FCT increase for some flows, and therefore increase tail latency. This increase in tail latency in distributed parallel computing systems decreases overall system performance since compute resources are stalled due to delayed communications from earlier compute stages. This may lead to a phenomenon known as "parallel slowdown", a condition where the addition of compute nodes doesn't increase or in some cases even decreases total compute completion time due to communication overhead.

Network Diameter and Radix

Parallel compute can be used to solve very complicated tasks in a more reasonable amount of time by simply adding more compute nodes to increase compute performance. However, in a distributed system, the network is a bottleneck.

The ability to add more compute nodes depends on the switch radix and the overall network diameter. Using traditional, small radix switches limits number of compute nodes that can be connected to a single switch, thus driving a requirement to add more and more switches only in order to add compute nodes. In turn, more switches lead to more complicated networks with more network hops that each flow must travel through, thus increasing the network diameter. An increase in the network diameter can cause an upsurge in both static and dynamic latency and FCT, altogether leading to a parallel slowdown phenomenon.

Network Infrastructure Challenges Summary

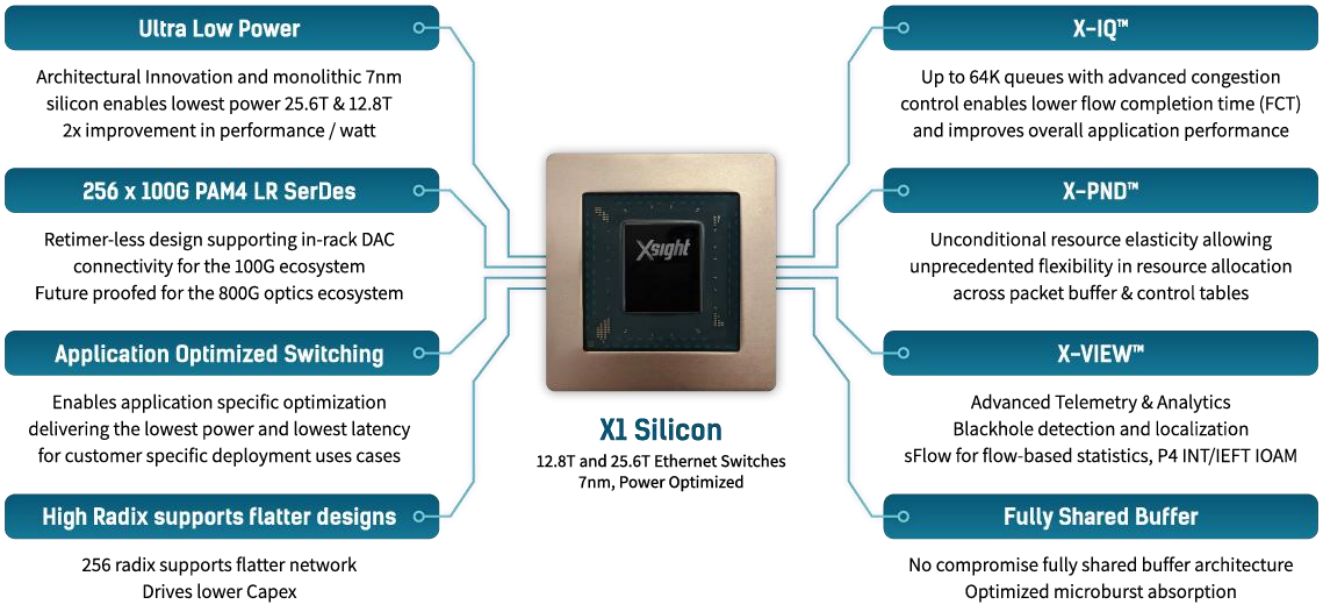
Distributed parallel computing, such as HPC and AI, are extremely compute and storage hungry systems. While parallelism unleashes the enormous potential of these applications, it at the same time, places tremendous stress on network operation. Network infrastructure must evolve beyond traditional architecture in order to address HPC and AI clusters needs. Switches must focus on decreasing FCT, managing their smart buffers (increasing the on-switch buffer doesn't singly mitigate the problem as it leads to an increase in queue time), large radix and throughput.

X1 Empowers AI and HPC Clusters

X1 is a family of fully programmable switching ASIC devices, optimized for AI and HPC cluster interconnect. It delivers best in class 25.6 Tbps (with a 12.8 Tbps variation also supported) full-duplex throughput, at ultra-low power and low latency with revolutionary traffic management mechanisms to reduce FCT and avoid parallel slowdown, as well as

Application Optimized Switching to empower evolution in compute and storage and to enable scalable distributed parallel computing systems.

existing infrastructure with support for 400G and 800G modules. It supports flexible port configurations using 100, 200, and 400 GbE speeds for port densities such as 256 x 100 GbE, 128 x 200 GbE or



X1 Architecture Highlights

100G LR SerDes

The X1 family of devices incorporates industry leading 100G LR PAM4 SerDes, enabling the design of in-rack DAC connectivity for the 100G ecosystem without the need for retimers while future-proofed for 800G optics. It enables in-rack passive copper attach and minimizes optical connectivity.

The X1 100G PAM4 and 50G NRZ LR SerDes enables interoperability and seamless integration into

64 x 100 GbE.

The devices deliver future-proofed systems by enabling the design of high scale systems with existing infrastructure that are ready to transition to 800G connectivity without the need to upgrade the network infrastructure.

Ultra-Low Power

The X1 monolithic die 25.6 Tbps design delivers a comprehensive feature set and large memories. The

combination of revolutionary architecture, monolithic die, and Application Optimized Switching delivers ultra-low power for typical parallel computing and data center use cases.

At less than 300W for 25.6T and under 200W for 12.8T (for typical parallel computing and data center use cases) X1 is the industry's lowest power data center switch silicon. X1 enables twice the improvement in performance per Watt in a 1 RU form factor compared to currently available solutions.

High Radix

High radix switching is an important factor for distributed parallel computing systems. High radix allows connecting a number of server racks to a single switch, thus enabling the system to scale. High radix also reduces the number of network nodes required to interconnect massive scale clusters, offering a flatter and reduced diameter networks. A flatter network reduces FCT and tail latency significantly, thereby effectively boosting overall system performance. As a result, smaller diameter networks mitigate parallel slowdown and as such, boost the compute power of distributed parallel computing systems.

X1's best-in-class radix of 256 ports allows creating massively scaled systems that contain tens of thousands of compute nodes. For example, X1's high radix allows connecting a maximum of 32,768 hosts in 2 network layers, and a massive 4,194,304 hosts in 3 layers.

X-IQ™

As described above, PFC's native lack of granularity, HoL Blocking, and possible deadlock significantly increase FCT in AI and HPC clusters all leading to an increase in tail latency and degraded system performance. This type of bottleneck leads to parallel slowdown in AI and HPC clusters.

X1 X-IQ™ introduces 64K on-chip queues and fine-grained channelized flow control protocol with XFC™. This unprecedented granularity of congestion control along with its comprehensive set of traffic management mechanisms significantly reduces queueing time, minimizes HoL blocking and fate-sharing in multi-tenant deployments.

Application Optimized Switching

X1 Application Optimized Switching enables ultra-low power (for typical AI and HPC interconnect use cases) along with low latency and fully optimized packet processing.

X-PND™: ELASTIC RESOURCES

Switches must be able to absorb and handle traffic loads and bursts gracefully. Packet memory and traffic management in HPC and AI network is critical.

There are two main memory components in any switch architecture: packet memory and control table memory. Some architectures dedicate separate memories for packet storage and controlled tables. The latter is further portioned into a set of control tables, each (or most of which) has its own dedicated memory. Such inflexible memory architecture poses two problems. First, the inability to increase packet

memory size at the expense of unused control tables, reducing the network's ability to sustain larger loads and bursts. Second, the lack of memory flexibility within control table blocks (inability to grow deployment critical tables at the expense of unused or under-utilized ones) decreases the network's ability to fully address deployment requirements and creates islands of unused critical memory resources. Certain switch architectures allow flexibility within the control table blocks. This approach addresses only the second problem, yet does not address the main problem.

X1's X-PND™ introduces complete and uncompromised resource elasticity. X1's fully shared and elastic memory can be partitioned without limitations. This approach enables a single architecture to tailor resource allocation to HPC and/or AI application needs and to avoid under-utilization.

Allocating resources to a packet buffer alone is not sufficient for the HPC and AI cluster environment. Thus, X-PND™ alongside smart buffer management, X-IQ™, and XFC™ deliver an optimized solution that enables parallel computing clusters to scale.

FULL PROGRAMMABILITY

Traditional, "hardcoded" pipeline packet processing switch architecture was created to address a feature-heavy enterprise environment at relatively small scales. Legacy architectures carry the same architecture, "a little bit of everything", approach into a cloud world. AI and HPC clusters need "lean and mean" networks that are fully utilized without legacy architectures and the overhead of unused memories and logic that carryover power, latency and cost penalties.

Some architectures took steps forward by introducing configurable pipeline stages and flexible memories. This approach mitigates an overhead of unused control memories and some logic, however, it still carries the penalty of power, latency, and packet memory inflexibility.

X1 architecture delivers tangible network programmability and introduces uncompromised programmability across its processing logic, memories, and queue management. Full programmability of the X1 device delivers Application Optimized Packet Processing without overheads that lead to tangible power and latency advantages.

X-VIEW™: Traffic Analytics and Telemetry

The network is a critical component in parallel compute clusters. It requires application optimized network nodes and application optimized traffic management subsystem configurations. In order to troubleshoot and optimize cluster performance, network analytics data and telemetry support is vital. X1's XVIEW™ delivers a comprehensive analytics and telemetry suite. It incorporates in-band telemetry, any-cause verbose mirroring, black hole detection and localization, real-time statistics histograms and microburst detection.

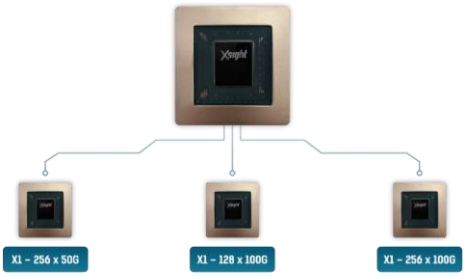
X-MON™: On-Chip Health Analytics

Data explosion drives massive growth of AI and HPC services. In its turn, it drives demand for enormous chip quantities, switches in particular, in order to address infrastructure needs. Networking devices with 25.6 Tbps throughput, comprehensive feature sets, and large memories (such as X1) are large and complicated. Quantities along with enormous scale of such devices makes quality and reliability more important than ever. Network reliability is critical as it directly affects parallel computing system operation. Chip vendors invest in comprehensive silicon level test coverage, screening processes, and quality and reliability monitoring processes, yet, test escapes and latent defects are a reality. Such issues often manifest themselves as in-field failures. Defective Parts Per Million (DPPM) is never 0 for chips at this scale. Prediction ability and root cause analysis of such in-field failures is practically non-existent today.

X1's X-MON™, powered by proteanTecs is a novel approach to this problem. It delivers in-field reliability assurance by providing readable data that enables predictive maintenance, alerts before failure, and extends system lifetime. In addition, X1's UCT™ dramatically reduces latent defects in deployment, significant improvement in DPPM, increases in-field reliability, and reduces network down time.

The X1 System

The X1 product family is comprised of 3 different device variations, all of which share the same software and feature set and are full interoperable.



X1 Family of Devices

Device Part Number	Maximum Throughput	Network Facing SerDes (Gbps)	Port Configuration Examples (GbE)
XLX1A256A	25.6 Tbps	256 x 100	• 256 x 25/50/100 • 128 x 200 • 64 x 400
XLX1A128A	12.8 Tbps	128 x 100	• 128 x 25/50 • 64 x 200 • 32 x 400
XLX1A256B	12.8 Tbps	256 x 50	• 256 x 25/50 • 128 x 100 • 64 x 200 • 32 x 400

X1 Hardware

The X1 family of devices is optimized for interconnecting AI, ML, storage and compute clusters within a data center's ecosystem. It enables building compact 1RU switches with large port densities of up to thirty-two 800G QSFP-DD/OSFP. The X1 based 1 RU system is built and backed by a leading ODM. A 1RU production-ready, cost effective system is available in multiple configurations: 12.8 Tbps (32 x 400G), 12.8 Tbps (16 x 800G), and 25.6 Tbps (32 x 800G) dual-face plate configurations — 32xOSFP or 32xQSFP-DD — enabling smooth infrastructure integration. The 1RU system's retimer–less design delivers improved power efficiency.

X1 1-RU Hardware

X1 Software

The Xsight Software Development Kit (X-SDK) delivers a comprehensive feature set. The multi-layered SDK design enables multiple integration models with different NOS types. The same X-SDK and feature set are consistent across the entire X1 family of devices. X1 software embraces open networking with SAI and SONiC integration.

Summary

The X1 family of devices are best-in-class ultra-low power switches providing throughputs of up to 25.6 Tbps. The X1s are powered by flexible 100G LR SerDes, unconditionally shared buffers, HPC and AI Application Optimized Switching, X-IQ™, comprehensive traffic management sub-system, X-PND™, X-VIEW™, and X-MON™ technologies. They deliver an extremely low power and low latency solution for AI and HPC cluster processing and programming, by addressing and minimizing current network infrastructure problems.

References

[1] Jon L. Jacobi. *NVMe SSDs: Everything you need to know about this insanely fast storage*. PCWorld. https://www.pcworld.com/article/2899351/everything-you-need-to-know-about-nvme.html Accessed on 10/28/2020

[2] https://habana.ai/home/. Habana (An Intel Company); Accessed on 10/31/2020

[3] Shuihai Hu, Yibo Zhu, Peng Cheng, Chuanxiong Guo, Kun Tan, Jitendra Padhye, Kai Chen. *Deadlocks in Datacenter Networks: Why Do They Form, and How to Avoid Them*. Microsoft, Hong Kong University of Science and Technology. https://www.microsoft.com/en-us/research/wp-content/uploads/2016/11/rdmahotnets16.pdf Accessed on 11/01/2020

About Xsight Labs

Xsight Labs is a fabless semiconductor company headquartered in Kiryat Gat, Israel with additional offices in Tel-Aviv and Binyamina. In the United States, Xsight Labs has offices in Boston, MA, Raleigh, NC, and San Jose, CA.

Founded in 2017, Xsight Labs has assembled a world-class engineering team to re-architect the foundation of cloud infrastructure by delivering a broad portfolio of products that enable end-to-end connectivity. Xsight Labs' technology delivers exponential bandwidth growth while reducing power and total cost of ownership.

Building on over 20 years of experience in developing and productizing multiple generations of cloud infrastructure products, the Xsight Labs executive team is focused on tackling modern data center challenges.

Xsight Labs
Leshem 1, Kiryat Gat, Israel
Contact a representative at sales@xsightlabs.com