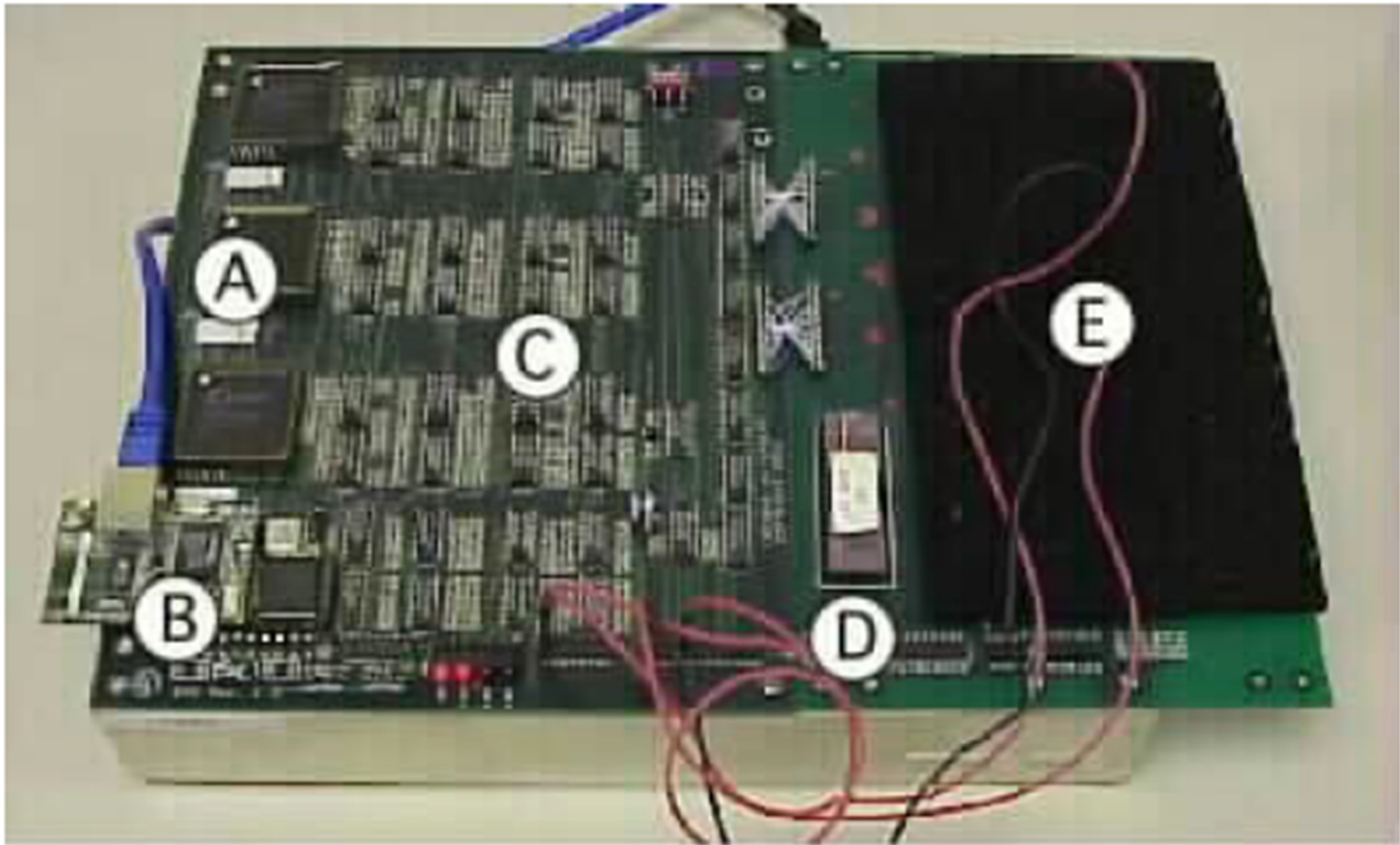


Building Blocks

CPUs, Memory and Accelerators







What is a computer?

- Device that manipulates data according to a set of instructions
 - Simple mathematical operations
 - Many operations per second
 - Can be programmed (i.e. instructions sets than can be saved and used when needed)
 - Embedded most common, Personal Computer what most think of as a computer
- Digital
 - Information stored in discrete quantities
 - 0 and 1 represented by presence or absence of electricity
 - Easy to create electronic circuits based on this



Outline

- Computer layout
 - CPU and Memory
 - What does performance depend on?
 - Limits to performance
- Silicon-level parallelism
 - Single Instruction Multiple Data (SIMD/Vector)
 - Multicore
 - Symmetric Multi-threading (SMT)
- Accelerators (GPGPU and Xeon Phi)
 - What are they good for?

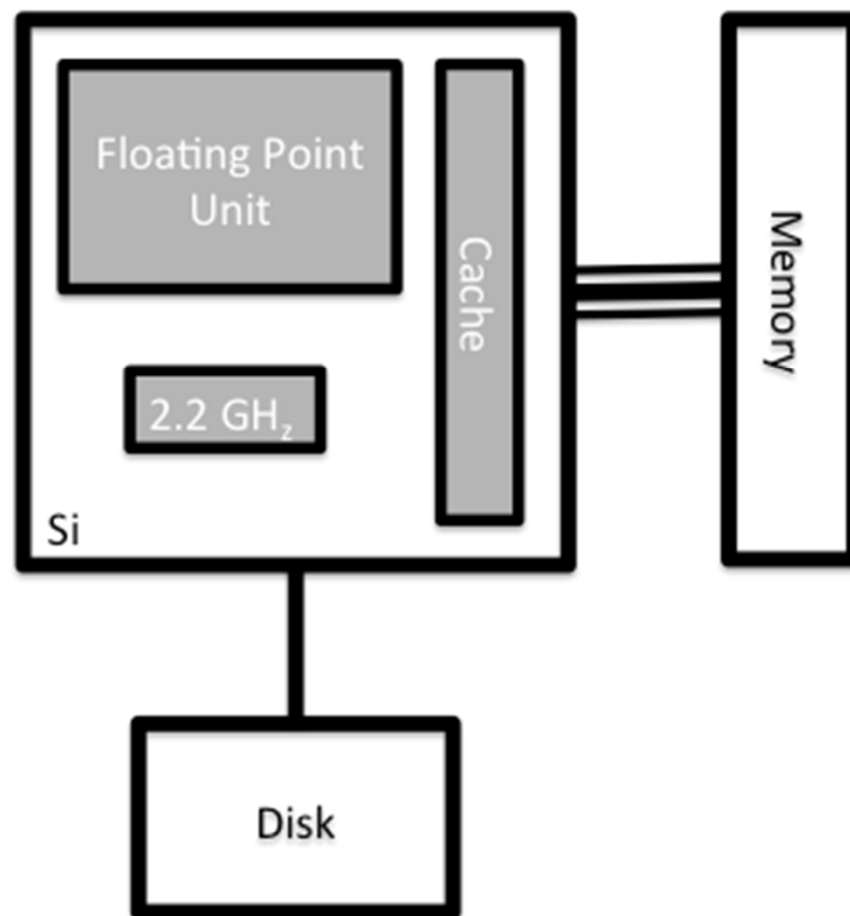


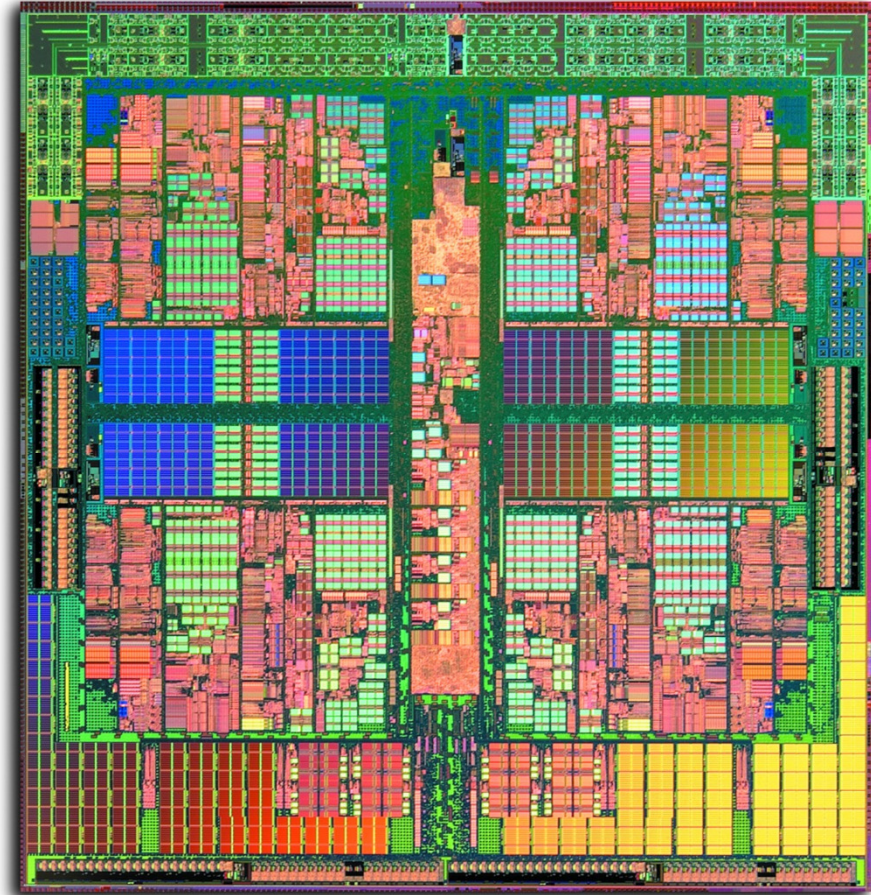
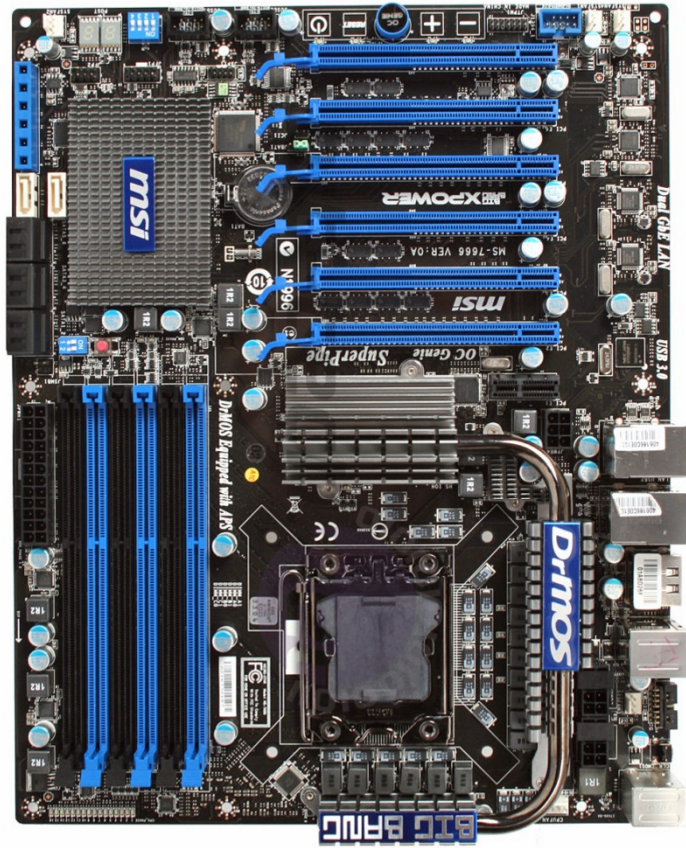
Computer Layout

How do all the bits interact and which ones matter?



Anatomy of a computer





Performance

- The performance (time to solution) on a single computer can depend on:
 - Clock speed – how fast the processor is
 - Floating point unit – how many operands can be operated on and what operations can be performed?
 - Memory latency – how fast can we access the data?
 - Memory bandwidth – how much data can we access in one go?
 - Input/Output (IO) to storage – how quickly can we access persistent data (files)?



Performance (cont.)

- Application performance often described as:
 - Compute bound
 - Memory bound
 - IO bound
 - (Communication bound – more on this later...)



Limits to performance

- Scientific simulation and modelling drive the need for greater computing power.
- Single systems can not be made that had enough resource for the simulations needed.
 - Making faster single chip is difficult due to both physical limitations and cost.
 - Adding more memory to single chip is expensive and leads to complexity.
- Solution: parallel computing – divide up the work among numerous linked systems.
 - HPC has become synonymous with *parallel computing*



Silicon-level parallelism

What does Moore's Law mean anyway?



Single Instruction Multiple Data (SIMD)

- For example, vector addition:

$$\begin{array}{|c|} \hline a_0 \\ \hline a_1 \\ \hline a_2 \\ \hline a_3 \\ \hline \end{array} + \begin{array}{|c|} \hline b_0 \\ \hline b_1 \\ \hline b_2 \\ \hline b_3 \\ \hline \end{array} = \begin{array}{|c|} \hline c_0 \\ \hline c_1 \\ \hline c_2 \\ \hline c_3 \\ \hline \end{array}$$

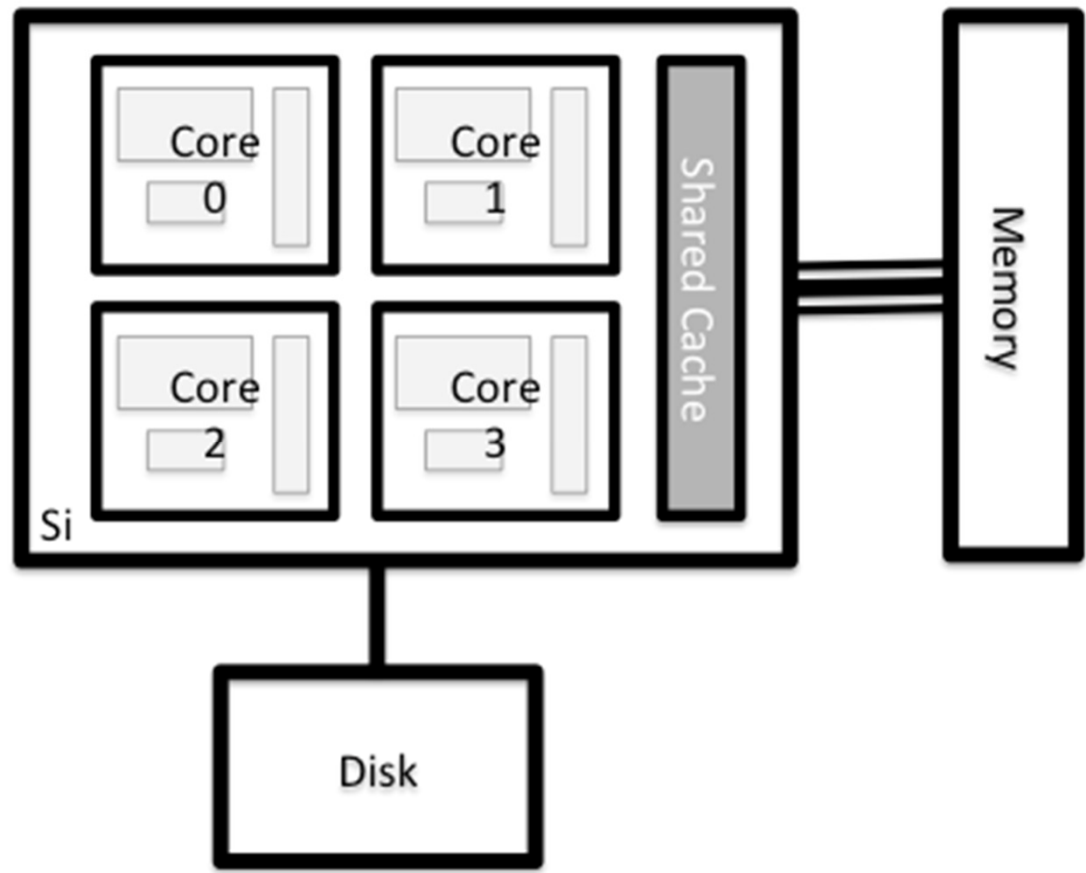


Symmetric Multi-threading (SMT)

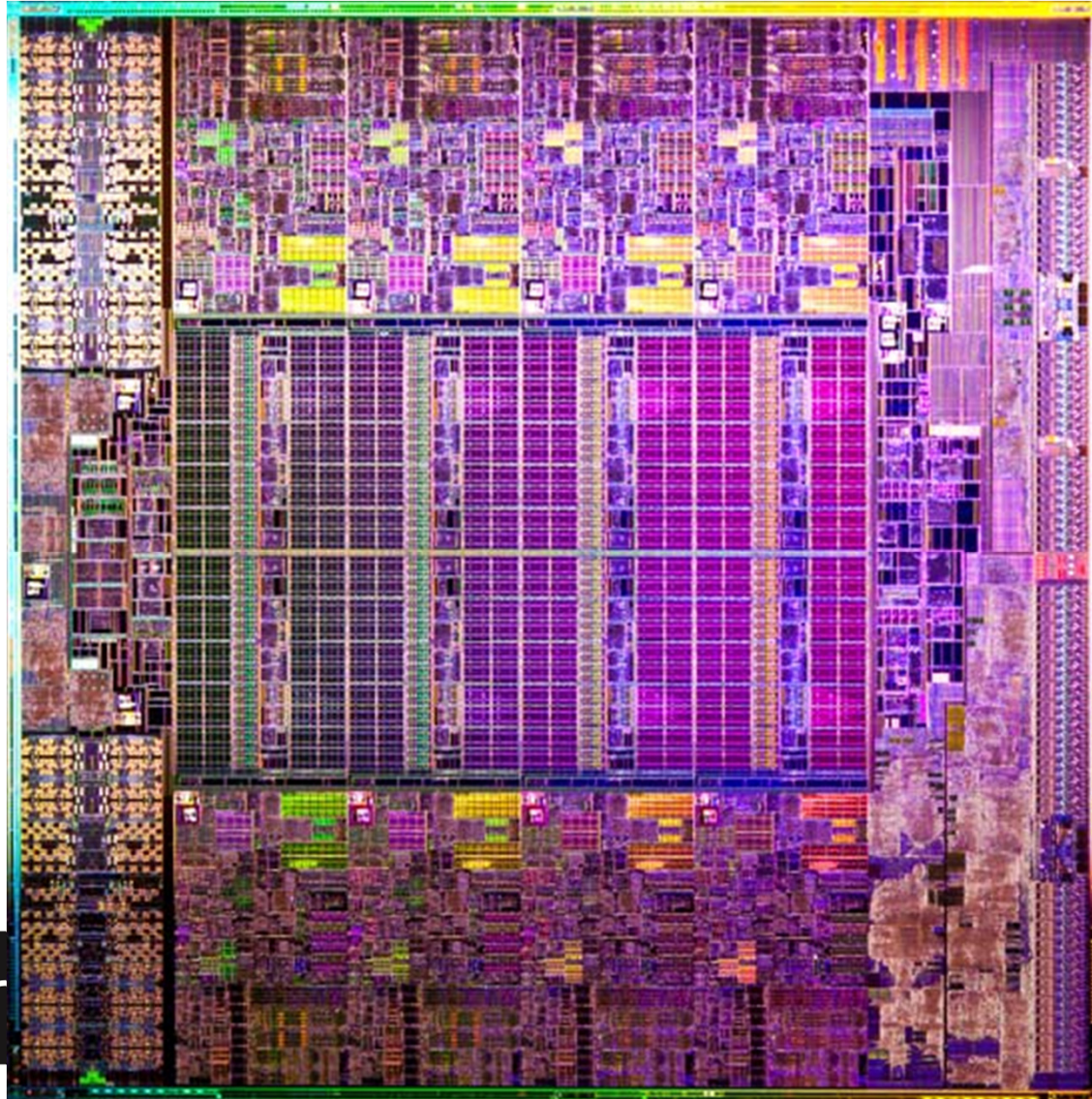
- Some hardware supports running more processes than there are physical cores
- Known as *Symmetric Multi-threading (SMT)* or *hyperthreading*
- Threading in this case can be a misnomer as it can refer to processes as well as threads
 - These are hardware threads, not software threads.
 - Intel Xeon supports 2-way SMT
 - IBM BlueGene/Q 4-way SMT



Multicore



Intel Xeon E5-2600 – 8 cores HT



Chip types and manufacturers

- x86 – Intel and AMD
 - “PC” commodity processors, SIMD (SSE, AVX) FPU, multicore, SMT (Intel), Intel currently dominate the HPC space.
- Power – IBM
 - Used in high-end HPC, high clock speed (direct water cooled), SIMD FPU, multicore, SMT, not as important anymore.
- PowerPC – IBM BlueGene
 - Low clock speed, SIMD FPU, multicore, high level of SMT.
- SPARC – Fujitsu
- ARM – Lots of manufacturers
 - Not yet relevant to HPC (weak FP Unit)



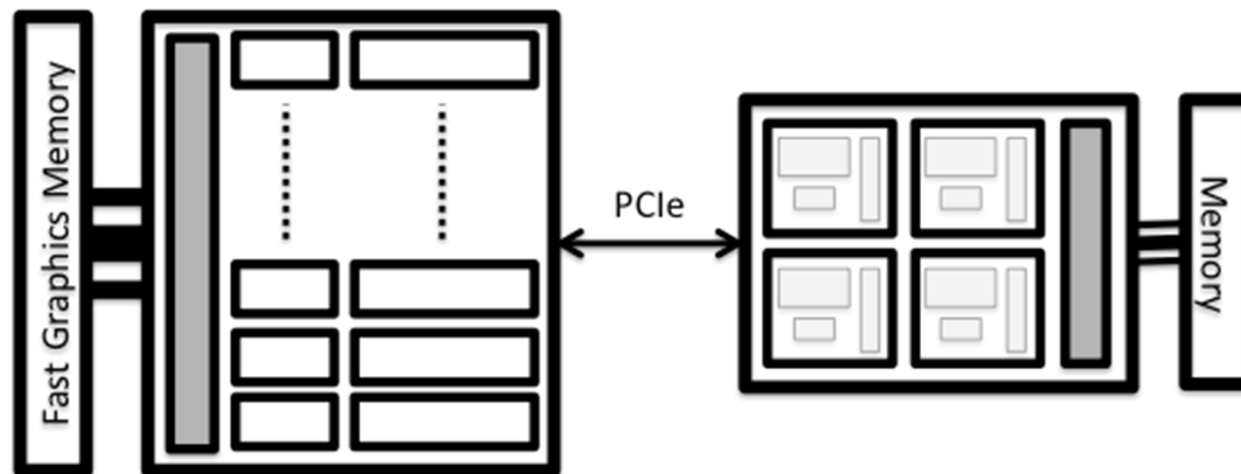
Accelerators

Go-faster stripes



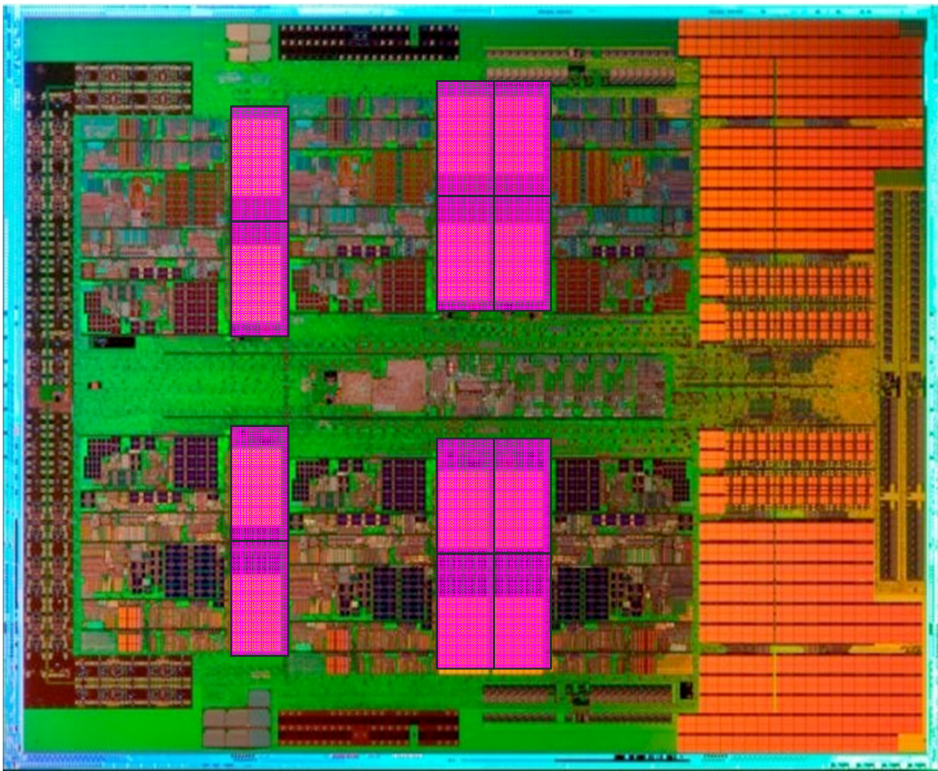
Anatomy


- An *Accelerator* is a additional resource that can be used to off-load heavy floating-point calculation
 - It is an additional processing engine that is attached to the standard processor
 - It has its own floating point units and memory



AMD 12-core CPU

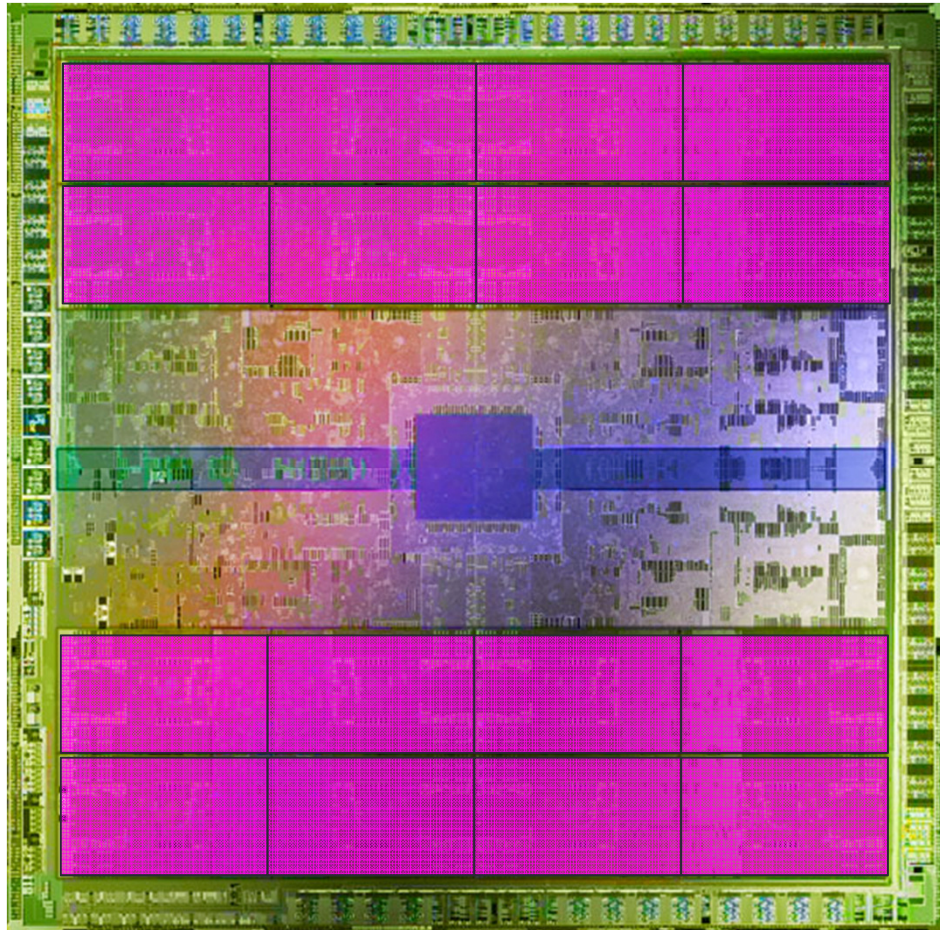
- Not much space on CPU is dedicated to compute




 = compute unit
(= core)



NVIDIA Fermi GPU

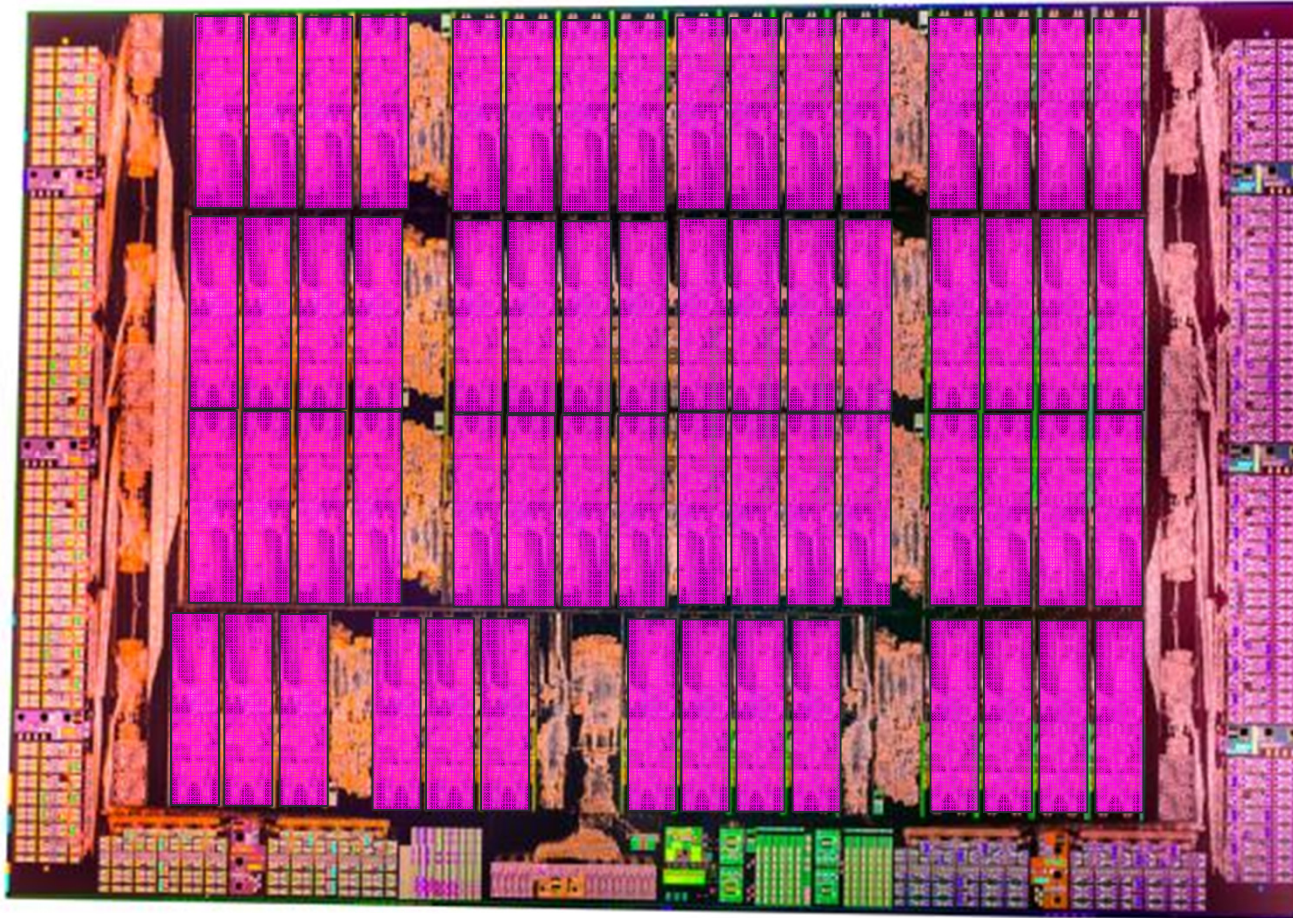


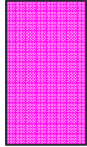
- GPU dedicates much more space to compute
 - At expense of caches, controllers, sophistication etc

 = compute unit
(= SM
= 32 CUDA cores)



Intel Xeon Phi

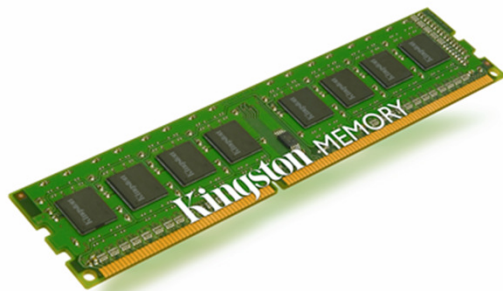


 = compute unit
(= core)



Memory

- For many applications, performance is very sensitive to memory bandwidth
- GPUs and Intel Phi both use Graphics memory: much higher bandwidth than standard CPU memory



CPUs use DRAM



GPUs and Xeon Phi use Graphics DRAM



Summary - What is automatic?

- Which features are managed by hardware/software and which does the user/programmer control?
 - Cache and memory – automatically managed
 - SIMD/Vector parallelism – automatically produced by compiler
 - SMT – automatically managed
 - Multicore parallelism – manually specified by the user
 - Use of accelerators – manually specified by the user

