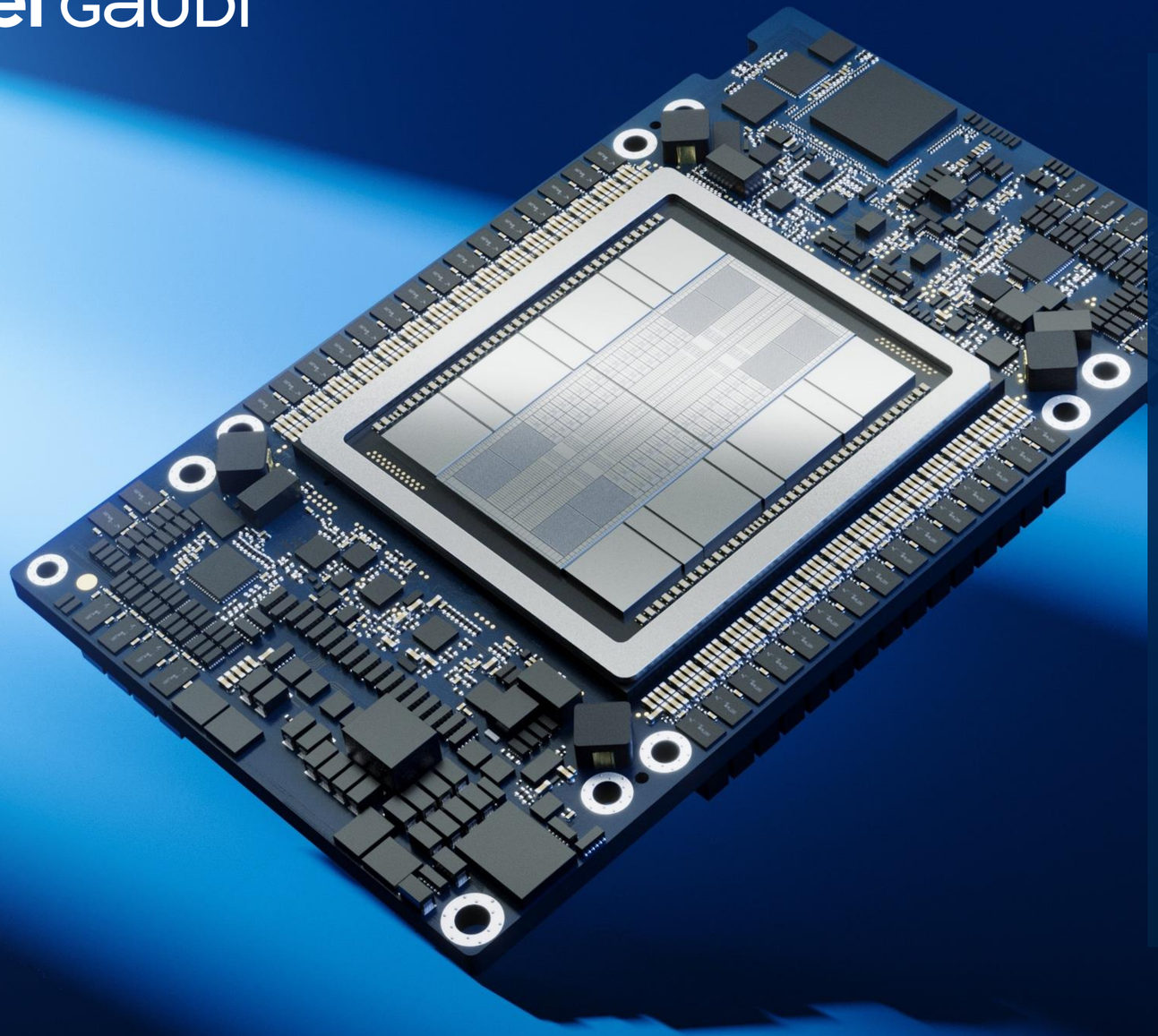
intel GAUDI

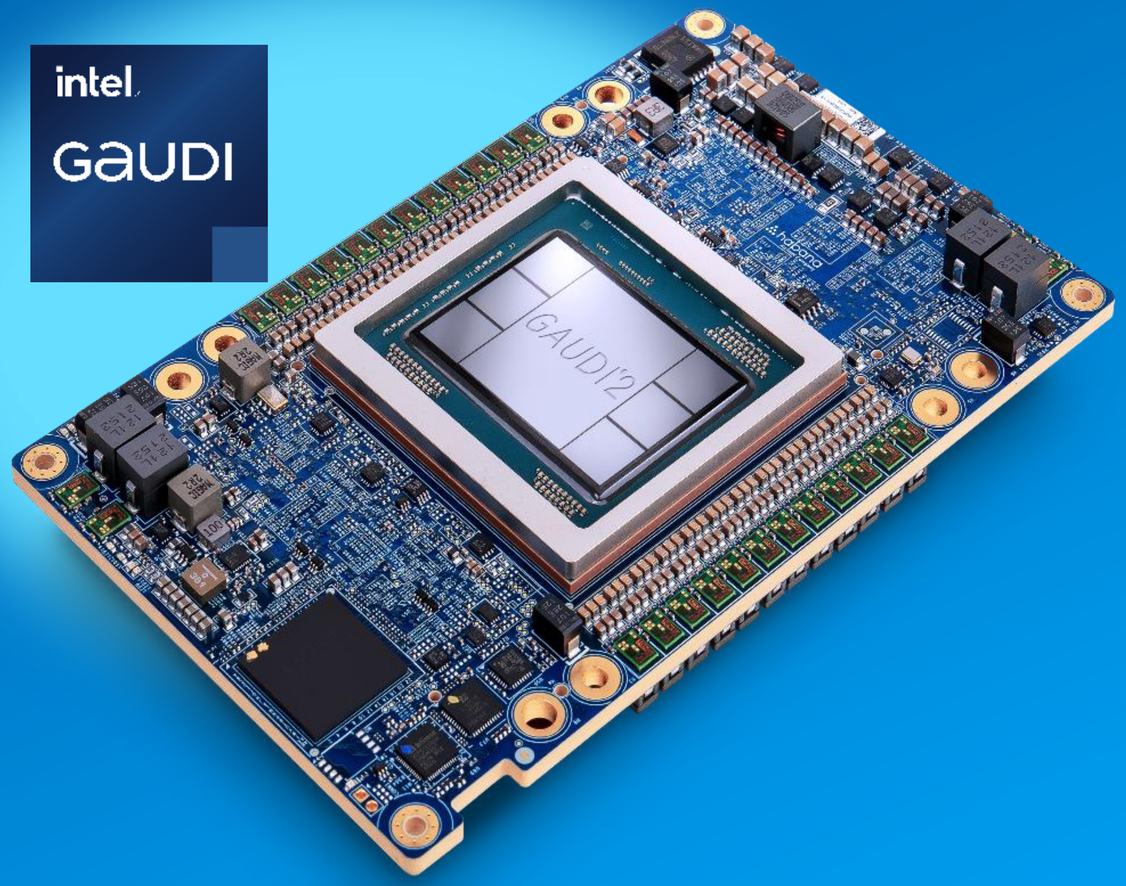


Programming Novel AI Accelerators for Scientific Computing Intel®

Intel® Gaudi® 2 AI accelerator

Buke Ao(buke.ao@intel.com)

Architected for Deep Learning Performance and Efficiency



Matrix Multiplication Engine &

24

Tensor Processor Cores

96 GB

On-Board HBM2e

7nm

Process Technology

24

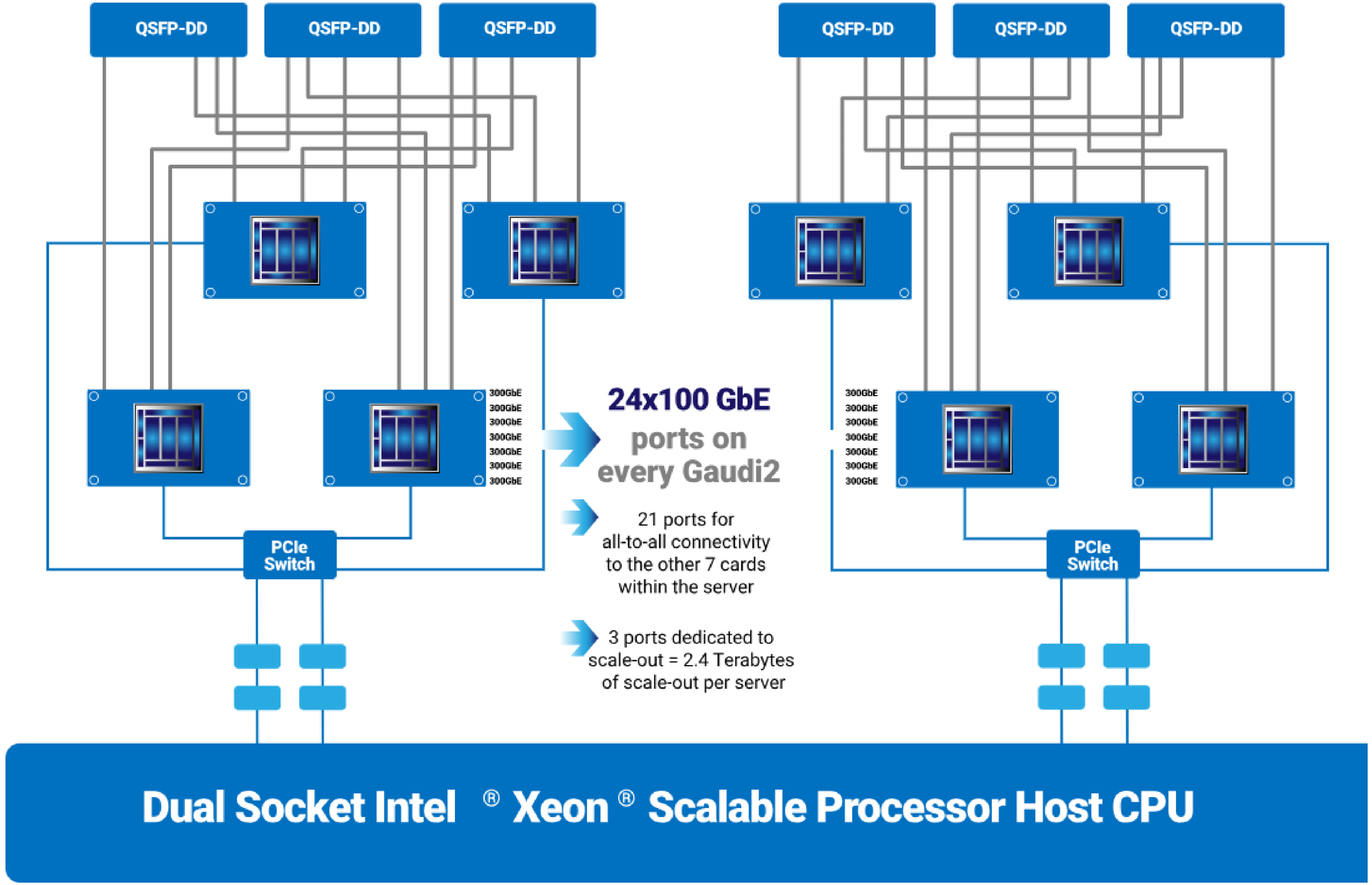
Integrated 100G Ethernet ports

48 MB

SRAM

Purpose Built for Deep Learning Acceleration at Scale

Intel® HLS-Gaudi®2 Server Architecture



Gaudi HL-225H

1x100GbE

PCIe

All-to-All Connectivity
21x100G Eth

Intel Gaudi Software Suite

Integrates the main Gen AI frameworks used today

Supports FP16/BF16 → FP8 quantization

Main proprietary SW layers

Graph Compiler: Handles all engine dependency and scheduling logic

Matrix operations: Configuring the MME

TPC kernels: All non-Matrix operations

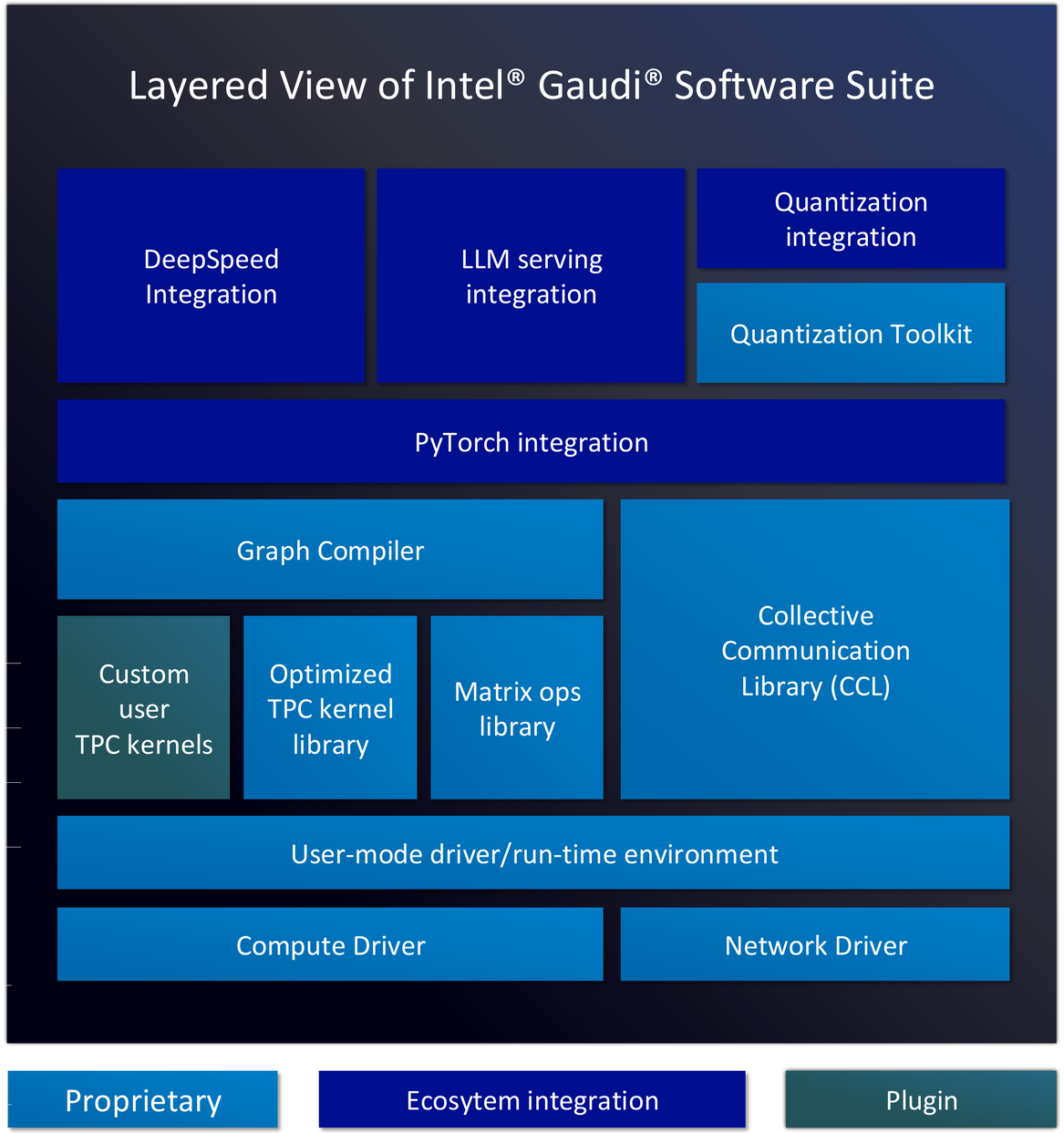
Collective Communication Library (CCL)

Several sources for TPC Kernels

Gaudi optimized TPC kernel library

Custom user kernels

MLIR-based fused kernels: generated during graph compilation



Model Migration Steps - Paths

Using Model References GitHub

Fully vetted starting point to validate existing performance or use examples to innovate.

Using Hugging Face

Start with existing examples or use Optimum Habana library with any transformer model

Public Models

Migrate models built for CPUs or GPUs.

PyTorch Manual Migration

1 Pre Work: Removing CUDA calls

```
import torch

# Import Habana Torch Library
import habana_frameworks.torch.core as htcore

# neural network model
class SimpleModel(nn.Module):
...
```

```
# training loop
def train(net,criterion,optimizer,trainloader,device):
...
        loss.backward()
        htcore.mark_step()

        optimizer.step()
        htcore.mark_step()

def main():
...
    # Target the Gaudi HPU device
    device = torch.device("hpu")
```

Autocast For Mixed Precision

```
with torch.autocast(device_type="hpu", dtype=torch.bfloat16):    output = model(input)
        loss = loss_fn(output, target)
loss.backward()
```

Distributed Training Setup Example

```
import habana_frameworks.torch.distributed.hccl
torch.distributed.init_process_group(backend='hccl')

...
# Use with PyTorch DDP Hook
ddp_model = DDP(model)

(model) loss_fn = nn.MSELoss()
optimizer = optim.SGD(ddp_model.parameters(), lr=0.001)

optimizer.zero_grad()
outputs = ddp_model(torch.randn(20, 10).to(device))
```

Migration from GPU

Intel Gaudi software maps specific API calls from Python libraries and modules like:

torch.cuda

Torch API w/ GPU related parameters like: torch.randn(device="cuda")

Apex. (check Limitations)

pynvml

Intel Gaudi software is preinstalled.

GPU Migration Logging allows investigation on what was changed

Logging feature can be enabled by setting the GPU_MIGRATION_LOG_LEVEL environment variable

PyTorch Migration from GPU

1 Pre Work: Removing CUDA calls

```
import torch
# Import Habana Torch Library
import habana_frameworks.torch.gpu_migration
import habana_frameworks.torch.core as htcore

# neural network model
class SimpleModel(nn.Module):
...
```

2

```
    # training loop
    def train(net,criterion,optimizer,trainloader,device):
    ...
        loss.backward()
        htcore.mark_step()

        optimizer.step()
        htcore.mark_step()

    def main():
    ...
        # Target the Gaudi HPU device
        device = torch.device("hpu")
```

3

Autocast For Mixed Precision

4

```
with torch.autocast(device_type="hpu", dtype=torch.bfloat16):    output = model(input)
        loss = loss_fn(output, target)
loss.backward()
```

Just 2 steps and go!

Distributed Training Setup Example

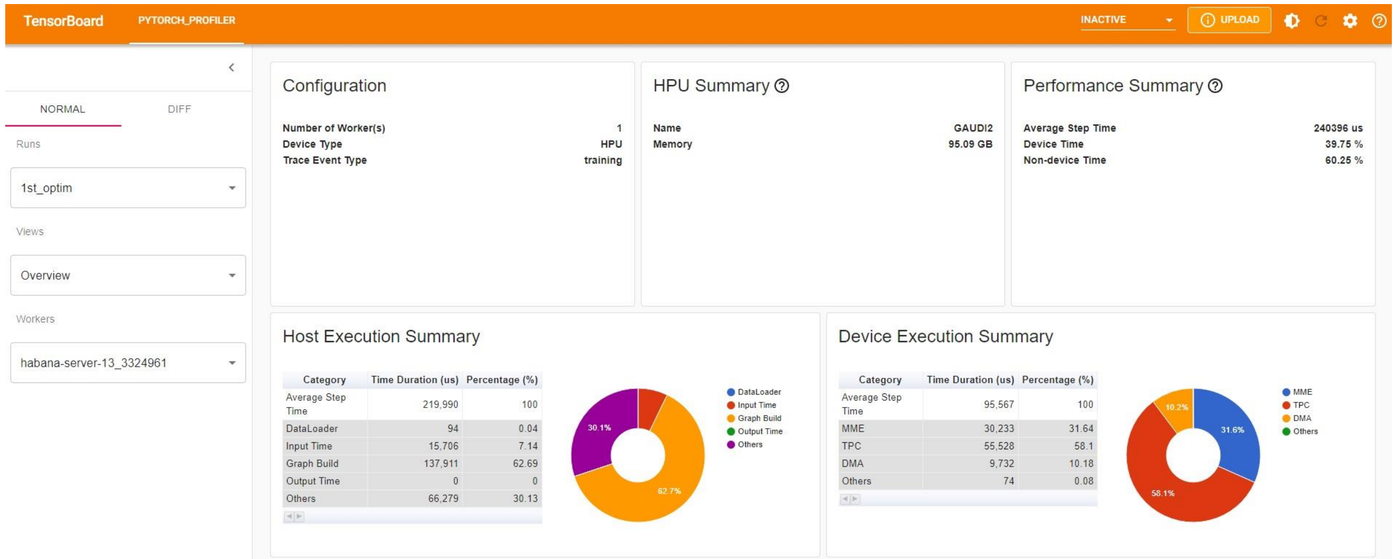
5

```
import habana_frameworks.torch.distributed.hccl
torch.distributed.init_process_group(backend='hccl')

...
# Use with PyTorch DDP Hook
ddp_model = DDP(model)

(model) loss_fn = nn.MSELoss()
optimizer = optim.SGD(ddp_model.parameters(), lr=0.001)

optimizer.zero_grad()
outputs = ddp_model(torch.randn(20, 10).to(device))
```

Profiling

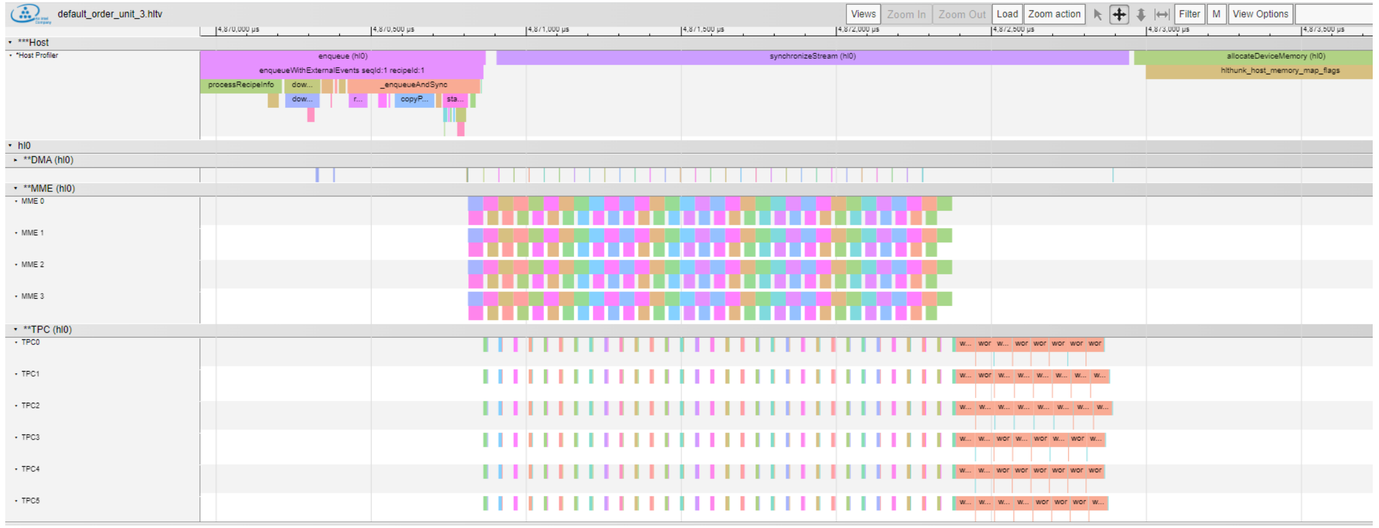
PT Tensorboard

- Gaudi Overview
- Gaudi Kernel View
- Gaudi Memory View
- TraceViewer
- Recommendations for HPU Optimization

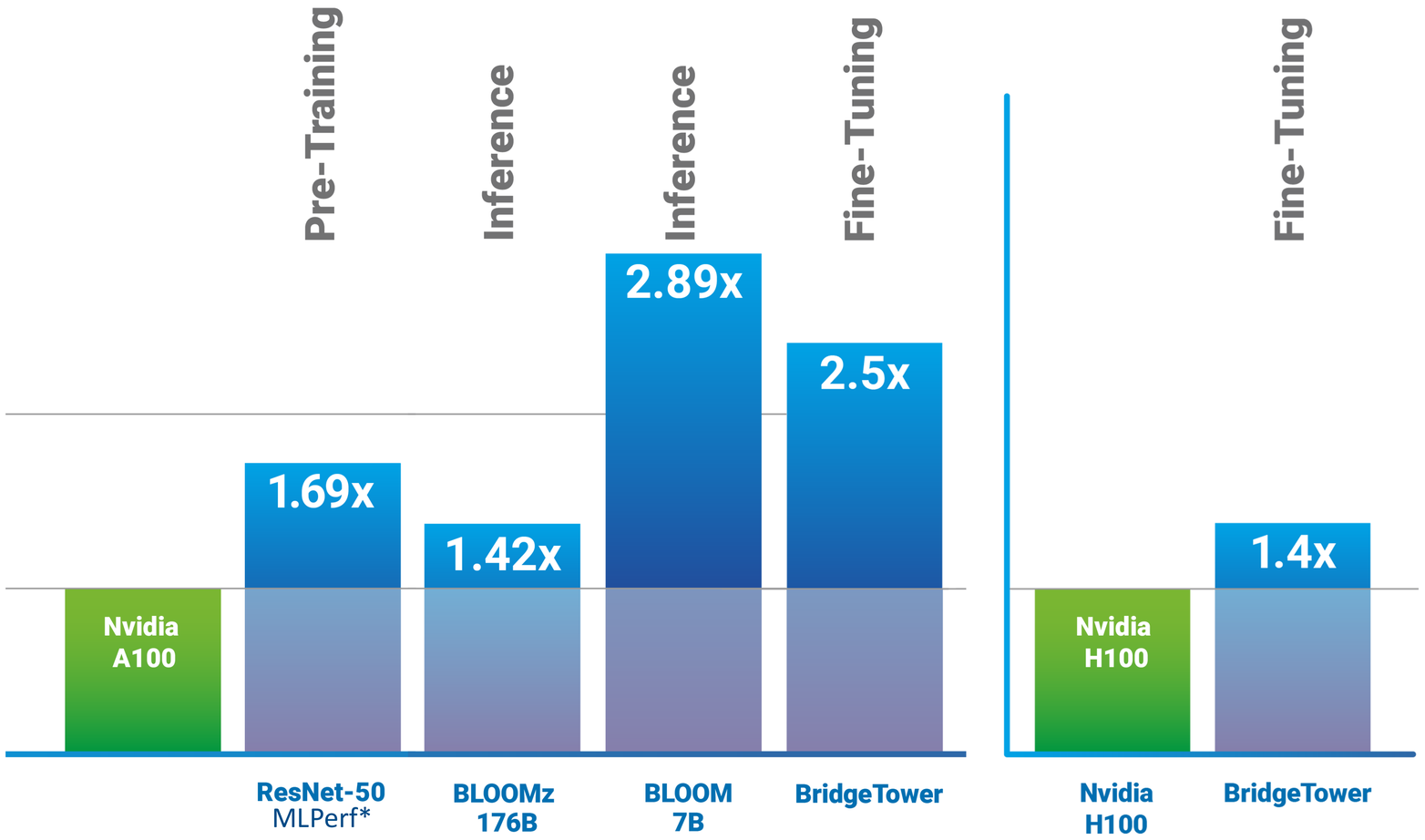


Intel Gaudi SW Profiling

- Advanced chip-level debugging
- Upload .hltv files to https://hltv.habana.ai/ or Perfetto UI
- Host and Gaudi analysis



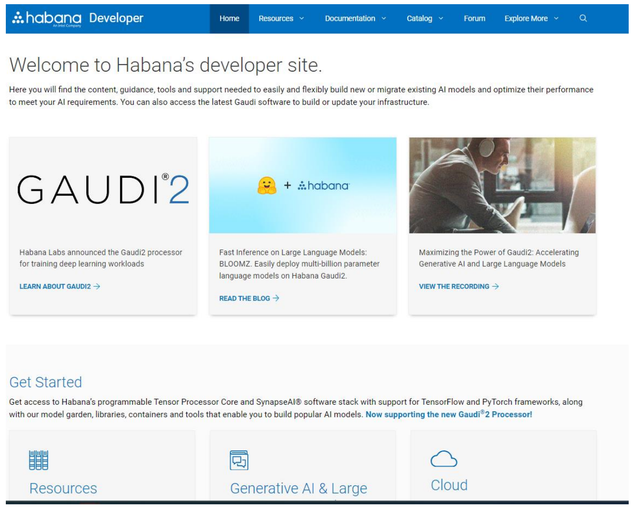
Hugging Face Evaluations of Intel® Gaudi®2 Performance vs. A100 and H100



Intel® Gaudi® Developer Platform

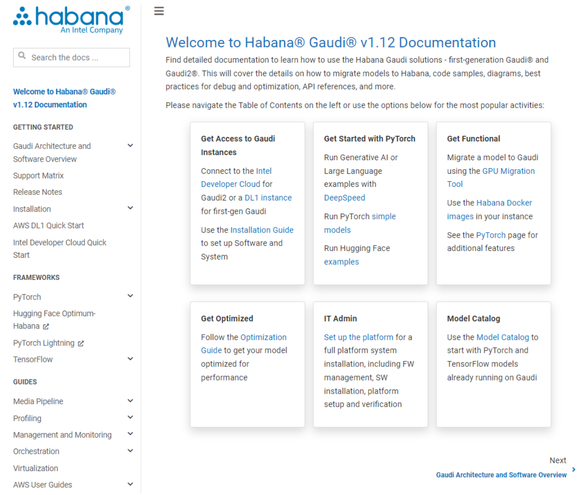
Habana Developer Portal

- Performance
- Catalog
- Tutorial



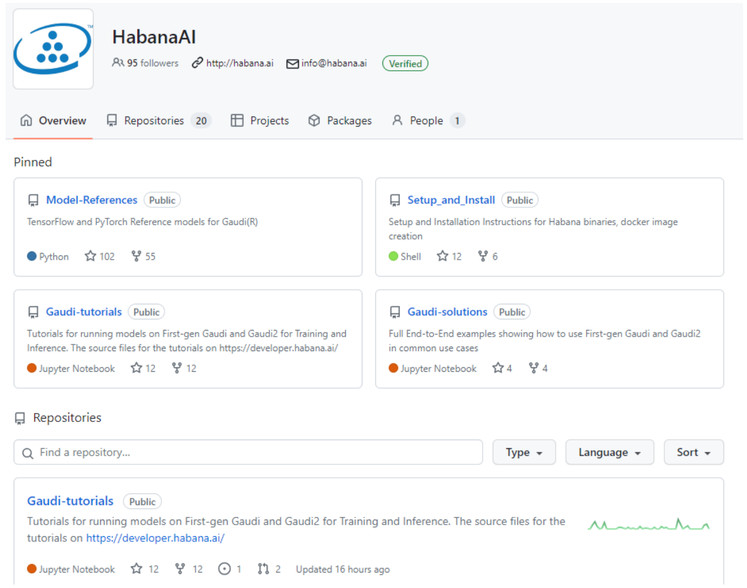
Habana Documentation

- Setup & Install
- User Guides
- Migration



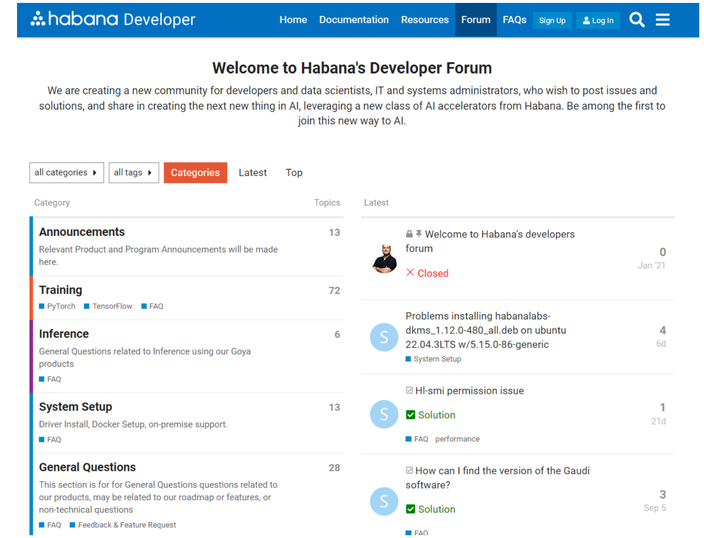
GitHub

- Model References
- Optimum Habana
- Hugging Face Community



Service Desk & Support Forum

- Announcements
- Community Support



Intel Gaudi AI Accelerator Roadmap

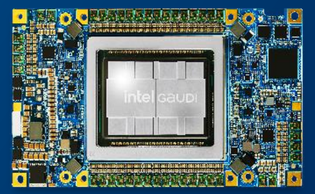
Intel Gaudi Accelerator

16nm

Intel Gaudi 2 Accelerator

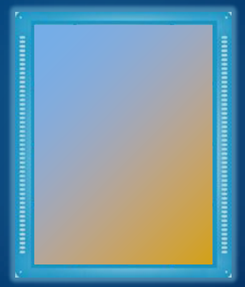
7nm

Intel Gaudi 3 Accelerator

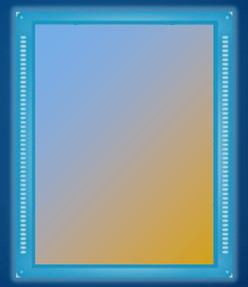


5nm

Next gen GPU Accelerator



Jaguar Shores



Thank You

www.habana.ai

habana
An Intel Company

Q&A

www.habana.ai

habana
An Intel Company