MN-Core 2: Second-generation processor of MN-Core architecture for AI and general-purpose HPC application

Jun Makino

Preferred Networks, Inc./Kobe University

Summary

- MN-Core is a series of AI-oriented accelerator developed by PFN
- The first-generation MN-Core has achieved very high performance-per-watt, ranked #1 in Green500 list 3 times.
- We completed MN-Core 2. Achieved higher performance per watt and also higher performance per silicon area.
- Performance on real applications, both AI and general HPC areas, is higher than that of other accelerator made with same process tech.
- We have already started the development of next-generation processors, both for learning and LLM inference.

Talk outline

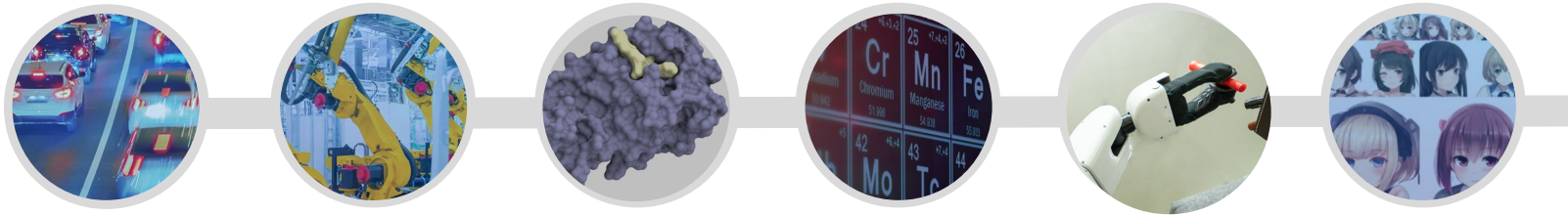
- Who are We?
- MN-Core
- MN-Core 2
- MN-Core 2 achieved performance
- Architecture of MN-Core 2
- Software for MN-Core 2
- Future direction
- Summary

Preferred Networks Inc.

Vision:
Make the real world computable

Founded	March 2014
Directors	CEO Toru Nishikawa CER Daisuke Okanohara
Located	Tokyo, Japan (HQ) Burlingame, CA., US (Preferred Networks America, Inc.)

Industry Domains

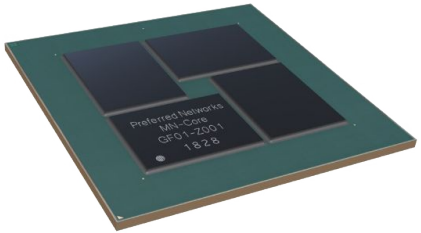


Transportation Manufacturing Life Sciences Materials Robots Entertainment

Preferred Networks

MN-Core: Accelerator for Efficient Deep Learning

Designed for efficient deep-learning training and backpropagation



Large-scale SIMD processor

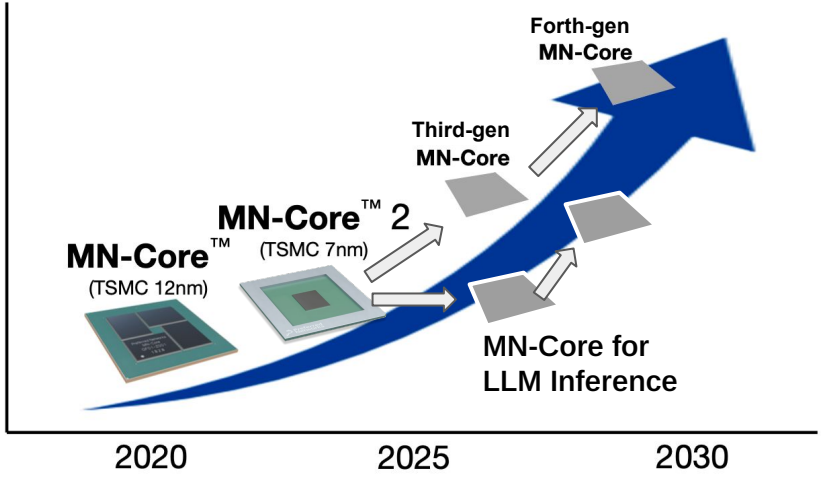
- Peak Performance: 32.8TFlops (double-precision)
- Chip Power: 500W
- Computing Efficiency: 65GFlops/W

Unique Architecture

- No data cache
- On-chip network wit tree topology and support for broadcast and reduction in all levels of the tree
- Designed for tensor-oriented operations of Deep Learning

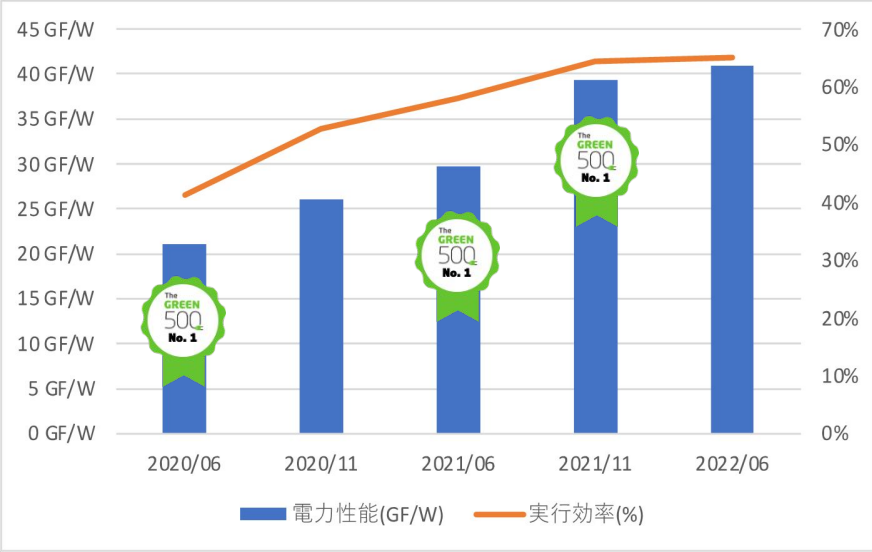
Our roadmap

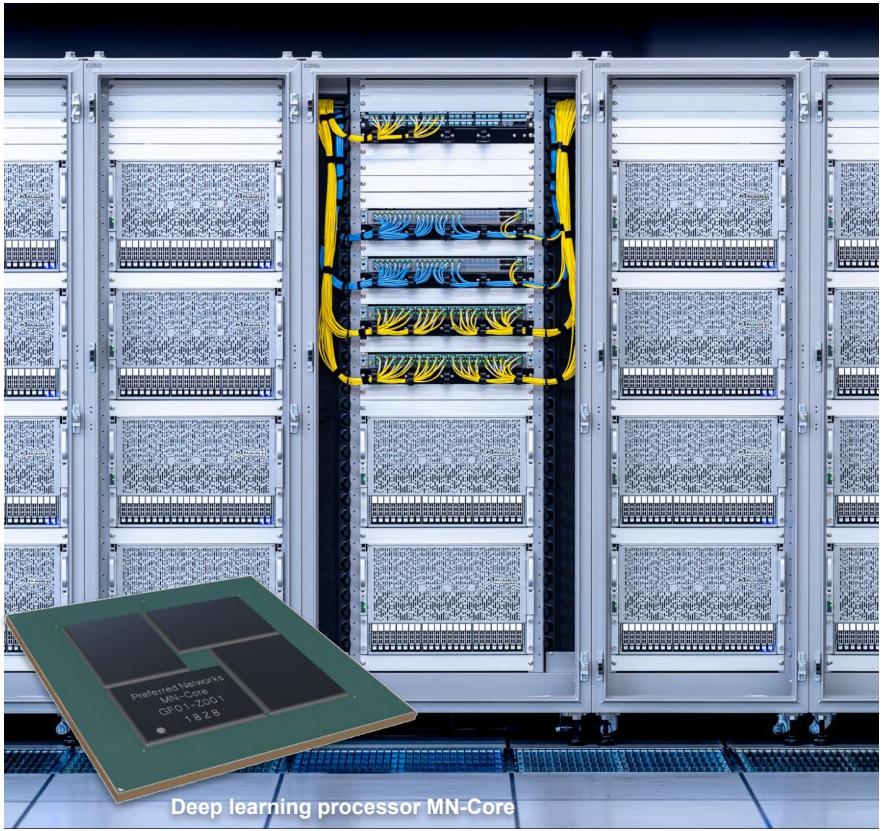
- Launched the project at 2017, we operate MN-3 supercomputer with the first generation accelerator MN-Core from 2020
- Second-generation MN-Core 2 is up and running.
- Future generations under development



Preferred Networks

MN-Core: #1 in Green500 list 3 times in 2 years





Our MN-3 supercomputer is certified as the world's most efficient supercomputer (Green500 List Nov. 2021).

Important effort towards achieving carbon neutral AI datacenter

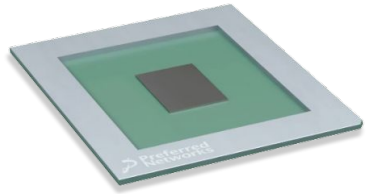
Preferred Networks

MN-Core 2

- Second-generation MN-Core processor
- Primary goal: Low production cost, while maintaining high performance and high performance-per-watt

	MN-Core	MN-Core 2
Technology	N12	N7
Die size and count	764mm^2 x 4	550mm^2
FP16 peak	532 TF	393 TF
Power Consumption	500W	330W
DRAM bandwidth	400GB/s	512GB/s

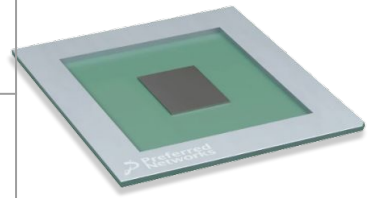
- Performance per die area improved by a factor of 4
- Performance per watt slightly improved
- TCO reduced by a large factor



Preferred Networks

Comparison with other processors

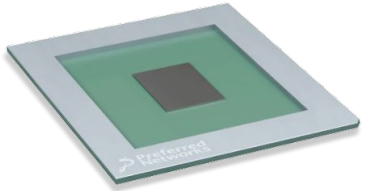
	A100	H100	MI300X	MN-Core 2
Technology	N7	N4	N5/N6	N7
Die size and count	826mm^2	814mm^2	1017mm^2 x 2	550mm^2
FP16 peak	312 TF	989 TF	1300TF	393 TF
TDP	400W	700W	750W	330W



- Highest performance/area for N7 process technology
- Superior performance/cost

Performance Comparison

	MN-Core 2	A100
GCN(PFN internal use)	5.41TF(FP32)	3.17TF
ResNet50 training	77TF(FP16)	33.2TF(BF16)
ResNet50 Inference	154TF(FP16)	33.7TF(BF16)
HIMENO benchmark	9.03TF(FP64)	0.634TF
OpenFDTD	0.655TF(FP32)	0.488TF



Comparison with A100

- Higher efficiency for ML workloads
- Significantly higher performance for memory-bound HPC workloads
  - Much larger on-chip memory
  - efficient algorithm such as temporal blocking

MN-Core architecture

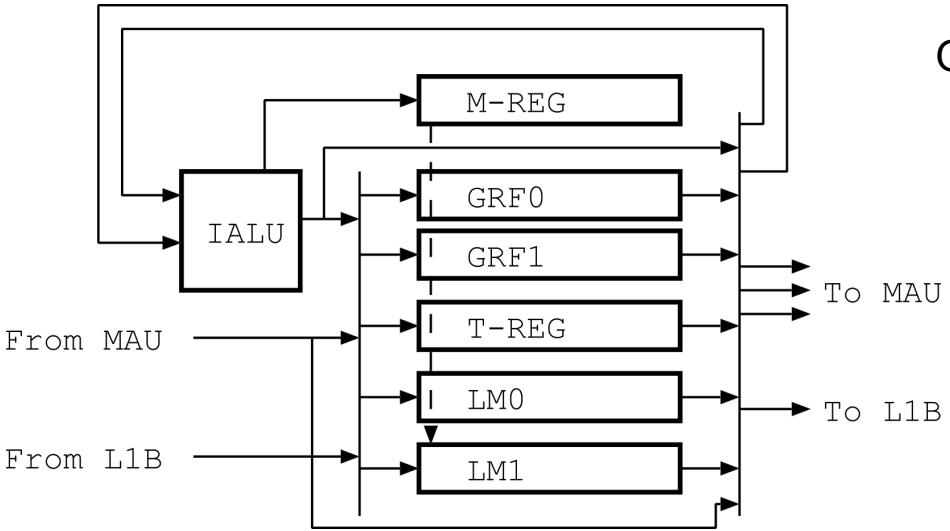
- Three levels: PE/MAB, L1B, L2B
  - PE: processing element
  - MAB: matrix arithmetic block
  - L1B: level-1 broadcast block
  - L2B: level-2 broadcast block
- MAB has matrix-vector multiply-and-add unit
- PE has local memory and registers
- All PEs work as a single large SIMD unit
- PEs communicate through L1 and L2 block
- L2Bs are connected to external DRAM (GDDR6)

Similar to large-scale SIMD machines like TMC CM-2 and MasPar MP-2

Network architecture simplified and optimized for deep learning (matrix/tensor operations)

Preferred Networks

MN-Core PE architecture



One PE (processing element) consists of

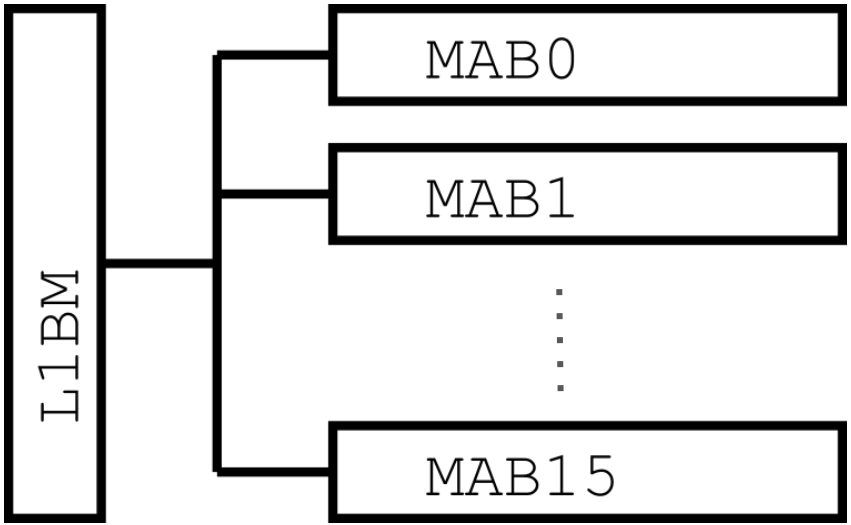
- IALU
- Ports to MAU and L1B
- two GRFs (256 64-bit words each)
- two Local memories (2048 64-bit words each)
- One work register (8 words). T-reg

Local memories can be specified as operands. Not a load-store architecture

Software-controlled multiple 1R1W GRFs, LMs, T-regs and feedback paths.

Compiler assigns variables to appropriate memory units to minimize the number of cycles

L1 (and L2) network



One L1B consists of

- One shared memory unit (L1BM)
- 16 MABs (each consists of 4 PEs+one MAU)
- Broadcast/reduction network

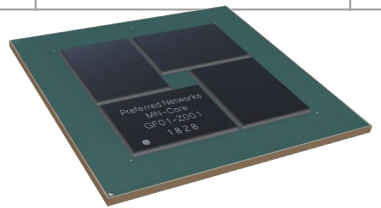
L1BM can broadcast data to all MABs and take summation (or other reduction operations) of data from all MABs

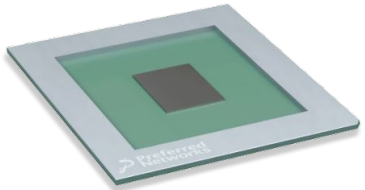
These are the two most important communication patterns for PEs used for tensor operations in deep learning. Compared to shared cache, energy consumption becomes much smaller

L2-L1 network has same logical structure

Architecture difference between MN-Core and MN-Core 2

	MN-Core	MN-Core 2
# or L2Bs	16	8
Clock Frequency	500MHz	750MHz
DRAM	32GB LPDDR4	16 GB GDDR6X
DRAM bandwidth	400GB/s	512GB/s
Host I/F	PCIe gen3 x16	PCIe gen5 x16
FP16 peak speed	532 TF	393 TF
Die size and count	764mm^2 x 4	550mm^2



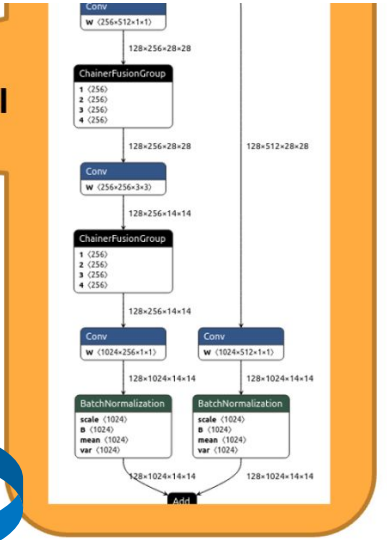


- Similar peak speed
- Die size 1/5
  = cost-effective
- Many architecture improvements to achieve higher efficiency
  - Second GRF added
  - L1B/L2B bandwidth increased
  - Direct feedback paths from MAU/IALU added

MN-Core Compiler for AI

PyTorch → PFVM

ONNX model



L3IR: DNN op level, SIMD parallelism strategy, ……

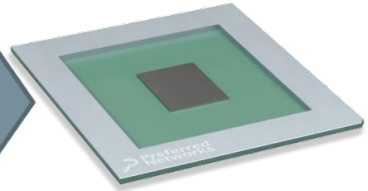
- MNGraph / MNNode / MNValue
- Global Layout Planner
- Re-computation Scheduler

L2IR/Generic Conv: ndarray op level, optimized DNN op impl, ……

- Generic Conv Impl
- MNTensor
- PEVector
- Generic Move Impl

L1IR: MN-core op level, memory allocation, scheduling, optimization, ……

- Layer Impl
- L1level instruction merger
- L1IR Scheduling Graph



Porting workloads MN-Core

Minimal change from Pytorch GPU code

Torch backend change

Use of special Trainer

```python
class MNISTCont(torch.nn.Module):
    def __init__(self):
        super().__init__()
        self._seq = torch.nn.Sequential(
            torch.nn.Conv2d(1, 32, 3, 1, padding=1),  # 32x32 -> 32x32
            torch.nn.ReLU(),
            torch.nn.Conv2d(32, 64, 3, 1, padding=1),  # 32x32 -> 32x32
            torch.nn.ReLU(),
            torch.nn.MaxPool2d(3, 2, padding=1),  # 32x32 -> 16x16
            torch.nn.ReLU(),
            torch.nn.Flatten(1),
            torch.nn.Linear(64 * 16 * 16, 128),
            torch.nn.ReLU(),
            torch.nn.Linear(128, 10),
        )

    def forword(self, x, t):
        y = self._seq(x)
        loss = torch.nn.functional.cross_entropy(y, t)
        return loss

m = MNISTCont()
opt = torch.optim.SGD(m.parameters(), lr=0.001, momentum=0.9)
model_opt = torch.compile(m, opt, backend=mncore.torch_backend)

for i in range(5):
    print("Epoch", i)
    for x, t in train_loader:
        print(model_opt(x, t))
```

MN-Core Version

```python
class MNISTCont(torch.nn.Module):
    def __init__(self):
        super().__init__()work
        self._seq = torch.nn.Sequential(
            torch.nn.Conv2d(1, 32, 3, 1, padding=1),  # 32x32 -> 32x32
            torch.nn.ReLU(),
            torch.nn.Conv2d(32, 64, 3, 1, padding=1),  # 32x32 -> 32x32
            torch.nn.ReLU(),
            torch.nn.MaxPool2d(3, 2, padding=1),  # 32x32 -> 16x16
            torch.nn.ReLU(),
            torch.nn.Flatten(1),
            torch.nn.Linear(64 * 16 * 16, 128),
            torch.nn.ReLU(),
            torch.nn.Linear(128, 10),
        )

    def forword(self, x, t):
        y = self._seq(x)
        loss = torch.nn.functional.cross_entropy(y, t)
        return loss

m = MNISTCont().cuda()
opt = torch.optim.SGD(m.parameters(), lr=0.001, momentum=0.9)
model_opt = torch.compile(m)

for i in range(5):
    print("Epoch", i)
    for x, t in train_loader:
        y = model_opt(x.cuda(), t.cuda())
        y.backward()
        opt.step()
```

GPU Version

Preferred Networks

MN-Core 2 Availability

- As of Aug. 23, 2024, MN-Core 2 is commercially available.
  https://projects.preferred.jp/mn-core/assets/MN-Core2-hardware-catalog.pdf

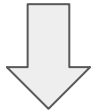
MN-Server 2 V1

MN-Core 2 Devkit

Preferred Networks

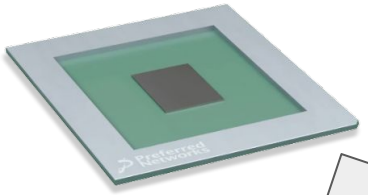
MN-Core Architecture Summary

- Matrix-vector unit in each PE (FP16, 32 and 64 operations)
  - logic circuit shared between different word lengths
- “register-memory” architecture to minimize data move and read/write
- hierarchical on-chip network with hardware support for broadcast/reduction to minimize data move between processor elements



- Very high performance per watt
- Very high performance per die area
- High application efficiency
- (rather unusual compiler)

MN-Core Future directions



MN-Core 2

MN-Core Next for learning
Higher computing power
> 10x peak performance
> 30x application performance
Samsung SF2

MN-Core Next for LLM Inference
Ultra-high memory bandwidth
> 20x inference performance

Preferred Networks

Summary

- MN-Core is a series of AI-oriented accelerator developed by PFN
- The first-generation MN-Core has achieved very high performance-per-watt, ranked #1 in Green500 list 3 times.
- We completed MN-Core 2. Achieved higher performance per watt and also higher performance per silicon area.
- Performance on real applications, both AI and general HPC areas, is higher than that of other accelerator made with same process tech.
- We have already started the development of next-generation processors, both for learning and LLM inference.

Preferred Networks

Making the real world computable

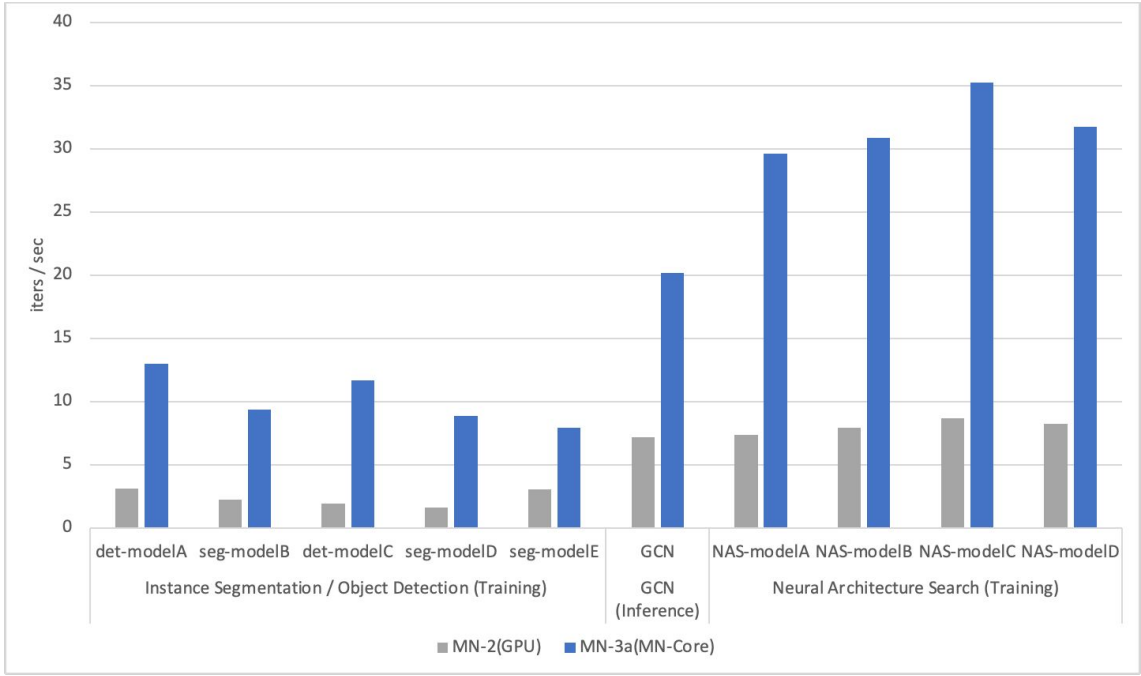
Preferred Networks

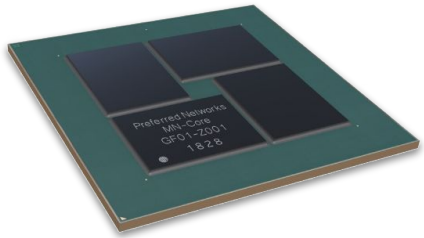
Backup Slides

MN-Core in Action

- We use MN-Core in R&D of deep learning applications
  - Neural Architecture Search (NAS) for optimized object recognition at edge devices
  - Material Informatics / Computational Chemistry. Etc.
    - Already has results of MD for more than 20mil. steps (>20ns)
- Average performance 3~5 times faster than existing GPUs

  ⇒ **shorter turnaround time**





Preferred Networks