Toward Latency-Aware Data Arrangement on Many-Core Processors

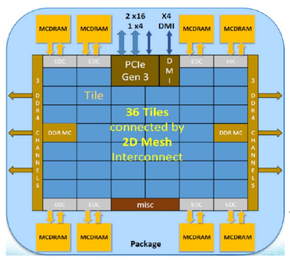
Tomoya Yuki[1], Toshio Endo[1] ([1]Tokyo Institute of Technology)

Background

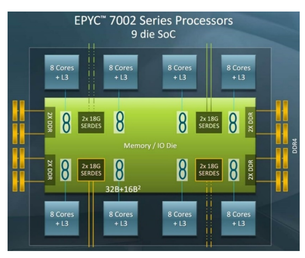
Recently, many processors which have a complex structure are coming out

- They are composed of unique structure: unique placement of cores and memory controllers
- They have many processor cores. Future architectures will have dozens or handreds cores.
- They and the whole systems using them may have a heterogeneous memory hierarchy composed of Cache, HPM (MCDRAM/HBM), DDR, NVM, SSD, HDD.

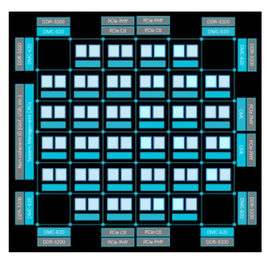
Intel Xeon Phi KNM [1]



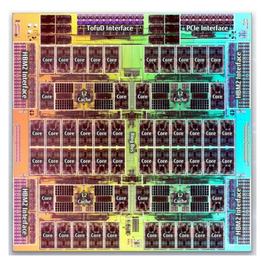
AMD EPYC Rome [2]



ARM Neoverse N1 [3]



Fujitsu A64FX [4]



Do **Latenty variance** caused by placement of cores and memory modules affect performance?

- Farther cores from a memory controller have high latency, nearer ones have low latency? If so, how much?
- How should we treat memory hierarchies properly? Interleaving many modules gets high bandwidth but on the other hand it causes high latency.
- What should we do to get the best performance on such processors or systems?

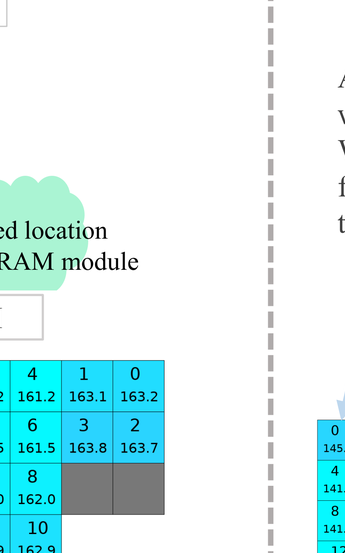
[1] https://www.anandtech.com/show/9802/supercomputing-15-intels-knights-landing-xeon-phi-silicon-on-display
[2] https://www.techspot.com/news/81350-amd-unveils-2nd-gen-epyc-cpus-world-fastest.html
[3] https://pc.watch.impress.co.jp/docs/column/kaigai/1202957.html
[4] https://www.nextplatform.com/2019/11/13/a64fx-arm-chip-gets-a-big-push-from-cray/

Latency Checker

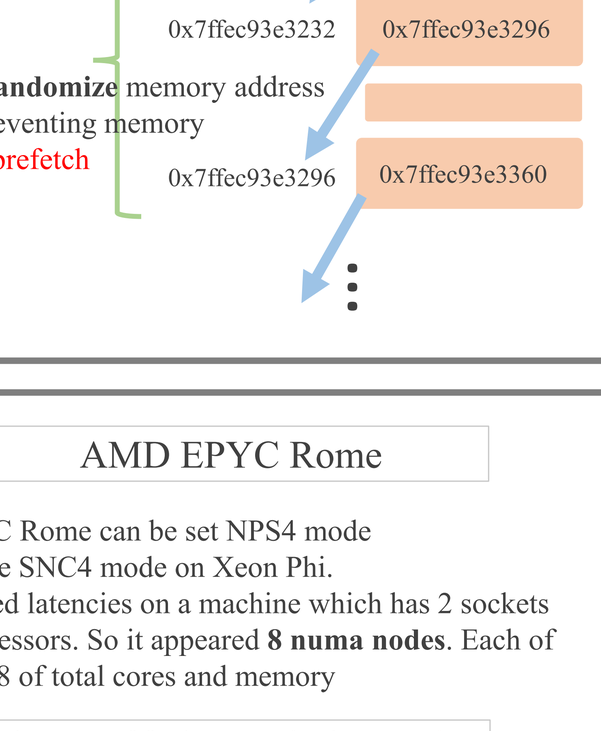
We made a **latency checker tool** for our preliminary experiments. It works on not only standard Xeon processors and also Xeon Phi and AMD EPYC processors.

Latency Checker functions
- ✓ Consider Cacheline size
- ✓ Linked List
- ✓ Random access mode
- ✓ Separate memory interleaving

To **separate memory interleaving**, the tool use stride access



Linked List prevents memory access from pipline



Can **randomize** memory address for preventing memory from prefetch

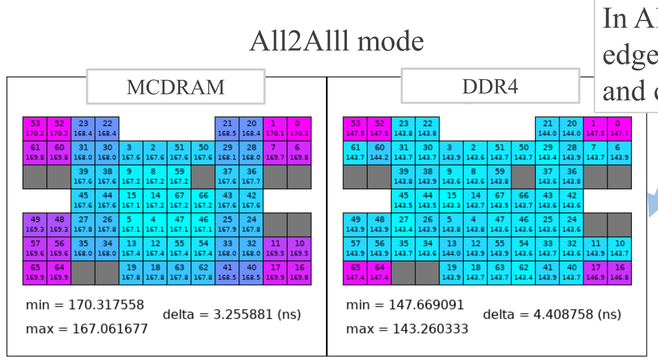
Preliminary Experiments

Intel Xeon Phi KNM

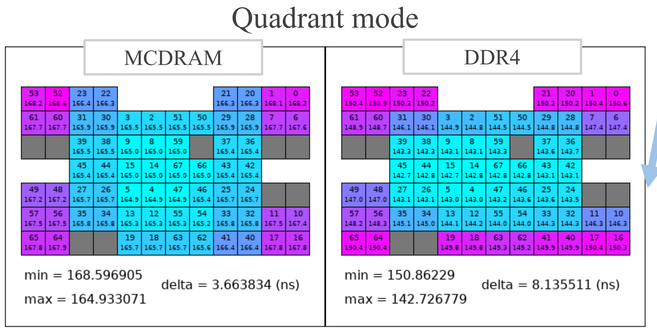
Toward our above objectives, we have measured memory access latency from every core to memory.
We presumed physical placement of cores with measured result.

In All2All and Quadrant modes, edge and border cores have high latency and cores around center have low one.

All2Alll mode

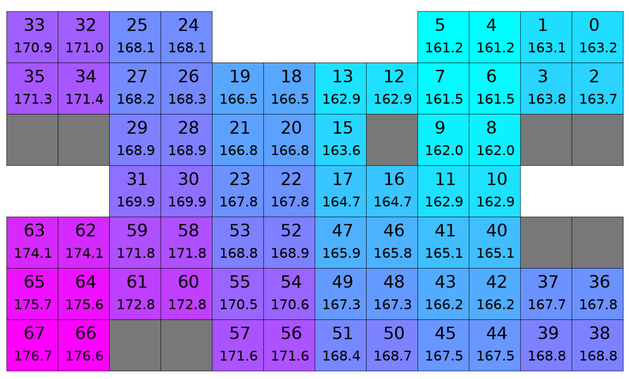


Quadrant mode

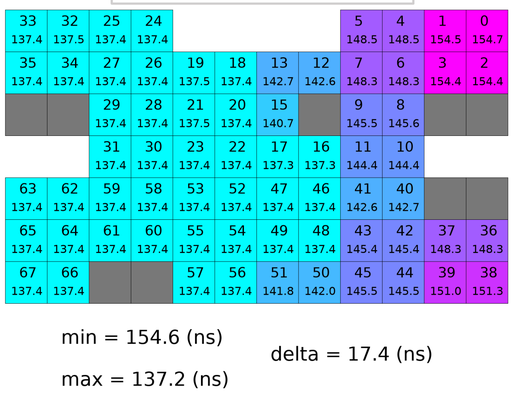


Expected location of a MCDRAM module

SNC4 mode MCDRAM



SNC4 mode DDR4



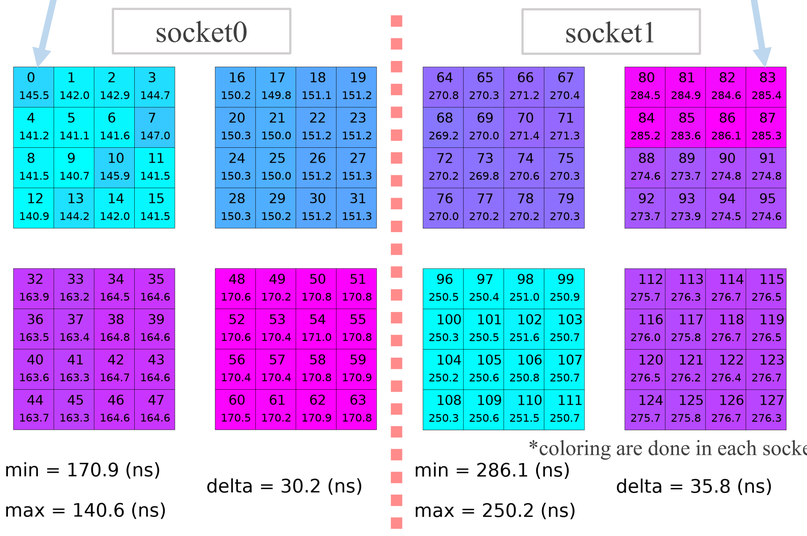
In SNC4 mode,
we can see a beautiful gradation from the nearest core to farthest one on not only MCDRAM but also DDR4.

Environment	
Processor	Intel® Xeon Phi™ Processor 7285 (Knights Mill)
Num of cores	68
Memory	MCDRAM*8, DDR4*6
Numa mode	All2All, Quadrant, SNC4
Conditions	Disabled h/w prefetch, Separated memory interleaving(SNC4)
Buffer Size	100 MiB

AMD EPYC Rome

AMD EPYC Rome can be set NPS4 mode which is like SNC4 mode on Xeon Phi.
We measured latencies on a machine which has 2 sockets for the processors. So it appeared **8 numa nodes**. Each of them has 1/8 of total cores and memory

The latency of farther core is almost twice as high as the one of nearer core!



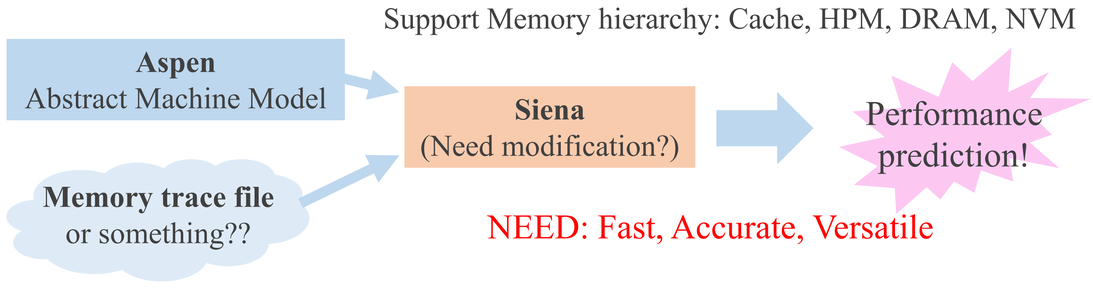
Access across sockets costs a lot (110ns).
But the **varience** of latencies is observed even in the same socket.

Environment	
Processor	AMD EPYC™ 7702 Processor (EPYC Rome)
Num of cores	64 cores * 2 socket = 128
Memory	DDR4
Numa mode	NPS4 (4 numa nodes)
Conditions	Ramdom access
Buffer Size	100 MiB

Future Directions

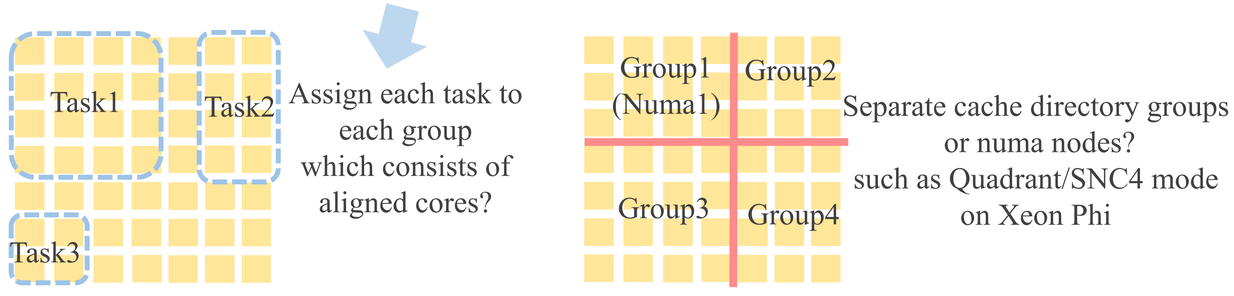
Development of **memory emulation technology** which consider core locations and their effect to performance on many core processors

Prior research: ORNL Siena[6]



Development of **latency-aware data arrangement technologies**

Prior research: Task Scheduling on Manycore Processors with Home Caches[5]



We predict these latencies will increase when the number of cores increase.
Cache coherency does not scale.
We have to consider how to treat dozens or handreds cores.

[5] Ananya Muddukrishna, Artur Podobas, Mats Brorsson, and Vladimir Vlassov. 2013. Task Scheduling on Manycore Processors with Home Caches. In Proceedings of Euro-Par'12. Springer-Verlag, Berlin, Heidelberg, 357–367. https://doi.org/10.1007/978-3-642-36949-0_39
[6] Ivy B. Peng and Jeffrey S. Vetter. 2018. Siena: Exploring the Design Space of Heterogeneous Memory Systems. In Proceedings of SC '18. IEEE Press, Piscataway, NJ, USA, Article 33, 14 pages. https://doi.org/10.1109/SC.2018.00036

This work is partly supported by a project commissioned by the New Energy and Industrial Technology Development Organization (NEDO)

Tokyo Institute of Technology