Parallel Computing with GPUs

Introduction Part 1 – Course Context

The University Of Sheffield.

Dr Paul Richmond

http://paulrichmond.shef.ac.uk/teaching/COM4521/

This Lecture (learning objectives)

- Introduce the course context
  - Identify the significance of GPU performance
  - Analyse the emergence of multi and many core architectures
  - Present accelerators as a co-processor

Context of course

8.74 TeraFLOPS

10.0 TFlops
9.0 TFlops
8.0 TFlops
7.0 TFlops
6.0 TFlops
5.0 TFlops
4.0 TFlops
3.0 TFlops
2.0 TFlops
1.0 TFlops
0.0 TFlops

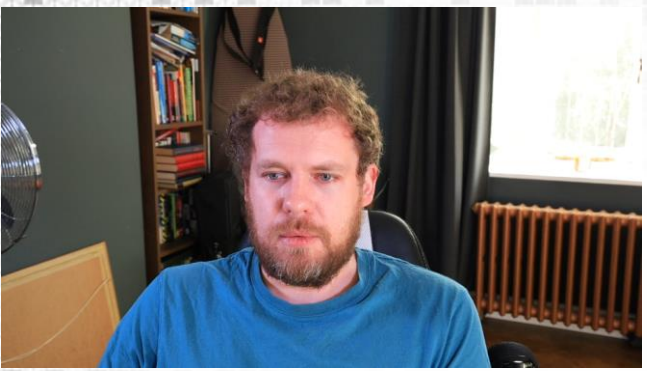
~40 GigaFLOPS

1 CPU Core

GPU (4992 cores)

6 hours *CPU* time
vs.
1 minute *GPU* time

Scale of Performance



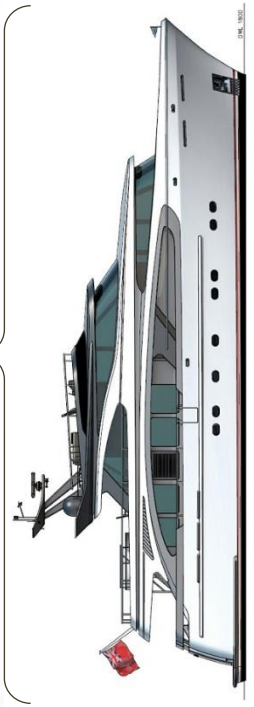
Serial Computing

1.8m

1 core

Parallel Computing

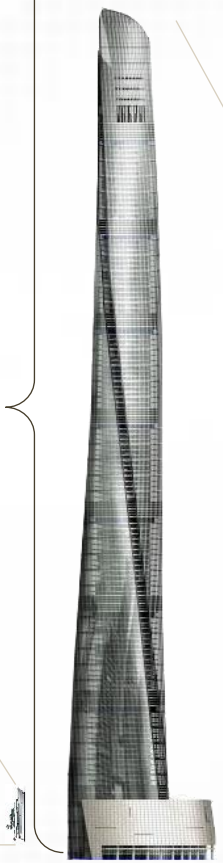
28m



16 cores

Accelerated Computing

650m



4992 GPU cores

Accelerated Workstation

2.6km

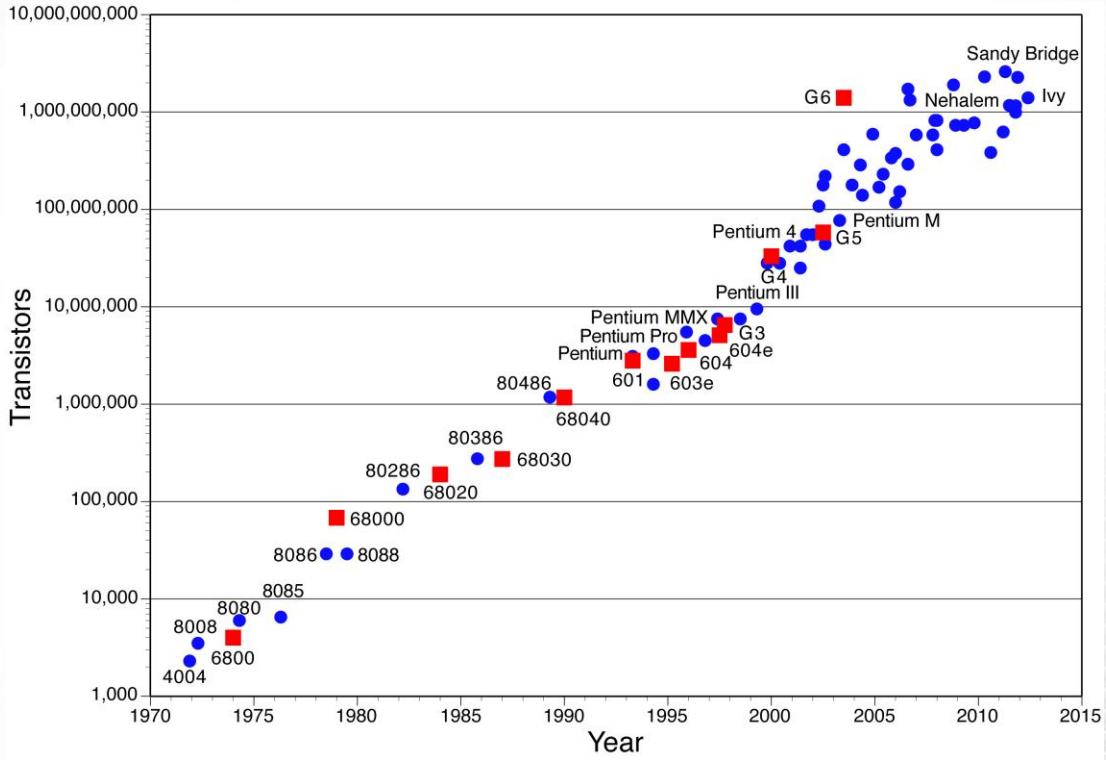
4x 4992 GPU cores +16 CPU cores

The University Of Sheffield.

Scale of Performance: Titan Supercomputer

Transistors != performance

- Moore's Law: A doubling of transistors every couple of years
  - Not a law actually an observation
  - Doesn't actually say anything about performance
- Future of Moore's Law
  - Moore's law is dead!
  - A bright future for Moore's Law



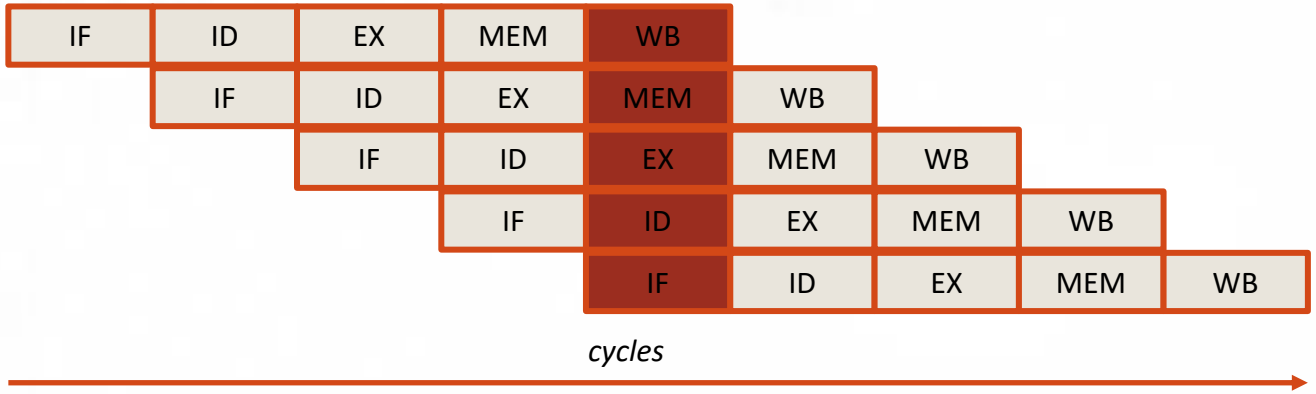
Dennard Scaling

"As transistors get smaller their power density stays constant"

$$\text{Power} = \text{Frequency} \times \text{Voltage}^2$$

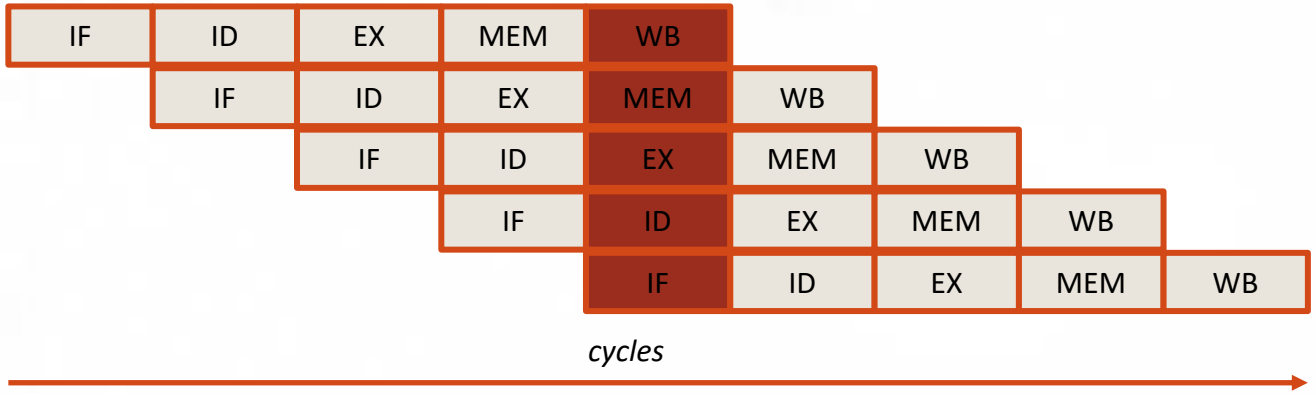
- Performance improvements for CPUs traditionally realised by increasing frequency
- Decrease voltage to maintain a steady power
  - Only works so far
- Increase Power
  - Disastrous implications for cooling

Instruction Level Parallelism



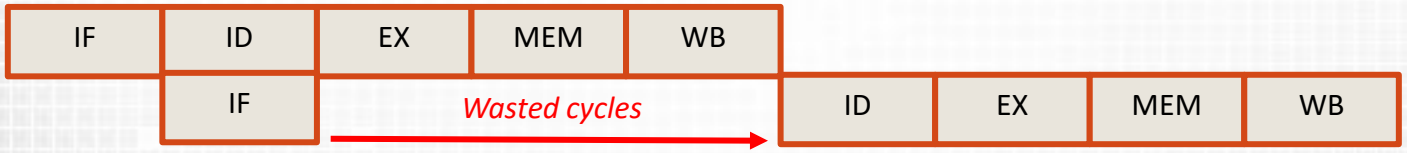
- Transistors used to build more complex architectures
- Use pipelining to overlap instruction execution

Instruction Level Parallelism

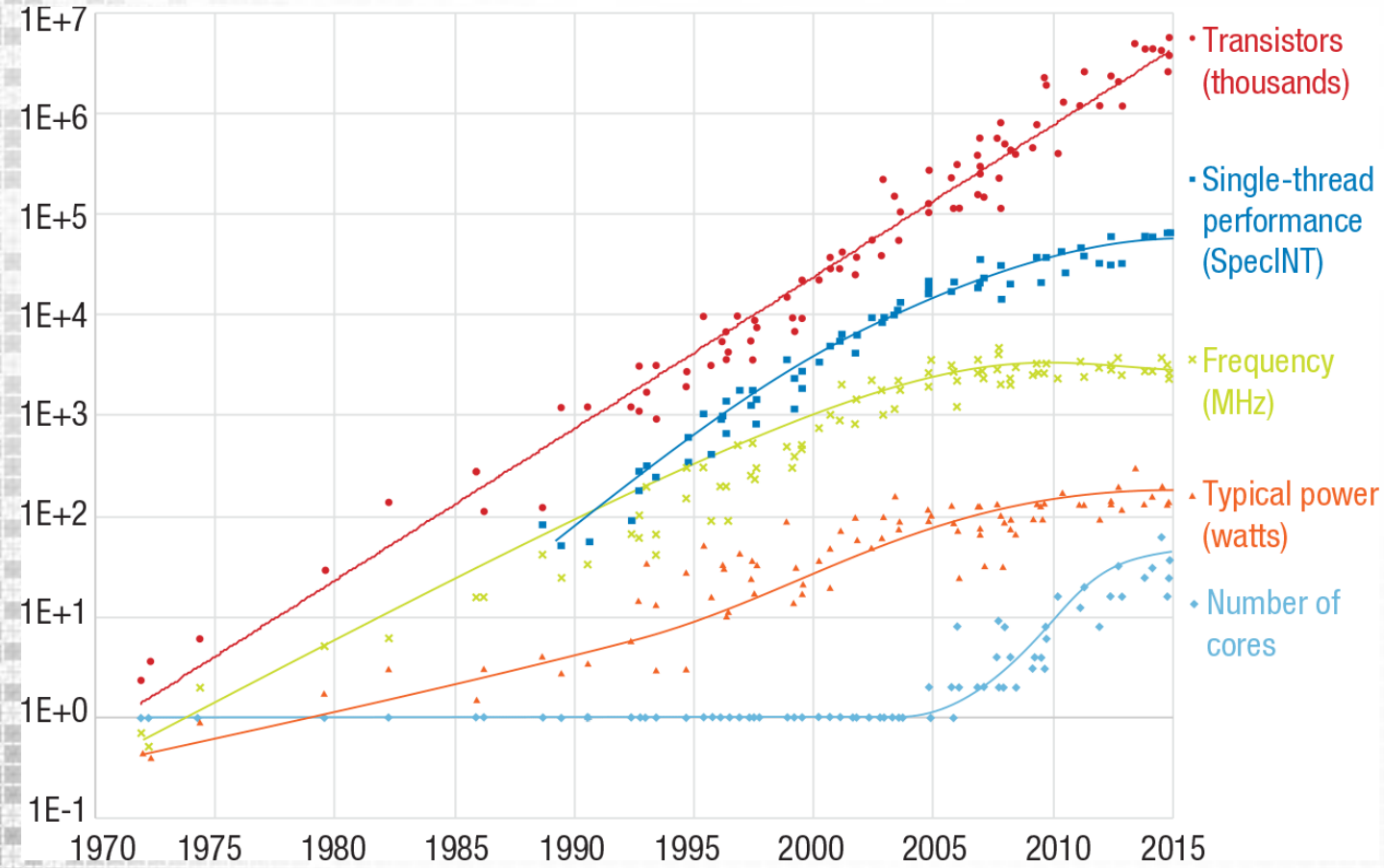


- Transistors used to build more complex architectures
- Use pipelining to overlap instruction execution

```
add 1 to R1
copy R1 to R2
```

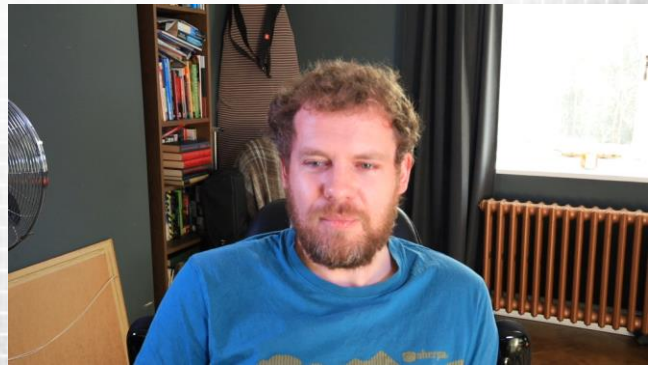


Golden Era of Performance



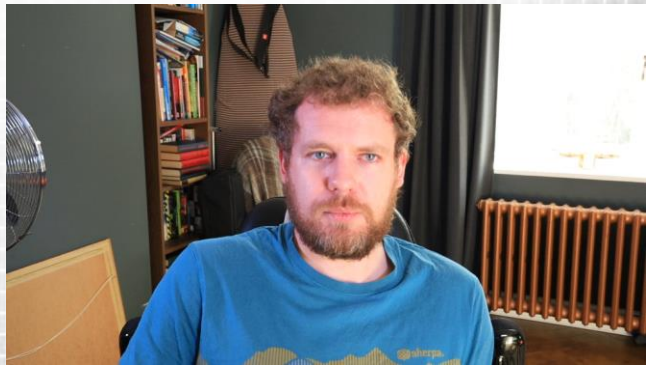
Adapting to Thrive in a New Economy of Memory Abundance, K Bresniker et al.

- 90s saw great improvements to single CPU performance
  - 1980s to 2002: 100% performance increase every 2 years
  - 2002 to now: ~40% every 2 years

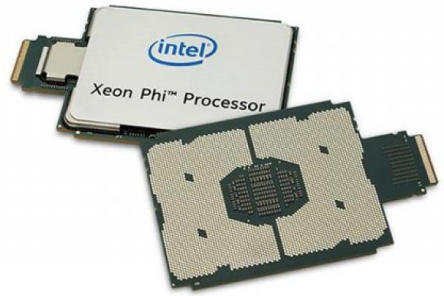


Why More Cores?

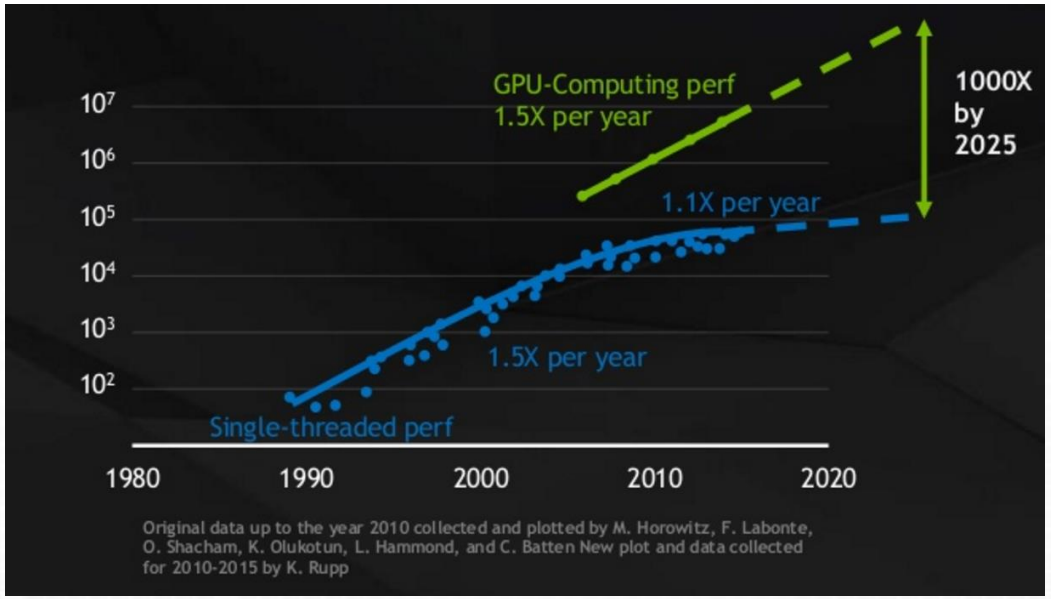
- Use extra transistors for multi/many core parallelism
  - More operations per clock cycle
  - Power can be kept low
  - Processor designs can be simple – shorter pipelines (RISC)



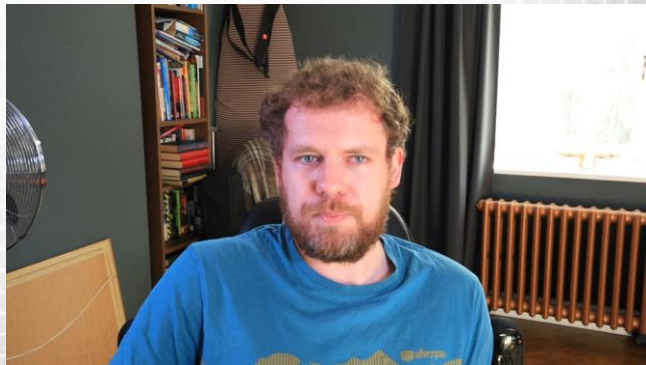
GPUs and Many Core Designs



- Take the idea of multiple cores to the extreme (many cores)
- Dedicate more die space to compute
  - At the expense of branch prediction, out of order execution, etc.
- Simple, Lower Power and Highly Parallel
  - Very effective for HPC applications

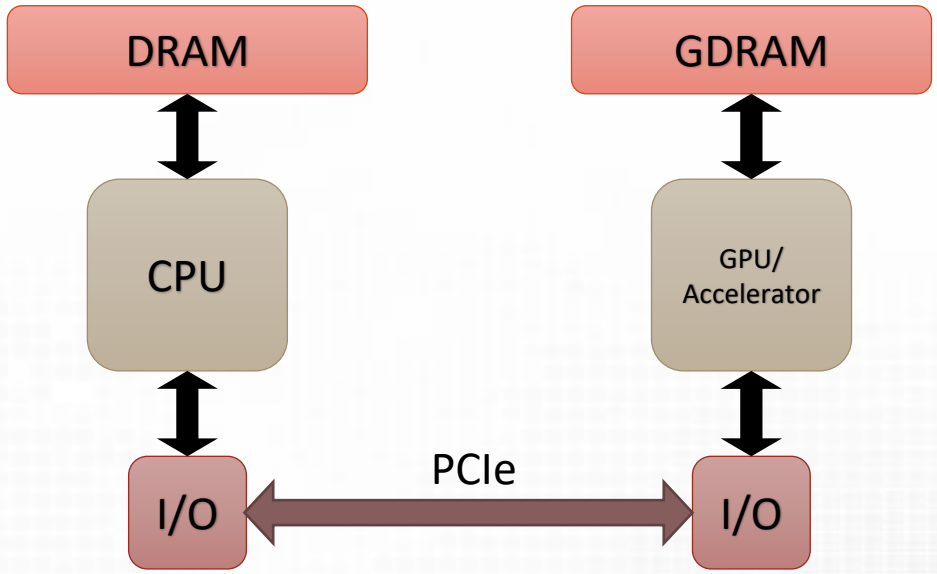


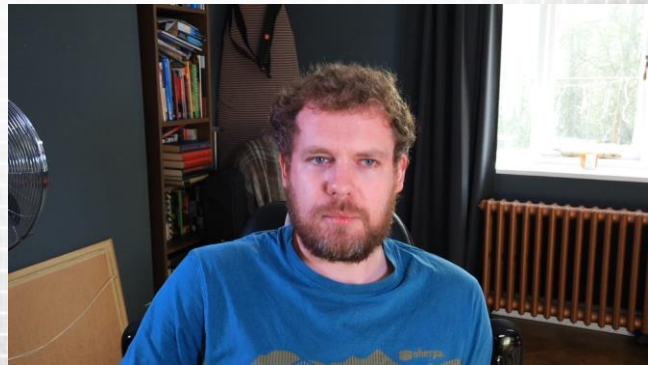
From GTC 2017 Keynote Talk, NVIDIA CEO Jensen Huang



Accelerators

- Problem: Still require OS, IO and scheduling
- Solution: “Hybrid System”,
  - CPU provides management and
  - “Accelerators” (or co-processors) such as GPUs provide compute power

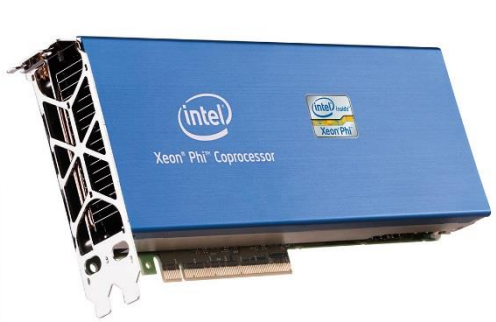


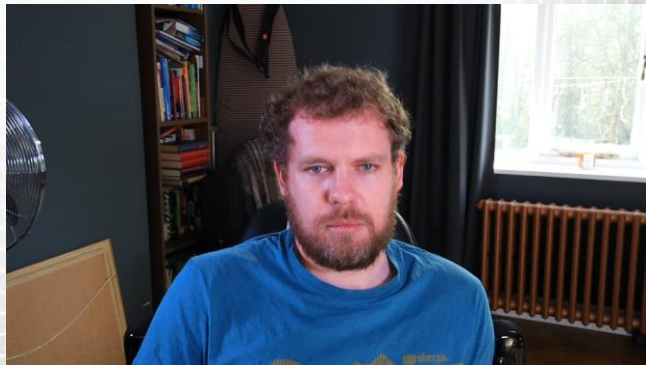


Types of Accelerator

- GPUs
  - Emerged from 3D graphics but now specialised for HPC
  - Readily available in workstations

- Xeon Phis
  - Many Integrated Cores (MIC) architecture
  - Based on Pentium 4 design (x86) with wide vector units
  - Closer to traditional multicore
  - Simpler programming and compilation





Summary

- Introduce the course context
  - Identify the significance of GPU performance
  - Analyse the emergence of multi and many core architectures
  - Present accelerators as a co-processor

- Next Lecture: Supercomputers and Software