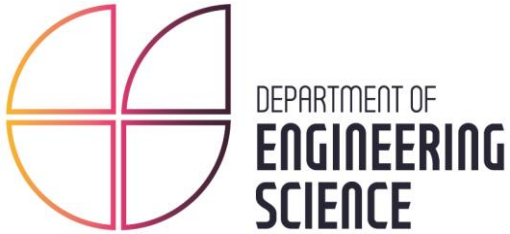


An Introduction to HPC and Scientific Computing

Lecture one: Introduction to Computer Architectures

Wes Armour

Oxford e-Research Centre,
Department of Engineering Science

Learning outcomes

In this lecture you will learn about:

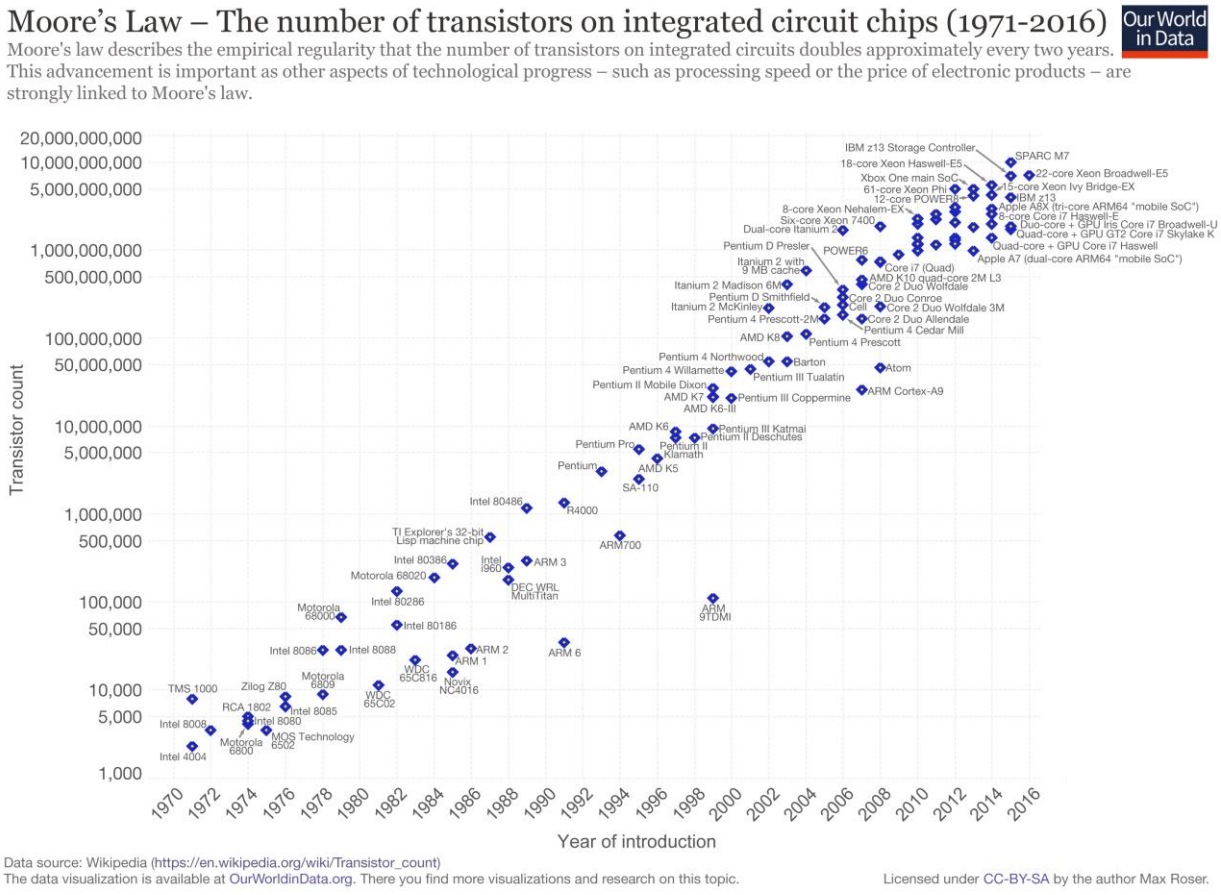
- Computer components and how they fit together.
- The basic concepts of memory hierarchy and how to achieve good bandwidth throughput.
- The design of a CPU and how you achieve good CPU performance.
- The design of a GPU.

Current computing landscape

The adoption of computing technology over the last decade has been pervasive. Many day-to-day objects employ light weight computers to function.

The ability of computers to perform complicated tasks has grown dramatically giving rise to new digital services and useful AI. This is largely due to Moore's law.

Moore's law (roughly) states that the number of transistors in an integrated circuit will double every two years. Moore made this prediction in 1965!



Data source: Wikipedia (https://en.wikipedia.org/wiki/Transistor_count)
The data visualization is available at OurWorldinData.org. There you find more visualizations and research on this topic.

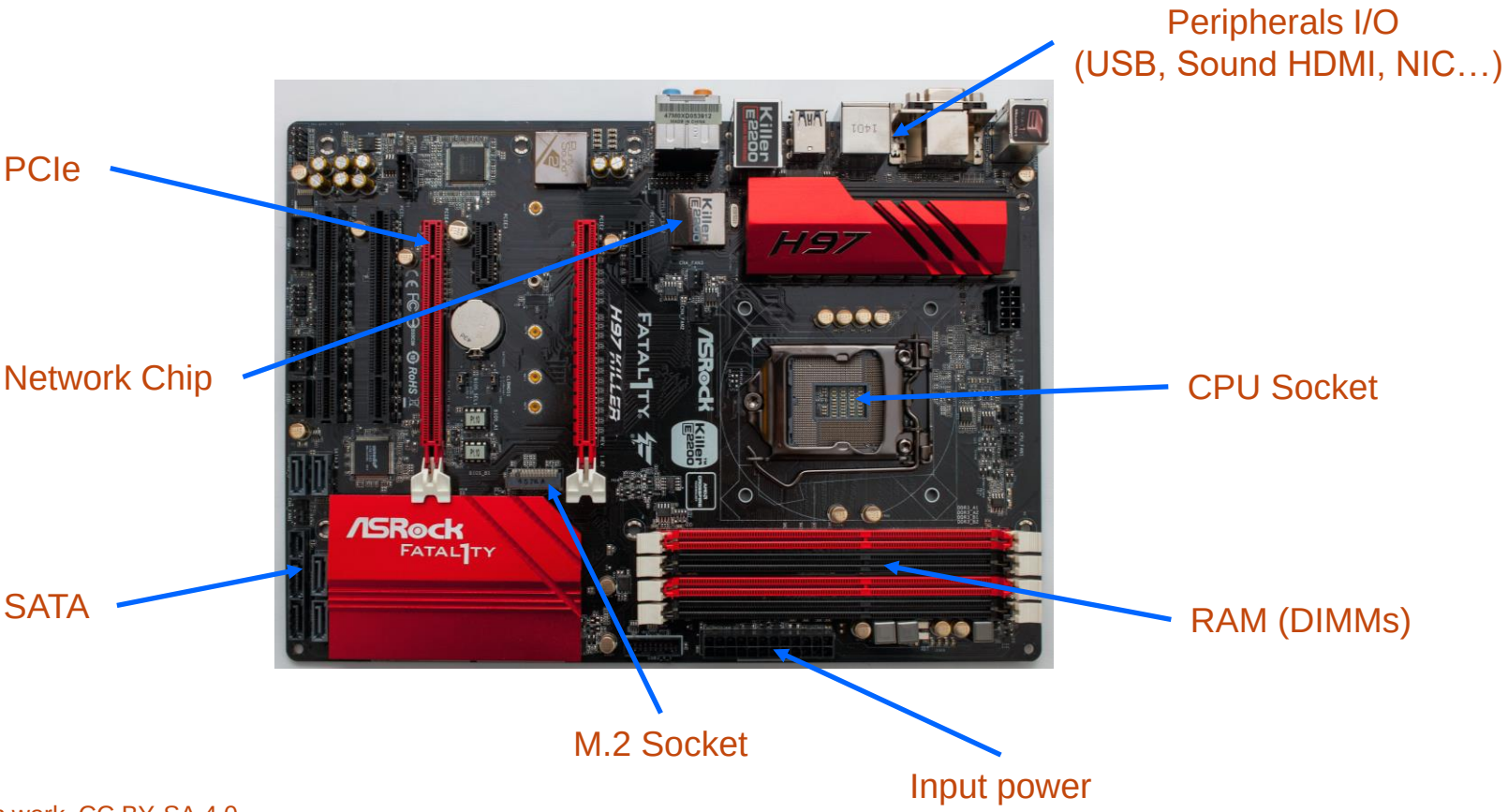
Quick system review

Modern computers are typically comprised of many different components.

- Motherboard – connects everything together.
- Central Processing Unit (CPU) – Command and control / computations.
- Random Access Memory (RAM) – memory that stores the current data we are working with.
- Graphics Processing Unit (GPU) – Hardware to display images, but over the last decade used as a computational accelerator.
- Hard Disk Drive (HDD) – a large storage area for our data files.
- Network Interface Controller (NIC) – connects your computer to the outside world.
- Other components include PSU, Chassis, DVD-ROM….

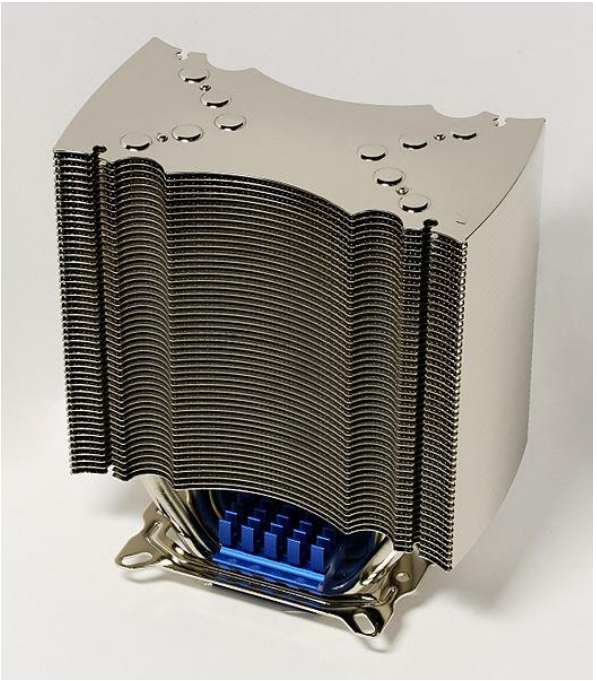
Quick system review - Motherboard

Motherboard – connects everything together



Quick system review - CPU

Central Processing Unit (CPU) and heatsink / fan, because CPUs get very hot!

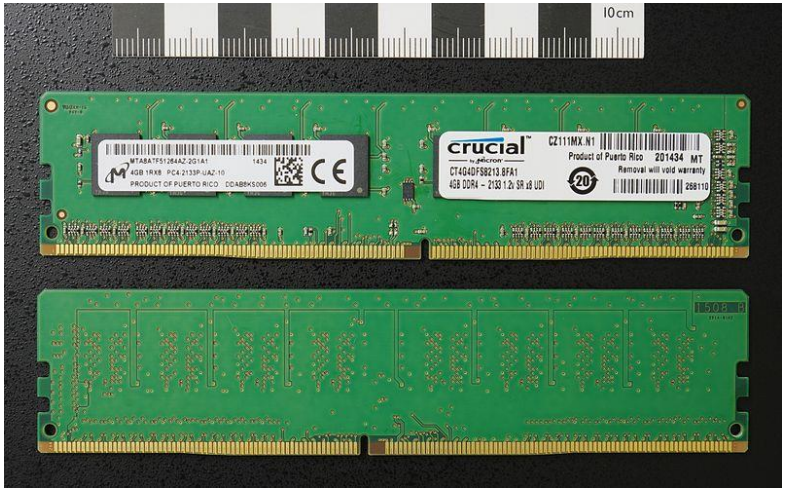


By D-Kuru [CC BY-SA 3.0 at (https://creativecommons.org/licenses/by-sa/3.0/at/deed.en)], from Wikimedia Commons

By Konstantin Lanzet - CPU collectionCamera: Canon EOS 400D, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=7152144

Quick system review – Memory/Storage

Random Access Memory (RAM) – memory that stores the current data we are working with.



By smial [FAL or GFDL 1.2 (http://www.gnu.org/licenses/old-licenses/fdl-1.2.html)], from Wikimedia Commons

By Darkone [CC BY-SA 3.0 (https://creativecommons.org/licenses/by-sa/3.0)], from Wikimedia Commons

Hard Disk Drive (HDD) – a large storage area for our data files.

Quick system review - GPU

Graphics Processing Unit (GPU) – Hardware to display images, but over the last decade used as a computational accelerator.

https://www.flickr.com/photos/gbpublic/13409246645

A closer look at a CPU – In a server

Modern datacentre servers typically have two (or four) CPUs per server (node).

The CPUs are connected by a fast interconnect such as Intel's Ultra Path Interconnect (UPI).

The CPUs have independent memory controllers that connect them to RAM.

They also have PCIe lanes to connect them to other components such as NVMe drives, GPUs or NICs.

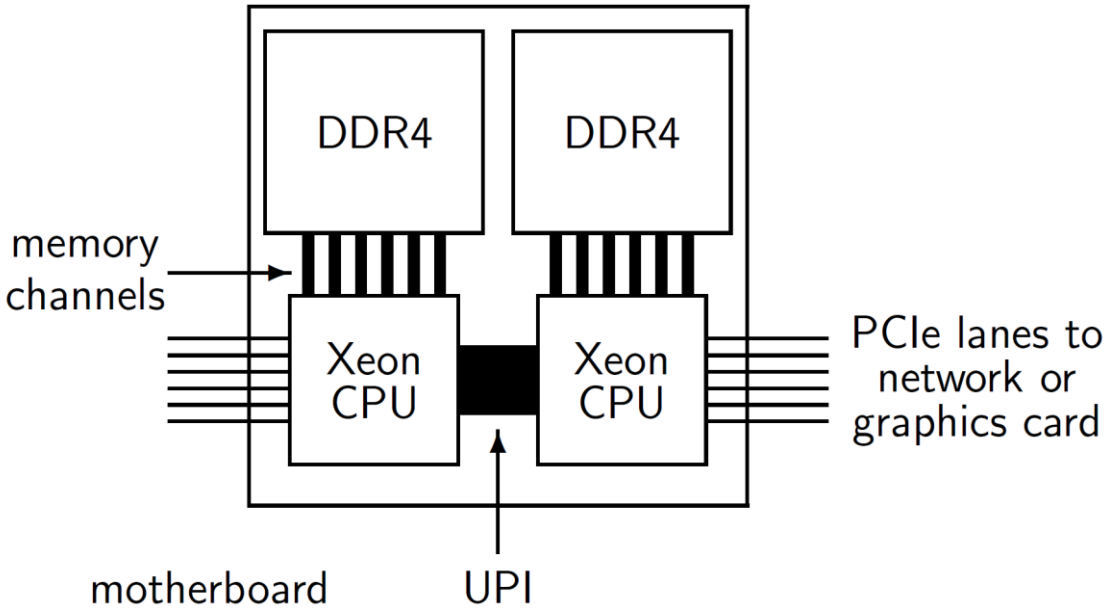


Image courtesy of Mike Giles

A closer look at a CPU – Single processor

A single CPU (processor) is comprised of many cores (anything from one or two, all the way up to twenty or more).

All cores are connected by the L3 (or Last Level Cache - LLC) which is shared amongst them.

The size of the L3 cache varies from processor to processor, but typically on Intel's latest architecture, Skylake, this will be between 1 and 1.5 MB.

To extract the maximum performance from modern CPUs work must be shared amongst the cores. This is where OpenMP is helpful.

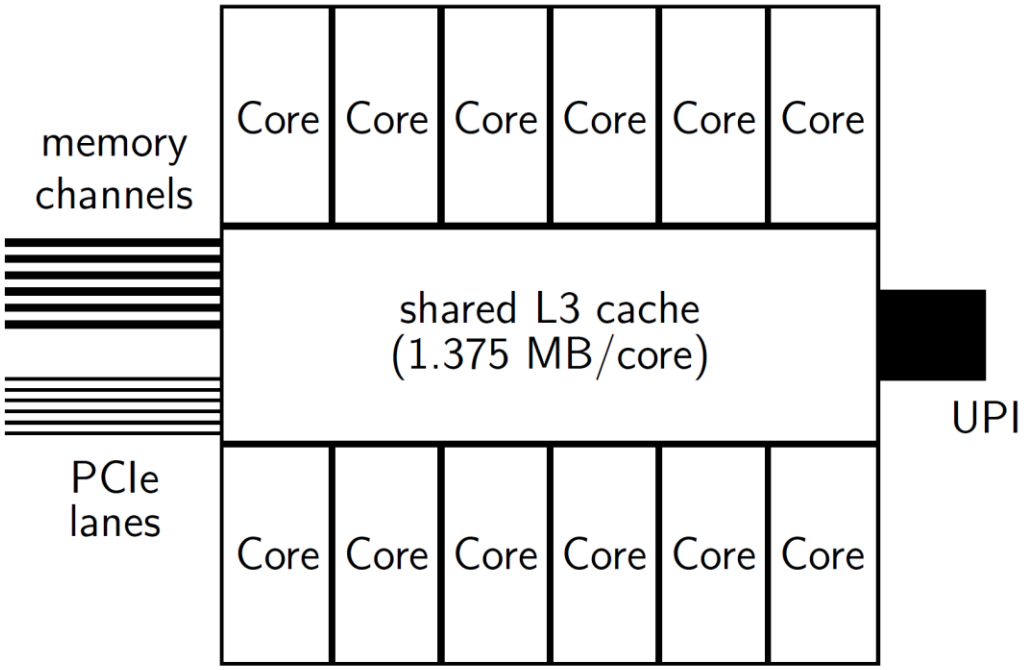


Image courtesy of Mike Giles

A closer look at a CPU – Single core

A single processor core is comprised of a level 2 (L2) cache which is small (1 MB on Skylake), but has fast read/write. It also has a much smaller, but even faster L1 cache. *Registers* are the fastest area of memory.

A core has lots of functional units which are used to load and store data, perform computations and control the order in which they are performed.

Computations come in the form of *Instructions*.

Modern processors are *Superscalar*, meaning they can execute more than one instruction at a time.

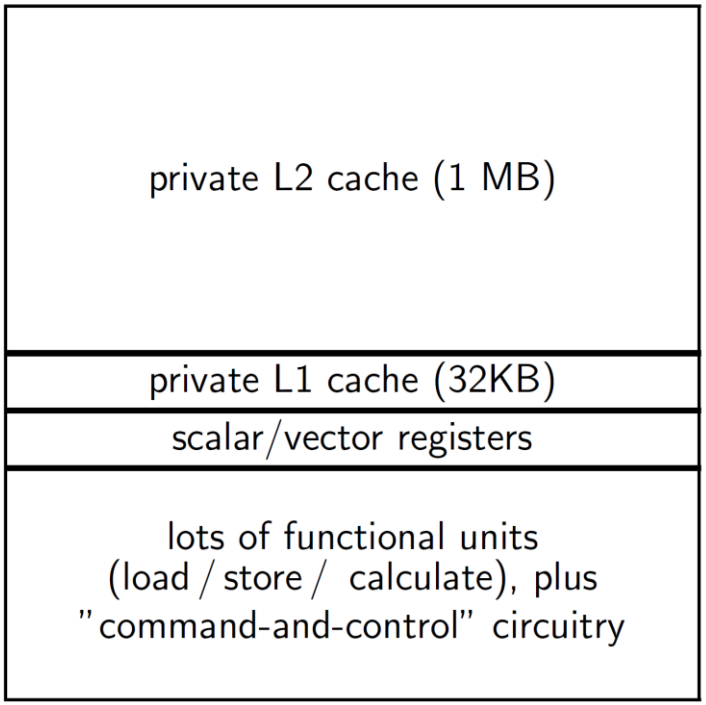


Image courtesy of Mike Giles

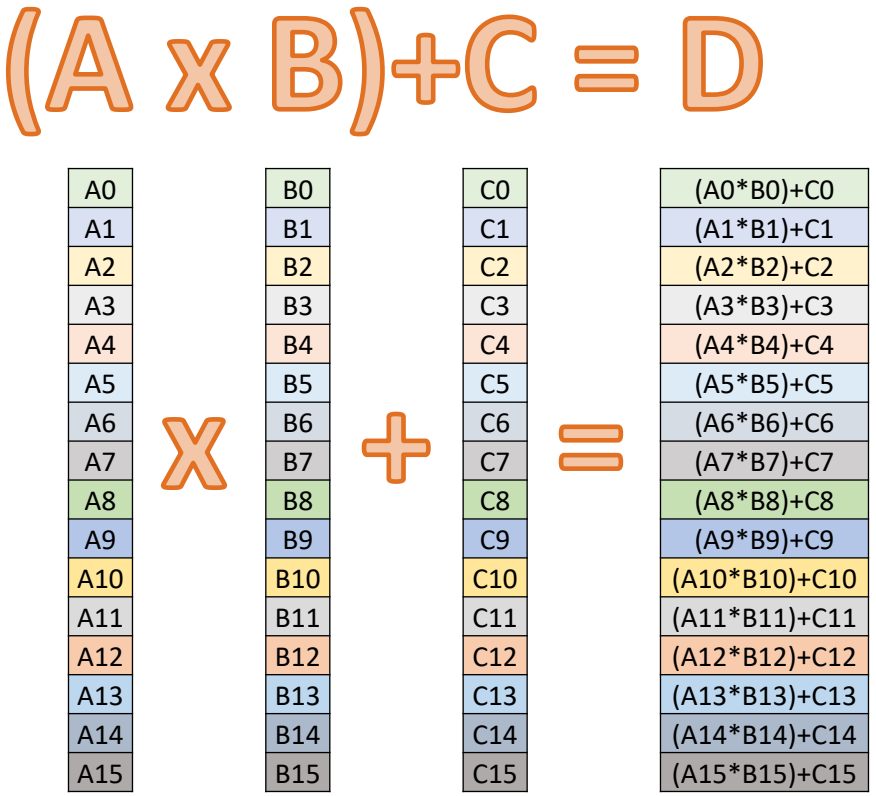
A closer look at a CPU – SIMD units

A core also has one (or two) *Advanced Vector eXtensions* (AVX) units.

These units are capable of executing a Single Instruction on Multiple Data (*SIMD*) elements at the same time (*in parallel*). A program that exploits these vector features is said to be *"vectorised"*

Skylake has AVX-512 vector units, meaning that the vectors are 512 bits long, so can store 16 single precision numbers in each. They can perform at most two instructions at once by issuing a *fused multiply add* (FMA) instruction (see right).

It's these units, along with the high core count that allow modern CPUs to perform very large and complex computations very quickly.



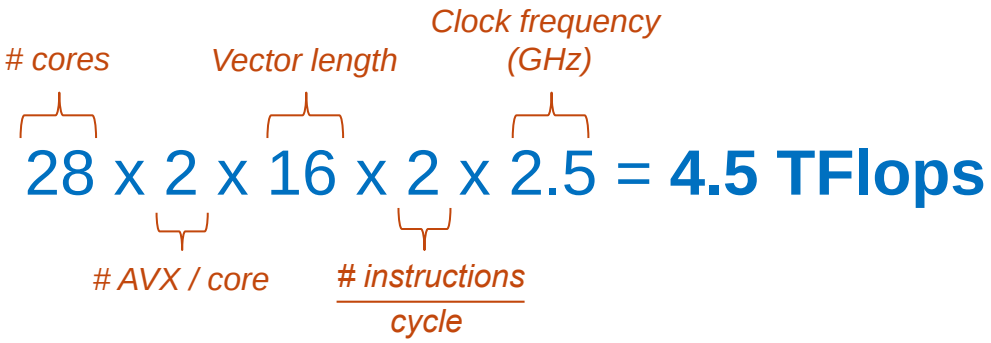
A closer look at a CPU – Throughput

One of the largest Intel CPUs at the moment is the Xeon Platinum 8180. This has:

- 28 Cores
- Runs at a frequency of 2.5 GHz
- Has 2x AVX512 units per core (so can hold 16 SP values each)
- Can execute 2x operations per execution
- Costs ~ £7,000.

The peak computational performance is measured by the number of FLoating Point Operations per second (*Flops*) that can be performed.

So it's peak single precision performance (more on this in our next lecture) is given by the calculation on the right.



A closer look at a CPU – Throughput

Processing cores become ever faster, but the rate that a computer can move data from memory to cores hasn't developed as quickly. This means that on a modern processor data movement is far more costly than computations.

So for a program to achieve *peak* performance on a modern processor, great care has to be taken to minimise data movement.

Memory Architecture

Modern computers can have several different types of memory all of which have different *bandwidths* (bandwidth is a measure of how quickly you can get a "unit", e.g. a GB of data from point A to point B).

They are arranged in a hierarchy. From lots of slower *RAM* down to very few, but very fast *registers* which are located closest to the CPU processing core.

The diagram on the right shows the different bandwidths for a modern Intel Skylake CPU. Note that all of the *caches* (smaller areas of fast memory) are located on the CPU.

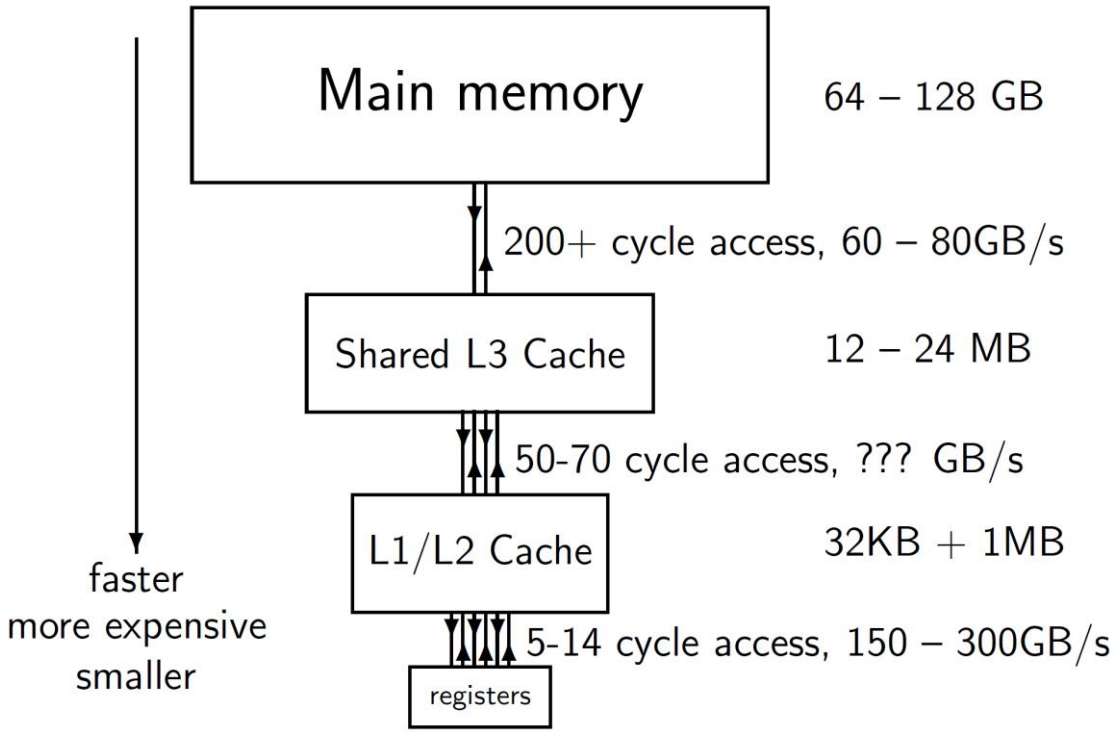


Image courtesy of Mike Giles

What's the point of a Cache?

As noted on our previous slide the area of memory where we can store the most data is the RAM, but this has lower bandwidth.

So if we need to move our data from RAM to processing cores many times our code would run slowly. This is where Cache helps. Caches exploit *data locality*.

- *temporal locality:* a data item just accessed is likely to be used again in the near future, so keep it in the cache.

- *spatial locality:* neighbouring data is also likely to be used soon, so load them into the cache at the same time using a 'wide' bus (like a multi-lane motorway).

It is these wide lanes that feed the AVX units with data.

Accelerator Computing

The use of many-core accelerators such as Intel's Xeon Phi, NVIDIA GPUs and now FPGAs have revolutionised modern supercomputing.

These accelerators are add-in PCIe cards that are used as extra computational resources for your program.

They typically have less complexity than CPUs and so use more of there transistors for computations.

Some of the motivations for using many-core technologies are the significant reductions in floor space, maintenance and power consumption that can be gained.

Top500	– 3 out of top 5
Green500	– 9 out of top 10

Examples of many-core accelerated supercomputers

Tianhe-2	16,000 nodes 2x CPU, 3x Phi per node 33.8 PFLOP/s
Titan	18,688 nodes 1x CPU, 1x K20x GPU 17.6 PFLOP/s

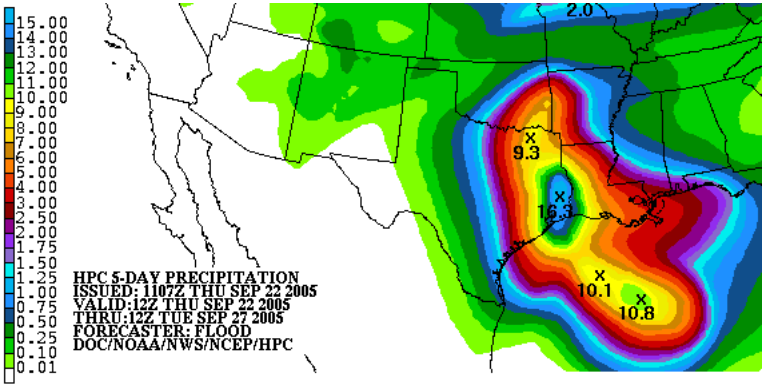
UNIVERSITY OF OXFORD

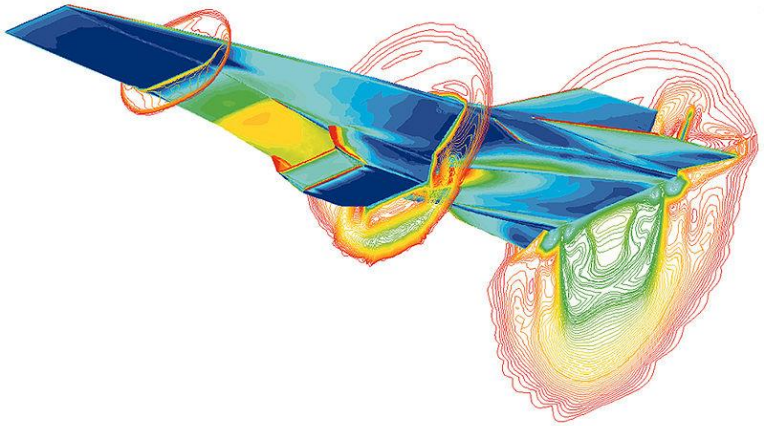
Oxford e-Research Centre

Accelerator Computing

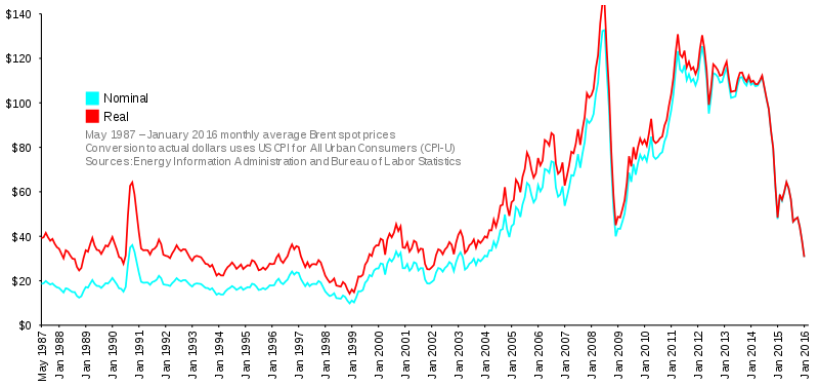
There are many uses for HPC and Scientific Computing. Some of the familiar ones are:

- Weather forecasting, Financial modelling, CFD, Drug discovery, materials modelling, AI, planetary formation, universe evolution…..





By NASA [Public domain], via Wikimedia Commons



By TomTheHand [CC BY-SA 3.0 (https://creativecommons.org/licenses/by-sa/3.0) or GFDL (http://www.gnu.org/copyleft/fdl.html)], from Wikimedia Commons

UNIVERSITY OF OXFORD

Oxford e-Research Centre

Graphics Processing Units

Graphics Processing Units (*GPUs*) have been available as add-in cards since the 1990's. There original purpose was to accelerate graphics processing, allowing computer games to give the player a better experience.

The use of GPUs widened in the early 2000's when NVIDIA released a GPU that had programmable *shaders*. This allowed programmers to use GPUs to execute short pieces of parallel code.

In 2007 NVIDIA launched *CUDA* (Compute Unified Device Architecture), it is a dedicated programming environment for NVIDIA GPUs. It comes with a rich ecosystem of tools and libraries that help programmers produce code for GPUs.

NVIDIA CUDA

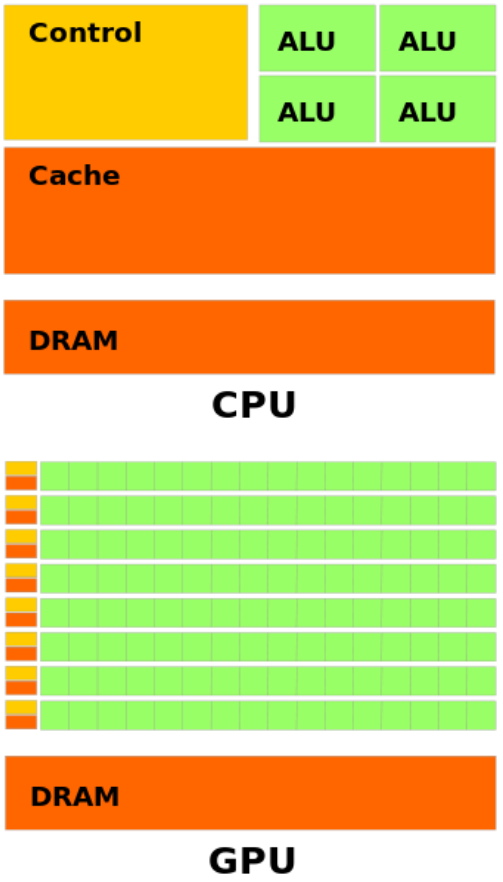
A closer look at a GPU - Design

GPUs have a different design to CPUs. Their complexity is greatly reduced (*no extensive data caching or command-and-control*). They dedicate most of there transistors to processing (right). This makes GPUs ideal for compute-intensive, highly parallel computation.

More specifically, GPUs excel at data-parallel computations. Just like the AVX (SIMD) units in CPUs.

GPUs also have access to a reasonable amount of *High Bandwidth Memory (HBM).* This can be as much as 32GB on flagship models. This memory has greater bandwidth than server RAM, but there is less of it.

However GPUs attach to a server via the PCIe bus, so this ultimately limits communication speed between a CPU and GPU.



A closer look at a GPU – Throughput

One of the largest NVIDIA GPUs at the moment is the V100.

This has:

- 5120 Cores
- Runs at a frequency of 1.37 GHz
- Can execute 2x operations per execution
- Costs ~ £8,800.

Comparing this to our Xeon CPU we see that the peak computational performance is about 3x higher, but costs just 25% more.

It's important to note though that the CPU is far more flexible that the GPU and can outperform the GPU for certain computational tasks.

 # cores — *Clock frequency (GHz)*

$$5120 \times \underbrace{2}_{\frac{\#\ instructions}{cycle}} \times 1.37 = 14\ \textbf{TFlops}$$

What have we learnt?

In this lecture you have learnt about modern computer architectures.

You have learnt about current CPUs and some of their design features.

We have covered what is needed to achieve good throughput on a CPU.

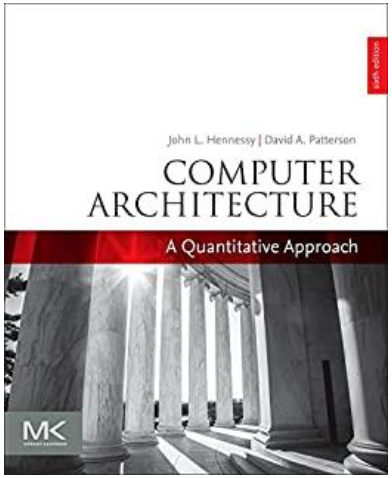
You have learnt about accelerator computing, GPUs and the differences between CPUs and GPUs.

Further reading

Computer Architecture: A Quantitative Approach

John L. Hennessy, David A. Patterson

The Morgan Kaufmann Series in Computer Architecture and Design



Now in it's sixth edition, this is considered the reference text for those learning about computer design.

In the next lecture…

In the next lecture you will learn about the C programming language. You will be introduced to the concept of High level computer languages, the basic components of a C computer program and how data is stored on a computer. By the end of the lecture you will be able to write your own C program.