habana GAUDI™

Gaudi™ Training Platform White Paper

June 2019
Ver 1.0

Gaudi™ Training Platform White Paper

Table of Contents

Table of Figures

Table of Tables

1. Introduction

Machine Learning (ML), a subfield of Artificial Intelligence (AI), is no longer science-fiction. One prominent field within ML is Deep Learning, in which the models are Deep Neural Networks (DNNs). Only several years ago, numerous problems from various domains were considered too difficult for machines to solve. For example, object detection in images, gesture recognition in videos, speech recognition, natural language processing and many more. That is no longer the case. Today, these problems are solved as accurately as by human experts and better, all using Deep Learning models. We have come to a time where Deep Learning is unanimously considered a transformational technology.

A typical Deep Learning algorithm comprises of multiple operators, such as matrix multiplication, convolutions and other tensor operations, 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, not to 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, a new class of software-programmable AI processors is emerging, designed bottom-up for DNN workloads – Pure AI™ processors.

Reaching impressive performance with Deep Neural Networks across a wide variety of domains requires performing a computationally demanding task known as training. During DNN training, the internal parameters of the neural network are tuned and optimized for the target application. A typical DNN contains billions of internal parameters, all tuned and changed many times while training, adding up to very long processing times even on large-scale multi-GPU systems. Further elaboration on the training process follows in the next sections.

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 a new system which serves as an infrastructure to distributing high-performance computing for DNNs.

2. Deep Learning Workflows – Training and Inference

A Deep Learning workflow consists of two stages:

- Tuning and optimizing a model for high performance on the desired task (training)
- Applying the trained model on new data (inference)

Both the training and inference processes involve similar workloads. However, their hardware requirements and performance characteristics are different.

During training, a large dataset is applied to a neural network with the objective to model statistically predetermined attributes of a large dataset. For example, recognizing images of apples from an arbitrary set of input images is a training function. Once the model meets the accuracy goals set for object recognition, the neural network is considered trained and can be used for Inference purposes, where the model can infer the predetermined attributes from any dataset.

Unlike Inference, training involves a backward propagation phase that transfers the error back through the network layers and updates their internal parameters in order to improve network's performance. It is common to batch hundreds of training inputs and operate on them simultaneously. Therefore, the training phase with its high computational demands, requires high memory bandwidth, large memory capacity and the ability to transfer data between compute entities quickly and efficiently.

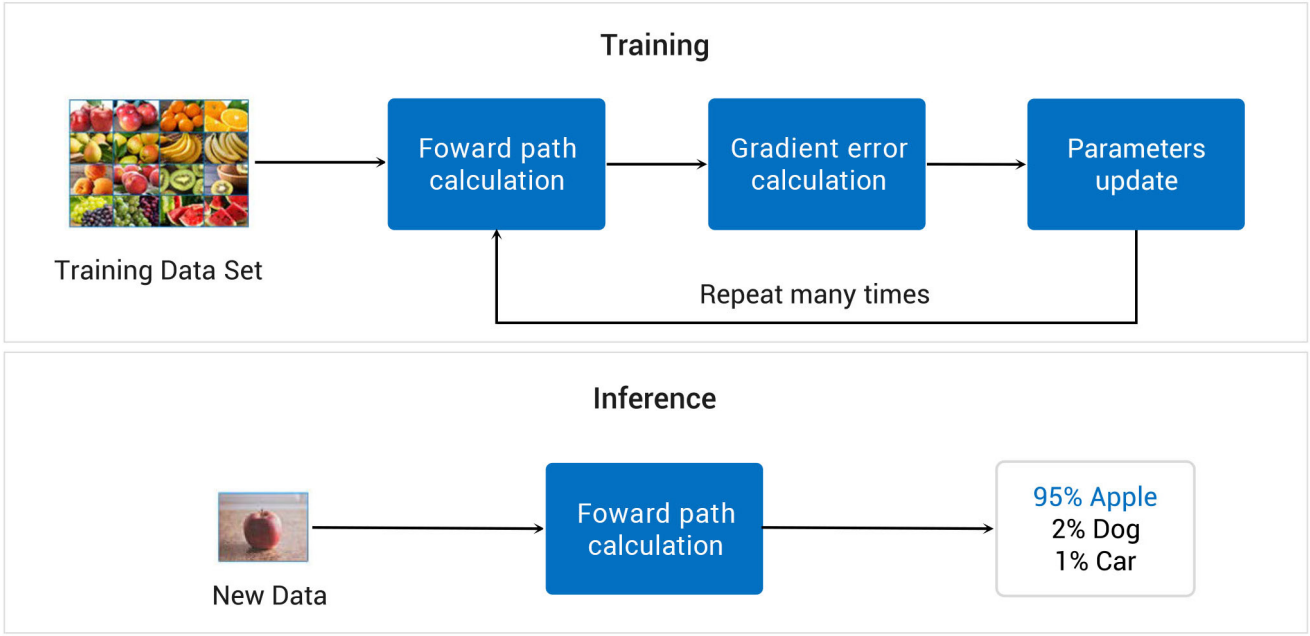

Figure 1: Deep Learning Workflows – Training and Inference

3. 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, libraries and compiler. All 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 unique scale out capability.

The Gaudi HL-2000 chip integrates its processor die with four HBM memories in a single-chip package. The chip is available in two card configuration types: a PCIe card and a Mezzanine card. These allow deployment in various systems from a small-scale server to medium and large-scale systems.
Table 1 describes the key differences between the cards.

	HL-205 Mezzanine Card	HL-200/202 PCIe Card	
PROCESSOR TECHNOLOGY	Gaudi HL-2000		
HOST INTERFACE	PCIe Gen 4.0 X 16		
MEMORY	32GB HBM2		
MEMORY BANDWIDTH	1TB/s		
ECC PROTECTED	Yes		
MAX. POWER CONSUMPTION	300W	200W	
SCALE-OUT INTERCONNECT	RDMA (RoCE v2)		
	10x100Gbps or 20x50Gbps	8x100Gbps or 16x50Gbps Dual QSFP-DD ports	
FORM FACTOR AND SKUS	OCP Accelerator Module Compliant	Full Height/Length HL-200 Passive Cooling	HL-202 Active Cooling

Table 1: Gaudi-based Cards

4. Gaudi Processor High-level Architecture

Gaudi is based on the scalable architecture of the (TPC™) Tensor Processor Core. Gaudi uses a cluster of eight TPC 2.0 cores. The first generation of TPC cores was introduced in the Goya inference processor. A high-level scheme of the Gaudi architecture is presented in Figure 2 below.

The TPC 2.0 core was designed to support Deep Learning training and inference workloads. It is a VLIW SIMD vector processor with instruction set and hardware that were tailored to serve training workloads efficiently.

The TPC 2.0 core 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 HBM devices, providing 32 GB of capacity and 1 TB/sec bandwidth.

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 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 HL-2000 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 for scaling 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 switches, are available from many vendors with bigger options in port-count to choose from (from small switches to 12.8Tbps with 128 ports of 100GbE in a single chip).

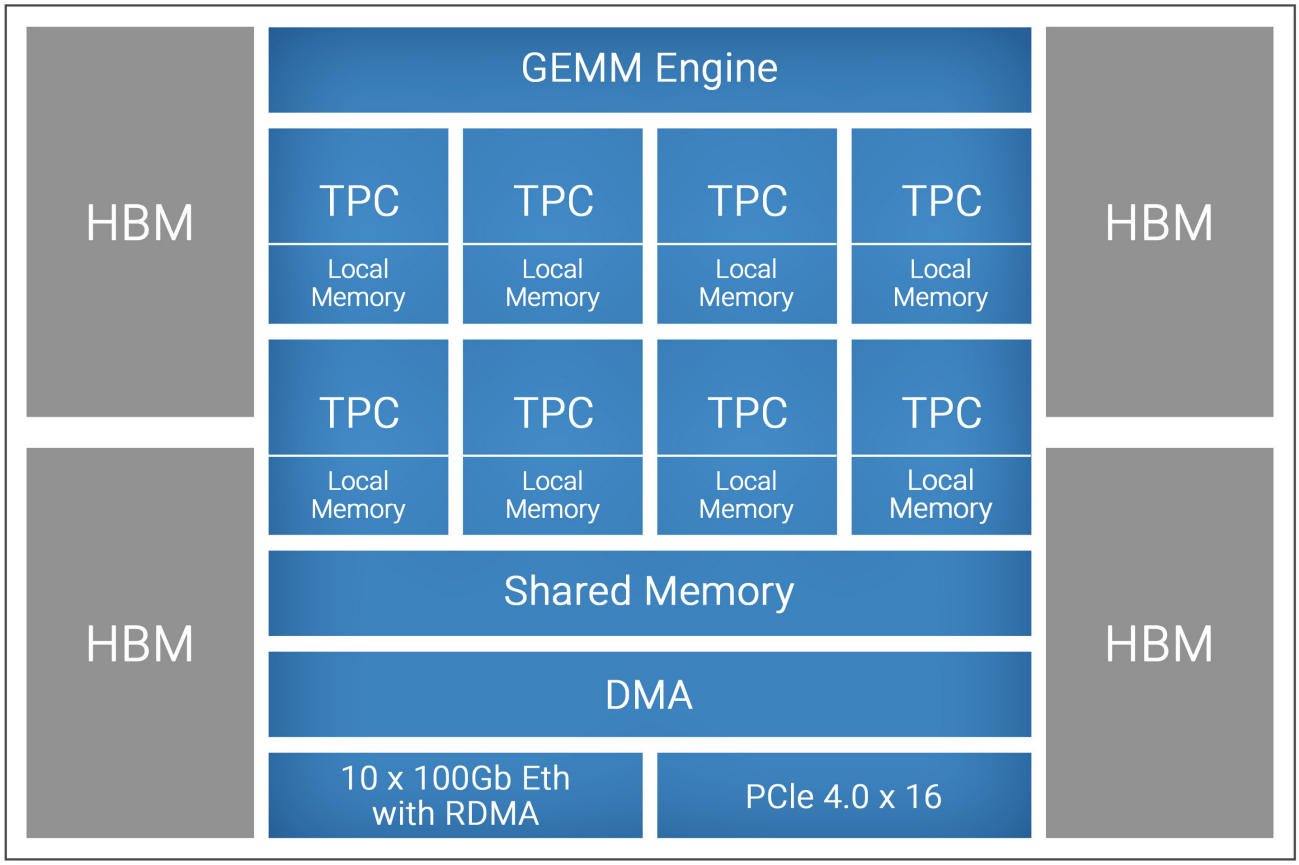

Figure 2: Gaudi High-level Architecture

5. SynapseAI® Software Development Tools

SynapseAI® is Habana's home-grown compiler and runtime. In its training incarnation, it 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 is also natively integrated into TensorFlow. Other Deep Learning Frameworks will be added at a later stage. By 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.

SynapseAI supports both Imperative/Eager mode and Graph mode. In Eager mode operations are evaluated immediately without building a computational graph. It is used for debug and prototyping and supports natural control flow making it easier to support dynamic networks.

SynapseAI is built out of the following main components:

- Multi-stream execution environment, exposing a multi-stream architecture to the framework. Streams can be synchronized with one another at high performance and with low run-time overhead;
- JIT compiler, capable of fusing and compiling multiple layers together, thereby increasing utilization and exploiting hardware resources;
- Including support for dynamic shapes;
- Habana Communication Library (HCL): a high-level communication library, tailor-made to Gaudi's high performance RDMA communication capabilities. HCL exposes all required primitives like Reduce, All Reduce, Gather, broadcast etc.;
- Rich rich set of pre-written and performance optimized TPC kernels.

The Gaudi platform comes with state-of-the-art development tools, including visual real-time performance profiling and TPC development tools for advanced users (including an LLVM-based C compiler, simulator, debugger and profiling capabilities). 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 with the SynapseAI software development tools.

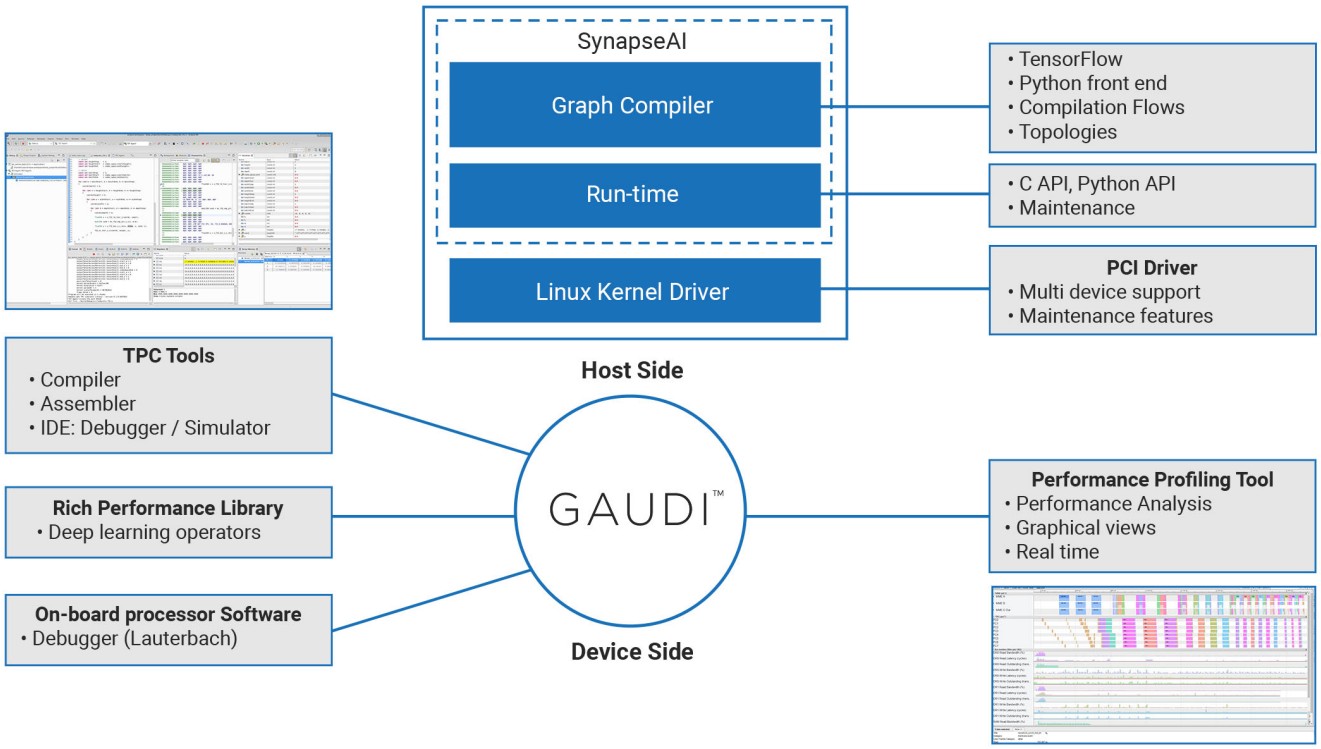

Figure 3: Gaudi Platform Software Development Tools

6. Training Algorithms

Figure 4 shows the two main distribution strategies used today when training Deep Learning models – Data Parallelism and Model Parallelism.

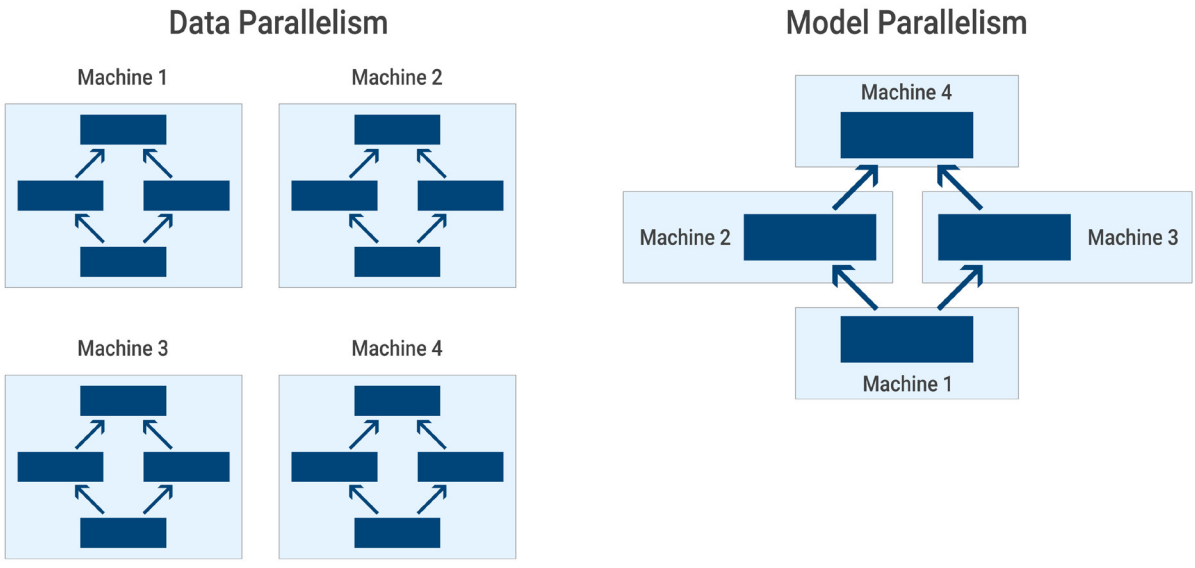

Figure 4: Data Parallelism and Model Parallelism

6.1 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.

6.1.1 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.

6.1.2 Hierarchical Reduction of Data

When performing Data Parallelism, gradients can be sum-reduced in a hierarchical manner. Figure 5 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.

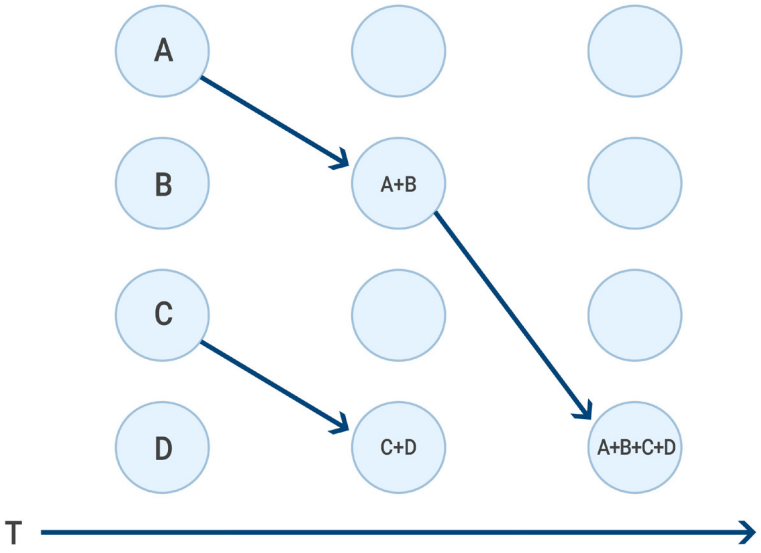

Figure 5: Hierarchical Reduction

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

6.2 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.

7. Building a Training Systems with Gaudi

7.1 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.

7.2 Habana Labs System-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 PCB, 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).

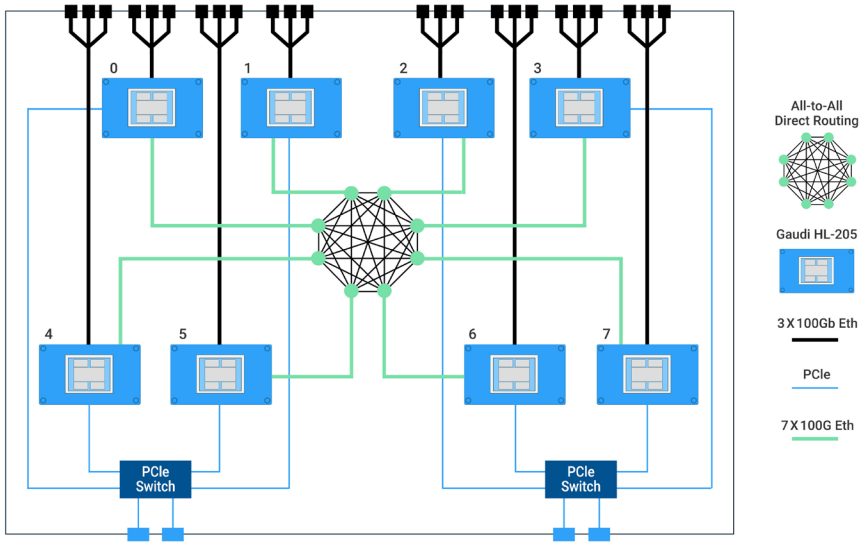

Figure 6: HLS-1: System Topology

7.3 Gaudi System with Maximum Scale-out

Figure 7 shows a system containing eight Gaudi chips and its interfaces. The interfaces are 4x16 PCIe Gen4 cables that can be connected to an external Host server, and up to 80X100Gb Ethernet links (using 20 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.

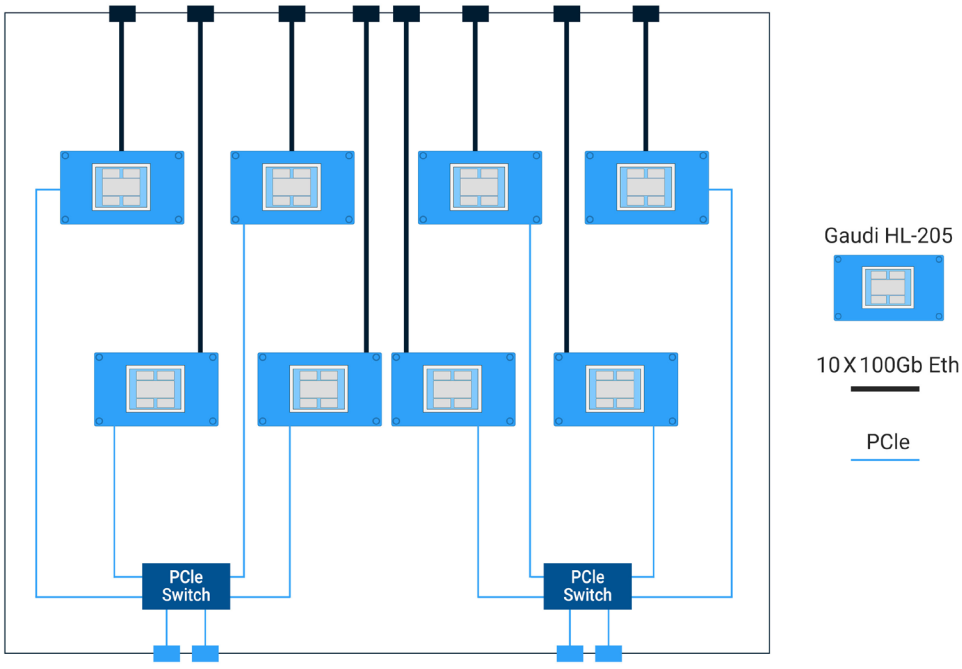

Figure 7: Gaudi System with Maximum Scale-out

7.4 Gaudi System with On-Board Ethernet Switch

Figure 8 shows a Gaudi system where the Ethernet switch is part of the integrated system.

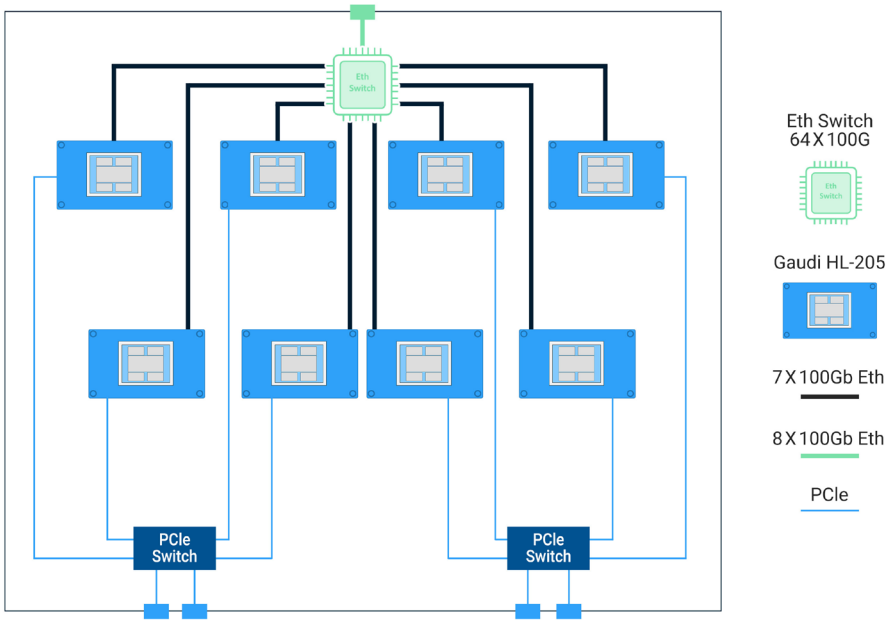

Figure 8: Gaudi System with an On-board Ethernet Switch

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.

7.5 Hyper-Cube Mesh System Topology

Some vendors who already designed a system to accommodate GPUs that had only six ports for scale out (like the V-100) may have already designed the interconnect for eight GPUs in a hyper-cube mesh to overcome the connectivity limitations of six ports. Such a system may be designed to accommodate OAM Mezzanine cards with the hope to accommodate cards from different vendors that may fit such a system.

As Gaudi has 20 Rx/Tx ports employing 56Gbps PAM4 SerDes, the HL-205 Mezzanine card supports the allocation of the 20 SerDes ports across the six channels defined in the OAM specification (3,3,4 on each of the two OAM connectors for a total of 20).

Gaudi devices are capable of passing Ethernet packets from one port to another, thereby enabling routing messages in a hyper-cube mesh.

While hyper-cube mesh is inferior to the HLS-1 in available throughput (HLS-1 has all-to-all direct connectivity), it is possible to use the hyper-cube mesh with most topologies. The partially blocking hyper-cube interconnect dictated by such designs may have some impact on actual training performance.

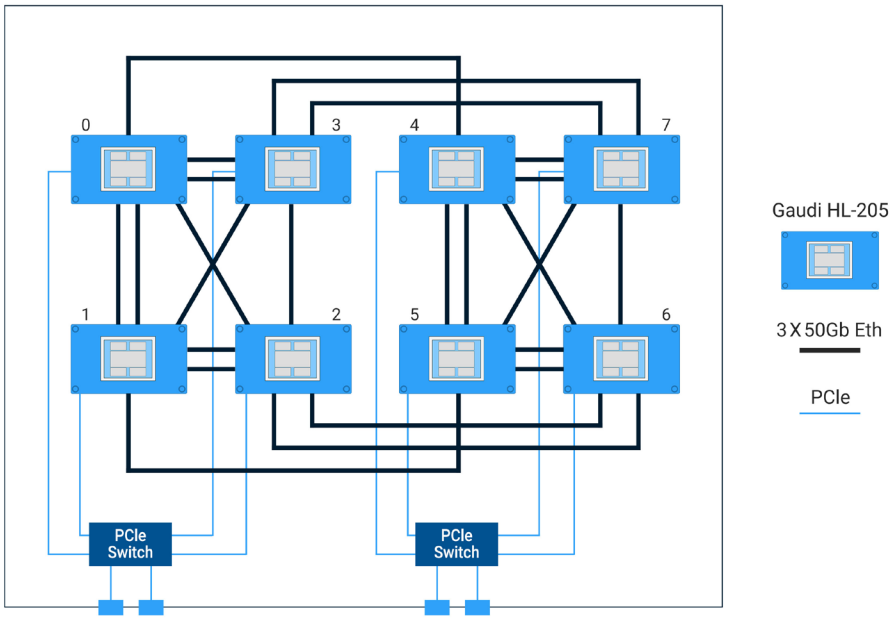

Figure 9: Hyper-mesh Cube Gaudi System

7.6 Very High-Performance System with 16 Gaudi Cards

Gaudi Ethernet ports can be configured to be 20 ports of 50GbE or any mix of 100GbE and 50GbE ports Therefore, it is possible to create a single motherboard interconnecting 16 Gaudi HL-205 Mezzanine cards directly, while still having five 50GbE ports per Gaudi for scale-out connectivity outside the box. If 100GbE is desired for external connectivity, then up-to two 100GbE ports can be provided from each HL-205 device (leaving one 50GbE port unused). The system topology that connects 16 Gaudi cards is shown in Figure 10.

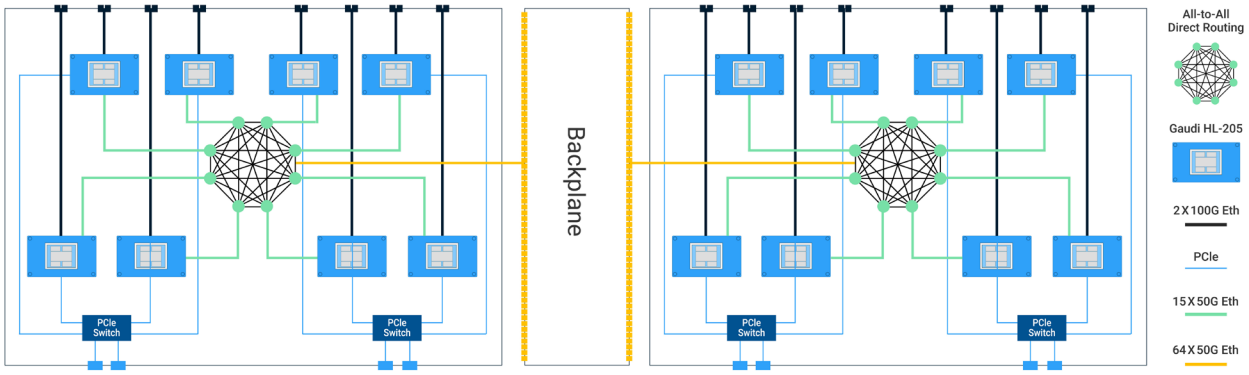

Figure 10: High-performance System with 16 Gaudi Cards

7.7 Gaudi-based Training Rack

Figure 11 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, such as described in previous examples. In this example, each HLS-1 is connected to the switch with four cables, carrying a total of 16x100GbE/8 = 2x100GbE per Gaudi system (a Gaudi system contains

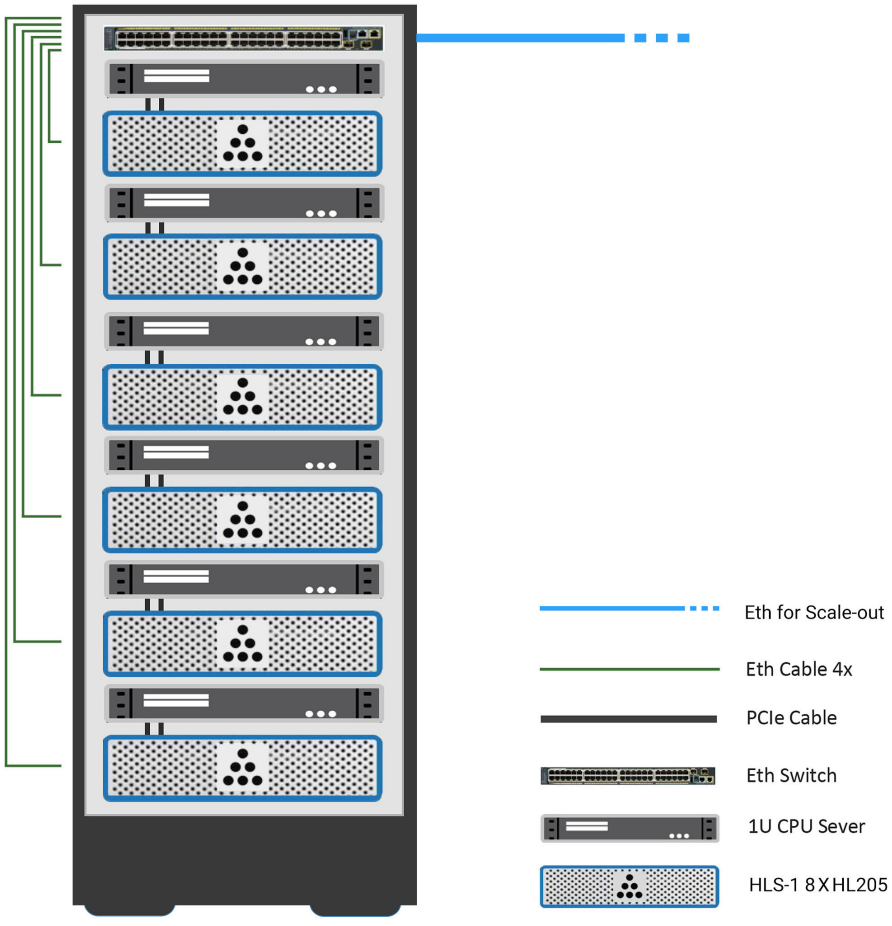

Figure 11: Example Rack

7.8 High-end 2K Gaudi System

In this example, each eight-Gaudi system has a companion 64-port Ethernet switch, as described in Figure 12. 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.

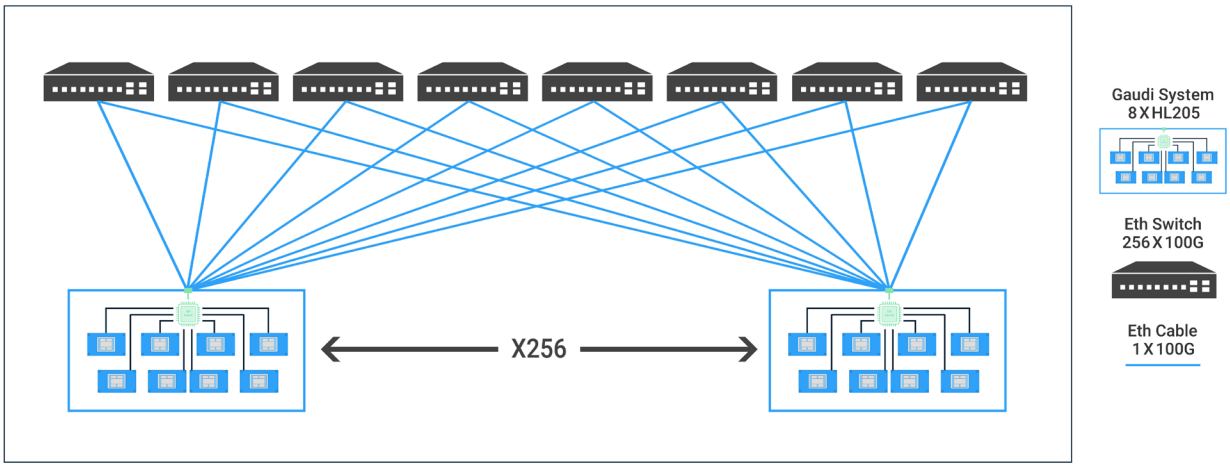

Figure 12: High-end 2K Gaudi System

7.9 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.

7.9.1 Topologies for Data Parallelism

Figure 13 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.

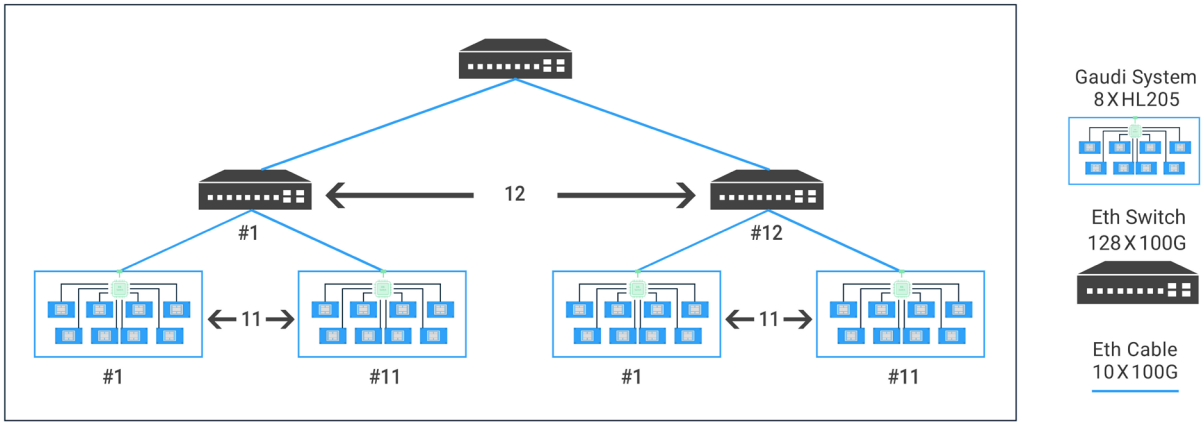

Figure 13: Hierarchical Fabric

7.9.2 Topology for a Combination of Data and Model Parallelism

Figure 13 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.

7.9.3 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 14 shows an illustration of such a large-scale system incorporating 64 Gaudi chips. A system with 128 Gaudi chips can be built in an analog manner, connecting a single 100GbE per Gaudi (thus 8x100 per Gaudi System) to a larger switching fabric, consisting of ten 128- port switches. Such a large-scale, single-hop system with full connectivity is only possible when connecting Ethernet directly to the Deep Learning Accelerator.

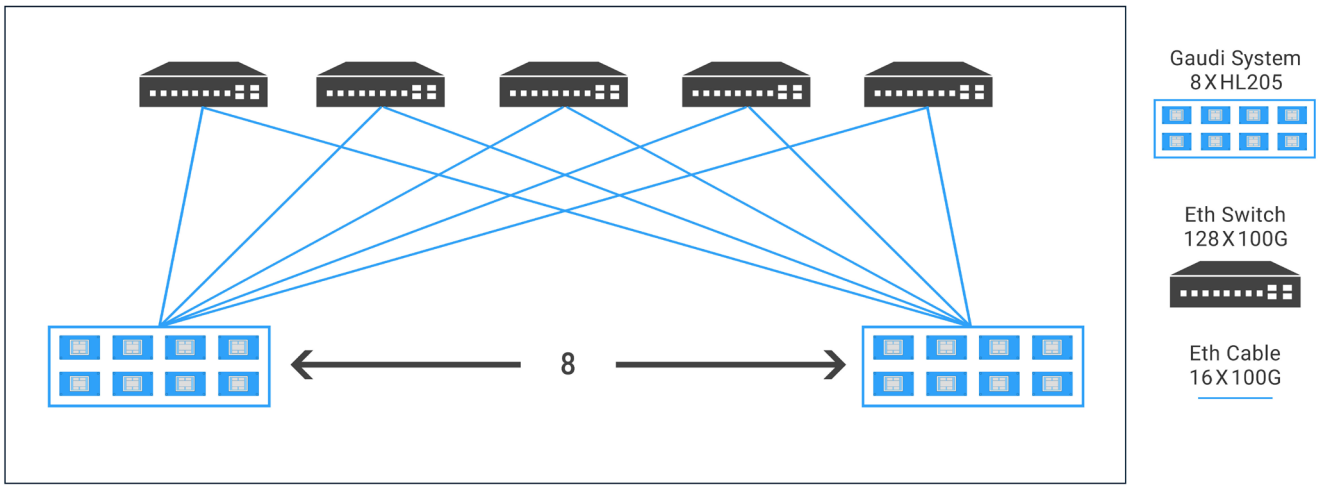

Figure 14: Fully Connected, Single-hop System

8. 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).

This whitepaper will be updated with benchmark results on Gaudi-based systems, for MLPERF and other benchmark tests, as new software releases are made available and new hardware systems based on Gaudi become available.

8.1 ResNet-50 Training 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 dissipating 140 Watt, delivers **1,650 images/second of training throughput**. The above throughput was achieved with a batch size of 64. Such a low batch size allows scaling the performance linearly to very large systems. Figure 15 shows the projected performance on Gaudi-based training systems, on the ResNet-50 benchmark as a function of the system size from an 8-Gaudi system to a 640-Gaudi system.

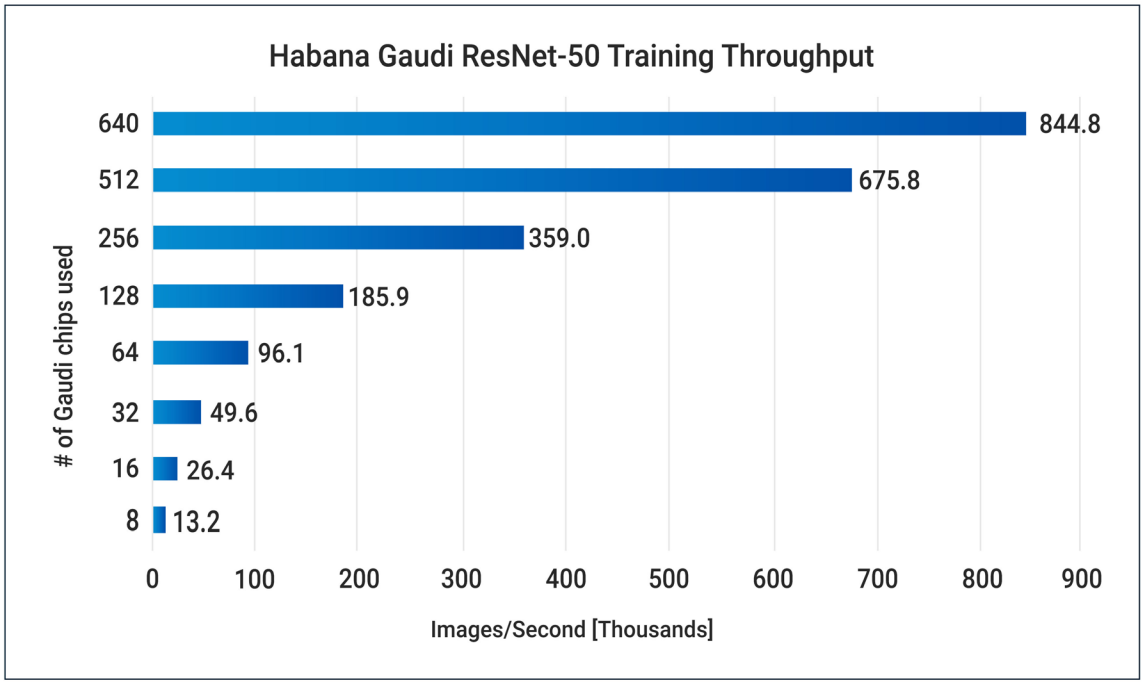

Figure 15: ResNet-50 Performance

One of the microarchitectural strengths of Gaudi is achieving record-setting throughput without using large batch size. This property is valuable when scaling the training system to a large number of processors where the Data Parallelism approach requires splitting the Minibatch to all the processors in every iteration. The more processors used, the smaller the batch will be needed to run on each individual processor.

For comparison, Figure 16 shows, based on nVidia's submitted results to MLPERF v0.5 for MxNet on 8,16, 512 (32 DGX-2h) and 640 (80 DGX-1) of V100 SXM based GPU systems, how reducing the batch size affected the V100's ability to sustain performance at scale.
A Gaudi-based system of 640 processors will deliver 3.8 times the throughput of a 640 V100-based system. This means that at-scale, HLS-1 based clusters will deliver 3.8x the throughput of DGX-1 systems on the ResNet-50 benchmark training.

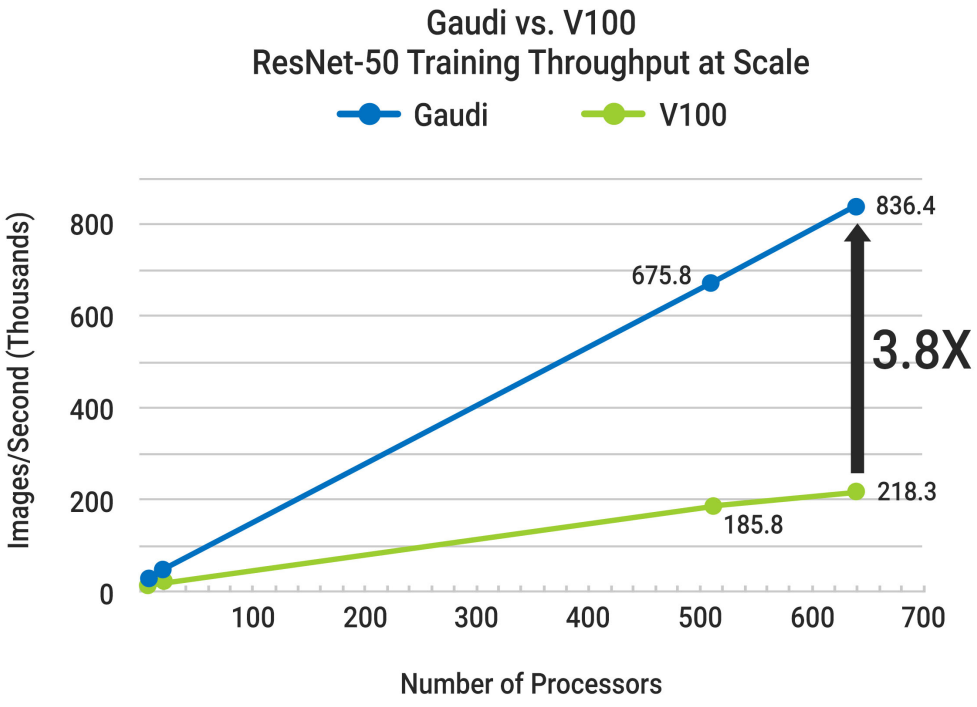

Figure 16: ResNet-50 Performance at Scale

The growing electricity usage in datacenters becomes a primary concern for datacenter operators and the general public. As datacenter operators are pressured to deploy more and increase their capacity of AI services worldwide by orders of magnitude, the total power consumption of AI training and inference systems must be taken into account when assessing processor performance and estimating the total cost of ownership. The following paragraphs address the power consumption of a single Gaudi card, as well as multi-Gaudi systems.

The actual power drawn by a single Gaudi-based PCIe card (HL-200) or Mezzanine card (HL-205), while running 1,650 frames per second in the ResNet-50 training benchmark, is just 140 Watts. Therefore, the power efficiency of a single Gaudi-based card is 1,650/140=11.9 Images/second/Watt.

9. 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 overnight or over lunch; second, to reduce energy consumption; third, to provide easier scalability with standard interfaces; and fourth, to achieve lower total cost of ownership.

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

- Performance leadership on key MLPERF benchmark that results in significantly lower training time 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;
  - End customers can avoid lock-in to any specific AI processor vendor, as with Gaudi they can use standard Ethernet as their scale-up and scale-out technology in the datacenter. This ensures that the datacenter customers retain the control over their infrastructure, rather than letting any single processor vendor control their future. With Habana's approach, Customers retain freedom to chose the best processor to meet their needs, over time.
  - On-chip integration of RDMA over Converged Ethernet (RoCE v2), in the processor itself, reducing system complexity, cost and power.
  - Lower system cost through wide availability of Ethernet switches of any size from multiple vendors.
  - Performance - Ability to scale to very large solutions as Ethernet has virtually limitless scalability.
  - 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).
  - Simplicity - Using the same technology to scale-up and scale-out from small systems to very large clusters
- Lower cost to maintain and operate compared to proprietary interfaces;
- Software stack designed to support popular frameworks;
- Rich TPC kernel library and user-friendly development tools that enable customization;
- 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 solution as they grow and evolve.