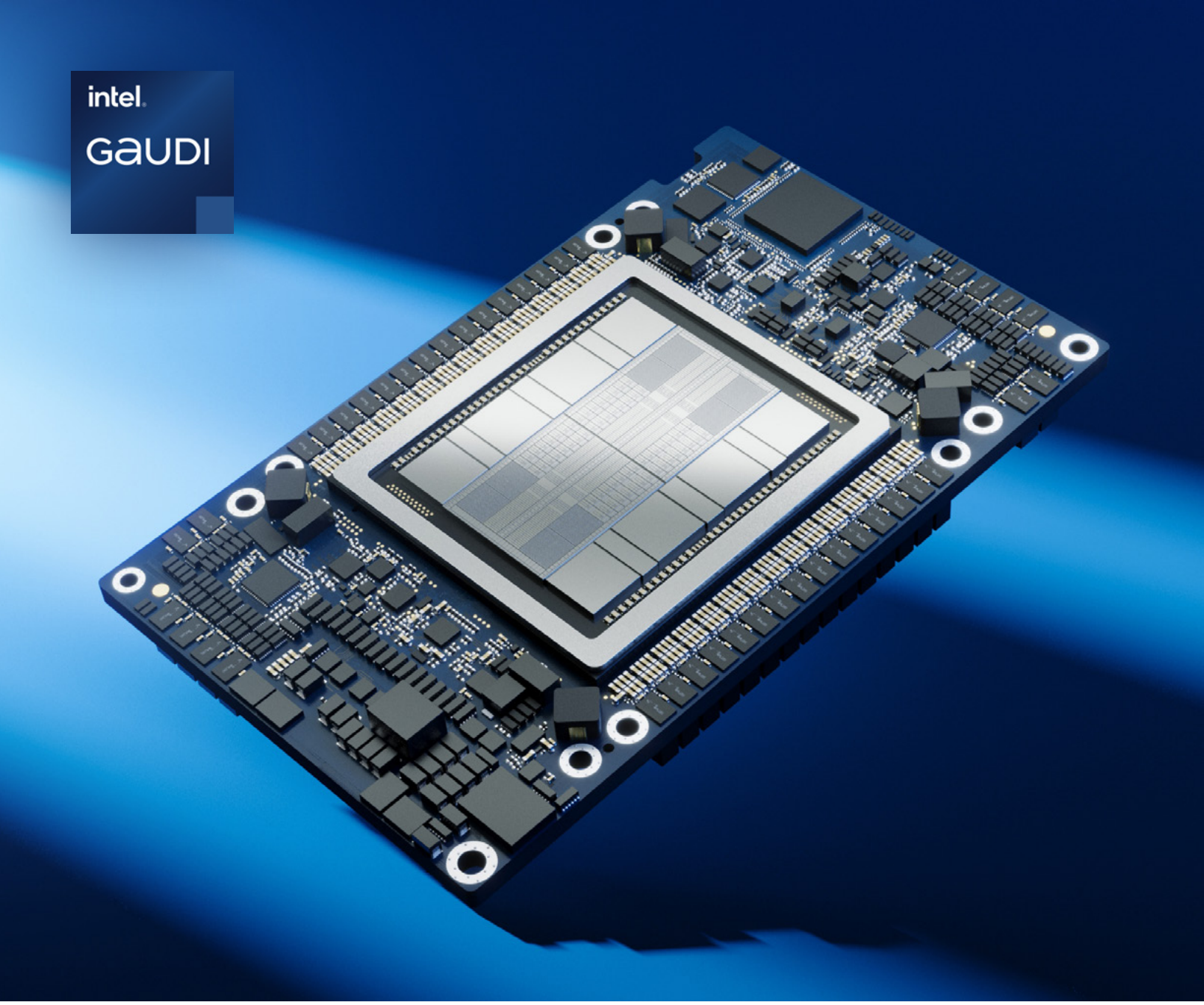


Technical Paper

Generative AI Compute
AI for Enterprise

Intel® Gaudi® 3 AI Accelerator

This technical paper introduces the next-generation AI accelerator from Intel®: The Intel® Gaudi® 3 AI Accelerator.

SEPTEMBER 2024 — Rev. 1

Table of Contents

Introduction

Deep Learning and Artificial Intelligence workloads continue to demand higher performance and lower power consumption. This technical paper introduces the next generation AI accelerator from Intel: the Intel® Gaudi® 3 AI Accelerator. The new accelerator features the 5th generation of heterogenous AI acceleration architecture. The Intel® Gaudi® 3 AI Accelerator was designed to provide state-of-the-art datacenter performance for all AI workloads, from generative applications such as large language models (LLMs) and diffusion models (image generation such as Stable Diffusion) to standard object recognition, classification, and voice dubbing.

The Intel® Gaudi® 2 AI Accelerator, introduced in 2022, is supported by the Intel® Gaudi® software suite, which integrates the PyTorch framework. With the Intel® Gaudi® 3 AI Accelerator we provide the next level of AI performance and power efficiency. Advancing from the Intel® Gaudi® 2 AI Accelerator 7nm process, the Intel® Gaudi® 3 AI Accelerator is manufactured in TSMC 5nm process, which provides improved area density and power efficiency.

Intel® Gaudi® 3 AI Accelerator continues to push the boundaries of what is possible in performance and power efficiency. Built on the Intel® Gaudi® 2 AI Accelerator architecture, Intel® Gaudi® 3 AI Accelerator provides significant boosts in compute, memory bandwidth, and architectural efficiency.

The Intel® Gaudi® 3 AI Accelerator features two compute dies, which together contain 8 MME engines, 64 TPC engines and 24x 200 Gbps RDMA NIC ports. In addition, the total of 8 HBM2e chips comprise a 128 GB unified High Bandwidth Memory (HBM).

The Intel® Gaudi® 3 AI Accelerator excels at training and inference with 1.8 PFlops of FP8 and BF16 compute, 128 GB of HBM2e memory capacity, and 3.7 TB/s of HBM bandwidth.

2x
FP8 GEMM FLOPs

4x
BF16 GEMM FLOPs

1.5x
Faster HBM Bandwidth

1.33x
Larger HBM Capacity

Gaudi® 3 AI Accelerator Overview

AI applications increasingly demand faster and more energy-efficient hardware solutions and the Intel® Gaudi® 3 AI Accelerator was designed to answer the demand. With more than 2x FP8 GEMM FLOPs and more than 4x BF16 GEMM FLOPs compared to the Intel® Gaudi® 2 AI Accelerator, Intel® Gaudi® 3 AI Accelerator continues to provide state-of-the-art AI training performance. With 1.5x faster HBM bandwidth and 1.33x larger HBM capacity, the Intel® Gaudi® 3 AI Accelerator provides an order-of-magnitude improvement in large language model inference performance compared to the Intel® Gaudi® 2 AI Accelerator.

The Intel® Gaudi® 3 AI Accelerator (Figure 1) features two identical compute dies, connected through a high-bandwidth, low-latency interconnect over an interposer bridge. The die-to-die connection is transparent to the software, providing performance and behavior equivalent to that of a large unified single die.

The Intel® Gaudi® 3 AI Accelerator compute architecture is heterogeneous and includes two main compute engines – a Matrix Multiplication Engine (MME) and a fully programmable Tensor Processor Core (TPC) cluster. The MME is responsible for doing all operations that can be lowered to Matrix Multiplication, like fully connected layers, convolutions and batched-GEMMs. The TPC, a Very Long Instruction Word (VLIW) Single-Instruction Multiple-Data (SIMD) processor tailor-made for deep learning applications, is used to accelerate all non-GEMM operations.

Intel® Gaudi® Accelerator Product Line

Intel® Gaudi® 2 to Intel® Gaudi® 3 AI Accelerator Feature Comparison.

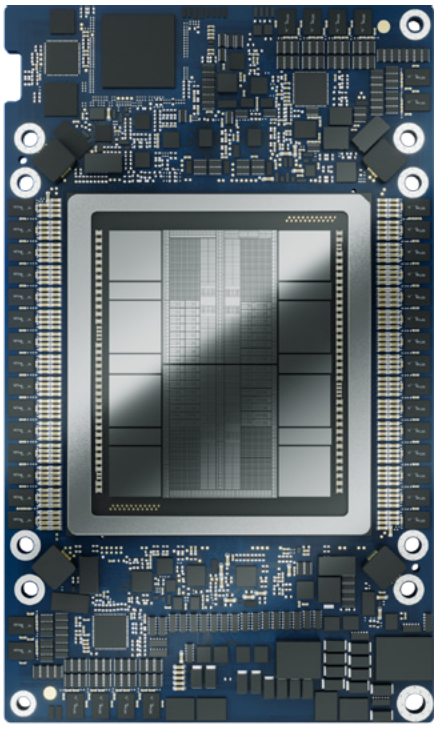
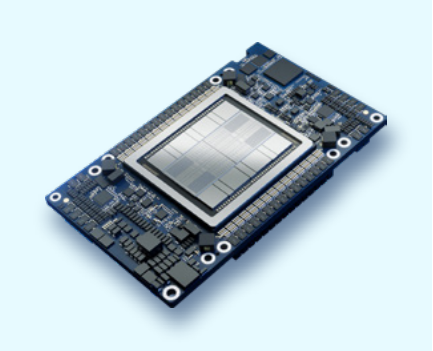


Fig 1. Intel® Gaudi® 3 OAM Module.

Feature/Product	Intel® Gaudi® 2 AI Accelerator	Intel® Gaudi® 3 AI Accelerator
BF16 MME TFLOPS	432	1835
FP8 MME TFLOPS	865	1835
BF16 Vector TFLOPS	11	28,7
MME Units	2	8
TPC Units	24	64
HBM Capacity	96 GB	128 GB
HBM Bandwidth	2.46 TB/s	3.7 TB/s
On-die SRAM Capacity	48 MB	96 MB
On-die SRAM Bandwidth	6.4 TB/s	12.8 TB/s
Networking (bidirectional)	600 GB/s	1200 GB/s
Host Interface	PCIe Gen4 X16	PCIe Gen5 X16
Host Interface Peak BW	64 GB/s (32 GB/s per direction)	128 GB/s (64 GB/s per direction)
Media Decoders	8	14

Table 1. Intel® Gaudi® 2 and Intel® Gaudi® 3 AI Accelerators.



HL-325L OCP Accelerator Module

HW System

HL-325L OCP Accelerator Module

The Intel® Gaudi®3 AI Accelerator OCP Accelerator Module (OAM) Card is offered to system designers in standard OCP OAM 2.0 Mezzanine card form and supports up to 900W Total Device Power (TDP) with passive cooling and up to 1.2KW TDP with liquid cooling.

Table 2 provides its key interfaces:

Interface	Description
Host Link	x16 PCIe Gen5
Networking:Card-to-Card & Scale-out	48 x 112 Gb/s PAM4 SerDes Links
JTAG	In-field CPLD programming and low-level ASIC debug
UART	Low level debug & BMC access
I2C Master	On/Off-board Peripherals
I2C Slave / SMBUS	BMC control and monitoring interface

Table 2. HL-325L OCP Accelerator Module Key Interfaces

HLB-325L Universal Baseboard

HLB-325L Universal Baseboard

The HLB-325 Universal Baseboard is another product inspired by Open Compute Project (OCP) and offered for simplifying system design with the Intel® Gaudi® 3 AI Accelerator. The HLB-325 supports eight Intel® Gaudi® 3 AI Accelerator cards that are passively interconnected on its PCB in a non-blocking, all-to-all configuration, using 21 NICs from each card (3x 200 GbE ports to every other of the 7 cards), as well as routing the 3 remaining 200 GbE NICs from every Intel® Gaudi® 3 AI Accelerator card (3x8=24) to the six on-board OSFP800 connectors for scaling-out.

The baseboard has standard interface/connectors to the HIB (Host Interface Board), which allows the system designer customization to design to specific needs and the flexibility to build systems of choice with a different ratio of CPUs to accelerators for different varieties of topologies and applications.

Block Diagram and Main Components

- HLB-325 has the following main components:
- 8 X dual B2B connectors for the HL-325 Mezzanine boards
- High speed connectors for x16 PCIe interconnect to HIB
- 2 Complex Programmable Logic Devices
- Power and reset control
- JTAG distribution to the mezzanines
- LED indications
- 6x OSFP connectors (6x800G using 112G PAM 8 SerDes)
- 3x PHY retimers
- 8x PCIe retimers
- USB connectors for Debug

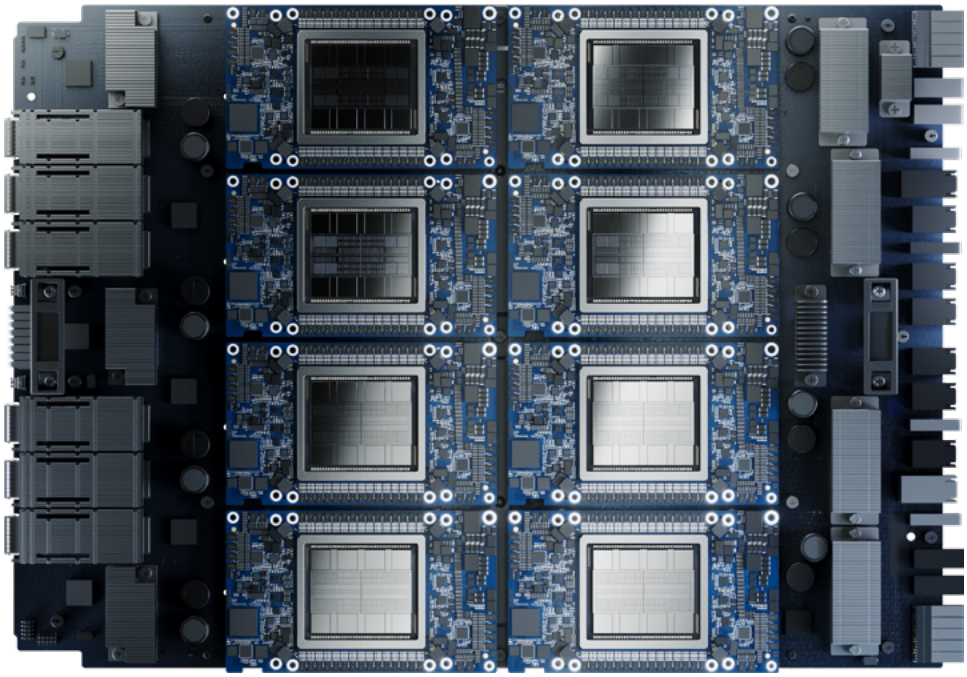


Fig 2. Key components of HLB-325.

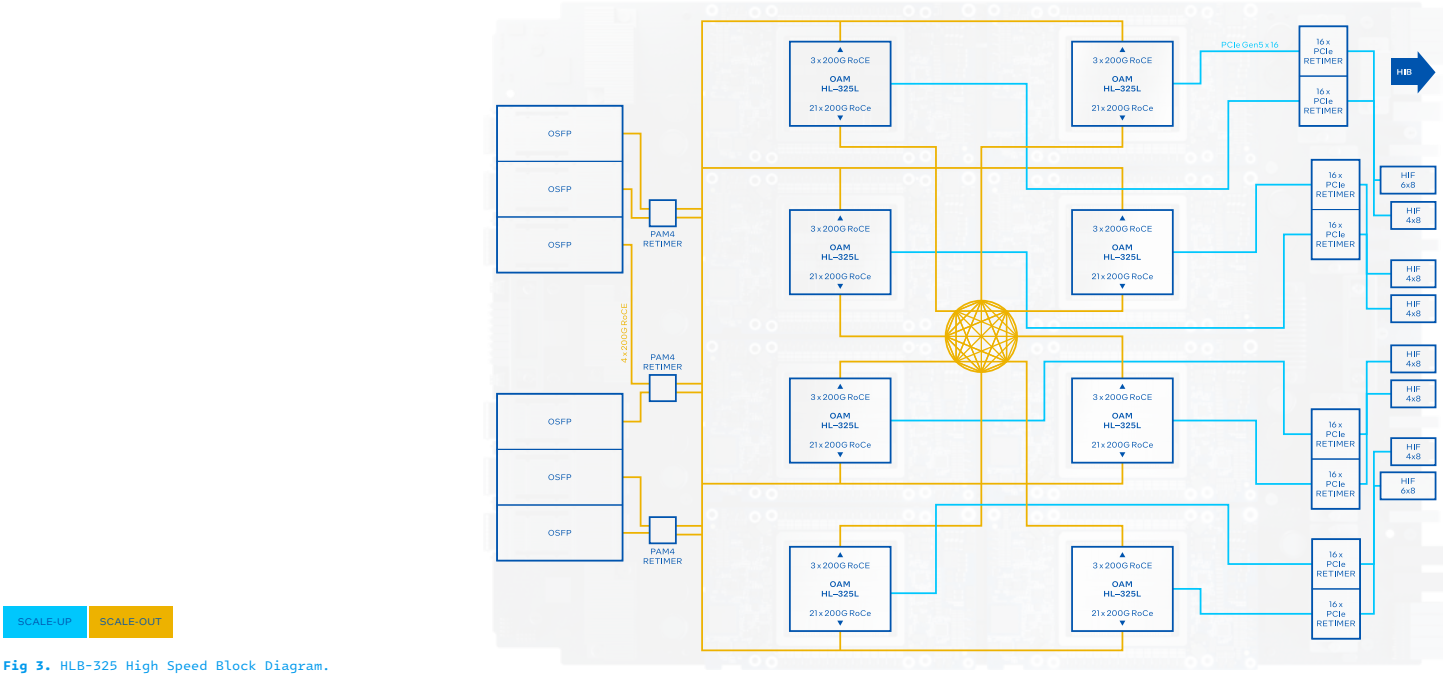
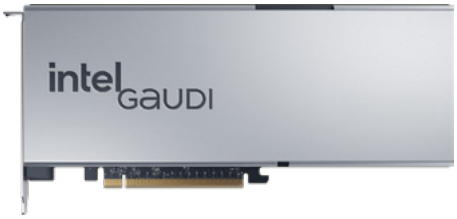


Fig 3. HLB-325 High Speed Block Diagram.

Feature	Description
OAM support	▪ OAM powered by 54V,12V and 3.3V ▪ Dual B2B connectors ▪ x16 PCIe Gen5 host interface per OAM ▪ 24x 200 GbE RoCE for scaleup and scale-out, via 48 112G PAM4 Serdes
Baseboard to HIB (Host Interface Board) Interface	▪ 8 X 16 PCIe Gen 5 connectors ▪ Power: 12V_Standby, 54V ▪ Side band signals: I2C, Reset, reference clocks, JTAG, UART, SGMII, USB ▪ Eight Amphenol connectors: 2x 160P (10131762-301LF) + 6 x 112P (10137002-101LF)
Networking: Card to Card & Scale-out	▪ Per OAM: 24x 200 GbE (48 112 Ghz PAM4 SerDes Links) split into: ▪ 21 x 200 GbE for OAM-to-OAM connections ▪ 3 x 200 GbE for scale-out ▪ Total Baseboard Scale-out: ▪ 8 x 3 x 200 GbE = 4.8 TbE connected to 6 OSFP800 ports
PCB dimension	▪ 585 mm x 417 mm x 4.6 mm

Table 3. HLB-325 Features



HL-338 PCIe Add-In Card

HL-338 PCIe Add-In Card

The Intel® Gaudi® 3 AI Accelerator PCIe Add-In Card is offered to system designers in accordance with PCIe CEM Spec. Revision 5.1 form and supports up to 600W TDP Power with passive cooling.

Table 4 provides HL-338's key interfaces:

Interface	Description
Host Link	x16 PCIe Gen5
Networking: ▪ Card-to-Card ▪ Scale-out	▪ 48 x 112 Gb/s PAM4 SerDes Links ▪ 2 x 400G QSFP112 ports
JTAG	In-field CPLD programming and low-level ASIC debug
I2C Slave / SMBUS	BMC control and monitoring interface

Table 4. HL-338 PCIe Key Interfaces

HLTB-304 x4 Top Board

The HLTB-304 board allows connectivity of 4 HL-338 cards, 6x 200 GbE links from each. HL-338 card to each of the other 3 HL-338 cards, 18 links of 200 GbE total per card.

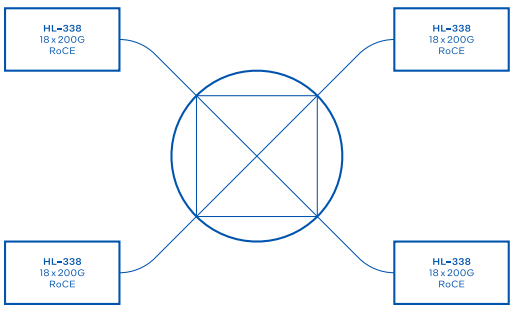


Fig 4. HLTB-304 Block Diagram.

Intel® Gaudi® 3 AI Accelerator Architecture

Parallel Execution of the Heterogenous Engines

Intel® Gaudi architecture was designed to allow activating all engines in parallel. This means that MME, TPC and NIC can all work at the same time.

The two main use-cases for running different engines in parallel are:

1. No dependency between the input and output of type of engine. In this case no special software intervention is needed. The Graph Compiler can simply trigger each engine to execute, providing the full input and output tensor sizes.
2. There is dependency between operations running on different engines: the output of one engine is used as the input of another engine.

The first case is simple and allows MME, TPC and NIC to be scheduled to run in parallel. When one engine has completed its executing operation, the engine can be scheduled to start working on the next operation (immediately upon readiness of its inputs).

The second case is more complex as it requires finer-grained scheduling, in addition to work size management that is done by the Intel® Gaudi software. In this case, the dependent engines are scheduled to execute in a pipelined manner with a producer-consumer relation. The engine scheduling and entire orchestration is done by the Graph Compiler. A more detailed explanation on how several software layers are combined to work together to achieve efficient engine scheduling and execution is presented in the following section.

Figure 5 shows the complete block diagram of the Intel® Gaudi® 3 AI Accelerator.

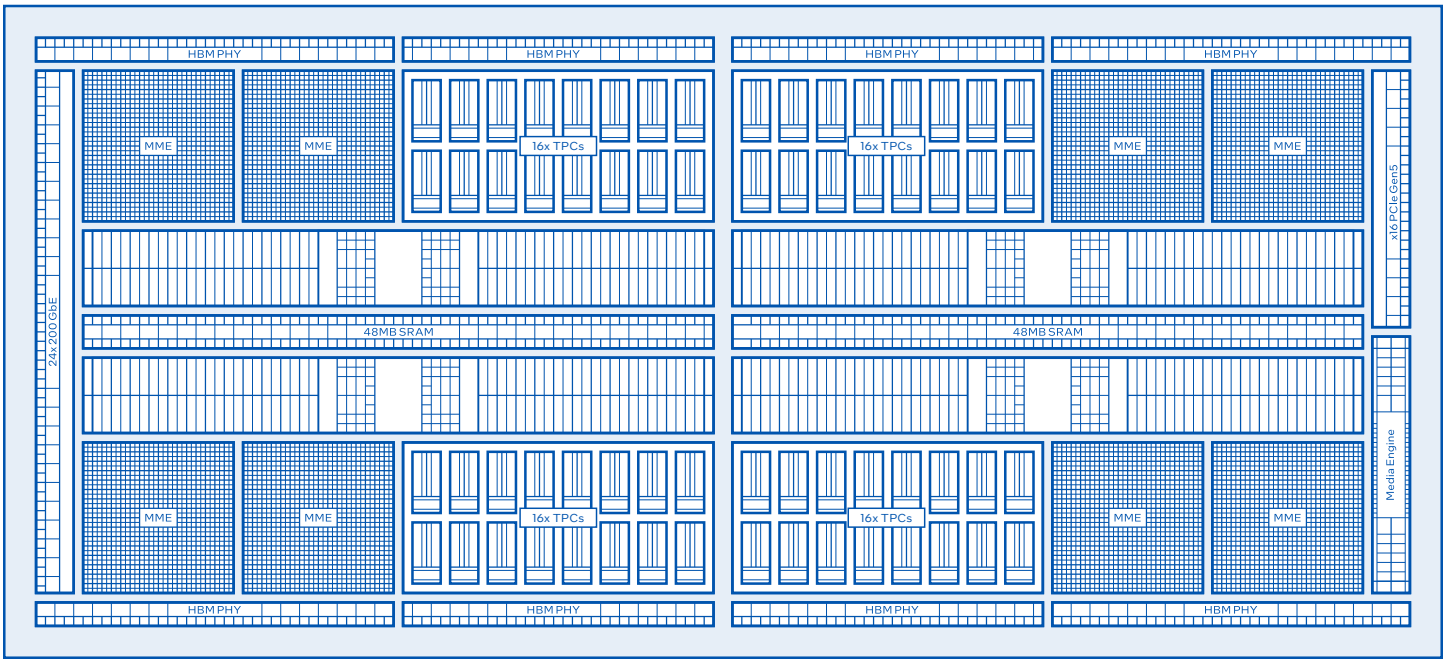
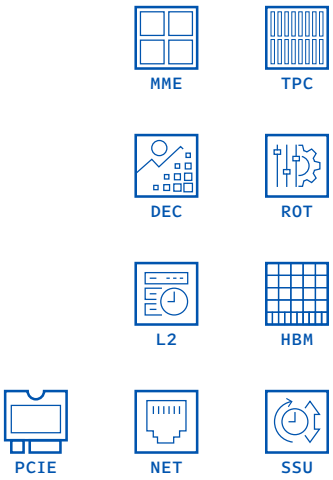


Fig 5. Intel® Gaudi® 3 AI Accelerator with L2 cache for every 2 MME and 16 TPC unit. Parts of L2 can be configured by the Graph Compiler to serve as shared L3.

Each of the components in the chip are explained in detail in the next chapters.

Full implementation of Intel® Gaudi® 3 AI Accelerator includes the following units:



Compute Engines
- 8 Matrix Multiplication Engines (MMEs)
- 64 Tensor Processor Cores (TPCs)

Media Engines
- 14 Media Decoder Engines (DECs)
- 4 Rotator Engines (ROT)

Memory
- 96 MB of L2 Cache
- 128 GB of 8 HBM2e stacks

Networking
- PCIe Gen5 X16 port for communicating with host
- 24 Network ports and the accompanied RDMA Engine
- Scheduling and Synchronization Unit

Physical partioning
Intel® Gaudi® 3 AI Accelerator compute engines are split into four clusters. Each cluster is referred to as a DCORE (Deep Learning Core) and contains:
- 2 Matrix Multiplication Engines (MMEs)
- 16 Tensor Processor Cores (TPCs)
- 24 MB of L2 Cache

Figure 6 reviews Intel® Gaudi® 3 AI Accelerator architectural elements with DCORE partition, Media Sub-system, Network sub-system and the connection with Host.

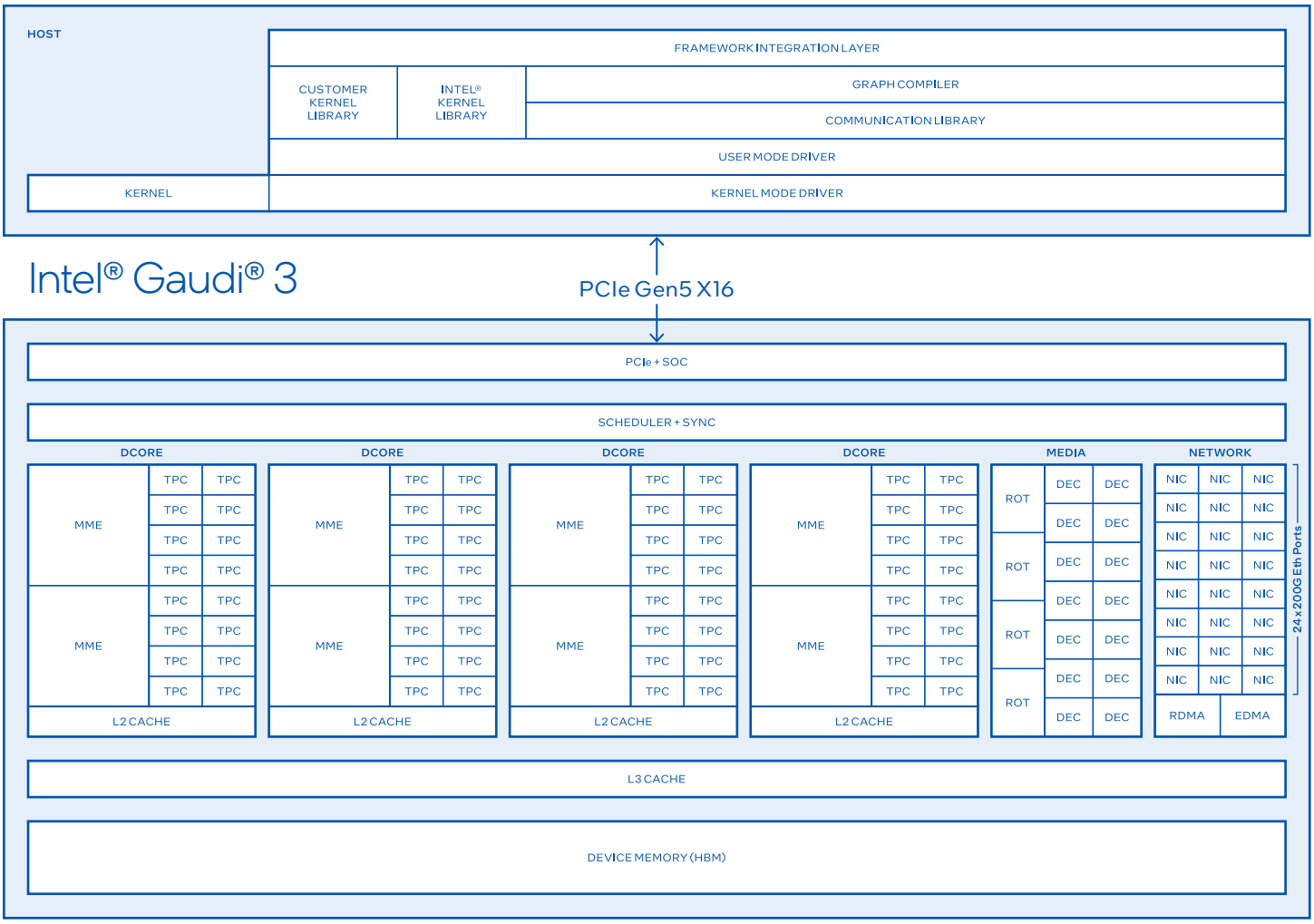


Fig 6. Intel® Gaudi® 3 architecture with DCORE point-of-view and supporting software layers.

Host Interface

Intel® Gaudi® 3 PCIe Card

Intel® Gaudi® 3 AI Accelerator is equipped with a state-of-the-art PCI Express Gen 5 x16 lane interface, a significant upgrade from the Gen 4 PCIe found in the prior generation accelerator. This advanced interface offers an impressive total bandwidth of 128 GB/sec, with 64 GB/sec available in each direction. This is a substantial improvement over the 64 GB/sec total bandwidth (32 GB/sec in each direction) provided by the Gen 4 PCIe.

The PCIe Gen 5 interface allows Intel® Gaudi® 3 AI Accelerator to seamlessly connect with the most powerful CPUs, external NICs, and SSDs available on the market. This ensures optimal performance and efficiency, making it a leading choice for high-performance computing solutions.

Intel® Gaudi® 3 Control Path

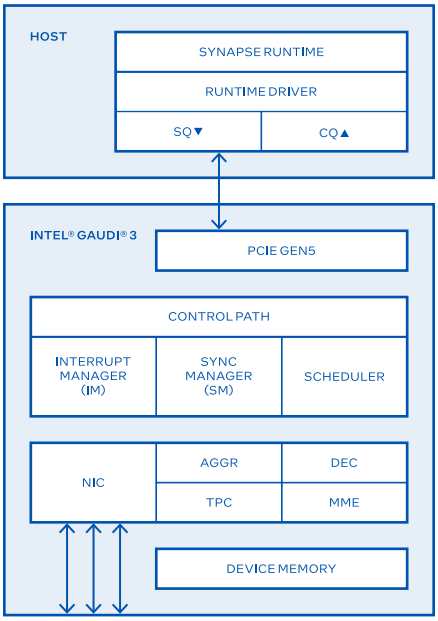


Fig 7. Control Path Block Diagram.

To manage the parallel and efficient execution of various engines, the Intel® Gaudi® 3 AI Accelerator incorporates a programmable Control Path entity. This entity is designed for high throughput and low latency. Figure 7 provides the primary components of this functionality.

The Control Path of Gaudi® 3 comprises the following elements:

- **Submission Queues (SQs):**
  These are issued by the runtime system.
- **Completion Queues (CQs):**
  These are used for job completion reporting.
- **Programmable Scheduling Mechanism:**
  This mechanism is utilized for task scheduling.
- **Programmable Hardware Synchronization Mechanism:**
  This is referred to as 'Sync Manager (SM)' in the diagram and is used for hardware synchronization.
- **Programmable Interrupt Service Mechanism:**
  This mechanism, referred to as 'Interrupt Manager (INTR)' in the diagram, enables the passing of asynchronous events to Habana Drivers.

Each of these components plays a crucial role in ensuring the smooth and efficient operation of Intel® Gaudi® 3 AI Accelerator engines.

For controlling parallel and efficient executions of the various engines, the Intel® Gaudi® 3 AI Accelerator includes a programmable low-latency, high throughput Control Path entity. Figure 7 illustrates the main building blocks of this functionality.

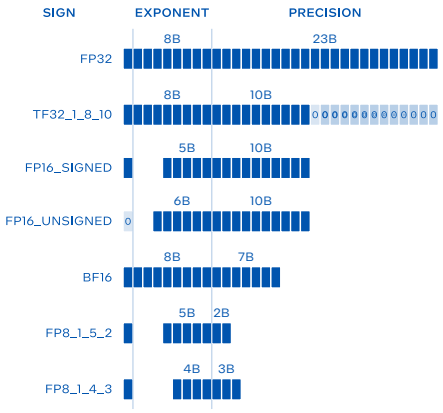


Fig 8. Supported floating-point datatypes.

Compute

Figure 8 showcases the range of Floating Point Data types that are supported by the Intel® Gaudi® 3 AI Accelerator engines, demonstrating its versatility and adaptability.

Table 5 provides a detailed breakdown of the peak operations per second for both matrix multiplication (performed by MME) and vector processing (performed by TPC). This information underscores the impressive computational power of the Intel® Gaudi® 3 AI Accelerator.

	Intel® Gaudi® 3 AI Accelerator		
Computation Type	Datatype	OAM Peak TFLOP/sec	PCIe Peak TFLOP/sec
MME (Matrix)	FP8	1835	1835
	BF16	1835	1835
	FP16 (signed)	459	459
	TF32	459	459
	FP32	229	229
TPC (Vector)	FP8	57.3	57.3
	BF16	28.7	28.7
	FP16	28.7	28.7
	FP32	14.3	14.3

Table 5. Intel® Gaudi® 3 OAM and PCIe matrix and vector compute capabilities.

Intel® Gaudi® 3 MME

MME Intro

The Intel® Gaudi® 3 AI Accelerator Matrix Multiplication Engine (MME) represents the 5th Generation of the Intel® Gaudi® Accelerator family MME Engines. These MMEs are specialized, high-performance compute cores, specifically designed for matrix operations, a type of computation that is fundamental to deep learning algorithms. The Intel® Gaudi® 3 AI Accelerator houses eight such MMEs, each capable of performing an impressive 64K parallel operations. This massive parallelism allows for a high degree of computational efficiency, making these MMEs particularly adept at handling the complex matrix operations prevalent in deep learning workloads.

The MMEs in Intel® Gaudi® 3 AI Accelerator have been tailored for efficiency in multiplication operations performed on current deep learning models. They feature a rich programmer's model that enables flexibility when distributing a job among the various MMEs and providing memory directives to maximize MACs utilization.

As deep learning models continue to increase in size and complexity, the demand for efficient, high-performance matrix multiplication engines is set to rise. The MMEs in solutions like Intel® Gaudi® 3 AI Accelerator are therefore of critical importance to the ongoing advancement of deep learning technologies.

MME Architecture

The Intel® Gaudi® 3 AI Accelerator is a powerhouse of computational capability, housing eight Matrix Multiplication Engines (MMEs). Each of these engines is equipped with 64K Multiply-Accumulate Units (MACs), which collectively enable a peak throughput of over 200 Teraflops per MME. This high throughput underscores the impressive performance potential of the Intel® Gaudi® 3 AI Accelerator.

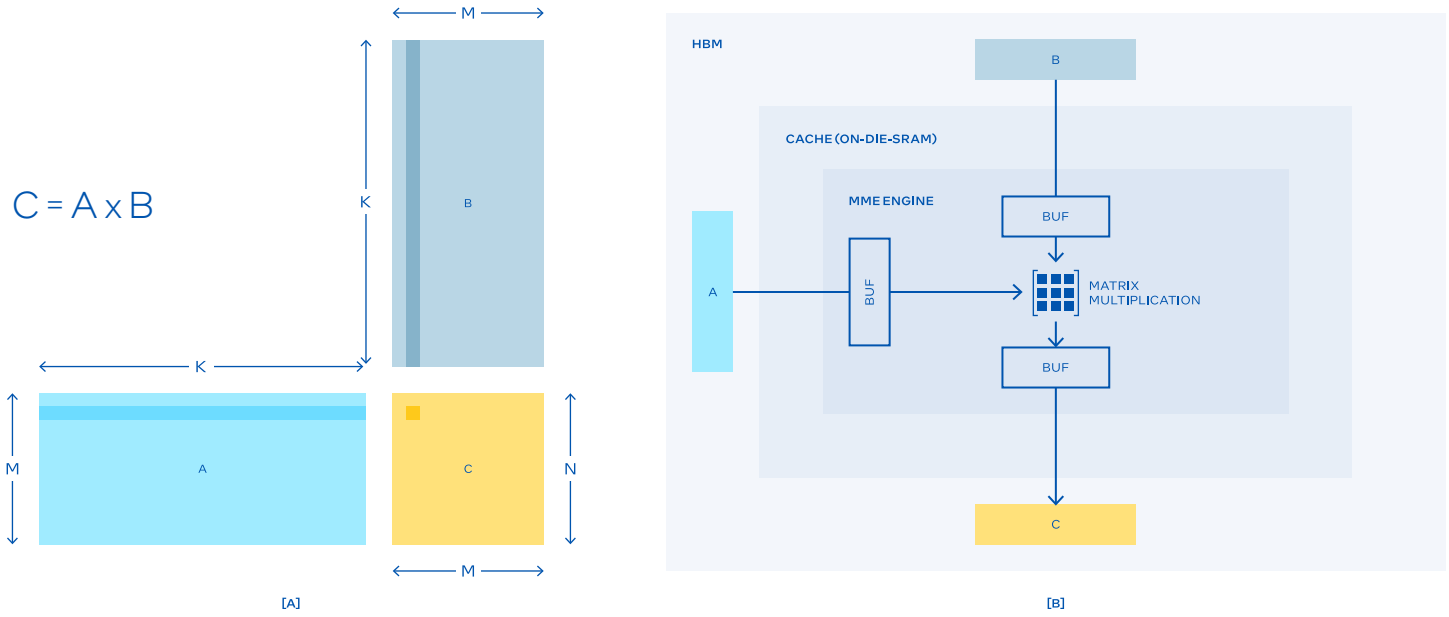


Fig 9. General matrix multiplication and its mapping to the MME engine block diagram.

Figure 9 offers a view of the functionality of a single engine. This visual representation can help users better understand the intricate workings of the MMEs and their role in the overall performance of the Intel® Gaudi® 3 AI Accelerator. With this knowledge, users can fully leverage the capabilities of the accelerator for their computational needs.

Figure 9A presents an algorithmic depiction of a General Matrix Multiplication (GEMM) operation, specifically an AxB multiplication. This operation generates tensor C[NxM] from two input tensors, A[NxK] and B[KxN]. Remember that in matrix multiplication, each computed element is the dot product of a row in A and a column in B, as demonstrated by the darker shades in the three tensors.

Figure 9B displays a block diagram detailing the data flows. The MME is programmed with the necessary dimensions, locations, data types, and various execution operands. It then retrieves tensors A and B from memory, pulling them into its streaming buffers for the matrix multiplication. The matrix multiplication can execute up to 64K Multiply and Accumulate operations in parallel. Upon completion, it will push tensor C back to memory. The memory system comprises a cache and the actual HBM memory. Each of these tensors can be independently pulled or pushed to the on-die SRAM, irrespective of the MME behavior. For more information, refer to the Memory section. The eight MME engines can be programmed together to perform a larger job. The following diagram represents 8 MMEs.

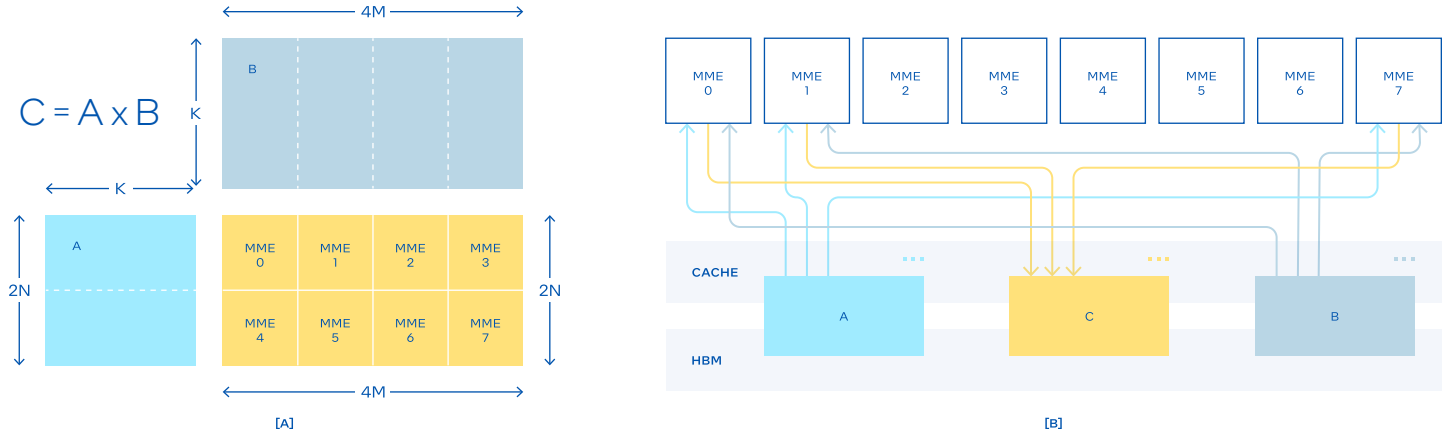


Fig 10. General matrix multiplication and its mapping to the MME engine block diagram.

Figure 10A represents 8 MMEs and illustrates an algorithmic process where an AxB matrix multiplication is divided among the eight MMEs. Each MME performs an NxM slice of the job, sharing a common dimension of K. The Intel® Gaudi® 3 AI Accelerator memory subsystem, along with its runtime optimizations, ensures that, when possible, any reused data is fetched only once from the HBM. For instance, mme0, mme2, mme4, and mme6 all pull from the upper part of tensor A, while mme0 and mme1 share a quarter of tensor B. The HL GC Runtime ensures that when needed, fetched data is stored in cache.

It's worth noting that other dimension splits are possible, and the Graph compiler analyzes the different options to choose the most efficient setting.

Figure 10B shows a block diagram detailing the data flows. The MMEs can operate in parallel, each fetching its required subset of A and B and producing its NxM subset within C. The eight MMEs in Intel® Gaudi® 3 AI Accelerator enable parallel performance of 0.5M operations, achieving up to 1.8 TB/s.

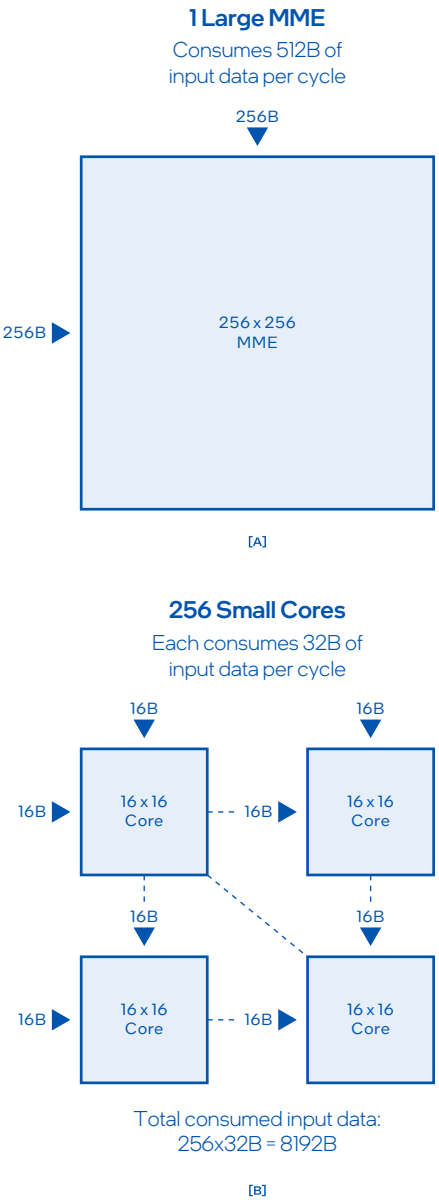


Fig 11. Comparison of one large MME to 256 small cores. Despite having the same compute capabilities, MME consumes 16× less input data than the smaller cores.

Benefits of One Large Matrix Multiplication Unit Over Multiple Small Units

As mentioned above, Intel® Gaudi® 3 AI Accelerator features eight large MMEs with each MME performing 64k MACs per cycle. Comparing the MME to modern GPUs that were modified for AI workloads, the Intel® Gaudi® 3 AI Accelerator features a small number of large matrix multiplication units, while GPUs contain a large number of small matrix multiplication units. The following diagram compares the two options for GEMM accelerators: one large unit vs. multiple small units.

Figure 11 compares a single Intel® Gaudi® 3 AI Accelerator MME with 64k Multiply-Accumulates (MACs) per cycle to 256 small GEMM cores each with 256 MACs/cycle, which amount to 64k MACs/cycle. This diagram assumes the MME and cores are organized as squared 2D matrices. The MME contains 256 rows over 256 columns, and the small cores contain 16 rows over 16 columns each. The comparison assumes input datatype of FP8, which requires 1 byte per input element.

The compute capabilities of the two options in Figure 11 are equivalent – both can perform 64k MACs/cycle. However, from a bandwidth perspective the two options significantly differ. Figure 11A shows that the large MME requires two sets of 256B inputs per cycle, summing up to 512B per cycle. On the other hand, Figure 11B shows each of the small cores requires two sets of 16B inputs per cycle, summing up to 32B per core per cycle. The total amount of input data that is required to feed all the 256 small cores is 256 times 32B, which amounts to 8192B. This is 16 times more than what a single large MME requires.

The smaller amount of required input data by the MME translates to multiple advantages. The 16x reduction in input bandwidth translates to less data transfers and higher energy efficiency. Second, the large requirement for input bandwidth puts constraints on the minimal GEMM dimensions that allow the system to reach high compute utilization. For example, to reach 80% compute utilization on modern GPUs with many small matrix multiplication cores, a GEMM dimension of m=n=k=~3K is required. In the Intel® Gaudi® 3 AI Accelerator, m=n=k=1K is sufficient to utilize 100% of the MACs. If activations are pipelined via 96 MB L2 cache (which is usually the case), m=n=k=512 is sufficient to utilize MME by 100%. In other words, Intel® Gaudi® 3 AI Accelerator requires between ~25x-~200x less MACs in a GEMM operation to reach 100% compute utilization compared to modern GPUs which reach only 80%. Paradoxically, we see that creating a relatively large matrix multiplication accelerator allows hardware to be efficiently utilized on smaller GEMM sizes compared to the alternative.

MME Data Types

The Intel® Gaudi® 3 AI Accelerator MME supports all the key AI compute datatypes: FP8 (both E4M3 and E5M2), BF16, FP16, TF32 and FP32. All datatypes are accumulated into an FP32 accumulator.

As FP8 becomes the favored compute datatype for training and inference, Intel® Gaudi® 3 AI Accelerator's 5th generation MME integrates on-the-fly FP8 input scaling, reducing the compute load requirements of the TPC for scaling to/from FP8.

Intel® Gaudi® 3 TPC

Tensor Processor Core Introduction

The Intel® Gaudi® 3 AI Accelerator integrates the 5th-generation Tensor Processor Core. The TPC is a general-purpose single instruction, multiple data (SIMD) VLIW processor. It is 256B wide and supports FP32, BF16, FP16 & FP8 (both E4M3 and E5M2) datatypes. In addition, the following integer datatypes are supported: UINT32, INT32, UINT16, INT16, UINT8 and INT8.

As opposed to common DSPs, which require a DMA to fetch in and out the operands to a local SRAM, the TPC exposes a DMA-free programming model, achieved by advanced micro-architectural techniques, which significantly eases software development. In addition, the same advanced microarchitecture allows consecutive execution, free of idle time, between kernels. This allows 100% runtime utilization of the TPC, even for micro-second scale kernels, regardless of the location of its inputs and outputs (cache or DRAM). Just like the MME, the TPC reaches high compute utilization even when working on small-sized inputs.

TPC Architecture

Figure 12 represents TPC Block diagram and illustrates its functionality.

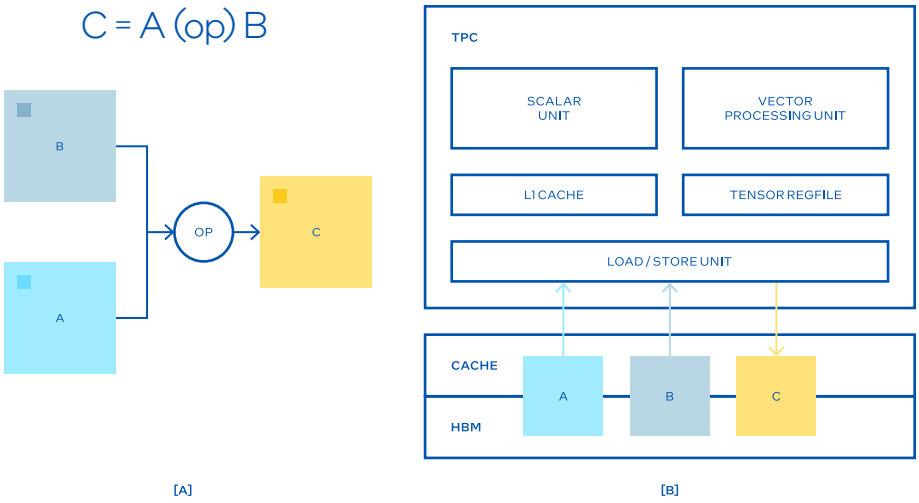


Fig 12. General matrix multiplication and its mapping to the MME engine block diagram.

Intel® Gaudi® 3 AI Accelerator Media Engine

DECODER ENGINE

Intel® Gaudi® 3 AI Accelerator has fourteen media decoding units. The following formats are supported.

Level 4 Video Formats
- HEVC up to 10 profiles, up to 8192x8192 resolution
- Progressive H.264 & SVC base layer & MVC up to 4096x4096 resolution
- VP9 up to profile 2 (10-bit) up to 8192x8192 resolution

Image Formats
- JPEG up to 8192x8192 resolution
- Progressive JPEG up to 8192x8192 resolution

In addition to decode, the block also supports the post processing of the streams.

Post Processing Features
- Image down-scaling (resizing the image):
- Vertical and horizontal scaling can use different scaling ratios
- Maximum output picture size of 4096x4096
- Image up-scaling (up to x3):
- Vertical and horizontal scaling can use different scaling ratios
- Maximum output picture size if 4096x2160
- Image cropping:
  - Use definable 4-pixel accuracy crop parameter setting of start, position width and height
  - Digital zoom
  - Supported by combining crop and upscaling.
- The PP supports bilinear scaling, Lancos scaling

Intel® Gaudi® 3 AI Accelerator implements two post processing channels per decoder block, one with scalar (up and down) and one just to output the original image.

Performance
Overall performance across all the hardware instances is show in the Table 7.

Formats Supported
Video decoder supports the following features:

Feature	Support
Input stream format	▪ YCbCr444, 422, 420 ▪ YCbCr440, 411, 400
Output stream format	YCbCr420 or RGB/BGR packet or per planer

Table 7. Decoder Formats.

Video Format*	1080p30 Streams
HEVC	250
VP9	300
H.264	200
Image Format*	**1080 img/sec**
Jpeg 420	12000

* Please note the actual performance of the decoder depends on various factors such as image resolution, image quality and format.

Table 6. Format performance.

ROTATOR ENGINE

The Intel® Gaudi® 3 AI Accelerator integrates a hardware rotator engine which allows performing the following transformations of an input image:

- 2D rotation
- 3D rotation
- Projection
- Mesh: Distort and undistort images
- Re-sampler: Re-samples input data at user-defined coordinates
- Re-scale with polyphase filter

Intel Gaudi® 3 AI Accelerator Memory Attributes

The Intel® Gaudi® 3 AI Accelerator on-die memory has two main advancements over its predecessor accelerator. The first is 2x size increase, from 48 MB to a total of 96 MB of on-die SRAM. The second advancement is the integration of two-level cache. The on-die 96 MB of SRAM can be used as a uniformly accessible last-level cache (L3) or split to 4 slices of 24 MB L2 cache each, with each slice accessible to 2 MMEs and 16 TPCs. L2 provides 2x higher cache I/O throughput compared to L3. Using the on-die memory as L2 or L3 cache is fully configurable by the Intel® Gaudi® software stack, which dynamically decides per I/O tensor its optimal cache allocation.

Intel® Gaudi® 3 AI Accelerator integrates 8 HBM2e devices running at 3.6GHz frequency, providing 3.7 TB/s peak HBM bandwidth, 50% higher than the Intel® Gaudi® 2 AI Accelerator. Each HBM2e device capacity is 16 GB, reaching a total 128 GB, 33% higher than the second generation accelerator and 1.6x higher than competing GPU solutions having only 80 GB of HBM memory.

The advantages of larger memory capacity are two-fold. One advantage is enablement of execution of configurations that require more devices with smaller HBM capacity; the other is use of configurations that are more compute-efficient, such as increased batch size or avoidance of precomputation.

In the rapidly evolving landscape of Deep Neural Network (DNN) acceleration, the Intel® Gaudi® 3 AI Accelerator stands out with its innovative memory subsystem. This subsystem is a critical component of our product, designed to work in harmony with Matrix Multiplication Engines (MMEs) and Tensor Processor Cores (TPCs) to deliver unparalleled performance.

	Gaudi® 2 OAM	Gaudi® 3 OAM
PCIe	Gen4 x 16	Gen5 x 16
PCIe Peak BW	64 GB/s bidirectional	128 GB/s bidirectional
HBM	6 x HBM2E	8 x HBM2E
HBM Capacity	96 GB	128 GB
HBM Peak BW	2.46 TB/s	3.7 TB/s
On-die-SRAM	48 MB	96 MB
On-die-SRAM BW	6.4 TB/s	19.2 TB/s
TDP	600 W	900 W

Table 8. Gaudi 2 OAM to Gaudi 3 memory attributes.

VIRTUAL SPACE ACCESSIBILITY

At the heart of the Intel® Gaudi® 3 AI Accelerator memory subsystem is a Memory Management Unit (MMU) that allows users to operate in a virtual space when accessing VRAM. This feature abstracts the complexities of memory management, providing a seamless user experience.

ADVANCED CACHING SYSTEM

The Intel® Gaudi® 3 AI Accelerator memory subsystem is equipped with L2 and L3 caches, which are coupled to each DCORE and HBM memory channels, respectively. The cache system is designed to optimize data access with several key features:

- High Throughput: The system provides a total throughput of up to 19.2 TB/s for L2 accesses and 6.4 TB/s for L3 accesses.
- Large Capacity & Set-Associativity: With a capacity of 96 MB and 12-way set-associativity, the cache system can handle large volumes of data effectively.
- Allocation Hints: Users can specify whether to cache in L2, in L3, or both, offering greater control over data management.
- Age Replacement Algorithm: The system uses an age replacement algorithm that considers user-defined classes and priorities, ensuring efficient use of cache resources.
- Maintenance Commands: These commands enhance cache utilization and prevent unnecessary data from consuming HBM resources.

HIGH BANDWIDTH MEMORY INSTANCES

The Intel® Gaudi® 3 AI Accelerator memory subsystem includes 8 High Bandwidth Memory (HBM) instances, providing a total capacity of up to 128 GB and a total bandwidth of 3.7 TB/s. This substantial capacity and throughput ensure that the system can handle large volumes of data effectively.

In conclusion, the Intel® Gaudi® 3 AI Accelerator memory subsystem is a testament to our commitment to pushing the boundaries of DNN acceleration. Its advanced features and high performance make it an integral part of our product, enabling us to deliver a solution that meets the demanding needs of today's DNN applications.

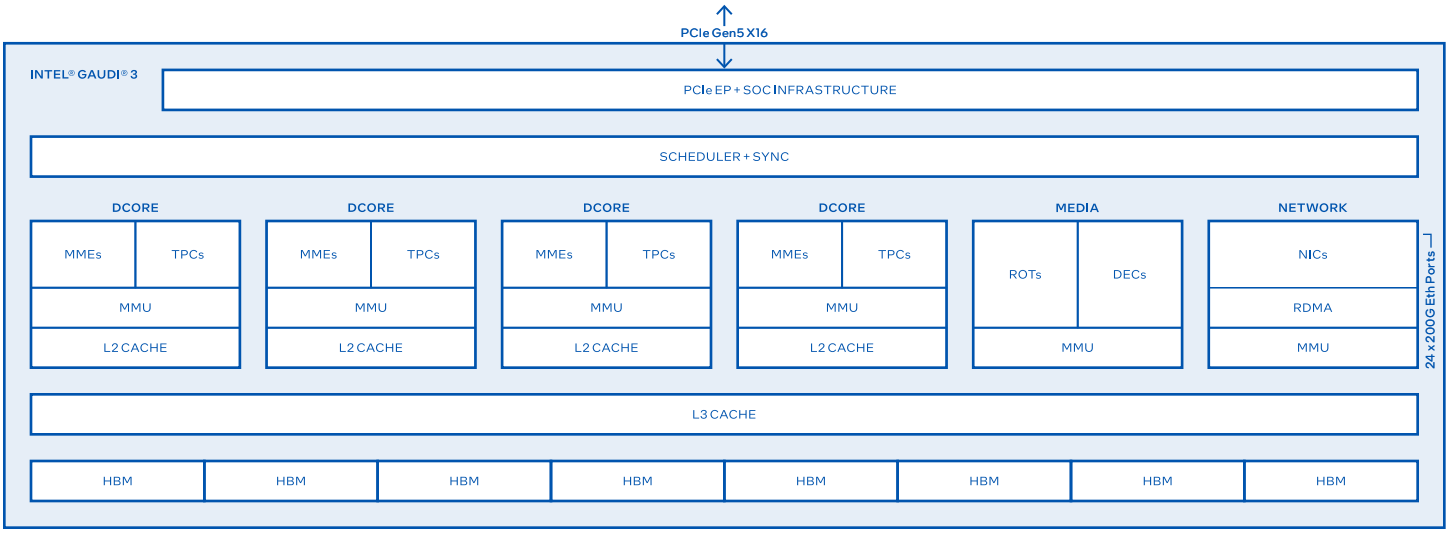


Fig 13. Intel® Gaudi® 3 architecture with DCORE point-of-view and supporting software layers.

Networking

The integration of RDMA over Converged Ethernet on the Intel® Gaudi® 3 AI Accelerator delivers distinct advantages enabling massive and flexible scaling from a single node to thousands. To express the advantages of the solution's scaling capabilities, it's essential to commence at the foundation of the network – the networking architecture contained in the Intel® Gaudi® 3 AI Accelerator.

The Intel® Gaudi® 3 AI Accelerator's revolutionary NW Sub-system, the powerhouse behind seamless data movement and efficient task management. At its core features the Intel® Gaudi® Communication Library (IGCL), a master conductor that orchestrates data movement. Our system is equipped with a programable scheduling mechanism, ensuring smooth activation of engines while maintaining task dependencies.

The Intel® Gaudi® 3 AI Accelerator networking sub-system boasts 24 200 Gigabit Ethernet NIC ports, a Layer2 MAC, and RDMA Engines. This robust setup supports high-speed data transfer and superior performance.

To top it all, the Intel® Gaudi® 3 AI Accelerator has four dedicated Aggregation Engines. These engines spring into action on behalf of the Communication Library, performing summing activities. This means faster computations and more efficient data processing.

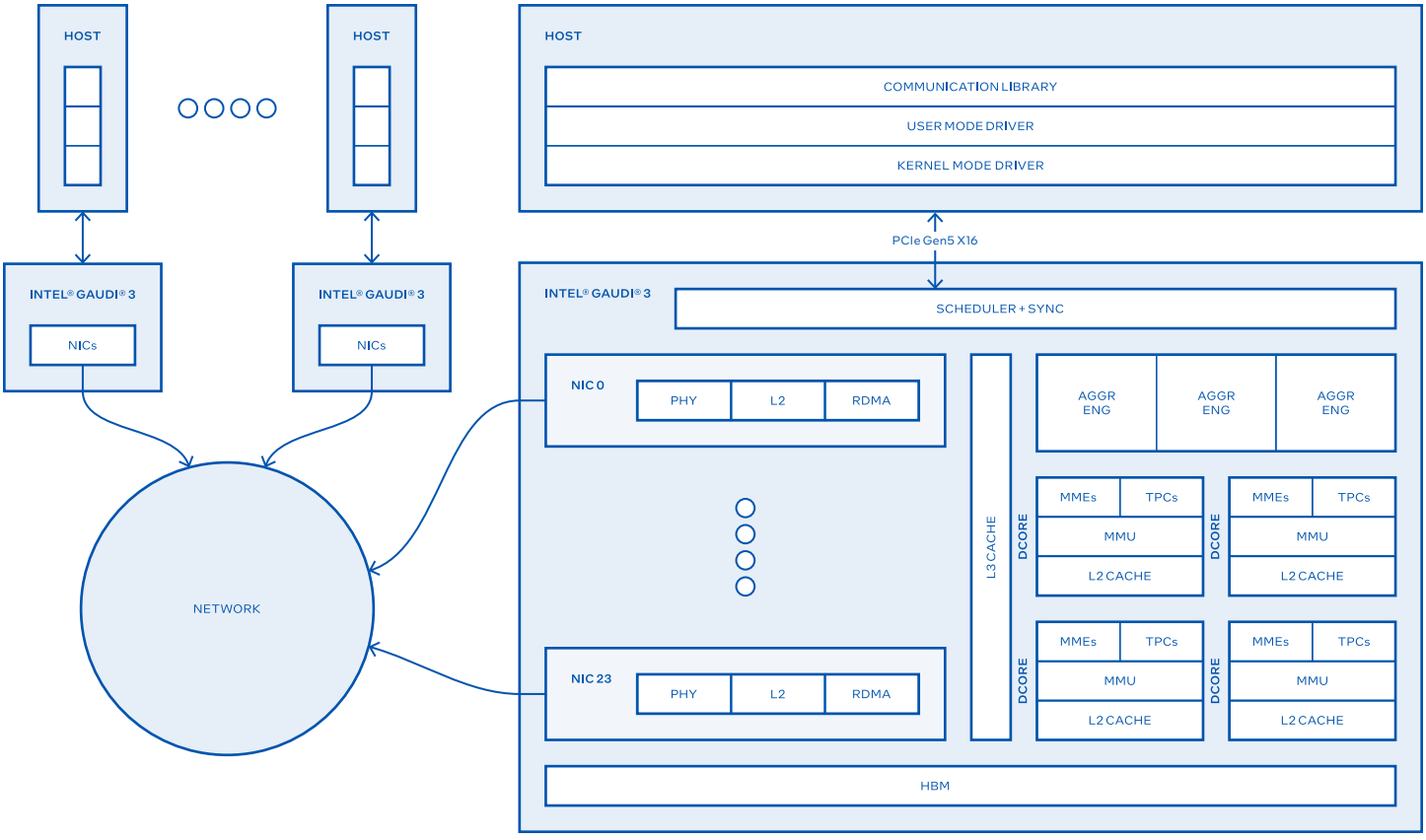
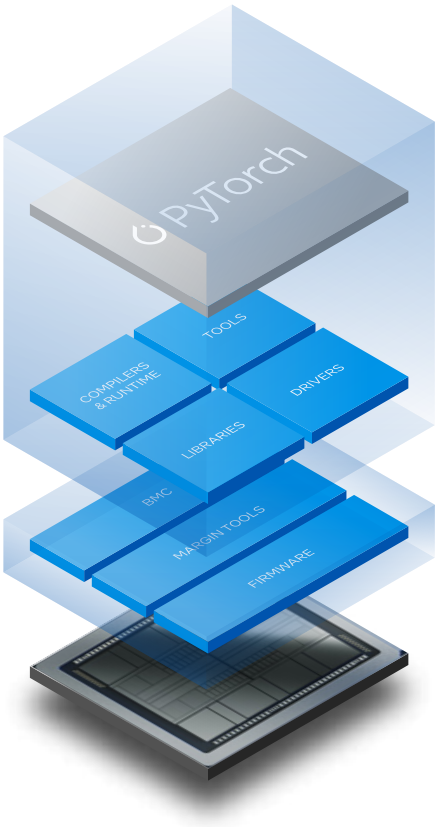


Fig 14. General matrix multiplication and its mapping to the MME engine block diagram.



Gaudi Software Suite

Intel® Gaudi® Software Suite

Designed to facilitate high-performance deep learning (DL) training and inference on Intel® 's AI Accelerators, the Intel® Gaudi® software suite efficiently maps neural network topologies to the Intel® Gaudi® hardware family. The software suite includes low-level components, such as a graph compiler, an automatic kernel fuser and a library of precompiled kernels, as well as integration to the AI ecosystem: PyTorch, DeepSpeed, Hugging Face, vLLM, Ray and more. The Intel® Gaudi software also includes custom implementations of popular algorithms such as Paged Attention, Flash Attention and more.

Graph Compiler and Runtime

The Intel® Gaudi Graph Compiler generates optimized binary code that implements the given model topology on Intel® Gaudi®AI Accelerators. It performs operator fusion, data layout management, parallelization, pipelining and memory management, and graph-level optimizations. The Graph Compiler uses the rich TPC kernel library, which contains a wide variety of performance-optimized operations (for example, elementwise, non-linear, non-GEMM operators). Given the heterogenous nature of Intel® Gaudi® 3 AI Accelerator hardware (MME, TPC and DMA), the Intel® Gaudi Graph Compiler enables effective utilization through parallel and pipelined execution of framework graphs. The Intel® Gaudi software uses stream architecture to manage concurrent execution of asynchronous tasks, supporting Intel® Gaudi's 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 minimal latency with no host involvement.

TPC Programming

The Intel® Gaudi software TPC SDK includes an LLVM-based TPC-C compiler, a simulator and debugger. These tools facilitate the development of custom TPC kernels. The SDK is used to build the high-performance kernels. Users can thereby develop customized deep learning models and algorithms on Intel® Gaudi® AI Accelerators to innovate and optimize to their unique requirements. The TPC programming language, TPC-C, is a derivative of C99 with added language data types to enable easy utilization of processor-unique SIMD capabilities. It natively supports wide vector data types to assist with programming of the SIMD engine (for example, float64, uchar256 and so on). It has many built-in instructions for deep learning, including tensor-based memory accesses, acceleration for special functions, random number generation and multiple data types.

Ecosystem Integration

The Intel® Gaudi software is natively integrated into PyTorch, both 1.x and 2.x. It's also integrated to many popular software packages: DeepSpeed for distributed training and inference, Hugging Face for using Transformers and Diffusers models, vLLM for cutting-edge LLM serving throughput, and more. The Intel® Gaudi software PyTorch Python packages expose several Gaudi optimized operations, such as Flash Attention, to leverage the existing ecosystem innovation around LLM training and inference.

Quantization

Intel® Gaudi® 3 AI Accelerator has even more support for the FP8 datatype than its predecessor. The Intel® Gaudi software exposes this to the user in the form of an automated quantization tool for converting existing models with high accuracy and improved throughput, as well as supporting a Transformer Engine-like API for compatibility with existing models. The Intel® Gaudi software also supports int4 weight-only quantization schemes, such as AWQ and GPTQ, and allows user innovation in those areas by open-sourcing its quantization tool under the umbrella of Intel® Neural Compressor.

Automatic Kernel Fusion

Kernel fusion has multiple benefits for training and inference, improving memory bandwidth, amortizing overheads and for inference also reducing the overall memory capacity and allowing an increase in the batch size for higher efficiency. The Intel® Gaudi software includes a cutting-edge, MLIR-based kernel fuser, capable of automatically generating fused kernels from sequences of primitive kernels in the user graph, without the need for user intervention. These kernels are then interfaced to the graph compiler to utilize Intel® Gaudi® Accelerator's heterogeneous architecture.

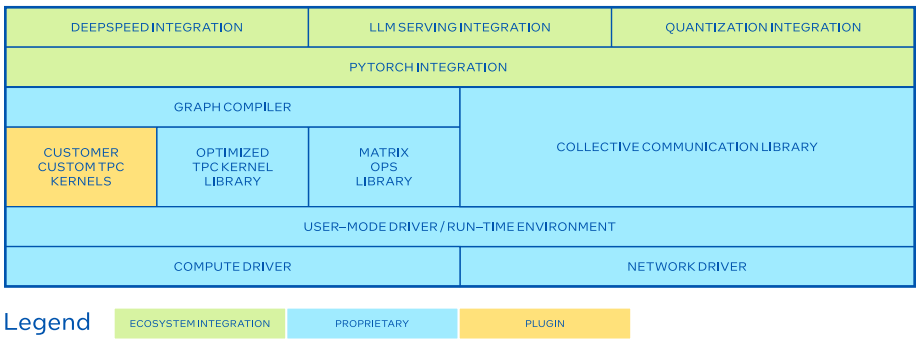


Fig 15. Intel® Gaudi® software stack.

Networking

As deep learning training is usually performed on multiple devices, the Intel® Gaudi® 3 AI Accelerator Network Interface Controllers (NICs) are an essential component in the overall Intel® Gaudi® third-generation training solution. Intel® Gaudi® 3 AI Accelerator NICs are customized to fit a distribution of a DNN graph between the devices in the network (scale-out). The NIC provides the compute engine with remote direct memory access (RDMA) featuring high bandwidth and low latency over reliable connection without any software intervention. To fit common cloud infrastructure, NIC ports use Ethernet (Eth) connectivity with an aggregated bandwidth of 4.8 Tb/s in each direction, supporting multiple port configurations. The NIC implements RoCE v2 specification, benefiting from the commonly used Ethernet infrastructure and the reliable and low latency RDMA of the InfiniBand (IB) protocol.

Intel® Gaudi® Accelerator implementation extends RoCE v2 specification to better fit it to DNN applications and large-scale deployments enabling linear scalability over thousands of Intel® Gaudi® Accelerators.

The upcoming sections highlight the main RoCE extension that Intel® Gaudi® 3 AI Accelerator supports.

Mapping MPI Collective Operations to RDMA

RDMA protocol supports remote memory access using natural read and write operations. RDMA read and write operations assume that the initiator has the pointers for both the local and remote memory. However, DNN applications commonly use MPI style collective operations that are based on a send-receive approach.

This approach defines three main elements: first, the sender side which has the pointer to the send buffer; second, the receiver side which has the pointer to the receive buffer; and third, a rendezvous flow to move data between the two sides. Therefore, MPI collective operations do not map naturally to RDMA read and write operations.

There are many ways to perform this mapping, each one with its own pros and cons. Mapping MPI operations to RDMA send-receive operations is one option. This option does not solve the rendezvous flow. When the sender sends data to the receiver before the receiver has posted the receive operation, a RNR NACK will be sent to initiate a retransmission, causing a significant performance drop.

Another option is to implement the rendezvous on the receiver side using a temporary buffer and later initiate a mem-copy once the receiver buffer is available. This option has many drawbacks such as high latency and high memory capacity.

Intel® Gaudi® Accelerator implements a hardware-based implementation that solves the rendezvous flow on the sender side and thus ensures the data is sent to the receiver only once and without the need for a mem-copy. This approach allows us to expose a simple collective API to the user, offloading the complexity from the CPU to NIC's hardware ensuring minimal latency and high message rate.

Offloading Collective Kernels to HW

In practice, execution of collective operations is done by splitting it to multiple send-receive operations between the ranks. The HLS-3 reference server includes eight different Intel® Gaudi® Accelerators. Each device is assigned with a unique rank ID. The connectivity between devices is done using multiple ports. Therefore, a collective operation needs to be split between the ranks and between the multiple ports connecting each rank. The splitting process consumes CPU resources, potentially lowering port utilization and its transmission bandwidth. Intel® Gaudi 3 AI Accelerator NICs offload the collective operation to the hardware, allowing the hardware to achieve full bandwidth with a buffer size as small as 300KBs.

Congestion Control - Timely Based

As DNN clusters become ever larger, congestion over the network becomes a more predominant problem. Congestion on lossy networks may cause significant performance degradation due to packet drop. Enabling Priority-based Flow Control (PFC) to achieve lossless networks, prevents packet drops but congestion may spread between the switching layers. RoCE v2 implements RoCE Congestion Management (RCM) based on Explicit Congestion Notification (ECN). However, RCM is a crude method, resulting in large throughput variability.

In the Intel® Gaudi® 3 AI Accelerator, congestion control was expanded to not only support ECNs but also support timely based congestion schemes such as SWIFT®. These algorithms use delay (RTT calculation) as their congestion indication signal and as such have a much more fine-grained control over ECN.

Multi-Path Load Balancing (Packet Spraying)

Connecting large clusters of nodes requires multi-layer switching topologies. In such cases, the network connectivity between nodes may include multiple paths. To fully utilize the network's bandwidth between two nodes and reduce congestion, traffic should be balanced between all possible equal cost paths. Deploying equal-cost multipath (ECMP) can provide a solution. However, as discussed above, large clusters may also suffer from congestion which impacts the different paths, reducing throughput and increasing flow's completion time.

To mitigate congestion buildups, we introduced a load balancing system. The system considers the path's load and adapts the cost function to keep the bandwidth utilization high and latency low. The load balancing system provides a method to re-order the packets traversed on different paths.

RDMA Reliable Connection (RC) Memory Footprint

Deploying a large cluster of all-to-all connectivity using RDMA reliable connection can suffer from unscalable memory footprint. Consider a cluster of N nodes, each with P processes. If all P processes wish to communicate with all processes on all the nodes, RDMA Reliable Connection service requires P^2×(N-1) QPs on each node. Each QP includes a context with size of O(100Bytes) and a work queue with size of O(10KBytes). In our implementation, the QPs are handled by the Intel® Gaudi® Accelerator Collective Communication Library (CCL). CCL most-commonly opens four QPs for each peer node in the cluster, so the total number of QPs on each node is 4×(N-1). Therefore, the memory footprint becomes scalable with the number of nodes.

In-network Reduction

To reduce compute requirements for reduction operations and to provide excellent overlap with the communication phase, the Intel® Gaudi® 3 AI Accelerator supports the capability of performing reduction operations on the network path. The supported operations are sum, min and max. In addition, the reductions support different data types including FP32, FP16, BF16 and FP8. Further, the BF16 and FP16 reductions can be performed with FP32 accumulation for better accuracy.

Network and Compute Synchronization

Some DNN accelerator systems use discrete NICs to communicate with other nodes in the cluster. In those systems, the synchronization between the networking that transmits data and compute engine that consume data suffer from high latency due to high host CPU utilization. The Intel® Gaudi® 3 AI Accelerator integrates both the NIC and compute engines and the synchronization between them is done within the chip with minimal latency and without host intervention.

Tensor Semantics

Standard RDMA operations are designed to work with contiguous buffer, but DNN applications are designed to work in tensor and sub-tensor semantics. Mapping sub-tensors to a contiguous buffer to work with RDMA operations can be very complex or even not scalable. Therefore, the Intel® Gaudi® 3 AI Accelerator introduces a tensor engine within the NIC that can access both local and remote memory in tensor semantics, much like all other engines in the chip.

Selective Retransmission and Out of Order Delivery

To provide high throughput and lower latency, the current RoCE implementation depends on networks being lossless. This is attributed to InfiniBand (IB) networks relying on credits while Ethernet based networks assume loss. Recovery implementation for packet loss in IB is go-back-N, i.e., retransmitting back from ONA once a NACK arrives. This greatly affects bandwidth and the flow completion time and tail latency, even in cases of sporadic drops. This is because all packets from ONA to NTS are re-transmitted.

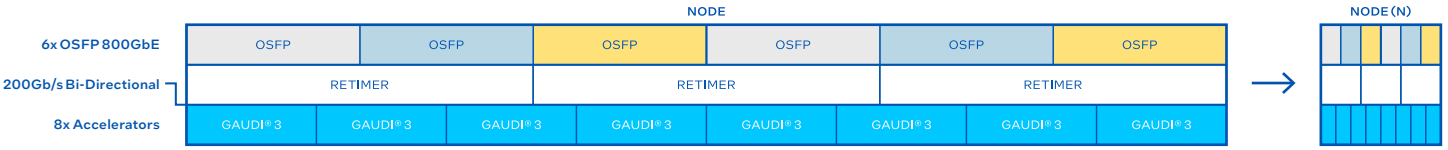
Since data centers are moving to lossy architecture, mainly because Priority-based Flow Control (PFC) has scalability limitations, Intel® Gaudi® 3 AI Accelerator RoCE implementation extends the IB transport layer spec and allows Selective ACKing by the responder and Selective Re-Transmission by the requester. For all other purposes, the IB spec is still valid. This allows Intel® Gaudi®3 AI Accelerator's RoCE to be even more scalable than TCP/IP with selective ACK implementation.

Cluster Architecture

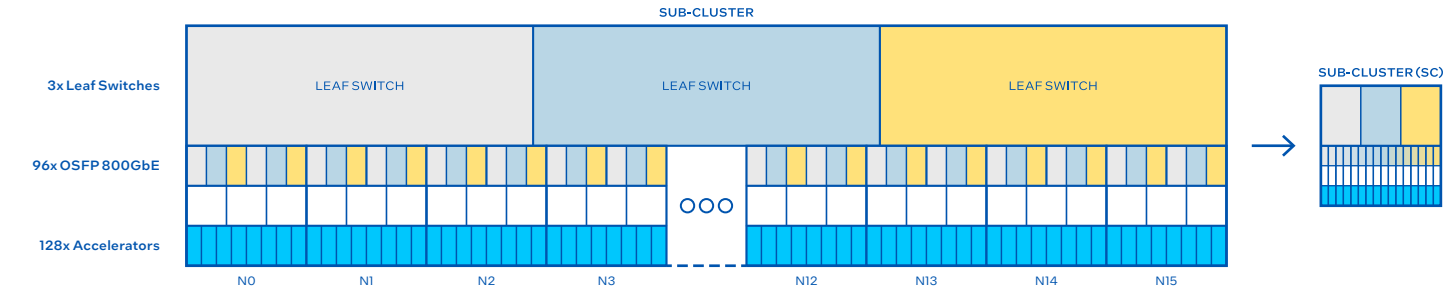
Using standard Ethernet switches, modular and high-performance clusters can be built to the desired scale. The following shows an example of building a 512-node cluster (4096 Intel® Gaudi® 3 AI Accelerators) using 16-node sub-cluster building blocks. In an Intel Gaudi 3 AI Accelerator-based server, each OAM card has a NIC port connected to 3 of the OSFP scale-out ports of the server. Then a sub-cluster is established by connecting 16 servers to 3 64-port 800 Gbps Ethernet leaf switches. In the sub-cluster, any card in a system can communicate with any other card in the other systems through all 3 of the leaf switches. Finally, 32 of the sub-clusters are networked together using 48 64-port 800 Gbps Ethernet spine switches. This topology forms a 3-ply network, where all 64-ports of each leaf and spine switch are utilized.

Figure 16 features representations of GenAI system scale out as a single node, 16-node sub-cluster, and 512-node cluster.

Node Level Architecture



Sub-Cluster Level Architecture (16 Nodes)



Cluster Level Architecture (32 Sub-Clusters, 512 Nodes)

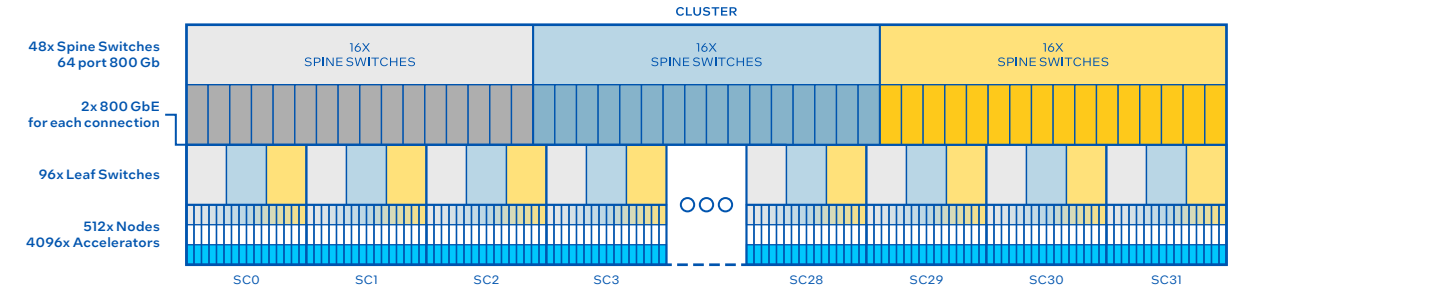


Fig 16. Intel® Gaudi® 3 AI Accelerator – Scale-out Cluster Architectures.

Putting It All Together: Combining Hardware and Software for a Unified AI Acceleration Solution

The Intel® Gaudi Software Suite offers a comprehensive set of capabilities that significantly enhance the Intel® Gaudi® 3 AI Accelerator hardware utilization. In this chapter, we delve into the seamless integration of various components from the Intel® Gaudi Software Suite, illustrating how they collaboratively reduce workload runtimes. By examining a practical example drawn from Large Language Models (LLMs), we highlight the impact of key software layers on hardware efficiency.

Naïve Execution of a Transformer Sub-Sequence

Language Models (LLMs) are composed of a series of repeating Transformer layers. Each Transformer layer involves an intricate sequence of operations, including:

1. General Matrix Multiplication (GEMM): A fundamental operation for linear transformations.
2. Batched-GEMM: An optimized variant of GEMM that efficiently processes multiple inputs.
3. Normalization: Encompasses techniques such as softmax, layer normalization, or RMSNorm.
4. Residual-Add: A crucial component for preserving information flow.
5. Non-Linear Activation Function: Choices include GELU or SwiGLU.
6. Dropout (Training Only): A regularization technique to prevent overfitting.

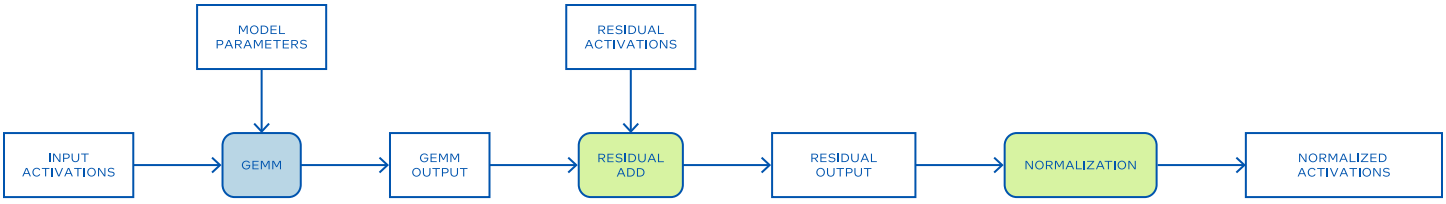


Fig 17. Illustration of a Transformer layer sub-graph from large-language model.

In Figure 17 we visualize a sub-sequence of operations that repeat twice within a transformer layer.

Blue colored graph nodes (rounded-corner rectangles) represent MME operations, while green colored graph nodes represent TPC operations. The sub-sequence of operations comprises the following steps:

1. GEMM: Executed by the MME
2. Residual-Add: Executed by the TPC
3. Normalization: Executed by the TPC

Executing the sequence of operations without any optimization results in the runtime illustrated in Figure 18. All eight MME units logically work as a single unit and execute the GEMM to completion. Each TPC kernel was written using the Intel® Gaudi® Accelerator TPC SDK, which uses TPC's ISA and microarchitecture efficiently.

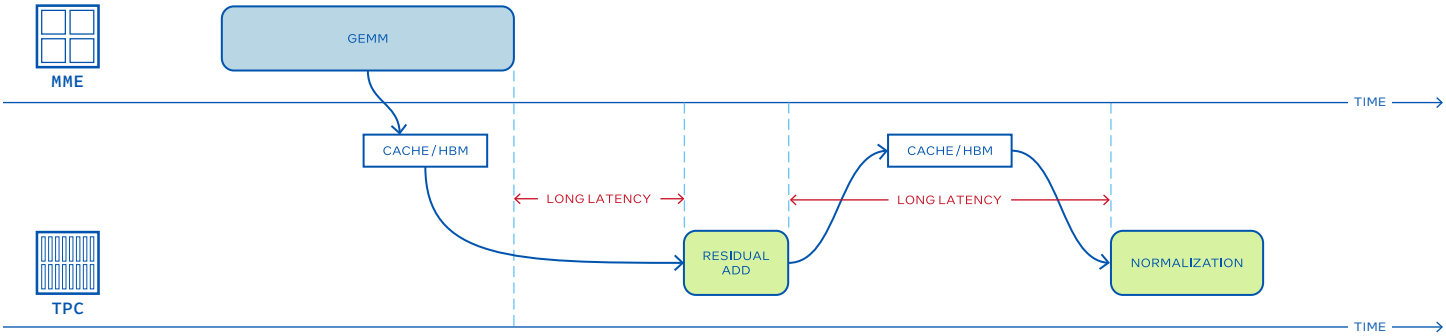


Fig 18. Illustration of timeline execution for naïve software implementation of the sub-graph in Figure 17.

In the execution shown in Figure 18, entire GEMM output is written to cache. However, since the full GEMM output can exceed the cache capacity, as is usually the case in large workloads, all GEMM outputs that do not fit in cache will be written to HBM. The TPC starts executing its first kernel after all GEMM result writes are completed. When TPC starts its execution, due to the large input size, some of the inputs will be read from the HBM, resulting in relatively long latency (1–2usec), which is determined by the HBM. The second TPC kernel experiences the same long latency as the first kernel.

Automatic Kernel Fusion

One immediate improvement that is delivered by the Intel® Gaudi® software suite is automatic kernel fusion. The two TPC kernels are automatically fused to generate a new kernel that contains the union of operations within the separate kernels. Fused kernel's inputs and outputs are the external inputs and outputs of the fused kernels. Fusing the kernel saves the I/O of reading or writing intermediate results between the original kernels.

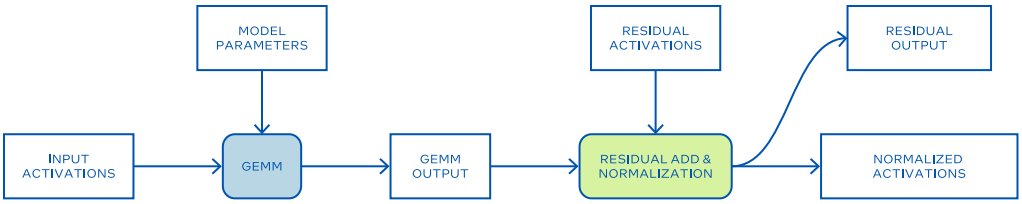


Fig 19. Illustration of the sub-graph from Figure 17 after the fuser has fused the two consecutive TPC kernels to one.

Fusing the sub-graph in Figure 17 results in the sub-graph illustrated in Figure 19.

1. Saving I/O time for writing a result from one kernel, then reading the same result in the following kernel.
2. Inter-kernel latency saving.

Executing the sub-graph of Figure 19 results in the execution illustrated in Figure 20. Runtime gain manifests in the two ways explained above. First, there is only one latency window between executing engines. Second, entire TPC runtime has decreased since the normalization part of the fused kernel reads its input internally from the TPC and not from an external I/O, thereby saving I/O time.

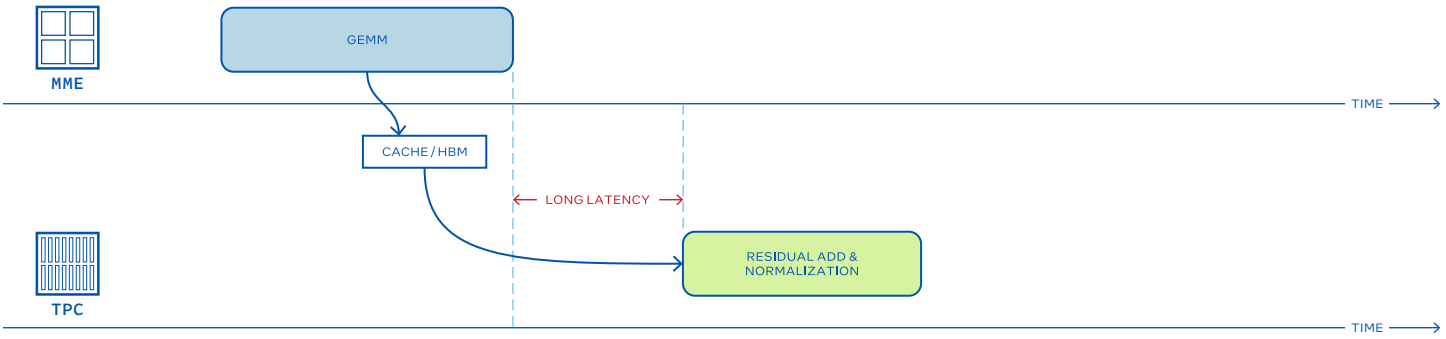


Fig 20. Illustration of device execution of the sub-graph from Figure 19.

Producer-Consumer Pipelining by the Graph Compiler

The Graph Compiler described in previous sections is specifically designed to optimize the execution of workloads on the Intel® Gaudi® 3 AI Accelerator's heterogeneous architecture. In cases where it is logically feasible, dependent engines are scheduled to operate in a pipelined manner, establishing a producer-consumer execution dependency. The producer writes its output to the L2 cache, the highest cache hierarchy shared by the MME and TPC. Once the output is fully written, the consumer reads this output from the cache as input. Intel® Gaudi® software suite ensures that the produced data fits within the cache, and the granularity of work allows for efficient device utilization.

By leveraging cache-based pipelining, we achieve minimal latency between the producer and the consumer, resulting in optimal utilization of all device engines. Figure 21 illustrates this producer-consumer relationship: a GEMM operation executed by the MME and a fused residual add & normalization operation executed by the TPC. The input and output of the GEMM operation are split into four slices, with each output slice fully produced by the MME and subsequently read as input by the TPC.

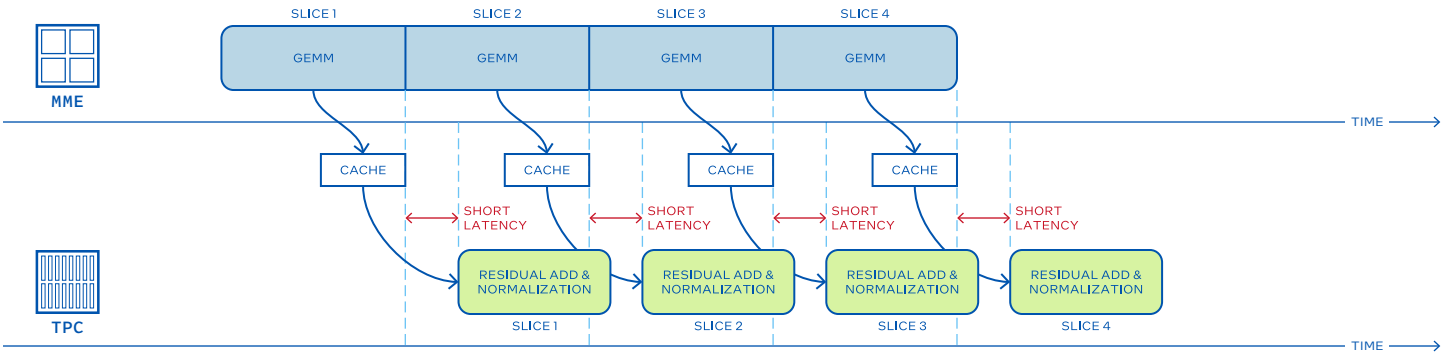


Fig 21. Illustration of pipelining between MME and TPC through the cache, when MME is the producer and TPC is the consumer.

Adding Network to the Mix

As stated above, Intel® Gaudi® 3 AI Accelerator architecture inherently supports running all engines in parallel, including the NIC. Expanding upon the example from Figure 19, we add all-reduce collective communication that follows the GEMM operation and before residual add. Figure 22 illustrates the sub-graph that is formed when adding the all-reduce collective operation. The all-reduce is required in cases of Tensor-Parallelism split of an LLM between multiple devices, for training and inference use cases.

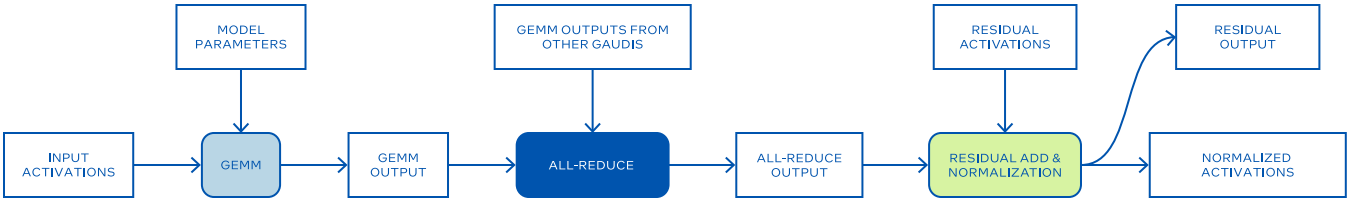


Fig 22. Illustration of the sub-graph from Figure 19 with an all-reduce collective communication operation between the GEMM and fused TPC kernel.

In a naïve execution, the all-reduce will interfere with the pipelining. The result is sequential execution of the GEMM, allreduce and fused TPC kernel, as illustrated in Figure 23. We see no parallelism between the engines and long latencies between the engine activations.

By structuring the LLM code in a manner that enables parallel execution of all-reduce operations, the Graph Compiler and HCL can efficiently distribute workloads across multiple engines. This approach maximizes device utilization. In Figure 24, we visualize the effective execution on Intel® Gaudi® 3 AI Accelerator of the sub-graph depicted in Figure 22. Specifically, the diagram illustrates the data flow: MME as the producer to the NIC, the NIC being a consumer of MME's output and producer to the TPC, and TPC being a consumer of NIC output.

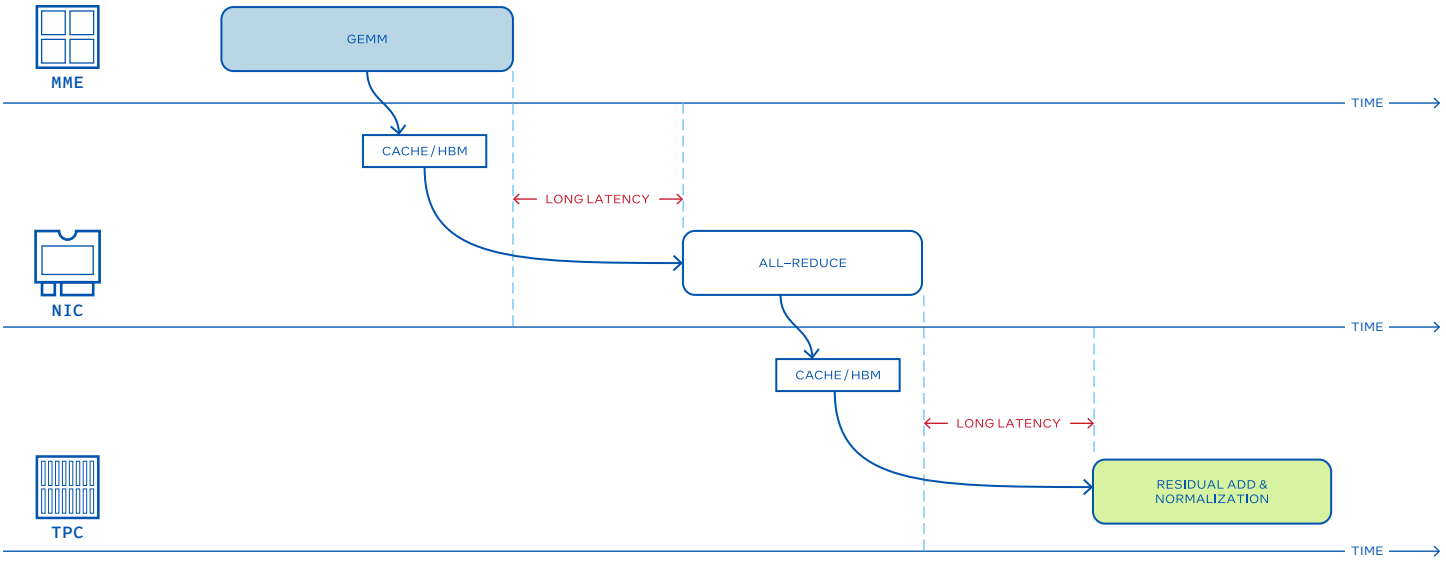


Fig 23. A naïve execution of the operations in the subgraph of Figure 22.

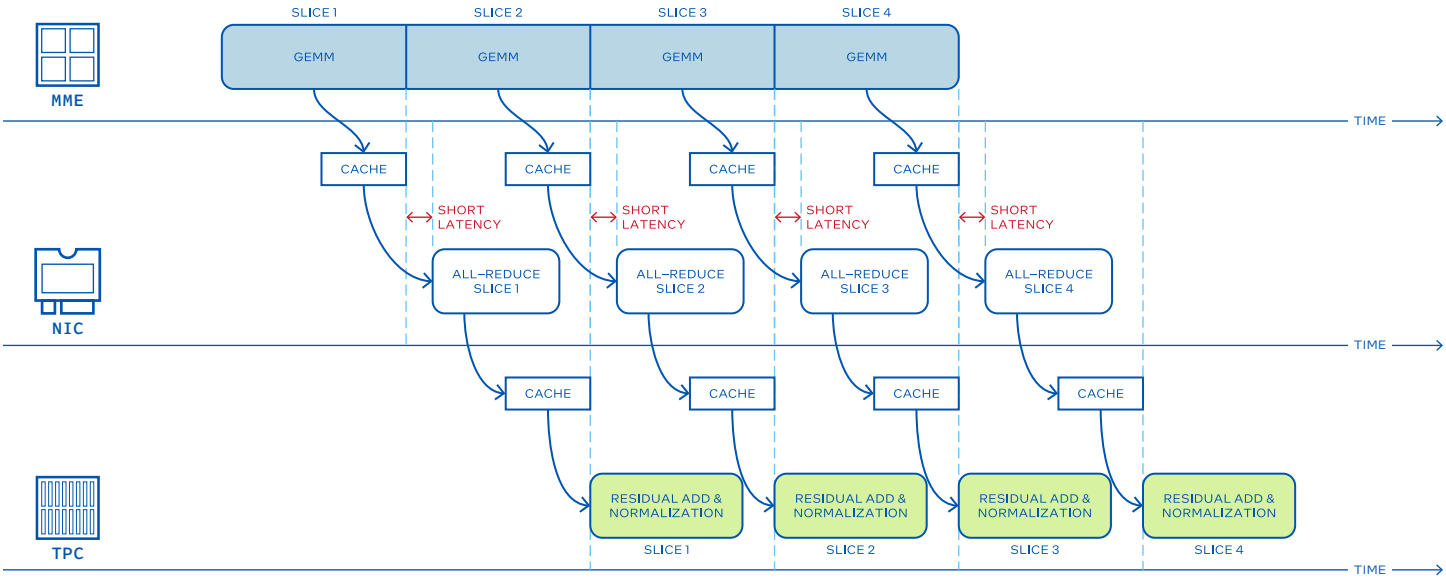


Fig 24. Optimized scheduling of the MME, NIC and TPC, with a producer-consumer relation between all three engines.

Fig 25. Intel® Gaudi® AI Accelerator Product Line.

Intel Gaudi® 3 Performance Improvements

Intel® Gaudi® 3 AI Accelerator is the third generation of the Intel Gaudi AI Accelerator family. The large HBM capacity and bandwidth allow the Intel® Gaudi® 3 AI Accelerator to achieve state-of-the-art GenAI training and inference performance.

In training scenarios, virtually all of the advanced capabilities of Intel® Gaudi® 3AI Accelerator over the previous generation come into play. Since training scenarios are compute-intensive, the increased compute ratio provides immediate gain. The increased HBM bandwidth allows larger compute to manifest the increased compute power. In addition, the larger HBM capacity also contributes to improved performance. Larger HBM capacity allows increased batch size, enabling higher compute utilization and allows avoiding re-computation of certain parts of the workload or avoiding model-parallel splits, which add networking operations during runtime.

In general, LLM inference throughput is determined by the available HBM bandwidth, which is used for reading the model parameters and context window. When comparing Intel® Gaudi® 3 AI Accelerator to Intel® Gaudi® 2 AI Accelerator, we expect that for small LLMs (13B-sized model or smaller), speedup is similar to the ratio of HBM bandwidths between the two generations of accelerators, roughly 1.5x. However, when comparing larger LLM models, like LLama-70B, improvements are expected to be greater than the HBM bandwidth ratios and surpass a 2x ratio. The larger improvement is due to the larger memory capacity that is available for Intel® Gaudi® 3 AI Accelerator This larger capacity allows use of increased batch size and therefore more samples processed per given amount of time.

Measured performance of Intel® Gaudi® 3 AI Accelerators will be updated and published at Model Performance for Intel® Gaudi® 3 AI Accelerators coinciding with the Intel Gaudi software releases.