Gaudi® Training Platform White Paper

Nov 2020
Ver 1.2

Table of Contents

1. Introduction

Demand for high-performance AI training compute has doubled in size rapidly and is accelerating with the growing number of applications and services based on image and gesture recognition in videos, speech recognition, natural language processing, recommendation systems and more. With this increased demand comes the need for greater training speed, throughput, and capacity, which translate into the growing need for efficient scaling of training systems.

Typical Deep Learning Training algorithms comprise multiple types of operators which add up to billions of operations. These massive amounts of operations can be accelerated by using the inherent parallel processing that advanced GPUs offer. However, GPUs are primarily designed to render graphics efficiently, rather than execute Deep Learning workloads. GPU inefficiency for Deep Learning workloads has a severe impact on the operational costs of cloud platforms and datacenters. To address this issue, Habana Labs has developed AI training processor solutions designed from the ground up to address the massive computation requirements of large DNN workloads and deliver unprecedented levels of efficiency to massive systems: the Gaudi Training Processor.

During training, the internal parameters of the neural network are tuned and optimized for the target application. A typical network contains billions of internal parameters, all tuned and changed many times while training, resulting in very long processing times, even on large-scale multi-GPU systems. Further elaboration on the training process follows in this paper.

Although in recent years significant advances have been made in GPU hardware, network architectures and training methods, the fact remains that network training can take an impractically long time on a single machine. Fortunately, we are not restricted to a single machine. A significant amount of research and development has been conducted toward enabling efficient distributed training of Deep Neural Networks. This whitepaper provides a technical review of the Gaudi Training system which serves as an infrastructure to distributing high- performance computing for DNNs.

2. Gaudi Deep Learning Training Solution

The Gaudi platform architecture has been designed from the ground up for Deep Learning training workloads in datacenters. It comprises a fully programmable Tensor Processing Core (TPC) cluster, along with its associated development tools and libraries. All are designed to collectively deliver a comprehensive, flexible platform that greatly simplifies the development and deployment of Deep Learning systems. The platform is capable of massive data crunching with its unique scale out capability.

The Gaudi chip integrates its processor die with four HBM2 memories in a single-chip package.

3. Gaudi Processor High-level Architecture

Gaudi is based on the scalable architecture of the TPC. The Gaudi processor features a cluster of eight TPC 2.0 cores. The first generation of TPC cores was introduced in the Goya inference processor and the TPC 2.0 was designed to support Deep Learning training and inference workloads. A high-level scheme of the Gaudi architecture is presented in Figure 1 below.

The TPC 2.0 core is a VLIW SIMD processor with instruction set and hardware that were tailored to serve training workloads efficiently. It is C-programmable, providing the user with maximum flexibility to innovate, coupled with many workload-oriented features, such as:

- GEMM operation acceleration
- Tensor addressing
- Latency hiding capabilities
- Random number generation
- Advanced implementation of Special Functions

The TPC core natively supports the following data types: FP32, BF16, INT32, INT16, INT8, UINT32, UINT16 and UINT8.

The Gaudi memory architecture includes on-die SRAM and local memories in each TPC. In addition, the chip package integrates four HBM2 devices, providing 32 GB of capacity.

The PCIe interface provides a host interface and supports both generation 3.0 and 4.0 modes.

Gaudi is the first AI Processor that also integrates on-chip RDMA over Converged Ethernet (RoCE v2) engines. These engines play a critical role in the inter-processor communication needed during the training process. By integrating this functionality and supporting bi-directional throughput of up to 2 Tb/sec, customers can build systems of any size, and adapt them to their requirements.

The Gaudi includes 20 pairs of 56Gbps Tx/Rx PAM4 serializers/de-serializers (SerDes) that can be configured as 10 ports of 100Gb Ethernet, 20 ports of 50Gb/25Gb Ethernet, or any combination in between. A Gaudi port operating at 100GbE can also be configured to use four SerDes operating at 25Gbps for connecting to legacy switches. These ports are designed to scale out the inter-Gaudi communication by integrating a complete communication engine on-die.

This native integration allows customers to use the same scaling technology, both inside the server and rack (termed as scale-up), as well as to scale across racks (scale-out). These can be connected directly between Gaudi processors, or through any number of standard Ethernet switches.

Compared with competing architectures, customers do not need to add an array of PCIe switches and dedicated NICs.

Ethernet switches, unlike proprietary connectivity switches, are available from many vendors with greater options in port- count to choose from (from small switches to 25.6Tbps with 128 ports of 100GbE in a single chip or bigger).

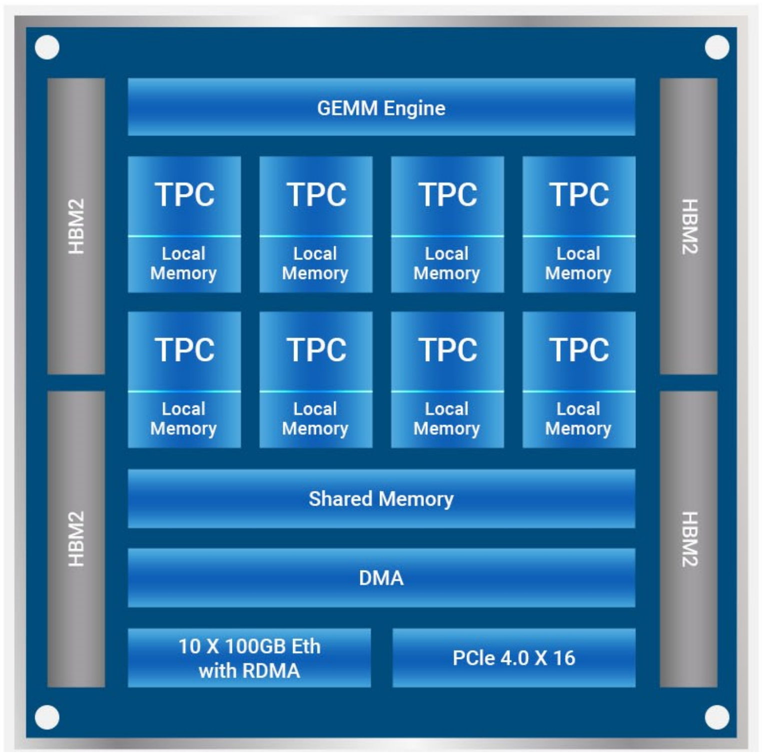

Figure 1: Gaudi High-level Architecture

4. Gaudi Software Stack

Habana's software platform is designed to provide a full software stack including flexible development capabilities of the programmable Tensor Processor Cores, the SynapseAI® - Habana's complete software stack custom designed to support Habana's Gaudi implementation. It is designed for flexibility and ease of development with its C-programmable Tensor Processor Core. custom-developed compiler and runtime and extensive, customizable kernel libraries.

Habana provides, as part of its SW package an extensive set of TPC kernel libraries (1400+ kernels) and opens its TPC for the user programming, providing a complete TPC tool suite (debugger, simulator, compiler). These tools facilitate the development of customized TPC kernels that can augment the kernels provided by Habana Labs. Thus, users can quickly and easily deploy a variety of network models and algorithms on Gaudi to innovate and optimize to any unique requirements.

The SynapseAI is built for seamless integration with existing frameworks, that both define a Neural Network for execution and manage the execution Runtime. SynapseAI can be interfaced directly using either C or Python API, It also natively supports TensorFlow 2.2 today and will be followed by native PyTorch support in the first half of 2021. Integrating natively into DNN frameworks like TensorFlow, SynapseAI enables users to unleash the power of Deep Learning by executing the algorithms efficiently using its high-level software abstraction.

Designed to facilitates high-performance Deep Learning training on Habana's Gaudi accelerators, Synapse AI provides the optimization, compilation, and runtime functionalities including multi-stream execution environment, supporting Gaudi unique combination of compute and networking, exposing a multi-stream architecture to the framework. streams of different types: compute, networking and DMA are synchronized with one another at high performance and with low run-time overheads.

As describes in detail in the next chapter, there are many scale-out trainings schemes and topologies that are built to support the different workloads requirements. To allow an efficient scale-out communication between any account of Gaudi processors and across the different model parallel, data parallel or combination of model and data parallel schemes, the Habana Communication Library (HCL) is provided: a high-level communication library, tailor-made for Gaudi's high performance RDMA communication capabilities. The HCL exposes all required primitives like Reduce-Scatter, All Reduce, All-Gather, and broadcast and manage the communication of all sizes of Gaudi clusters per a given cluster topology.

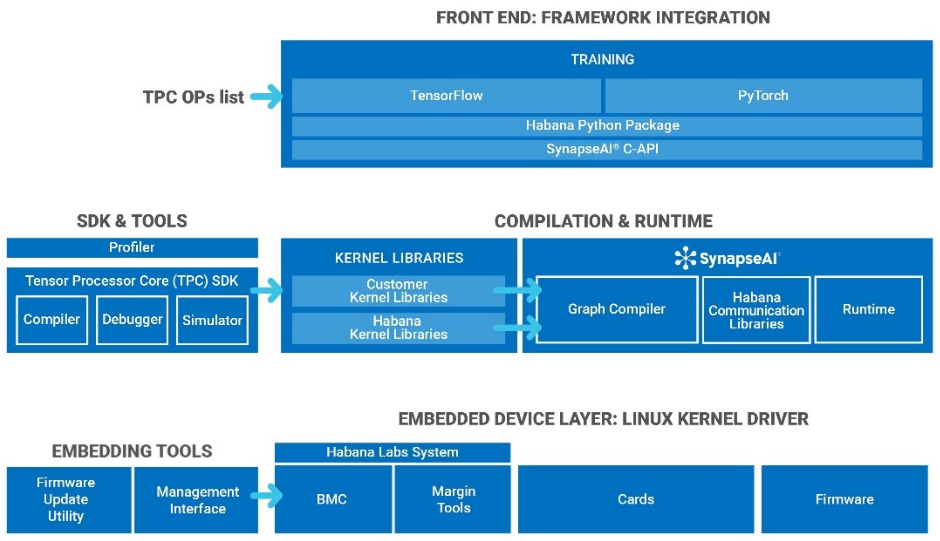

Figure 2: Gaudi Platform Software Development Tools

5. Training Algorithms

Figure 3 shows the two main distribution strategies used today when training Deep Learning models – Data Parallelism and Model Parallelism.

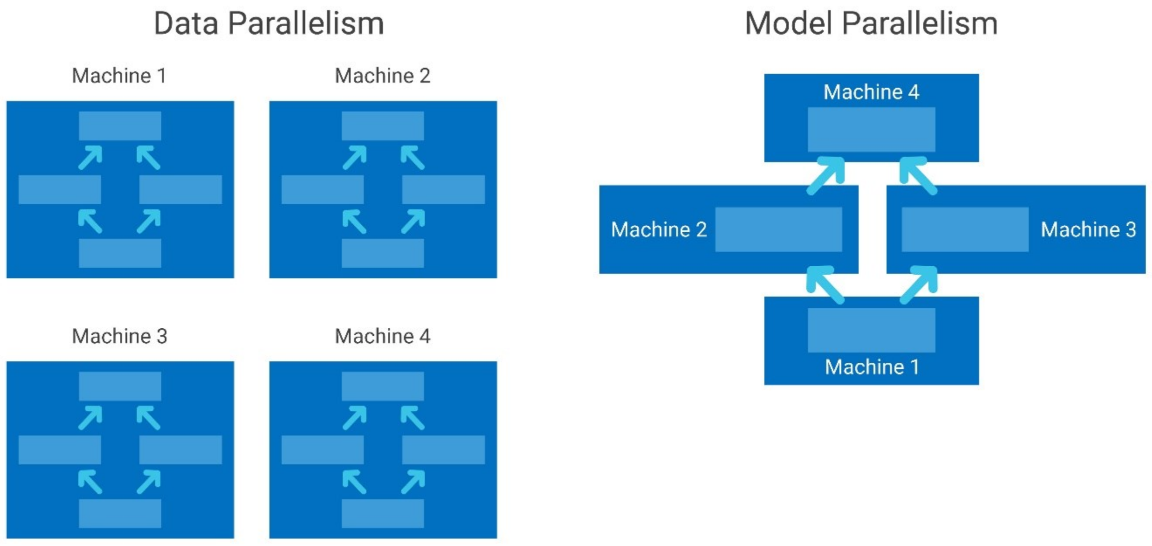

Figure 3: Data Parallelism and Model Parallelism

5.1 Data Parallelism Training

In Data Parallelism, every machine has a complete copy of the Deep Learning model. Each machine receives a different portion of the data, performs the training locally, then transmits its parameter updates to a Parameter server to share its training output with the other machines.

5.1.1 Training Bandwidth Requirements

The most widely used algorithm for training a DNN model is Stochastic Gradient Descent (SGD). SGD works on small batches (Minibatches) of input samples at a time. After each Minibatch, internal summation of gradients derived from all examples within the Minibatch occurs.

As of today, Data Parallelism is the mainstay of Deep Learning. With Data Parallelism, the Minibatch is distributed between the different workers. For example, for a Minibatch of 1,024 images and 16 workers, each worker is assigned a worker-batch of 1024/16=64 examples. After performing the Forward and Backward paths on all 64 examples, each worker performs internal summation of the gradients that resulted from the 64 examples, then sends a single copy of the gradients to the Parameter server. The size of the message sent to the Parameter server is constant and proportional to the number of internal parameters within the model. The size of the Minibatch does not affect the size of the message sent to the Parameter server.

Note that the parameter server can be centralized or distributed.

Generally, the larger the Minibatch the longer it takes to perform the Forward and Backward paths. Thus, larger Minibatches require lower network bandwidth. There is an inverse correlation between the Minibatch size and the interconnect bandwidth requirements.

5.1.2 Hierarchical Reduction of Data

When performing Data Parallelism, gradients can be sum-reduced in a hierarchical manner. Figure 4 shows four workers. The first worker computes the gradient tensor A, the second worker computes the gradient tensor B and so on. After the first reduction phase, half of the workers hold sum-reduced results of A+B and C+D. After the third step, a quarter of workers hold the sum-reduced result A+B+C+D. In the example of Figure 5, a single copy of the gradients holding A+B+C+D can be sent to the next reduction hop.

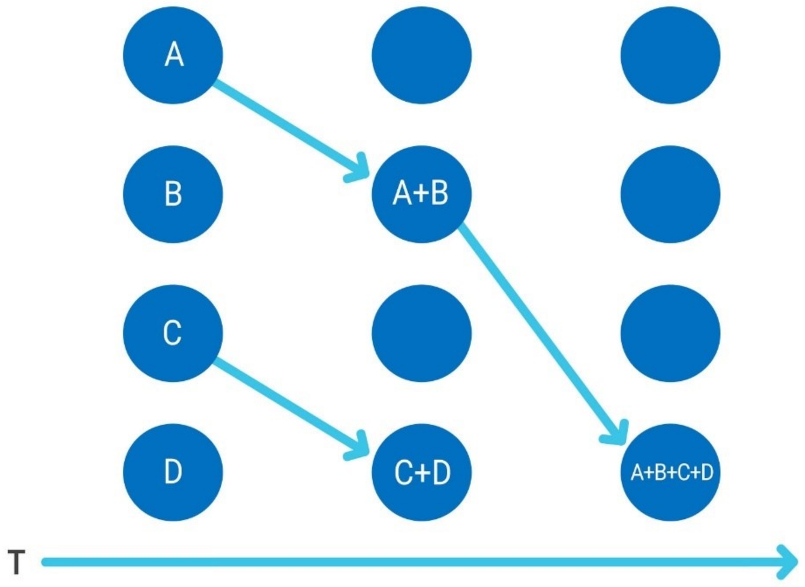

Figure 4: Hierarchical Reduction

Because of this reduction property, when building a network infrastructure, reduction-domains are aggregated in a hierarchical manner.

5.2 Model Parallelism Training

In Model Parallelism, different machines in the distributed system are responsible for the computations of different parts of a single network. For example, each layer in the neural network may be assigned to a different machine. In another example of Model Parallelism, parameters of a single layer are distributed between different machines.

Model Parallelism requires low-latency and high-throughput connectivity between chips. Model Parallelism does not require reduction to always take place. For example, if four Gaudi chips are working on the same problem and different layers are assigned per Gaudi chip, results from one of the Gaudi chips must be transferred to at least one, or even to all, other Gaudi chips. Using this example, results between two different Gaudi chips are not combined or reduced in any way.

6. Building a Training Systems with Gaudi

6.1 System Building Blocks

Scaling-out an AI workload has never been easier.

Gaudi leverages a superior, open-standard networking technology for scale-out. Each Gaudi chip implements 10 ports of standard 100Gbit Ethernet (or 20 ports of 50 GbE/25 GbE). Integrating networking directly into the AI processor chip creates a nimble system without bandwidth bottlenecks. By combining multiple Gaudi chips with Ethernet switching, limitless possibilities are available for distributing training across 8, 16, 32, 64, 128, 1K, 2K, 8K and more Gaudi chips.

As Gaudi uses off-the-shelf Ethernet, many different systems and network configurations can be used.

The following sections describe several potential system implementations that can be built using the Gaudi chip.

6.2 Habana Labs Systems

6.2.1 HLS-1

HLS-1 is a system provided by Habana Labs, containing eight HL-205 OCP Accelerator Module (OAM) Mezzanine cards and dual PCIe switches. The all-to-all connectivity allows training across all eight Gaudi processors without requiring an external Ethernet switch.

In HLS-1, the Gaudi chips are connected all-to-all on the main board, using seven 100GbE ports of each Gaudi. The remaining three ports from each Gaudi are available to scale out the solution over Ethernet ports on the HLS-1. Any host can manage the Gaudi system through the PCIe ports. Such a system topology is optimal for both Data parallelism, where HLS-1 serves as the first reduction hierarchy, and the Model-Data Parallelism hybrid, by using all-to-all connectivity within the HLS-1 for Model Parallelism together with intra-HLS-1 connectivity for Data Parallelism (intra- and inter-card connectivity).

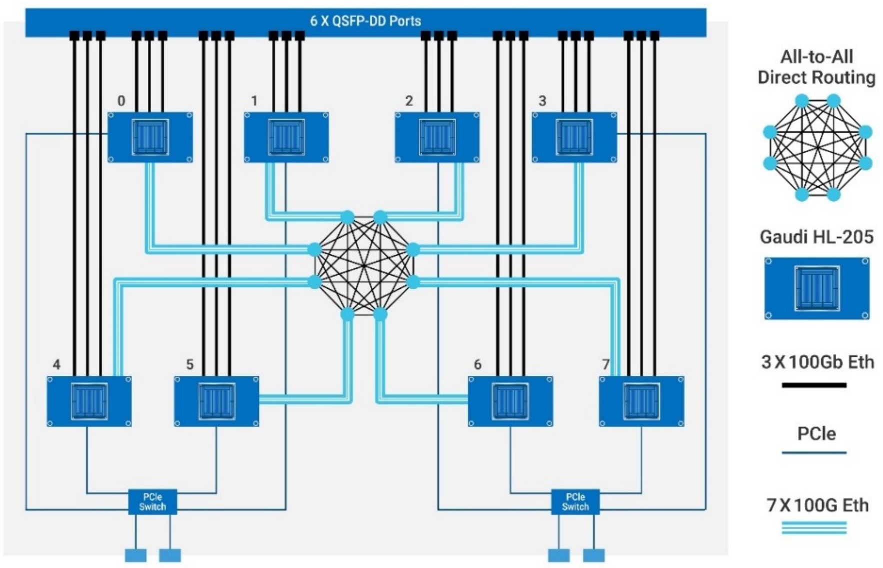

Figure 5: HLS-1: System Topology

6.2.2 HLS-1H

HLS-1H is a system provided by Habana Labs, containing four HL-205 OCP Accelerator Module (OAM) Mezzanine cards and is built for massive scale out. The massive scale-out enable training of large models across any size of Gaudi processors clusters using off-the-shelf external standard Ethernet switches.

The Figure below shows the HLS-1H system containing four Gaudi HL-205 cards and its interfaces. The interfaces are 2x16 PCIe Gen4 cables that can be connected to an external host server, and up to 40X100Gb Ethernet links (using 10 QSFP-DD connectors). The external Ethernet links can be connected to any switching hierarchy. Such configuration can be optimized to implement extra-large Model Parallelism in large scale and can easily handle Data Parallelism or a combination of Model and Data parallelism.

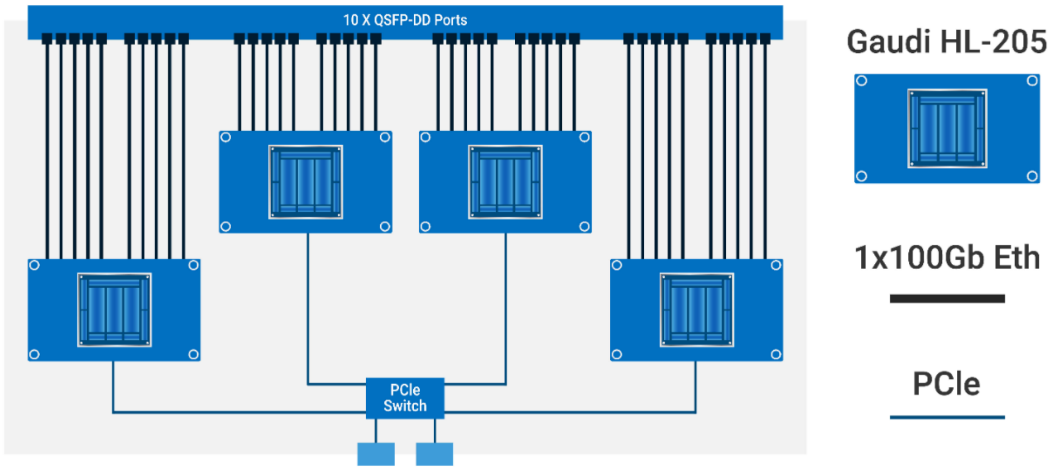

Figure 6: Gaudi System with Maximum Scale-out

6.3 Gaudi System with On-Board Ethernet Switch

The below Figure present another example of a Gaudi system where the Ethernet switch is part of the integrated system.

Note that a solution with more networking BW is possible: it is possible to use a 128x100G Ethernet switch, and connect 10 ports of 100G from each Gaudi to the ethernet switch. In that case the Ethernet switch will be connected to the aggregation fabric with 10x100G ports, leading to total utilization of 90x100G ports on the switch.

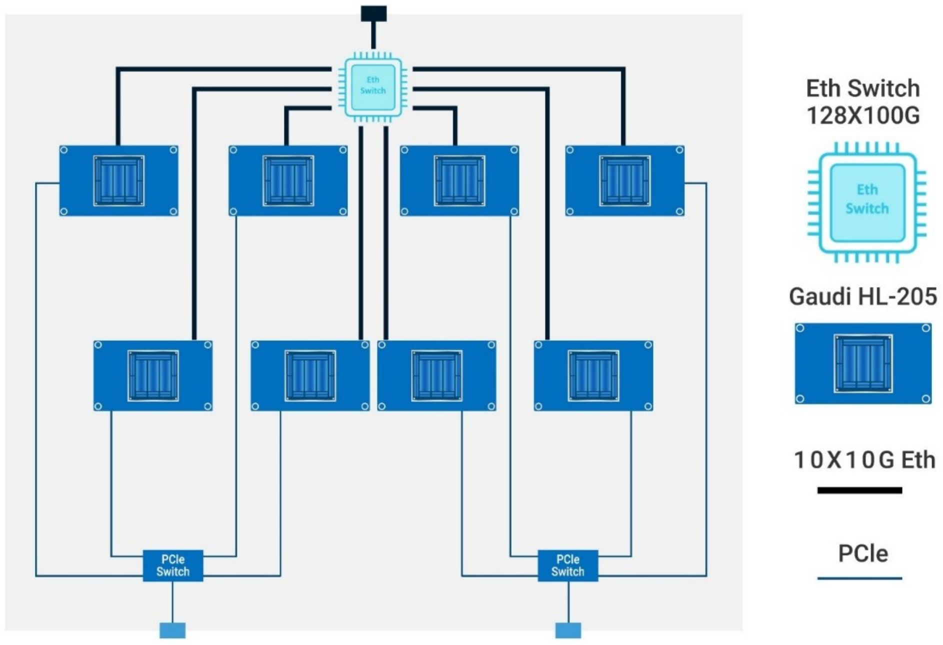

Figure 7: Gaudi System with an On-board Ethernet Switch

6.4 Gaudi-based Training Rack

The below exemplary figure shows a configuration where six Gaudi systems containing 48 Gaudi devices in total (eight Gaudi devices per Gaudi system), are connected to a single Ethernet switch. The switch can be further connected to other racks in order to form a much larger training farm that can hold hundreds or thousands of Gaudi processors.

The eight-Gaudi system can be connected with either internal connectivity (such as HLS-1) or with completely exposed ports, or any other partitioning of internal connectivity and external scale-out. In this example, each HLS-1, apart of the internal all-to-all connectivity of the eight Gaudi is connected to a switch with four cables, carrying a total of 16x100GbE (8x 2x100GbE per Gaudi system)

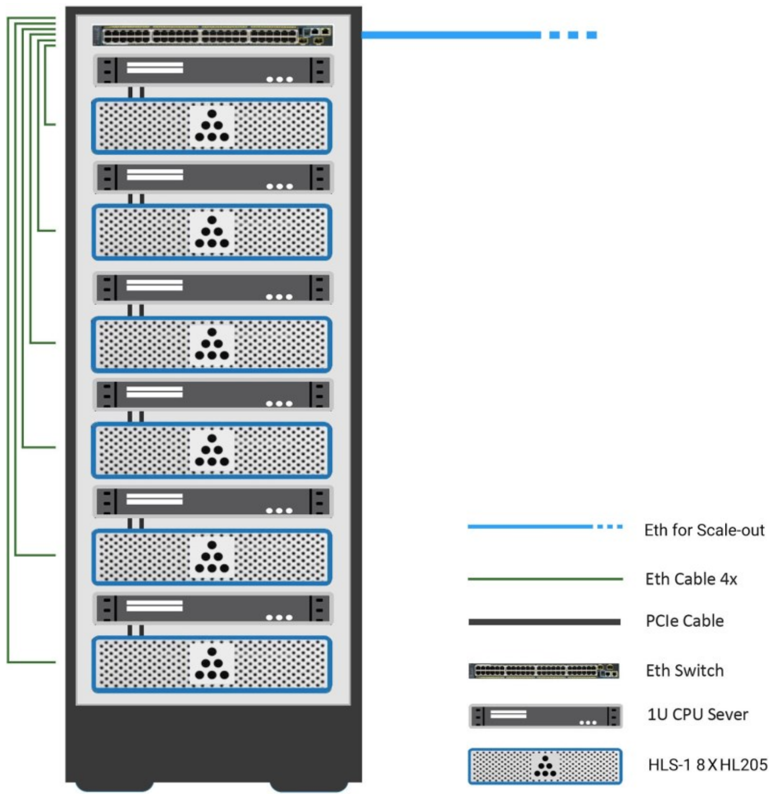

Figure 9: Example Rack

1.1 Gaudi-based Training POD

The figure below presents an exemplary diagram of an HLS-1H based POD build for massive scale out in datacenter. It comprises of 128 HL-205 Gaudi training processor cards with 10 links of 100Gb/s per card, this translates into a cluster of 32 HLS-1H systems (4 cards per system), each with 10 QSFP-DD ports of 400Gb/s.
Ten standard Ethernet switches (32 QSFP-DD) enable the connection of a fully connected non-blocking CLOS topology of the 128 Gaudi processors.

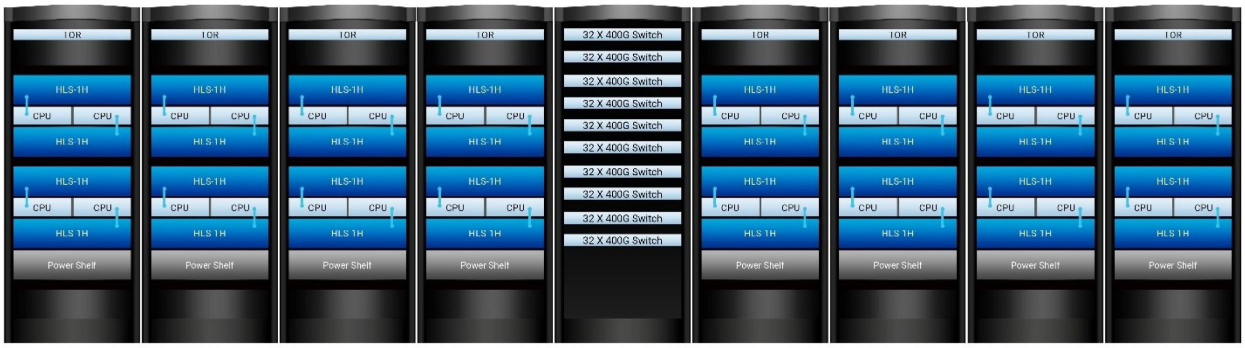

Figure 9: HLS-1H Based POD

1.2 High-end 2K Gaudi System

In this example, each eight-Gaudi system has a companion 64-port Ethernet switch, as described in the below diagram. Each such switch is connected to an aggregation fabric built from eight 256x100GbE switches, using a Clos topology. All Gaudi chips are connected to all other Gaudi chips with only three networking hops.

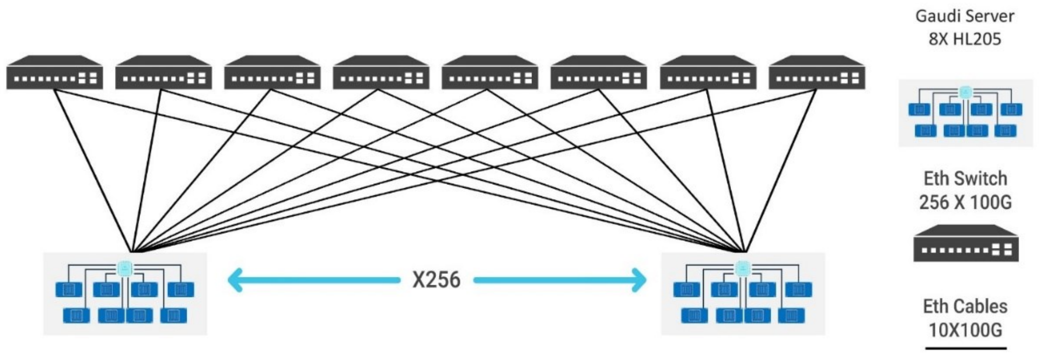

Figure 10: High-end 2K Gaudi System

1.3 Topologies for Different Types of Parallelism

In this section we explore how to construct topologies for Data Parallelism, Model Parallelism or a combination of both.

1.3.1 Topologies for Data Parallelism

The Figure below shows how a larger system is built using the Gaudi system as a basic component. It shows three reduction levels – one within the system, another between 11 Gaudi systems and another between 12 islands. Altogether, this system hosts 8*11*12 = 1056 Gaudi cards. Larger systems can be built with an additional aggregation layer or with less bandwidth per Gaudi.

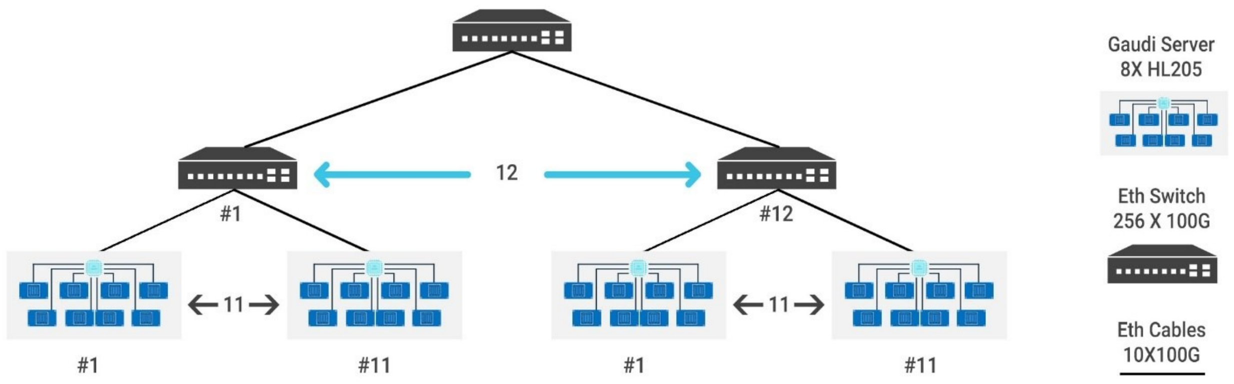

Figure 11: Hierarchical Fabric

1.3.2 Topology for a Combination of Data and Model Parallelism

Figure 11 can also serve for a combination of Data and Model Parallelism. Each Gaudi system can be used for Model parallelism, while Data Parallelism can be used between Gaudi systems. Normally, Model Parallelism requires higher bandwidth and lower latency than Data Parallelism. A tightly coupled Gaudi system with hierarchical connectivity between boxes can serve this requirement.

1.3.3 Topologies for Model Parallelism

Since Gaudi uses standard Ethernet connectivity, very large-scale systems can be built with all-to-all connectivity utilizing a single networking hop.
Figure 12 shows an illustration of such a large-scale system incorporating 128 Gaudi chips. A system with different count of Gaudi chips can be built in the same manner, connecting 1TbE per Gaudi (thus 4TbE per HLS-1H System) to a larger switching fabric, consisting of ten 32- port switches of 400G. Such a large-scale, single-hop system with full connectivity is only possible when connecting Ethernet directly to the Deep Learning Accelerator.

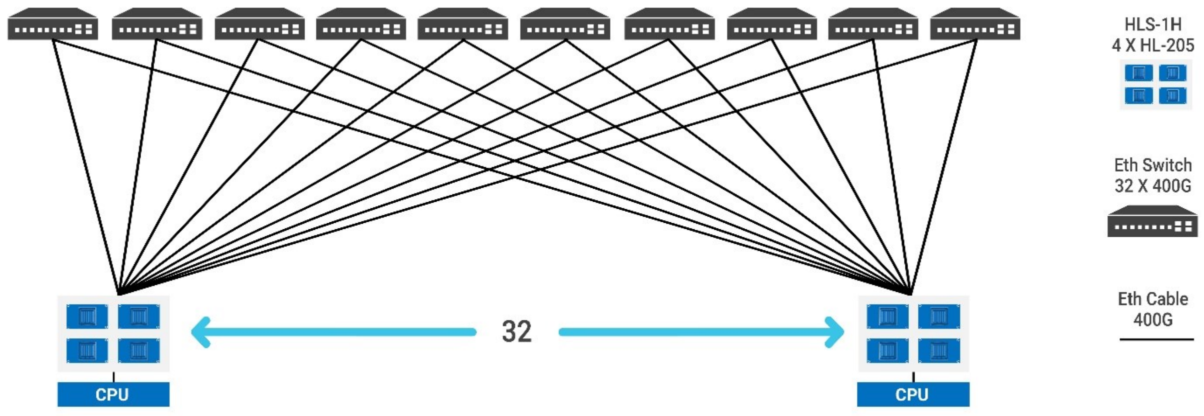

Figure 12: Mass scale-out configuration

2. Gaudi Training Performance

The key factors that are used in assessing the performance of a training solution are as follows:

- Deep Learning Benchmarks – Measure of the time needed to train a given DNN topology on the target system to meet a given accuracy goal. MLPERF.org is the primary organization developing such benchmarks. The training time is also correlated to measuring throughput in samples/second
- Scalability – Does it scale well as more training processors are added? Ideally, the throughput should scale linearly, to ensure optimal cost efficiency at the system level
- Power Consumption at the system level (not just the accelerator)
- Total cost of ownership (capital and operational costs)

2.1 Performance

The following presents the performance of a single Gaudi chip for the ResNet-50 image classification benchmark. ResNet-50 is one of the MLPERF benchmarks. A single Gaudi running natively a TensorFlow resnet50 model delivers 1590 images per second of training throughput and scaling to eight Gaudi cards running the resnet50 model it is scale near linearly to 12,008 images per second.

Training systems are typically not sensitive to latency in the sense of cycle execution, they care about the model converge time, however a single processor latency is extremely important to accelerate the over whole training.

In many cases the trained data is split across multiple processors and run concurrently to accelerate the training time to converge, this will work only if the architecture will allow linear or close to liner scaling. Low latency and the ability to run efficiently in low batch size allow this linear scaling.

- Performance numbers are for SynapseAI v0.11 release, Nov 2020

3. Summary

Deep Learning revolutionizes computing, impacting enterprises across multiple industrial sectors. The computational complexity of deep neural networks is becoming exponentially larger, driving massive demand for compute power. The challenge of deep neural network training is to improve upon multiple criteria at once: first, to shorten execution time and complete the job faster; second, to reduce energy consumption; third, to provide flexible scalability with standard interfaces; and fourth, to lower the total cost of ownership.

Gaudi training platforms present several key advantages over GPU-based solutions:

- Performance leadership that results in significantly lower training time, improved efficiency and lower system size
- High throughput at low batch size – showing speedup at scale, over GPU solutions
- High power efficiency
- Native integration of Ethernet for scaling, instead of GPU proprietary interfaces, which offers several major advantages
- Using standard Ethernet for scale up/out, end customers can avoid lock-in to any specific AI processor vendor
- Lower system cost through wide availability of Ethernet switches of any size from multiple vendors
- Model Parallel Training - Ability to train models that require splitting to many more processors than today's limits with GPU based systems (hitting major bottleneck beyond 16 GPUs)
- Open Compute Project (OCP) Accelerator Module (OAM) compliant

Gaudi-based training solutions provide the opportunity for organizations to lower the total cost of ownership (TCO), as well as provide a flexible path to easily scale their systems as they grow and evolve.