Accelerator for LLM inference in the Chiplet era

Jun Makino

Kobe University/Preferred Networks

Summary

- We have finally come into the “Post-Moore” era, where the “shrink” of transistors does not give automatic improvement of the performance of computers.
- From the viewpoint of the computer architecture, the real problem is not the “Post-Moore” era, but the fact that the designs of CPUs or GPUs cannot take advantage of the advance of the semiconductor technology.
- This is similar to what happened to shared-memory parallel-vector machines 30 years ago. Distributed-memory machines took over.
- I’d like to discuss what is necessary to improve performance and how “Chiplets” can or cannot help.

Chiplet

Chiplet technology landscape

	2(2.5)D	3D
Logic-Logic	AMD/Intel CPU/GPU and many others	?
Logic-SRAM	?	AMD CPU/GPU
Logic-I/O	many ongoing?	?
Logic-DRAM	HBM	3D Stacked DRAM

- 2(2.5D) Chiplet: essentially a PCB with very fine pattern and pad pitches.
- 3D Chiplet: New technology not much utilized or even understood.

Chiplet

Chiplet technology landscape

	2(2.5)D	3D
Logic-Logic	AMD/Intel CPU/GPU and many others	?
Logic-SRAM	?	AMD CPU/GPU
Logic-I/O	many ongoing?	?
Logic-DRAM	HBM	3D Stacked DRAM

- 2(2.5D) Chiplet: essentially a PCB with very fine pattern and pad pitches.
- 3D Chiplet: New technology not much utilized or even understood.
  - We discuss the potential of 3D Logic-DRAM in this talk.

Talk Structure

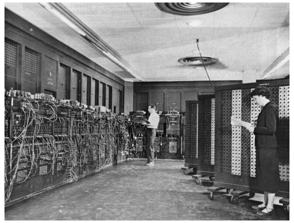
- How computers have evolved and why the evolution has slowed down
- (Non)-similarity between system-level and chip-level architectures.
- Requirement of LLM inference
- MN-Core
- “Chiplets”
- Summary

The evolution of computers

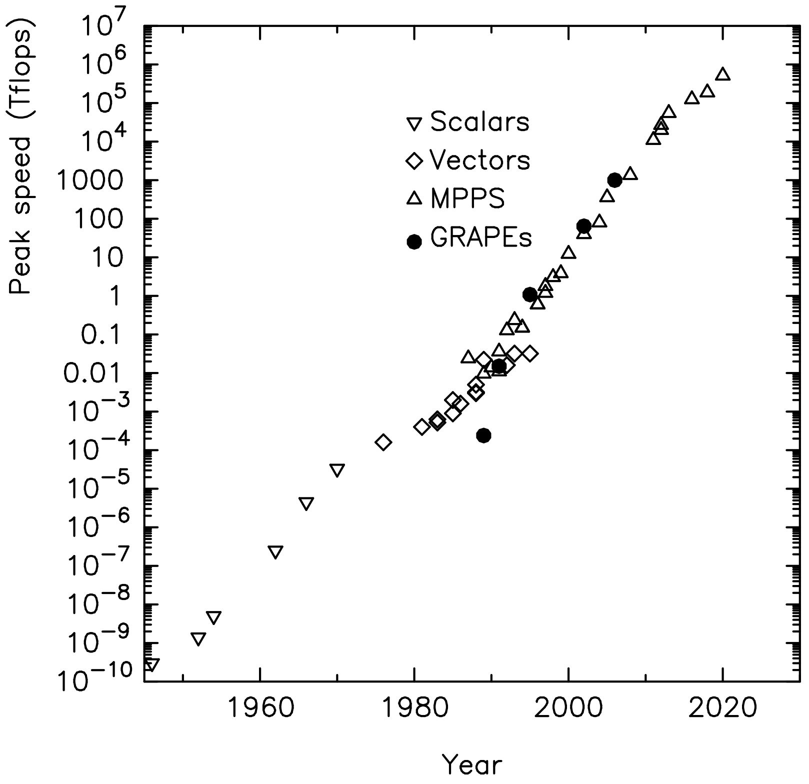
From 1940s to 2010s, the speed of computers have improved by roughly a factor of 100 in every decades. Why such an exponential growth was possible for such a long time?

Basic reasons:

- **Switching elements became faster**
- **Switching elements became smaller and less power-consuming**
- **Switching elements became cheaper**



Evolution of Supercomputers



- **Mostly machines in Top-500 list (except GRAPEs and machines before 1993)**
- **A bit less than 16 orders of magnitude in 80 years.**
- **How such improvement has actually achieved?**

(In 2020s, Top 500 list no longer represent the fastest or largest machines... Huge GPU clusters are used for LLM training.)

Improvement of the switching speed

- Actually vacuum tubes were already pretty fast.
- Signal propagation delay through wires have been always more important.
- Long time ago signals on wires propagate with the speed close to the speed of light, which is not so fast... (30cm/1ns)
- “Wires” in modern processor chips are so thin, the resistance is very large and RC delay becomes easily significant ($\propto l^2/d^2$ where l: wire length and d: wire width and height)
  - If the wire width is shrinked by a factor of k, the siginal delay for a given length INCREASES by a factor of k^2 — main problem with modern LSI design.

Miniaturization of switching elements

- The change of Vacuum tube $\rightarrow$ discrete transistors $\rightarrow$ IC was the driving factor until 1970s.
- Reduction of the power consumption was as important as size reduction.
- Since 1980, CMOS scaling has been the driving force. Roughly a factor of 10 shrink in 10 years.
- CMOS scaling law, which means the speed improves and power is reduced when CMOS transistor becomes smaller, had been the true driving force of computer evolution.
- CMOS scaling has reached the limit by around 2000, when the reduction of Vdd stopped.
- The miniaturization itself has become difficult and the reduction of the cost per transistor has almost stopped. (The end of the Moore's Law)

What is CMOS scaling?

Dennard Scaling

When we make the size (in all three dimensions) of a CMOS transistor k times smaller, and Vdd also k times smaller and increase the impurity concentration by the same factor of k, the switching speed is improved by k and power consumption per switching $1/k^2$.

This means, with d being the line width,

Clock speed $\propto 1/d$
Operating voltage $\propto d$
Performance $\propto 1/d^3$ (For same power and die area)

Semiconductor evolution

- Transistor size had shrunk exponentially until 2015.
- CMOS scaling law has broken by around 2000 (90nm)

What happened?

- Down to 90nm, clock was improved by keeping voltage high.
- This resulted in near-exponential increase in power consumption.
- So the evolution changed the direction to multicore and SIMD units.
- By around 2015, multicore and wide SIMD approach has become difficult, and power consumption started to increase again...

This does not look like the most clever way...

Looking back the evolution of computer architecture

- Until 1976: Scalar computers. The last one: CDC 7600
- 1976 to 1992: Shared memory parallel vector processors. Cray-1 to C-90.
- 1993 to 2008: Distributed-memory parallel microprocessors. Cray T3D to Cray XT4.
- Since 2008: CPU + GPU (or some other accelerator). IBM Roadrunner

Roughly in every 15 years big change architecture occurred.
The successor of GPU has not appeared yet.

Why the change was necessary?

Basic reason: Existing architecture became unable to make use of the advance in the semiconductor technology

- **advance in the semiconductor technology**
- **limit in the scalability of the architecture**

Scalar to vector

- Scalar computer: use the increased number of transistors to make a faster arithmetic unit.
- Magnetic core memory, much slower than transistors
- CDC 7600 reached the limit: fully pipelined arithmetic unit
- S. Cray started the development of 4-processor CDC 8600, but project canceled and Cray established Cray Research.

Scalar machines could not make use of

- Available gate count much larger than that for fully pipelined arithmetic unit
- Very fast SRAM main memory

Vector to MPP

- Assumption of vector archtecture: Memory bandwidth can be made large enough to support arithemetic units.
- Advance of vector processors: increase in pipeline per processor and number of processors which share the physical memory.
- Wires and switches increase faster than number of pipelines. 64 pipelines seem to be the practical limit — assumption broke down.
- Need to move to a system made of large number of simple processors, each with small memory, connected with relatively thin network.
- Early examples: Intel Touchstone Delta, Cray T3D

MPP to GPU

- It becomes possible to fit a large number of processors (pipelines) in one chip.
- The same situation as that of vector-parallel processors.
- Many-core processors have hierarchical cache with coherency.
- hardware and power consumption to maintain coherency becomes dominant.
- GPU relaxes/removes coherency and thus push up the limit a bit.

GPU to ???

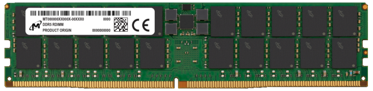
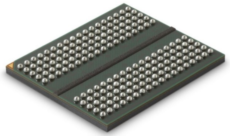
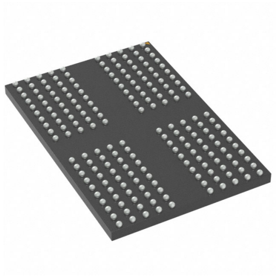
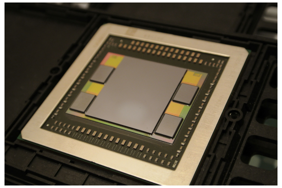
- Even without coherency, the data movement between off-chip DRAM and multiple levels of cache memory becomes the bottleneck.
- The “obvious” solution is to give up the cache hierarchy completely and move the main memory physically close to processors.
- In other words, we need to move to on-chip distributed-memory processor.
- However, we do not know how to make such a thing, unless we give up DRAM memory completely.

(Non)-Similarities between system-level and chip-level evolution

System	Chip	
Up to CDC 7600	Up to Intel 80860	single pipe
Cray 1 to T90	to Intel KNL	multiple pipe, shared memory
Stanford DASH/SGI Altex	GPUs? NUMA CPUs?	distributed-shared memory(??)
Cray T3D to K/Fugaku	—	distributed memory
GPU clusters	—	non-coherent cache

No analogue of distributed-memory systems in the chip side. Needless to say about GPU clusters...

What we did instead

DDRx(2000〜) **GDDRx(2003〜)** **LPDDRx(2008〜)** **HBMx(2015〜)**

- **DRAM cell structure and power consumption has not change much.**
- **Main change: wire length between DRAM chips and XPU.**
- **Access energy: DDRx: 20pJ/bit, HBMx: 4pJ/bit. Very large reduction. But not enough.**

Why the horizontal data move so hard?

Delay and power consumption.

The capacitance of a wire per unit length is independent of the line width d.

On the other hand, the resistance per unit length is proportional to d^{-2}.

- **The propagation delay of a wire $\propto d^{-2}$**
- **The power consumption is constant.**
- **Thus, it is very difficult to increase the bandwidth of data move for a fixed length.**

Some exercise

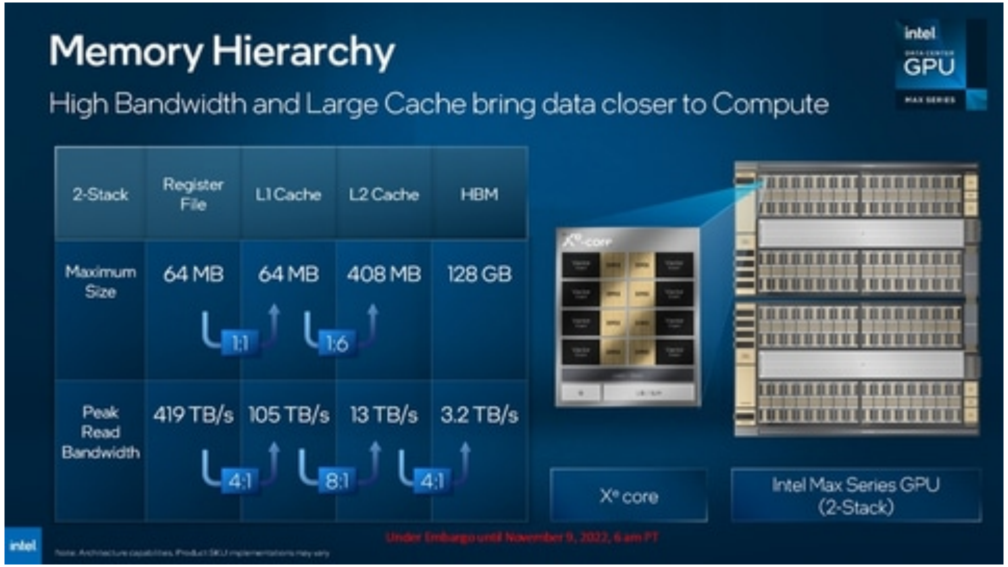
- A wire of length 10cm ($10^5 \mu\mathrm{m}$) has 20pF capacitance. For 1V swing, it consumes 10pJ/bit.
- Actual power consumption of a DDR5 is around 20pJ/bit. Voltage is around 1.3V.
- LPDDR5 and GDDR6x: $\sim$ 10 pJ/bit Wire length is shorter than that of DDR5 modules. and swing voltage is smaller.
- HBMx: 3-4 pJ/bit. Wire length is around 25mm.

Modern GPUs spend around 50% of total power to move data from HBM to L2D$.

Intel GPU example

B/F: number of bytes you can move between memory and arithmetic unit while you do on FP(usually FP64) operation.

B/F values



Memory Hierarchy

High Bandwidth and Large Cache bring data closer to Compute

2-Stack	Register File	L1 Cache	L2 Cache	HBM
Maximum Size	64 MB	64 MB	408 MB	128 GB
Peak Read Bandwidth	419 TB/s	105 TB/s	13 TB/s	3.2 TB/s

1:1 1:6

4:1 8:1 4:1

intel DATA CENTER GPU MAX SERIES
X^e core
Intel Max Series GPU (2-Stack)

Note: Architecture capabilities, Product SKU implementations may vary

Under Embargo until November 9, 2022, 6 am PT

Registers: 8? (One usually need 16...)

L1: 2

L2: 0.25

Memory: 0.06

These are very small numbers, but yet caused the excessive power consumption...

Other GPUs

	FP64 perf. (TF)	L2 BW(B/F)	DRAM (B/F)
V100	7.8	0.32	0.12
A100	19.5	0.25	0.08
AMD MI250X	47.9	0.27	0.07
H100	34	0.17	0.10
Intel MAX	52(31)	0.25	0.06

- **L2 bandwidth numbers are very small.**
- **B/F numbers are low and that results in rather low application efficiency (except for "traditional" ML applications where all you need is matrix multiplication)**
- **LLM requires extremely high memory bandwidth. Very different from "traditional" ML applications.**

Characteristics of LLM (or transformers)

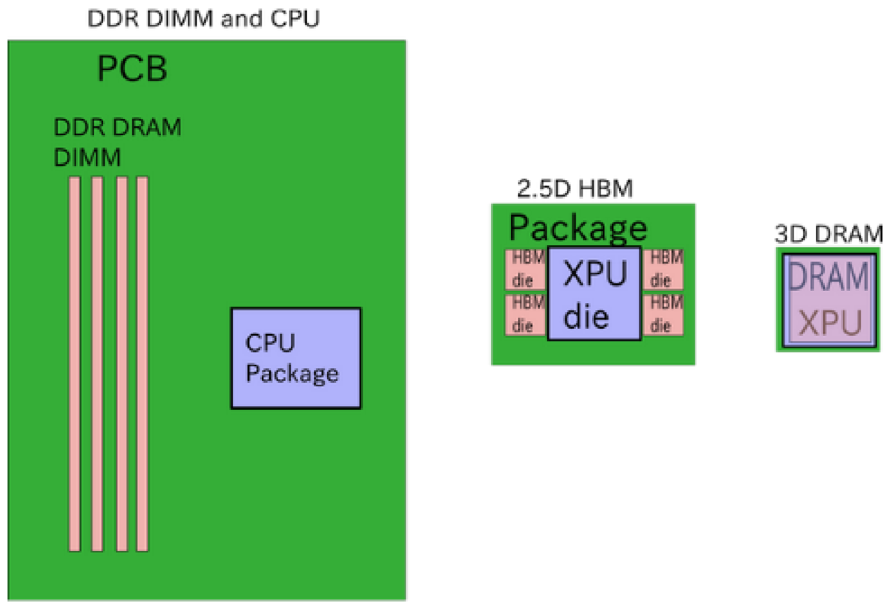
- Models have a very large number of parameters, such as 70B, 400B etc.
- These parameters are used to generate one token for one user (with many users parameters are shared, but contexts are not)
- Edge or private uses require rather small batch size.
- If we want to achieve 500 tokens/s with 400B model, we need 200TB/s of memory bandwidth for 400GB or memory on a tightly coupled machine (currently impossible).

Post-GPU architecture or LLM accelerator architecture

Processors with Post-GPU architecture for LLM should have very large memory bandwidth for a fairly large memory and with a reasonably low power.
In order to increase the memory bandwidth without increasing the power consumption too much, we must further reduce the memory access energy per bit.
This means we need to reduce the physical distance between processor cores and memories.

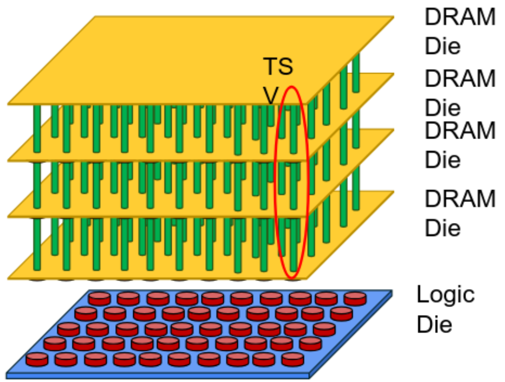
But how?

What we can do and how will it look like.



We need to realize "3D memory", in other words, to put DRAM on top (or bottom) of the processor die.
We also need to change the memory hierarchy.

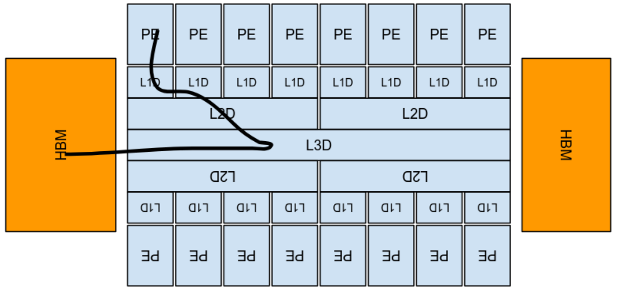
3D Stacking (Chiplet?)

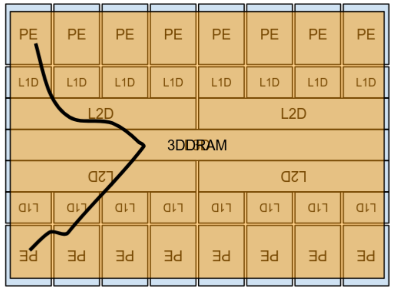


Connect multiple DRAM dies and a logic die using a very large number of TSVs and pads.
Microbumps or Hybrid bonding are used for connection.
Several (small) DRAM companies are working on this technology.
Taiwan: Powerchip, Winbond, Nanya
China: CXMT, YMTC, XMC.

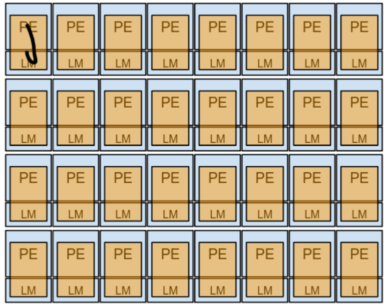
This might be the second (and last...) time that the large jump in the memory bandwidth occurred in the history of computer architecture.
First time: semiconductor memory.

Stacked DRAM with shared and distributed arch





Shared memory: Data move distance not much different from that of HBM



Distributed memory: Data move minimized

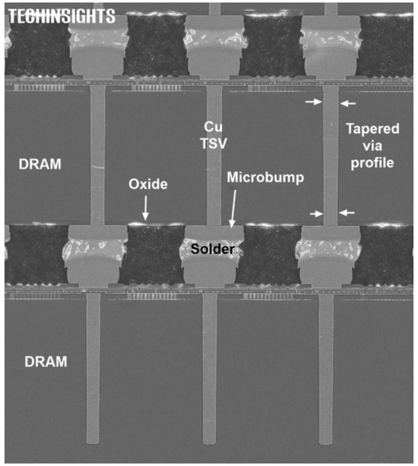
Now we can thing of chip-level analogue of distributed memory parallel computer.

Similarities between system-level and chip-level evolution

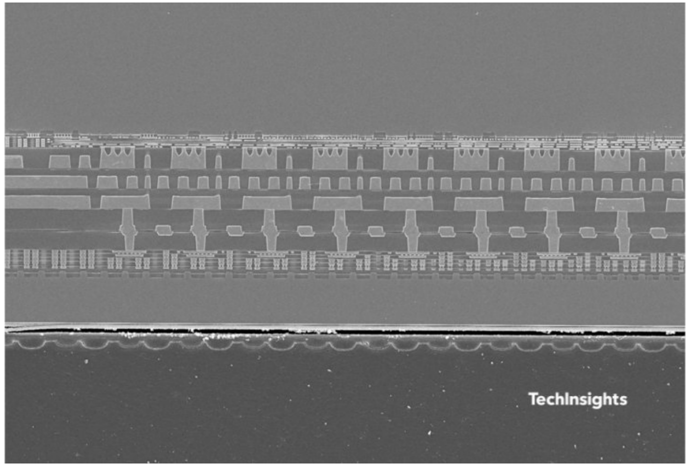
System	Chip	
Up to CDC 7600	Up to Intel 80860	single pipe
Cray 1 to T90	to Intel KNL	multiple pipe, shared memo
Stanford DASH/SGI Altex	GPUs?NUMA CPUs?	distributed-shared memory(
Cray T3D to now	On-chip distributed memory processor with 3D DRAM	distributed memory

Now there might be an analogue of distributed-memory systems in the chip side.

Microbumps and Hybrid Bonding



Microbumps



Hybrid Bonding

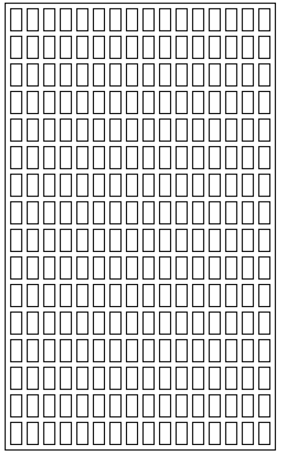
Microbump

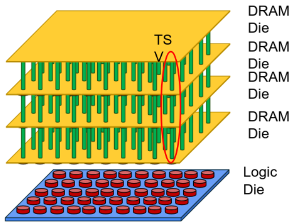
- **Extension of solder balls and “C4” bumps.**
- $\sim 40\mu\mathrm{m}$ **pitch is available. In Intel MAX GPU** $37\mu\mathrm{m}$ **pitch has been used.**
- **Will go below** $10\mu\mathrm{m}$ **in future.**
- **Used in all HBM memories shipped so far up to HBM3e.**
- **High yield is possible since we stack dies after they are cut out from wafer.**
- **Heat resistance is large = cooling is problem.**

Hybrid bonding

- Cu pad on dies are bonded through thermal process, after dies are bonded (no adhesive or whatever is used. SiO2-SiO2 “direct bonding”)
- First used in Sony’s CMOS image sensors.
- $\sim 5\mu\mathrm{m}$ pitch is available now.
- $< 1\mu\mathrm{m}$ will be available.
- Roughly 100x more pads can be used compared to microbumps.
- Heat resistance is quite low. “DRAM on top” structure is possible.
- Bonding process is “Wafer-on-Wafer”. So the cost should be low. However, there is no way to remove defective dies before bonding. So some new design method which can achieve near 100% yield is essential.

DRAM design image





Very large number of “small” DRAM blocks.

16x16, 32x32 etc.

Each block has its control input, address input and data I/O pads.

Example: $200\mathrm{Mbit}/\mathrm{mm}^2$ density DRAM,

$800\mathrm{mm}^2$ die = 160Gbit (effective 144Gbit)

1024 144Mbit blocks (with ECC). 144bit I/O.

Total pads/die = 150K. With 4 dies 600K pads.

500MHz data rate gives 40TB/s.

So extremely high memory bandwidth is mechanically possible.

Question: power consumption.

Power consumption and capacity

Current goal with 3D DRAM: 0.5pJ/bit (around 1/15 of the actual power consumption of HBM)
This means 40TB/s = 320 Tb/s = 160W. NVIDIA Rubin Ultra: 32TB/s, 3-4kW.
Can be 20x more energy efficient compared to Rubin Ultra.
Practical problem: 4 DRAM dies of $800\mathrm{mm}^2$ size gives only 72GB. We need much more.

- **Use more dies per package (possible)**
- **Stack more DRAM dies (possible)**
- **Use DRAMs with more advanced process technology (maybe in future — $200\mathrm{Mbits}/\mathrm{mm}^2$ is what is available now from second-tier DRAM companies)**

How far can we go?

- With hybrid bonding a very large number of pads is possible. This means that the DRAM design can be greatly simplified.
- For example, almost all of the digital logic circuits on current synchronous DRAM design can be moved to the logic die side.
- 0.1-0.2 pJ/bit is within reach (1/20 — 1/40 of HBMx)
- DRAMs with IGZO FET might further reduce the access energy (cell capacitance might be reduced by several orders of magnitude)

Design challenges

Many, many challenges...

- **Physical design of DRAM-logic interface: what buffer cells can be used, how to interface signals with different voltages, how to actually connect the DRAM pads and logic IF cells if their physical locations do not exactly match (need RDL), how to design power grid. and so on...**
- **Yield: Hybrid bonding is “Wafer-on-wafer” bonding. No way to select good dies before stacking. (Almost) all dies must be “good”.**

I *believe* **we can come up with reasonable solutions for all of these practical issues.**

Impact on processor architecture

Again a number of them, mostly not well understood.

- Processor architecture. SIMD/MIMD/something in between?
- Memory hierarchy
- Inter-processor network architecture
- DRAM interface design

In other words: EVERYTHING we learned about computer architecture in the last 40 years is based on the assumption that memory hierarchy is necessary because DRAM is slow. This assumption might be no longer true.

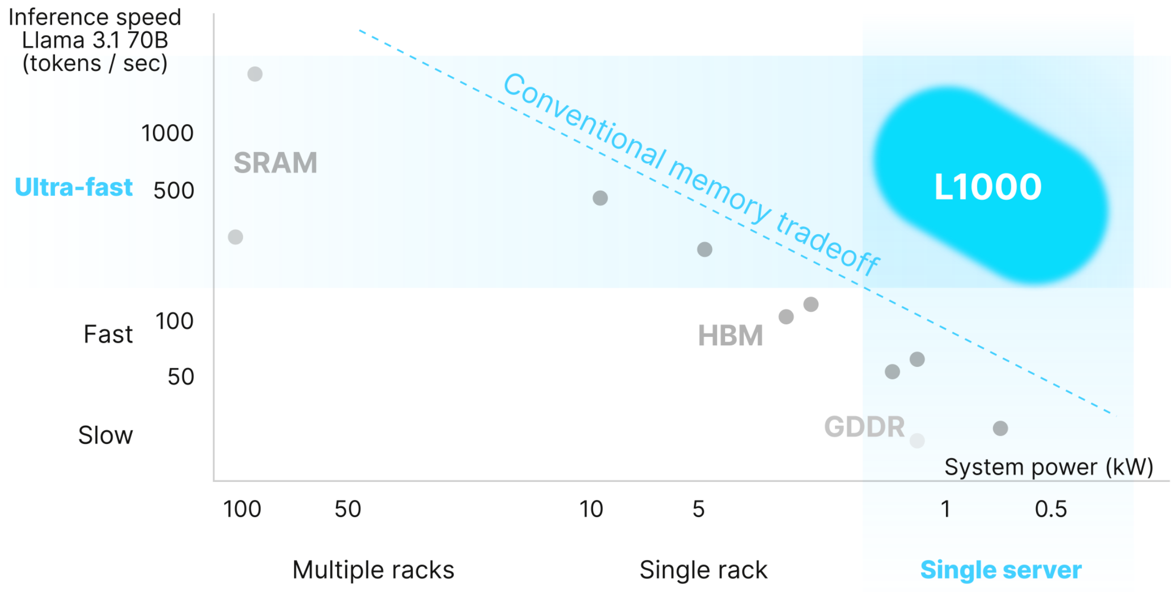
Impact on algorithms/applications

- **0.1pJ/bit corresponds to 312.5GF/W(FP64) for B/F=4.**
  - **Maybe we can forget about cache-efficient (and thus cache-oblivious) algorithms and go back to the era of vector processors with B/F=4.**
- **Further reduction of DRAM access energy seems possible, while the reduction of the logic power consumption seems difficult.**
  - **Algorithms which reduce the arithmetic operations while increasing the memory access might become important.**

MN-Core L1000

Preferred Networks"s answer to the challenge of designing on-chip distributed-memory processor

Preferred networks is currently developing MN-Core L1000, which will provide 10x more LLM inference performance compared to GPUs, by 2026.



https://mn-core.com

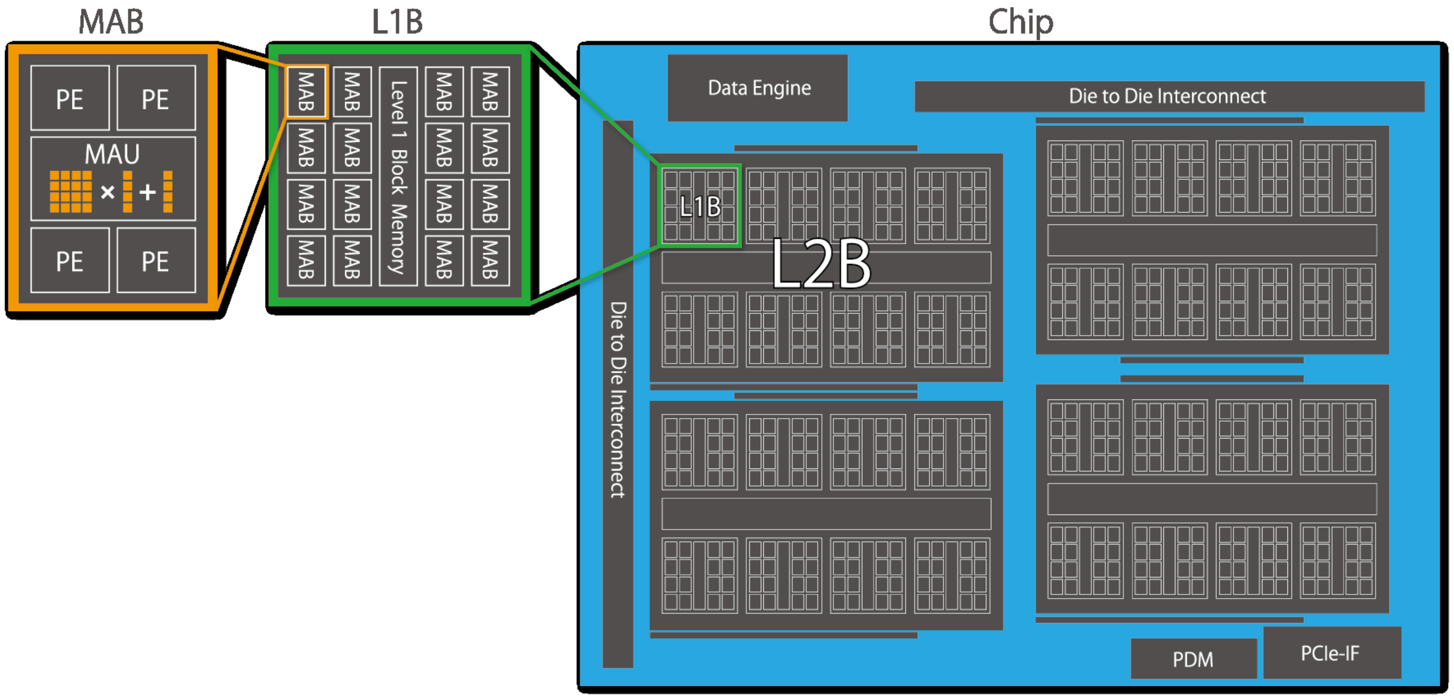
MN-Core

- AI-oriented processor developped by Preferred Networks (PFN) and JM.
- (JM moved to the cross-appointment position between Kobe U and PFN as of Nov 2023)
- First gen completed in 2020
  - At the time of completion, achieved highest FP16 performance per board and highest performance per watt number.
  - FP16 Peak 524TF
  - Power consumption less than 500W, 1.2TF/W
  - 2.5x higher performance per watt compared to NVIDIA V100 (both made with TSMC N12 process)

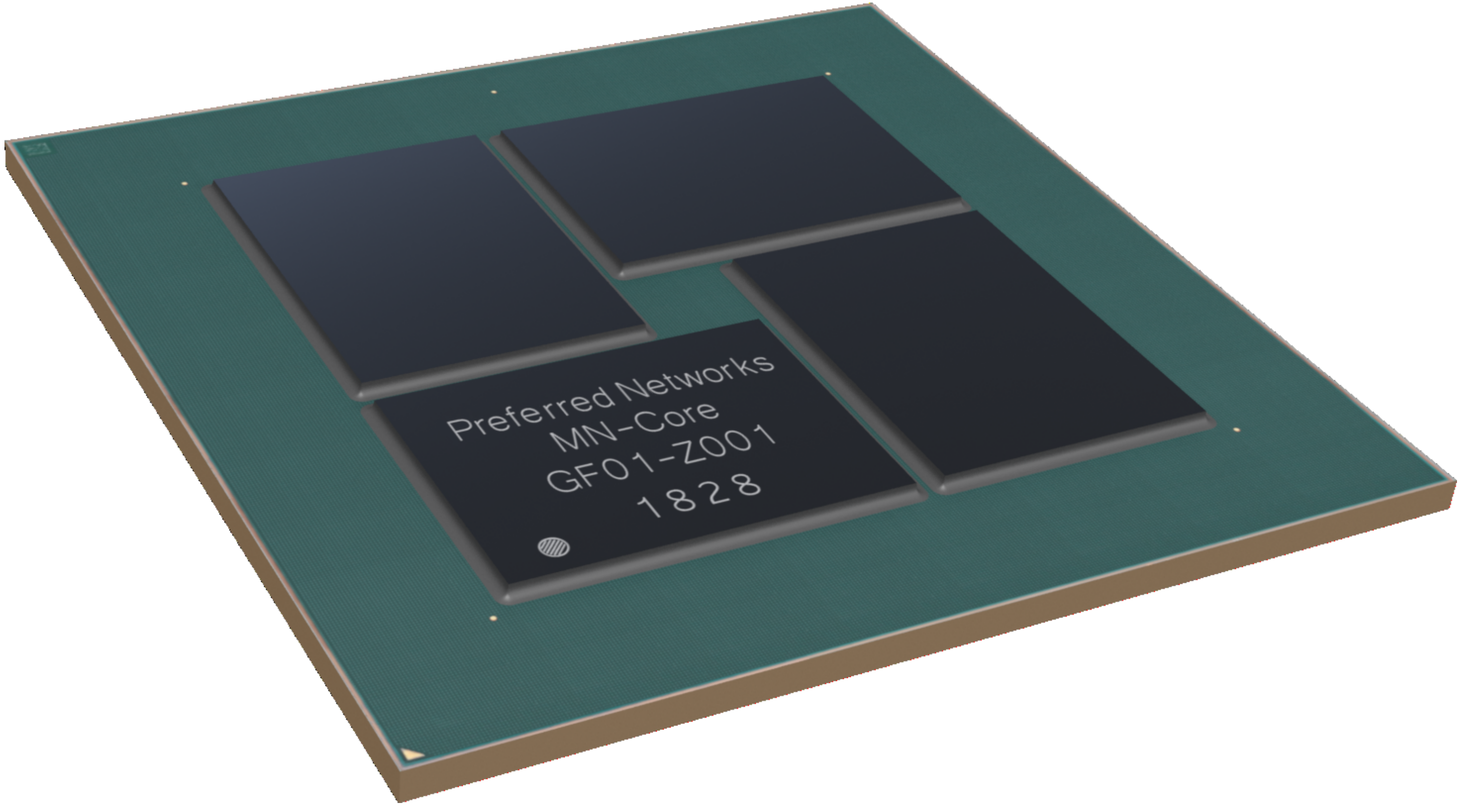
MN-Core overview

- 1 card : 1 module
- 1module : 4-die MCM
- 1 die: PCIe (gen4, x16), LPDDR4 memory, 4 “Level-2 broadcast blocks” (L2Bs)
- 1 L2B: 8 L1Bs
- 1 L1B: 16 MABs (Matrix Arithmetic Blocks)
- 1 MAB: 4 Processor Elements and one Matrix arithmetic unit
- FP64:FP32:FP16 performance ratio is 1:4:16
- Entire module operates as one huge SIMD processor with single instruction stream.

MN-Core Structure



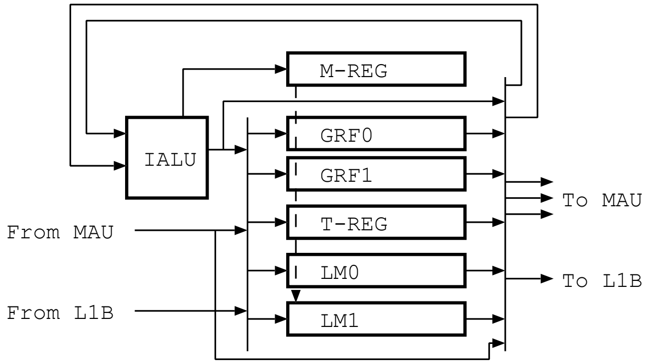
MN-Core



Details

- PE(Processing element)
- L1B(Level 1 broadcast block)
- L2B(Level 2 broadcast block)

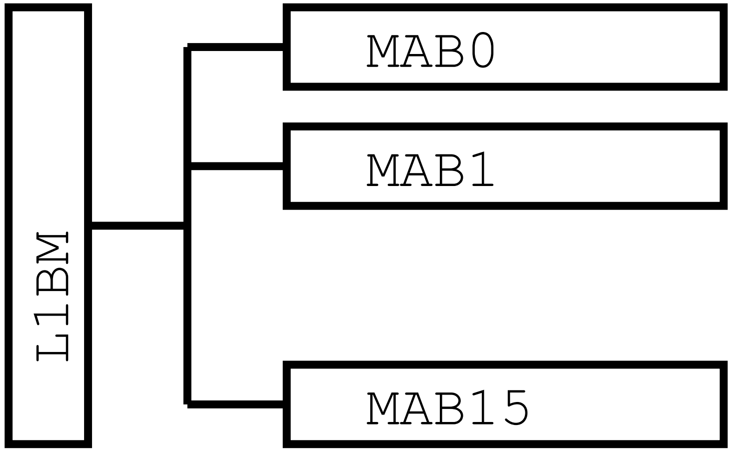
PE(Processing element)



- IALU and MAU as arithmetic units
- MAU performs FP64, FP32 and FP16 matrix-vector product

- GRF are 1R1W 2-port memories. LMx are single-port
- T-reg: additional register, 1R1W 4 words (vector length)
- LMx (local memory): 2048 64bit-words x 2, GRF: 512 words
- all instructions are vector instructions with length 4

L1B(Level 1 broadcast block)



- 16 MABs are connected to one L1BM(level 1 broadcast memory)
- Data read from L1BM are broad-casted to all PE (or MAB).

- Data read parallel from all PEs/MABs can be summed up and stored to L1BM with full speed.
- No direct connection between PEs.

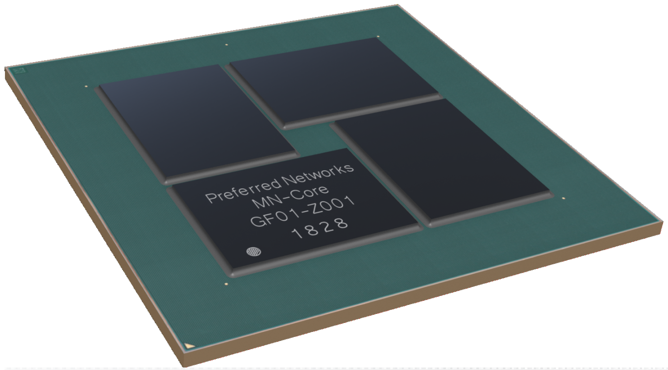
L1B characteristics

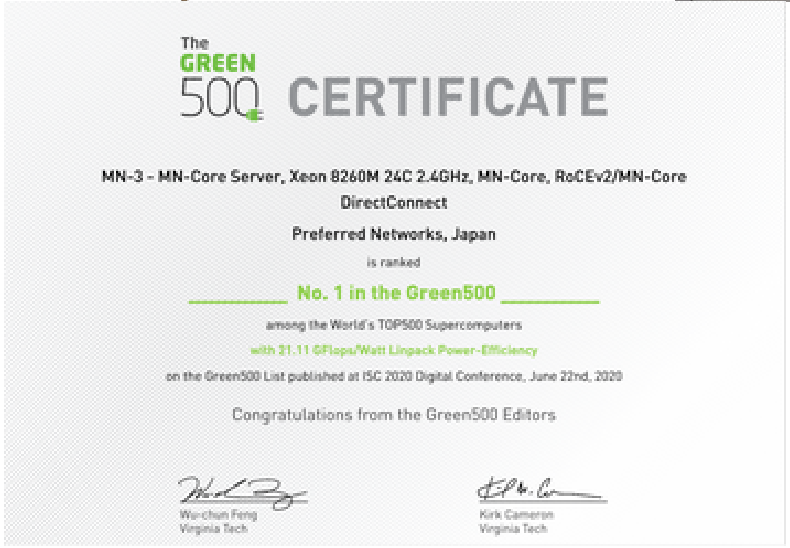
- Using explicit broadcast/reduction, fine-grained parallel operations can be performed with very low overhead.
- In particular, reduction operations over multiple PEs. which are very slow on GPUs, can be done very quickly.
- As a result, large number of PEs can be used to parallelize relatively small matrix product. This feature is actually very important for the inference performance of both CNN and LLM.
- L2B and top-level structures are largely the same as that of L1B.

Difference from usual processor architecture

- With present-day high-performance CPUs, from the machine code one need to restore the parallelism by means of register renaming and OoO execution.
- With MN-Core architecture, very large number of registers are all visible and there is no need for register renaming.
- By using fixed-length vector instruction, we removed the need of out-of-order execution as well. There is no need of real-time instruction scheduling since the result of one instruction is available to the next instruction.
- A large number of visible registers requires very long instruction words, which is okay because of chip-wide SIMD architecture.

MN-Core/MN-3 system




The GREEN 500 CERTIFICATE

MN-3 - MN-Core Server, Xeon 8260M 24C 2.4GHz, MN-Core, RoCEv2/MN-Core DirectConnect

Preferred Networks, Japan

is ranked

No. 1 in the Green500

among the World's TOP500 Supercomputers

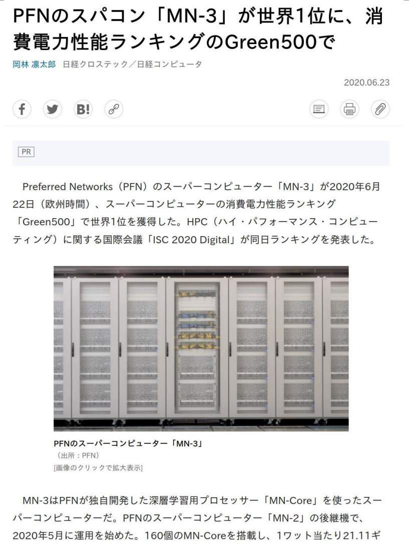
with 21.11 GFlops/Watt Linpack Power-Efficiency

on the Green500 List published at ISC 2020 Digital Conference, June 22nd, 2020

Congratulations from the Green500 Editors

Wu-chun Feng
Virginia Tech

Kirk Cameron
Virginia Tech


PFNのスパコン「MN-3」が世界1位に、消費電力性能ランキングのGreen500で

岡林 凛太郎　日経クロステック／日経コンピュータ

2020.06.23

PR

Preferred Networks（PFN）のスーパーコンピューター「MN-3」が2020年6月22日（欧州時間）、スーパーコンピューターの消費電力性能ランキング「Green500」で世界1位を獲得した。HPC（ハイ・パフォーマンス・コンピューティング）に関する国際会議「ISC 2020 Digital」が同日ランキングを発表した。

PFNのスーパーコンピューター「MN-3」
（出所：PFN）
[画像のクリックで拡大表示]

MN-3はPFNが独自開発した深層学習用プロセッサー「MN-Core」を使ったスーパーコンピューターだ。PFNのスーパーコンピューター「MN-2」の後継機で、2020年5月に運用を始めた。160個のMN-Coreを搭載し、1ワット当たり21.11ギ

MN-Core 2 and next generations

- MN-Core2 was completed in 2023. Now commercially available.
- Performance comparable to MN-Core with 1/5 of die area.
- Development of next generations already started.
  - Samsung 2nm, should achieve highest performance for training.
  - Also started the development of new processor for LLM inference.

Software for MN-Core 2

- MLSDK: AI-oriented
  - PyTorch — ONNX — actual machine code.
  - Existing PyTorch code (should) work with small changes.
- HPCSDK: For General-purpose HPC
  - Dialects of OpenCL and OpenACC
  - OpenACC direct resembles HPF.

Application performance of MN-Core 2

	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(FP32)	0.634TF
OpenFDTD	0.655TF(FP32)	0.488TF

- Performance 1.5-5 times higher than that of A100
- Very high performance for finite-difference applications. (OpenFDTD implementation used the temporal blocking and HIMENO benchmark fits to the on-chip SRAM).

MN-Core and 3D DRAM

- With MN-Core architecture, it is natural to add DRAM units to each PE.
- Physical latency of DRAM (in particular in the same page) is very small, and the total bandwidth is very large. So we can connect DRAM and arithmetic units “directly”. No need for caches.
- Registers might be of some help.
- Very similar to large-scale SIMD machines like TMC CM-2 and MasPar MP-2. So the programming model will be similar. Data parallel languages like HPF (OpenACC) can be used.

For anyone who still remembers CM and Maspar and Garma Zabi

Q: CM and MasPar are dead. Why?

A: Because off-chip bandwidth became insufficient to support arithmetic performance (the Memory Wall/the Attack of Killer Micros). The same reason as vector processors died.

Q: So why do you think they can be resurrected?

A: Because DRAM is no longer "off-chip" with 3D stacking. With 3D staching, DRAM can offer bandwidth high enough to support arithmetic performance.

Chiplet

Chiplet technology landscape

	2(2.5)D	3D
Logic-Logic	AMD/Intel CPU/GPU and many others	?
Logic-SRAM	?	AMD CPU/GPU
Logic-I/O	CPO?	?
Logic-DRAM	HBM	3D Stacked DRAM

- **2(2.5D) Chiplet: essentially a PCB with very fine pattern and pad pitches.**
- **3D Chiplet: can realize a huge reduction in the energy to move data. Opens up new dimensions in the design space of computer architecture.**
  - **We discussed the potential of 3D Logic-DRAM in this talk.**
  - **Logic-Logic integration might give interesting possibilities.**

Summary

- We have finally come into the “Post-Moore” era, where the “shrink” of transistors does not give automatic improvement of the performance of computers.
- From the viewpoint of the computer architecture, the real problem is not the “Post-Moore” era, but the fact that the designs of CPUs or GPUs cannot take advantage of the advance of the semiconductor technology.
- This is similar to what happened to shared-memory parallel-vector machines 30 years ago. Distributed-memory machines took over.
- In this talk, I focused on the potential of 3D integration of DRAM and logic, and its impact on computer architecture. Logic-logic 3D integration might open up new possibilities as well.