HPC architectures: Processors

Simon McIntosh-Smith

simonm@cs.bris.ac.uk

University of BRISTOL

Introduction

We're going to take a look at **cutting edge technology** in processors

A recent major shift in microprocessors resulted in ***commodity hardware*** dominating HPC

Other hardware trends mean that HPC technology is now pervasive

The next major technology trend is always just around the corner, so you need to keep your eyes and ears open…

The “commoditisation” of HPC

- The HPC industry used to design its own special-purpose processors and interconnects
  - Vendors included SGI, Cray, NEC, Sun, DEC, Fujitsu, …
- What are the drawbacks of this approach?
  - Expensive
  - Risky – bugs, market windows, single source suppliers …
  - Market fragmentation
- By the mid 1990’s x86-based PC processors were “good enough” for HPC [Thomas Sterling’s Beowulf]
  - Fast
  - Cheap
  - Multiple vendors of the same architecture (Intel, AMD, …)
  - ***Eventually*** added fast floating point, 64-bit memory addressing, hardware error correction, …

#1 1993, CM-5 from Thinking Machines

#1 1994, Paragon, Intel i860 RISC

#1 2000, ASCI white, IBM Power3

#1 2004, Earth Simulator NEC SX-6

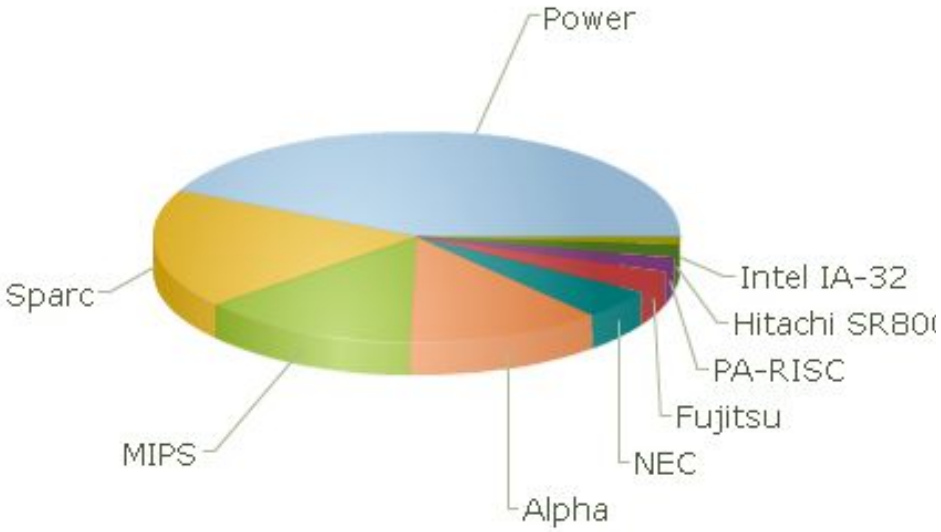
#1 2007, BlueGene/L IBM Power

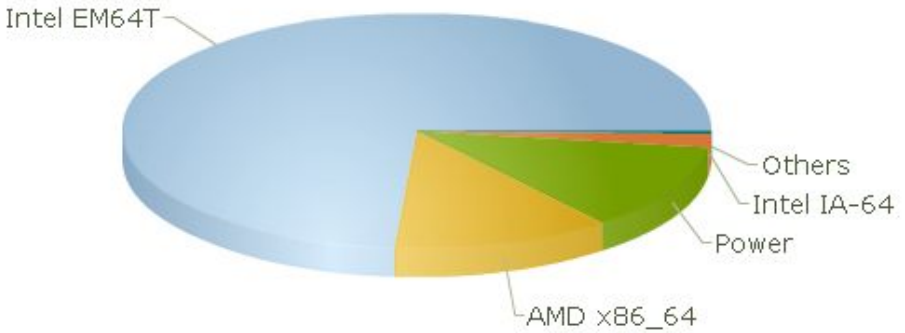
#1 2011, K Computer Fujitsu SPARC64

The commodity revolution in HPC

- HPC has been exploiting the commodity cluster trend since 2000

Top500 systems by Processor Family

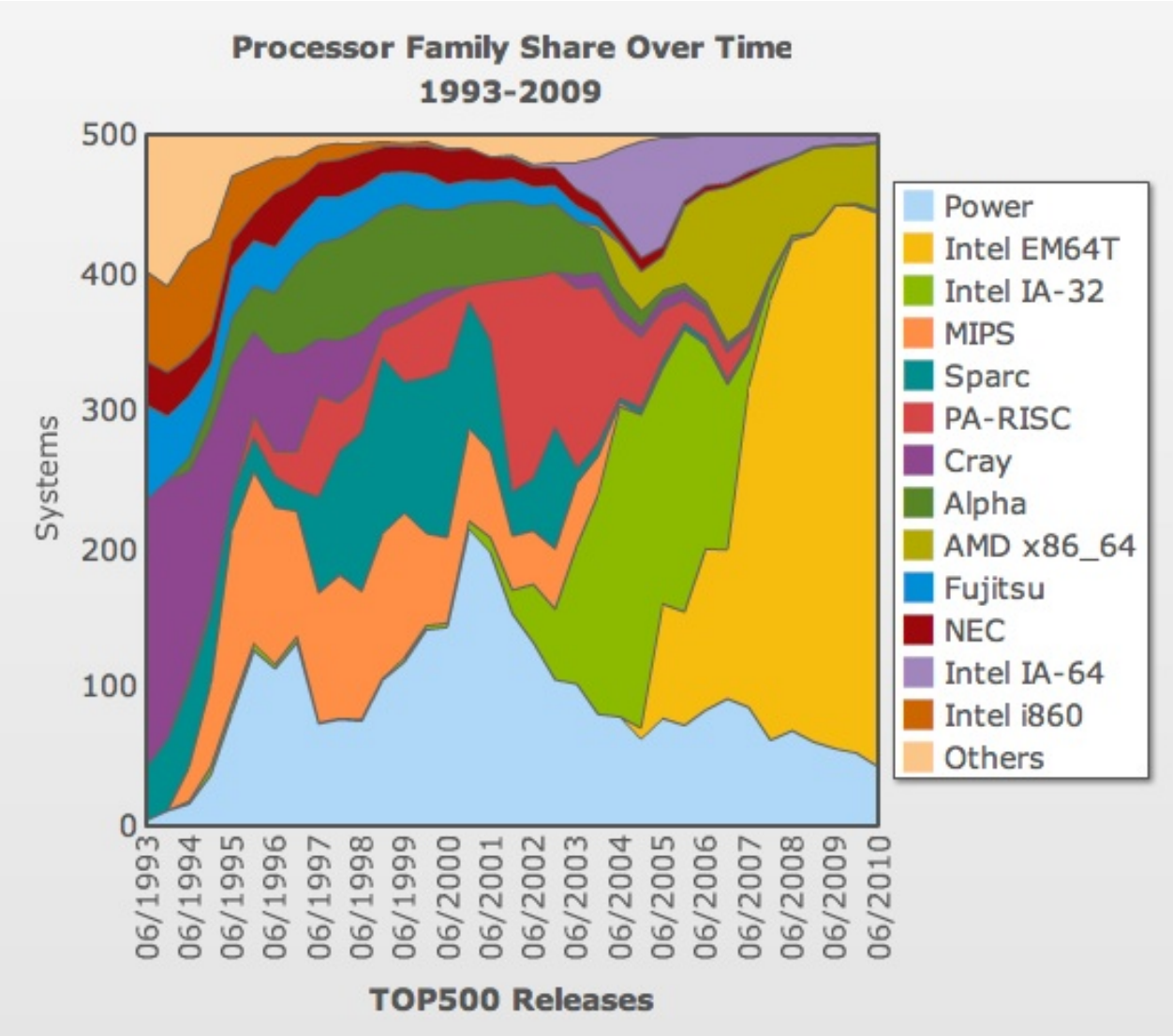




November 2000, x86 accounts for 1.2% of systems. By November 2008 this had grown to 85.8%.

http://www.top500.org/

Top500 HPC processor progression



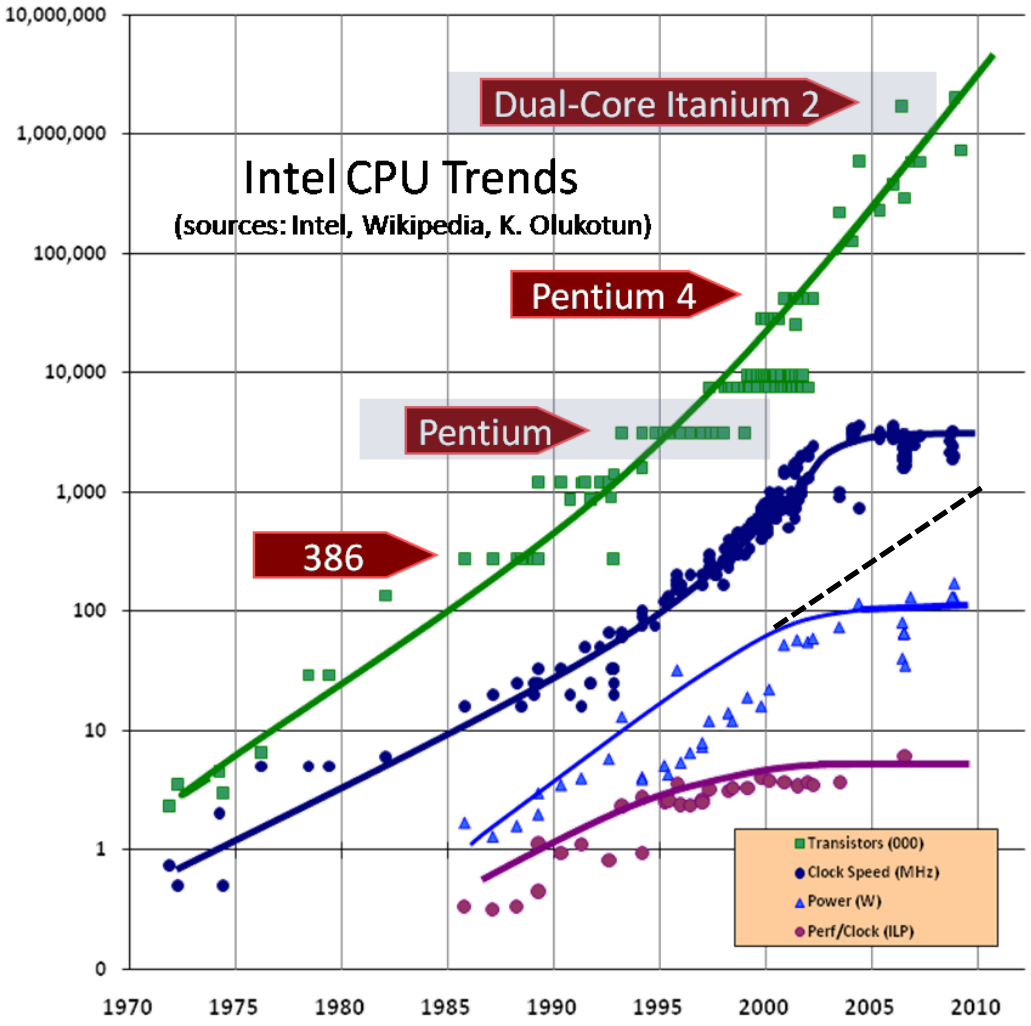
Processors are now all multi-core

- Intel Skylake x86 – up to 28 cores
- AMD Naples x86 – up to 32 cores
- AMD Jaguar x86 (PS4) – 8 cores + GPU
- IBM PowerPC BlueGene/Q – 16+2 cores
- IBM Cell (PS3) – 1+8 cores

- Intel Xeon Phi – up to 72 cores
- Graphics processors – >>hundreds "cores"

Why?

Because of Moore's Law



The real Moore's Law

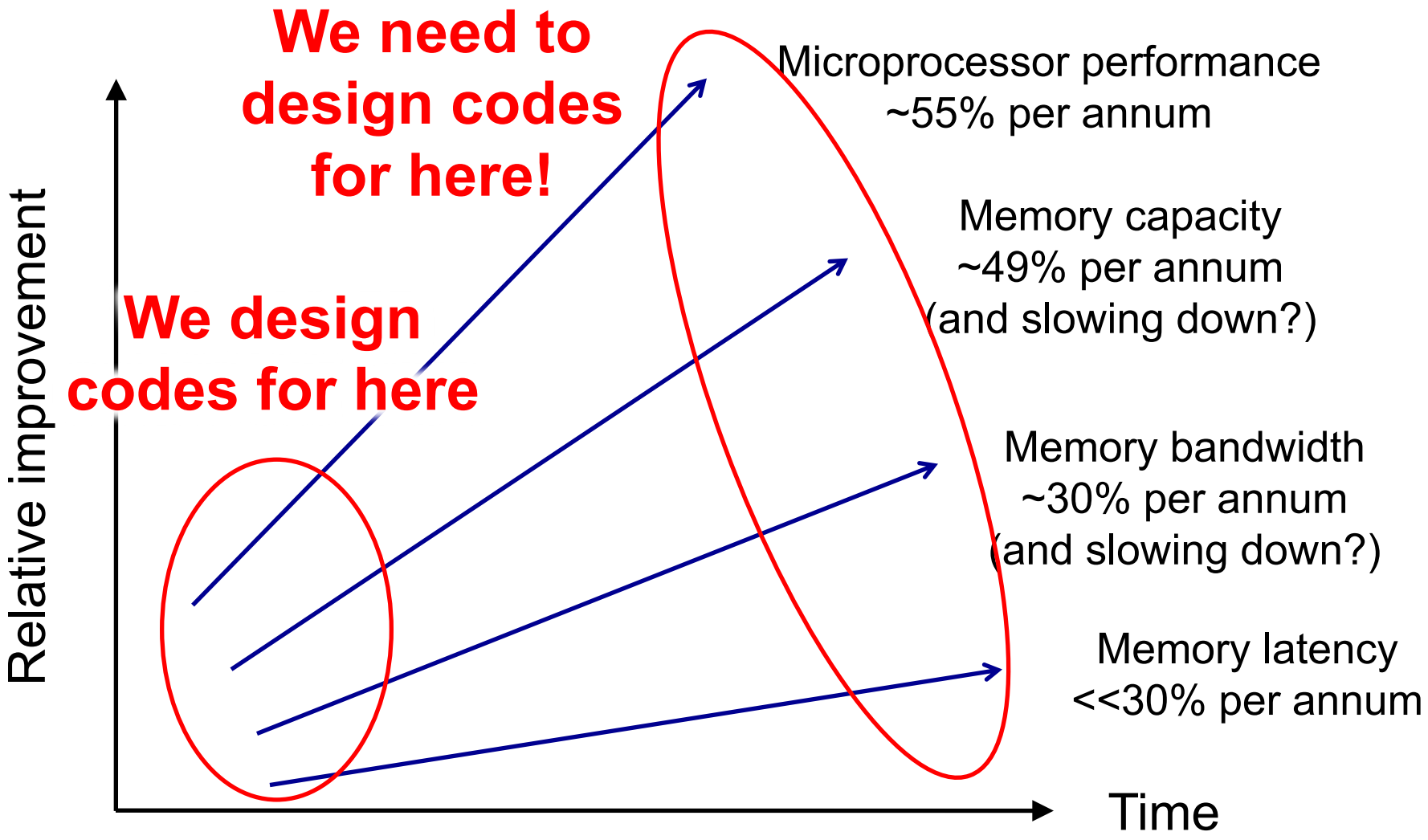
The clock speed plateau

The power ceiling

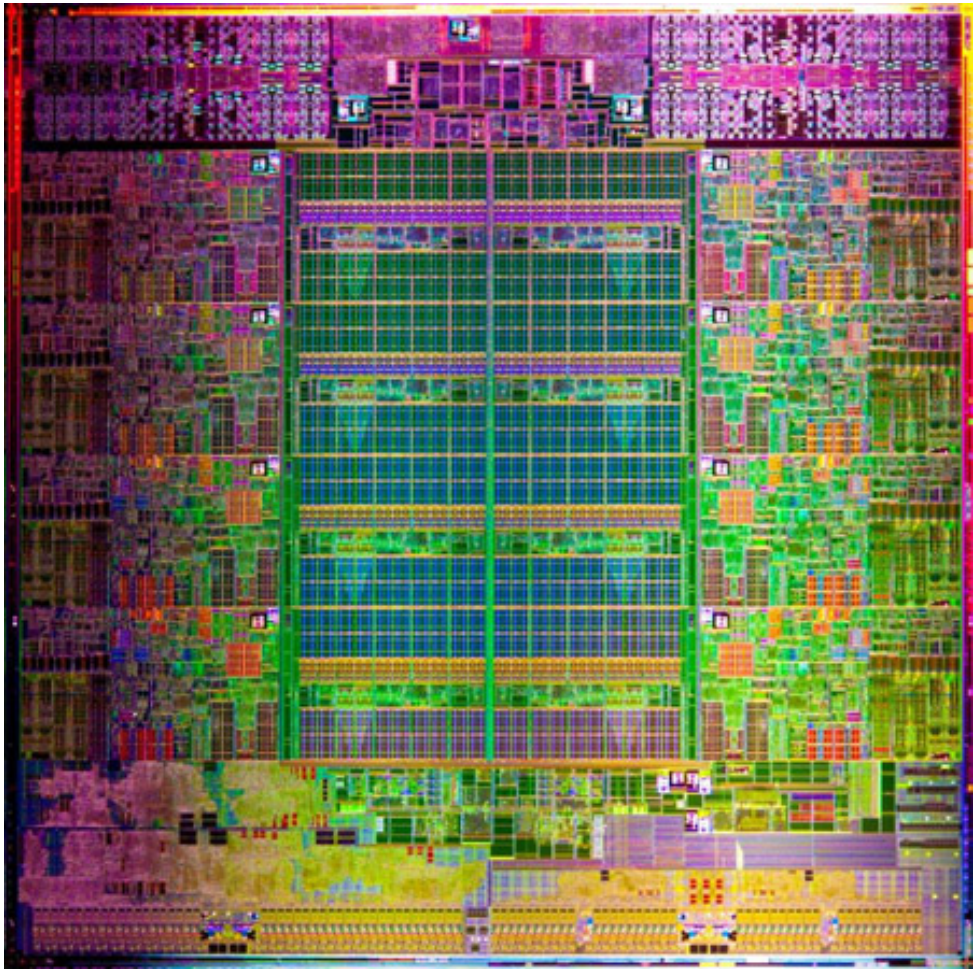
Instruction level parallelism limit

Herb Sutter, "The free lunch is over", Dr. Dobb's Journal, 30(3), March 2005. On-line version, August 2009.
http://www.gotw.ca/publications/concurrency-ddj.htm

The most important HPC trends



Intel's Sandy Bridge (in BCp3!)



- Xeon E5-2670 2.6 GHz
- 8 cores per CPU
- Two CPUs per node

- 20 MBytes of on-chip L3 cache per CPU
- 256KB L2 cache per core
- 32KB L1 data cache per core

- Supports AVX SIMD instructions
- 51.2 GBytes/s bandwidth to DRAM (main memory) per socket

- 32nm
- ~115W per CPU

A chassis / node of phase 3

Dell C8000 chassis

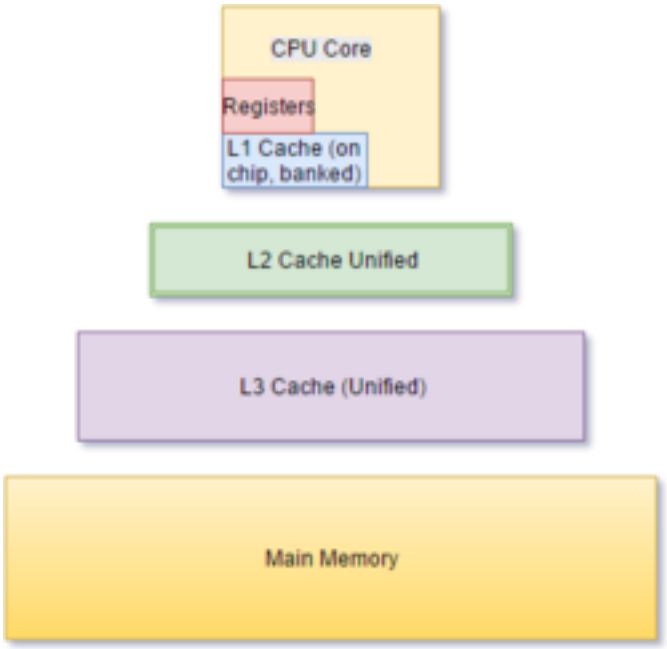
A CPU blade (left) and GPU blade (right)

What you need to know about a modern CPU

- They're complicated
- They rely on the **compiler** to present your code the best way possible
- They have **deep pipelines** that get messed up by certain instructions
  - **Branches**, long latency operations (e.g. **divide**) …
- They rely on exploiting multiple levels of fast but small, on-chip memories, called ***caches***

- The following graphs are from Subhash, Chang, and Jin, "*Performance evaluation of the Intel Sandy Bridge based NASA Pleiades using scientific and engineering applications*." *PMBS*, 2013

On-chip cache hierarchy

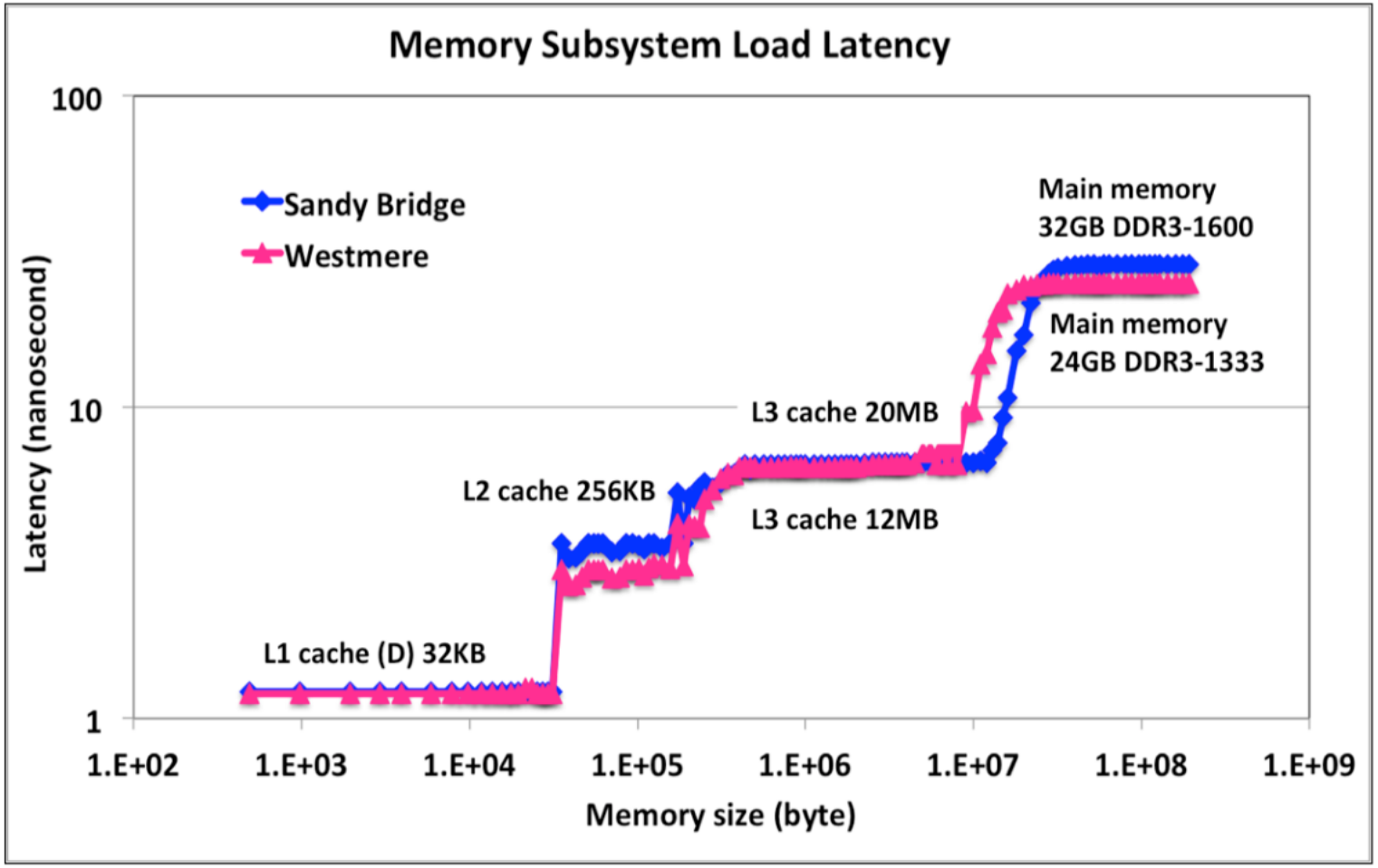


Level 1 (L1) cache (per core):
- Tens of KiloBytes (KB) in size
- 1-4 cycles to access

L2 cache (per core):
- Hundreds of KB in size
- 8-12 cycles to access

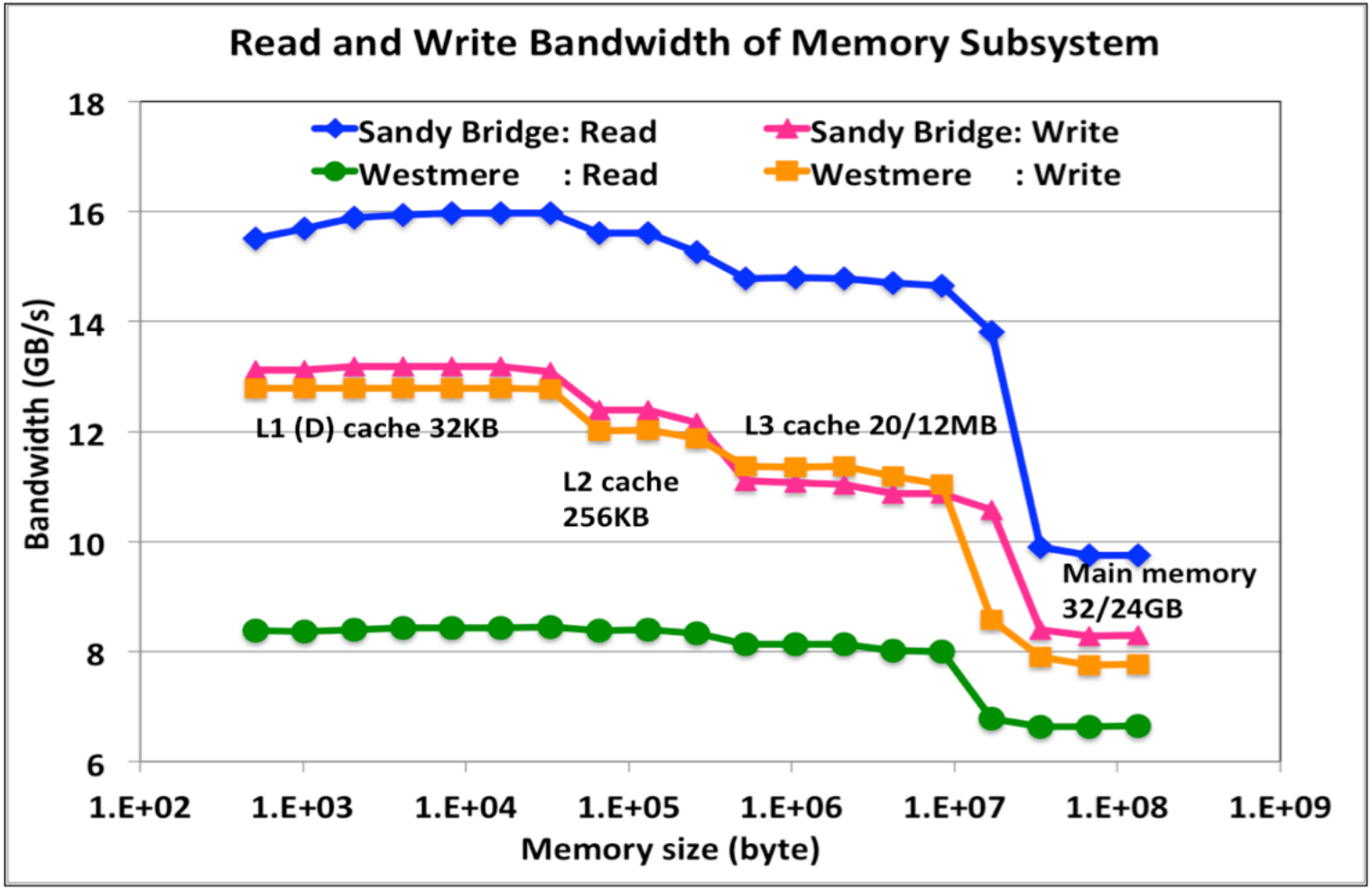
L3 cache (shared):
- Tens of MegaBytes (MB) in size
- 20-30 cycles to access

Main memory (DRAM, shared):
- Tens of GigaBytes (GB) in size
- 200-400 cycles to access



Sandy Bridge cache latencies (2.6GHz CPU):

L1 1.2ns, L2 3ns, L3 6.5ns, DRAM 28ns



Sandy Bridge cache bandwidths:

L1 16GB/s, L2 15.5GB/s, L3 15GB/s, DRAM 10GB/s

Heterogeneous computing is taking off

- Most systems are *already* heterogeneous
  - PCs have CPU, GPU, network processor, I/O processor, …
  - Has been a common approach in embedded systems since the early ‘90s

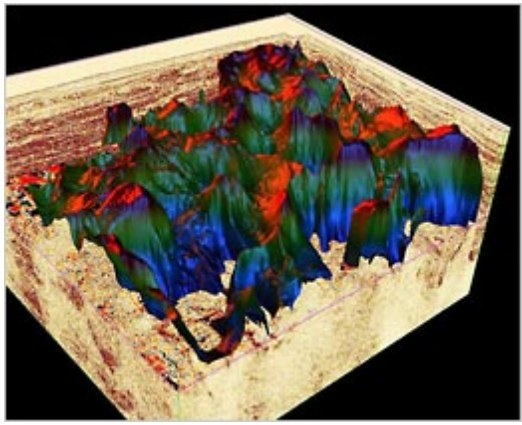
- But now heterogeneous systems are starting to include several different types of *general-purpose, programmable* processors
  - Users have to programme more than one type of processor to get the most out of a system

GPGPU computing

GPGPU (General-Purpose computation on Graphics Processing Units)

- Term first coined by Mark Harris in 2002
- http://gpgpu.org/ GPGPU

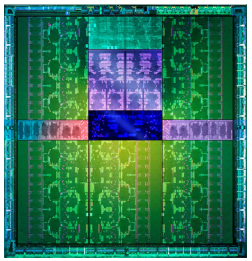
- The first GPGPU applications were still graphics-oriented (ray tracing, video, …)
- Found early use in Seismic Processing
  - FFT intensive, something GPUs are good at
- Also BLAS, PDEs, RNGs, …



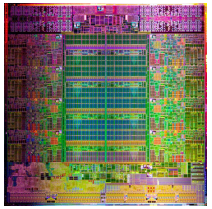
Why have GPUs become good at HPC?

- Graphics is a very **parallel problem**
  - Many vertices, polygons, pixels…
- Graphics APIs have become more and more *programmable* (OpenGL, Direct3D, …)
- Driven by a mass market with huge volumes
  - Lots of game players
  - Drives large-scale Research and Development
- Graphics market now saturated, vendors looking for new markets (esp. Nvidia)

Comparing GPUs and CPUs



- Nvidia Kepler K20
- 2,496 simple cores
- 7.1 billion transistors
- 706 MHz
- 3,520 GFLOPS S.P.
- 1,170 GFLOPS D.P.
- 208 GBytes/s
- 225W (whole board)
- 561mm^2 die size



- Intel Sandy Bridge
- 8 complex cores
- 2.3 billion transistors
- 2.6 GHz
- 332 GFLOPS S.P.
- 166 GFLOPS D.P.
- 51 GBytes/s
- 115W (CPU only)
- 435mm^2 die size

Heterogeneous systems in the TOP500 JUNE 2010

- Tokyo Tech's TSUBAME was first in 2006
  - Started with ClearSpeed, now using GPUs

- Now dozens of GPU-based systems in existence, more on their way
  - Former #1 system Titan, 17.6 PetaFLOPS using 18,000 GPUs

Conclusions

- Performance per watt = performance
- Heterogeneous computing is here to stay
- Even single chips will contain thousands of cores
- Hierarchies will become deeper
  - Processing, interconnect, memory, software
- Compute >> Memory >> Bandwidth >> Latency
- Parallelism is now increasing exponentially and will continue to do so
- **All** progamming is now **parallel programming!!**

Before next week

- Read up how commodity clusters came to dominate HPC: start by researching the history of Thomas Sterling's *Beowulf* project

- Find out more about the CPUs in Blue Crystal phase 3, especially about how caches work
  - This knowledge should help you with a future assignment

Further reading

- Reading list available on the website
- History of the commodity cluster – Thomas Sterling et al's invention of the cheap, x86-based Linux cluster, 'Beowulf', circa 1994: http://www.beowulf.org/overview/history.html
- Herb Sutter, "The free lunch is over", Dr. Dobb's Journal, 30(3), March 2005. On-line version, August 2009. http://www.gotw.ca/publications/concurrency-ddj.htm
- Lots more in my 4th year unit 'Advanced Computer Architecture', COMSM0109