EURO 4SEE

Introduction to HPC using GPUs,

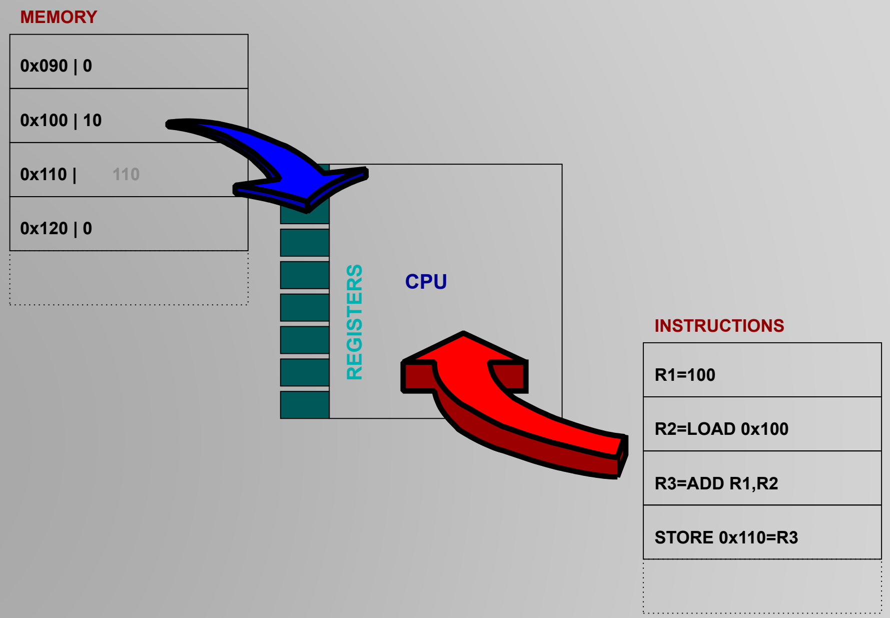
Denizhan Tutar, Istanbul Technical University

ncc@ulakbim.gov.tr

Lesson 1: Remarks on performance for CPUs

What is a computer

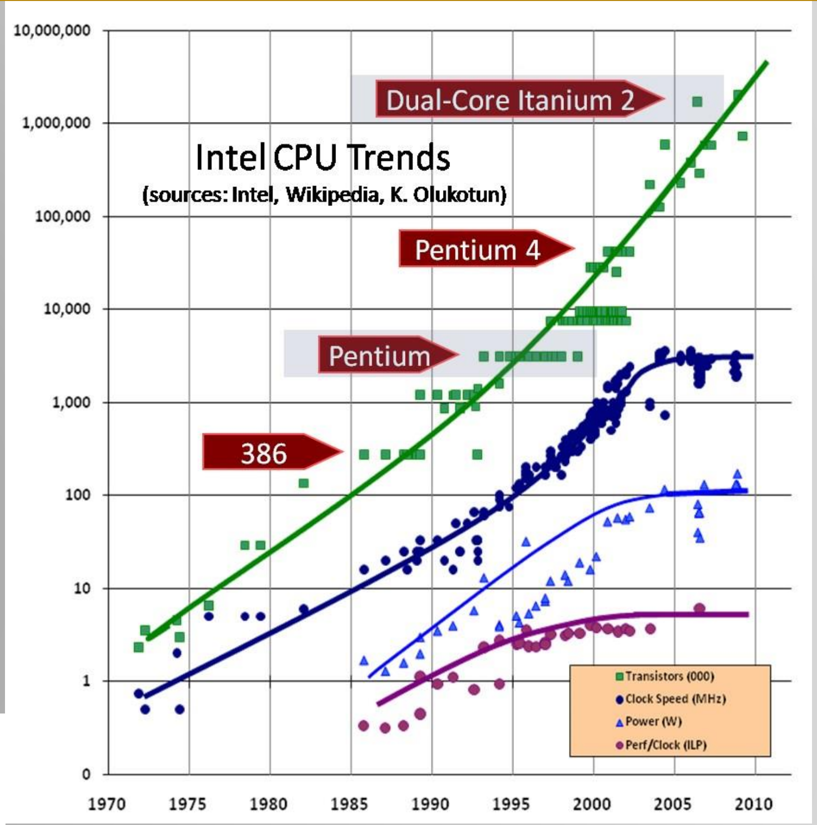
- A machine that performs operations on some data

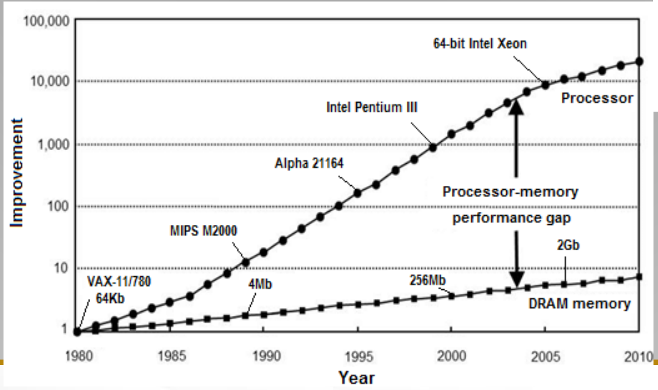


Lesson 1: Remarks on performance for CPUs

Moore's law

- Both clock speed and transistor count of CPUs are doubling roughly every 2 years
- Memory components couldn't reach same speed of development (about %50 difference in improvement every year)
- Result: computation cores are often hungry for data

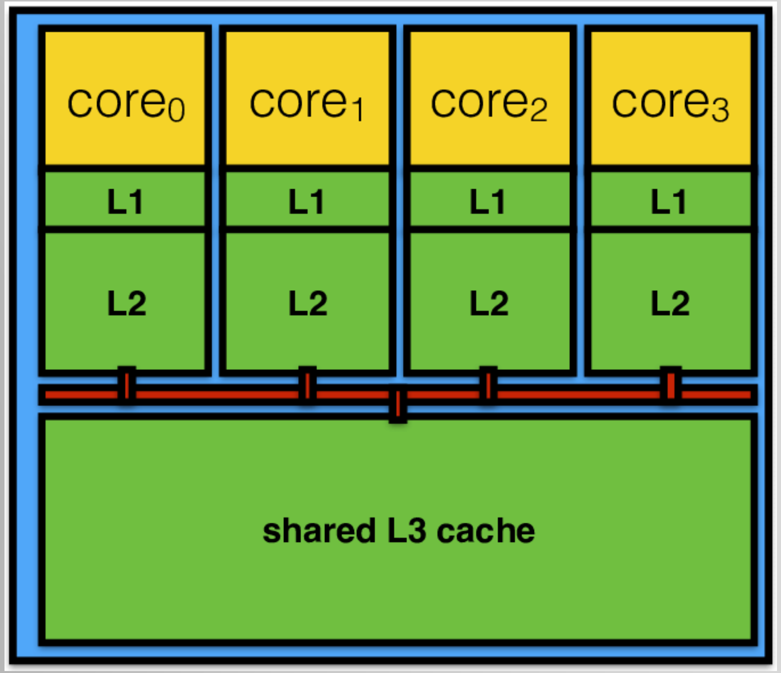




Lesson 1: Remarks on performance for CPUs

Cache

- Problem: computation cores are often hungry for data
- Solution:
  - add faster and smaller memory units (cache)
  - Try to use it as much as possible

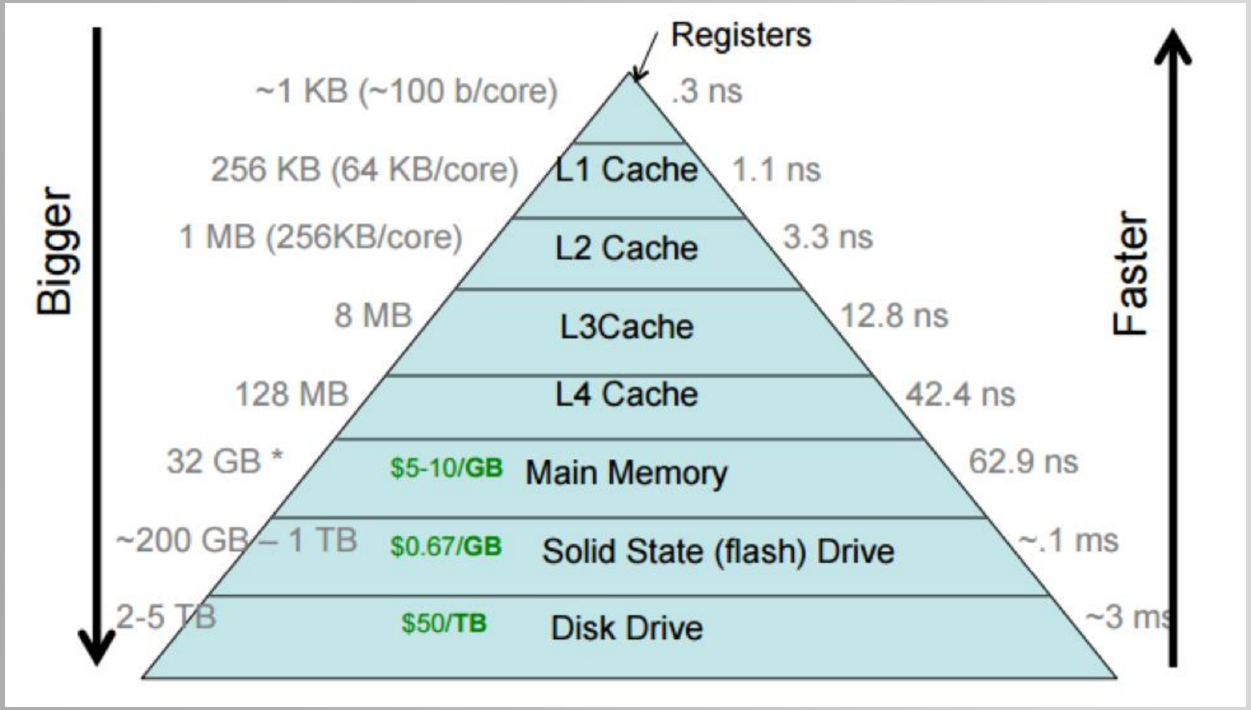


Lesson 1: Remarks on performance for CPUs

Memory Hierarchy

- Nearest memory units are much faster (low latency, high bandwidth) but also much smaller

- They also tend to be accessible by fewer cores



Lesson 1: Remarks on performance for CPUs

Cache Hit/Miss

- If the next data item to be processed is found in a faster memory unit (**cache hit**), CPU needs to wait for data shorter and waste less clock-cycles

- If it's not found (**cache miss**) the penalty is higher

- So how well our program will perform is highly dependent on **cache hit/miss ratio**

Lesson 1: Remarks on performance for CPUs

Compiler Optimizations

- The main reason C/C++ and Fortran still dominate HPC field is compilers
- Compilers rewrite the code so it would have higher cache hit/miss ratio (optimization level flag: -O0, -O1, -O2, -O3)
- Programmer cannot dictate how to use cache, the OS and hardware are responsible for it
- Compilers and hardware relies on **heuristics** to try keep needed data closer

Lesson 1: Remarks on performance for CPUs

Locality

- Locality is probably the most important heuristic used by compilers

- Temporal Locality: if a variable is recently used, it is likely to be used again
- Spatial Locality: if a variable is close (in the memory) to requested item, it's likely to be requested next

```
for(int i=0; i<N; i++) sum += a[i];
```

Lesson 1: Remarks on performance for CPUs

Locality

- Computer fetches a block of elements of an array into cache
- Access pattern is thus very important: better performance if aligned with heuristics
- Ex: Loop ordering in nested loops, summing an array by reaching elements in order or out of order (sum+=a[i] vs sum+=a[i*37%N])

Lesson 1: Remarks on performance for CPUs

Multitasking

- Computer holds some other computation ready if it can, with its instructions and data on the cache, so that it can work on this instead of waiting idle if a cache miss happens

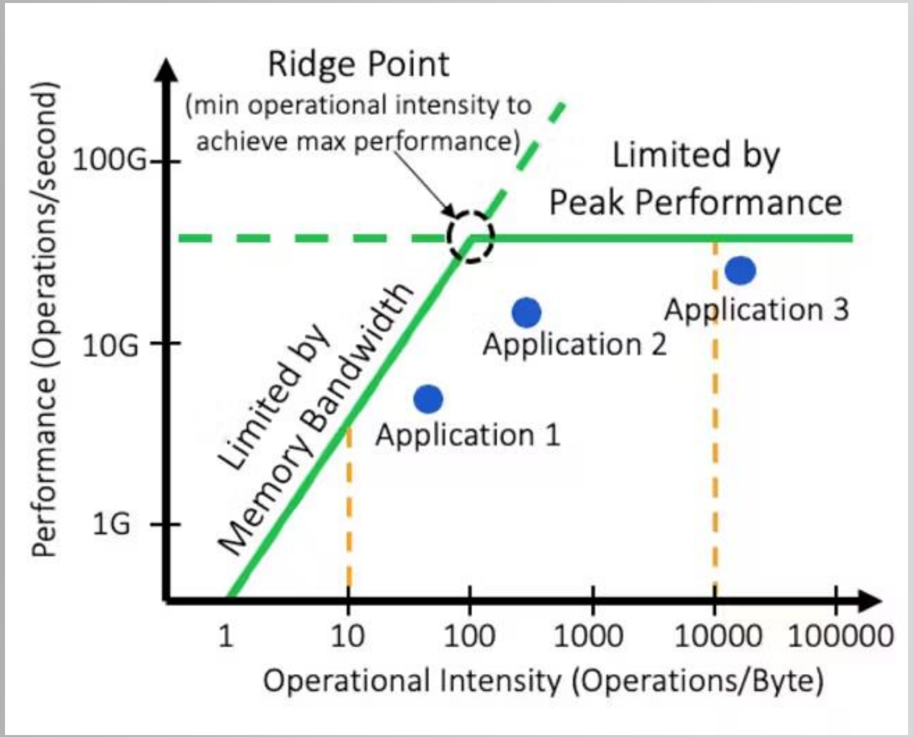
- **Occupancy**: shows how much of the clock-cycles are used for a computation, one of the most important **metric** about performance

- Other metrics are **data bandwidth** and **instruction throughput** (how much data/instruction we deal with in a second)
- Higher score on metrics doesn't guarantee but often correlate with higher performance

Lesson 1: Remarks on performance for CPUs

Know your application: compute bound vs memory bound

- Some applications perform a lot of operations per data item, and their performance becomes limited by hardware's peak performance. (compute heavy)
- In other applications cores would need to wait idle for data (memory bound)
- Bottlenecks and how well an optimization step would benefit you depends on where your application is in this chart



Roofline model

Next lecture

Remarks on parallelism (using OpenMP)

References

- https://www.bottomupcs.com/ch03.html
- https://ms.codes/blogs/computer-hardware/moore-s-law-cpu-speed?srsltid=AfmBOooZ8EZBE7qHDWok1gROFz6dozR3rBQB2Egw4AaZEuRX55whNyxV
- Efnusheva, Danijela & Cholakoska, Ana & Tentov, Aristotel. (2017). A Survey of Different Approaches for Overcoming the Processor - Memory Bottleneck. International Journal of Information Technology and Computer Science. 9. 151. 10.5121/ijcsit.2017.9214.
- https://shadesofcode.hashnode.dev/where-the-hell-is-my-data
- https://www.electronicdesign.com/technologies/embedded/article/21157756/rambus-memory-for-ai-two-edges-and-a-roofline

Copyright

EURO 4SEE

Thanks!

Co-funded by
the European Union

EuroHPC
Joint Undertaking

This project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No 101191697. The JU receives support from the Digital Europe Programme and Germany, Türkiye, Republic of North Macedonia, Montenegro, Serbia, Bosnia and Herzegovina.