PROGRAMMING THE XEON PHI

ARCHER and DiRAC Course

EPCC and Cambridge
Adrian Jackson
adrianj@epcc.ed.ac.uk

Intel’s IPCC program

- Collaboration between Intel and leading Universities around the world
- “Intel® Parallel Computing Centers are universities, institutions, and labs that are leaders in their field, focusing on modernizing applications to increase parallelism and scalability through optimizations that leverage cores, caches, threads, and vector capabilities of microprocessors and coprocessors.”

Reusing this material

Course Parameters

- Pre-requisites
  - Some level of C or FORTRAN programming knowledge
- Hands-on practicals form an integral part of the course.
  - We will help with these

Aims

- On completion of this course students should be able to:
  - Describe the Xeon Phi architecture.
  - Use the Intel compiler and associated tools to exploit its full computational potential.
  - Use a variety of programming models to accelerate code.
  - Understand how the Xeon Phi operates as part of a larger HPC system.

Timetable

Day 1

09.30 - 10.00: Introduction to the Xeon Phi

10.00 - 10.45: Programming models for the Xeon Phi

10.45 - 11.00: Practical: Introduction to Xeon Phi

11.00 - 11.30: Break

11.30 - 12:00: Achievable performance on Xeon Phi

12.00 - 12.30: Native model programming

12.30 - 13.00: Practical: Native mode

13.00 - 14.00: Lunch

14.00 - 15.00: Off-loading to the Xeon Phi

15.00 - 15.30: Practical: Off-loading

15.30 - 16.00: Break

16.00 - 16.30: Case studies: Porting to the Xeon Phi

16.30 - 17.30: Practical: Continue the practicals

Day 2

09.30 - 09.45: Recap of Xeon Phi

09.45 - 10.30: Vectorisation

10.30 - 11.00: Practical: Vectorisation

11.00 - 11.30: Break

11.30 - 12.00: Practical: Vectorisation

12:00 - 13.00: Serial Optimisation

13.00 - 14.00: Lunch

14.00 - 15.00: Practical: Serial optimisation

15.00 - 15.30: Optimising MPI and off-loading

15.30 - 16.00: Break

16.00 - 17.00: Continue practicals or port own code

17.00 - 17.30: Summary and finish

Course materials

- Everything online:
  - Slides, exercise notes, code to use

`http://www.archer.ac.uk/training/course-material/2015/04/XeonPhi_EPCC/`

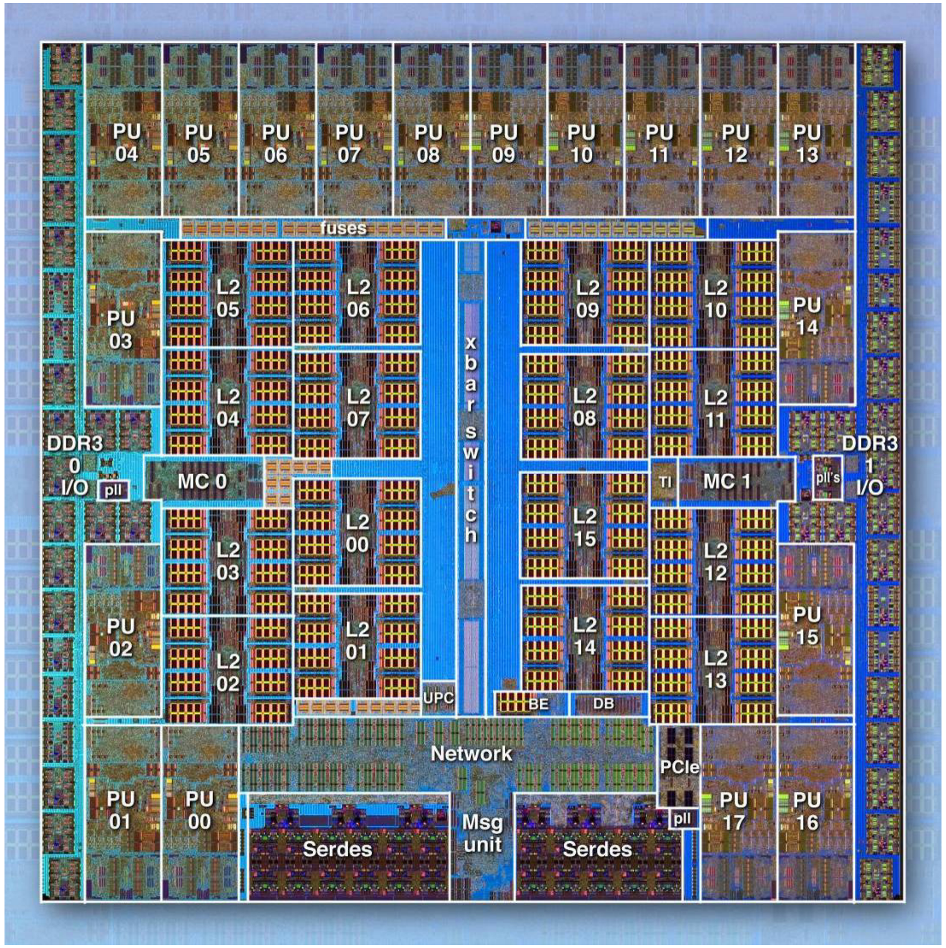
Feedback and follow-up

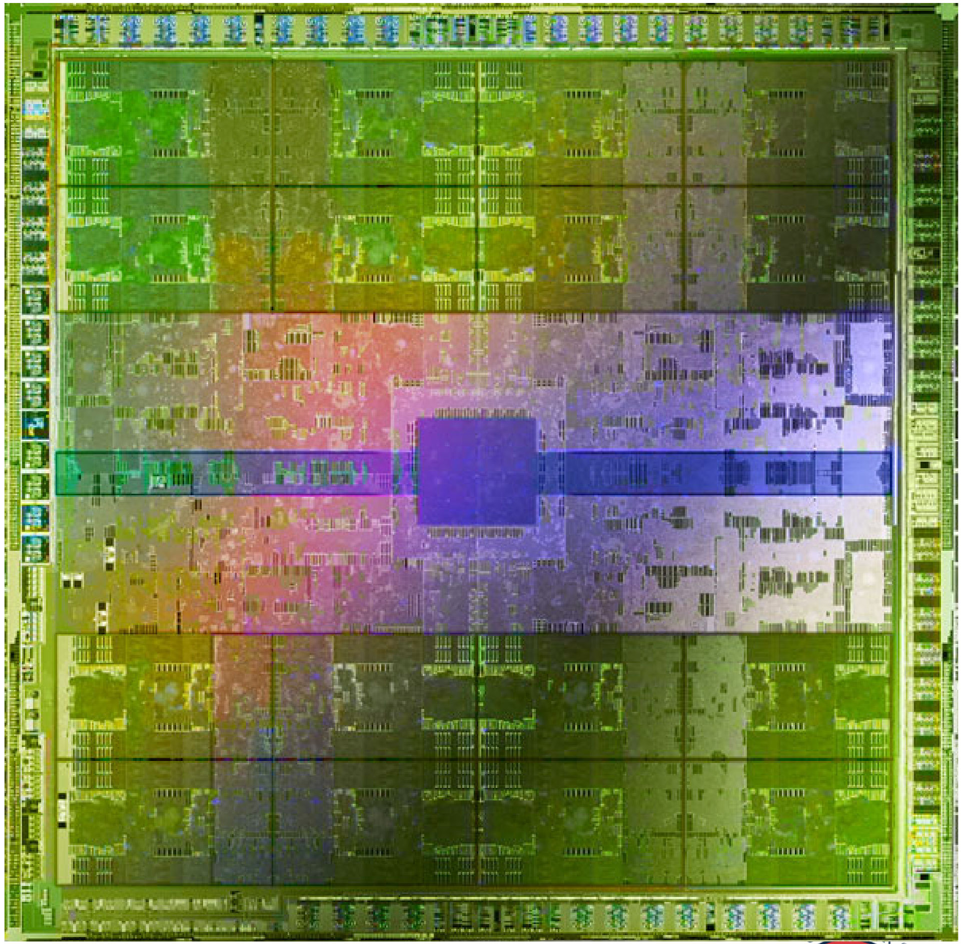
- http://www.archer.ac.uk/training/feedback/
- Virtual Tutorials
  - Online every second Wednesday of the month
  - http://www.archer.ac.uk/training/virtual/

Processors

- The power used by a CPU core is proportional to Clock Frequency x Voltage^2
- In the past, computers got faster by increasing the frequency
  - Voltage was decreased to keep power reasonable.
- Now, voltage cannot be decreased any further
  - 1s and 0s in a system are represented by different voltages
  - Reducing overall voltage further would reduce this difference to a point where 0s and 1s cannot be properly distinguished
- Other performance issues too…
  - Capacitance increases with complexity
  - Speed of light, size of atoms, dissipation of heat
- And practical issues
  - Developing new chips is incredibly expensive
- Must make maximum use of existing technology
- Now parallelism explicit in chip design
  - Beyond implicit parallelism of pipelines, multi-issue and vector units

Multicore processors





Accelerators

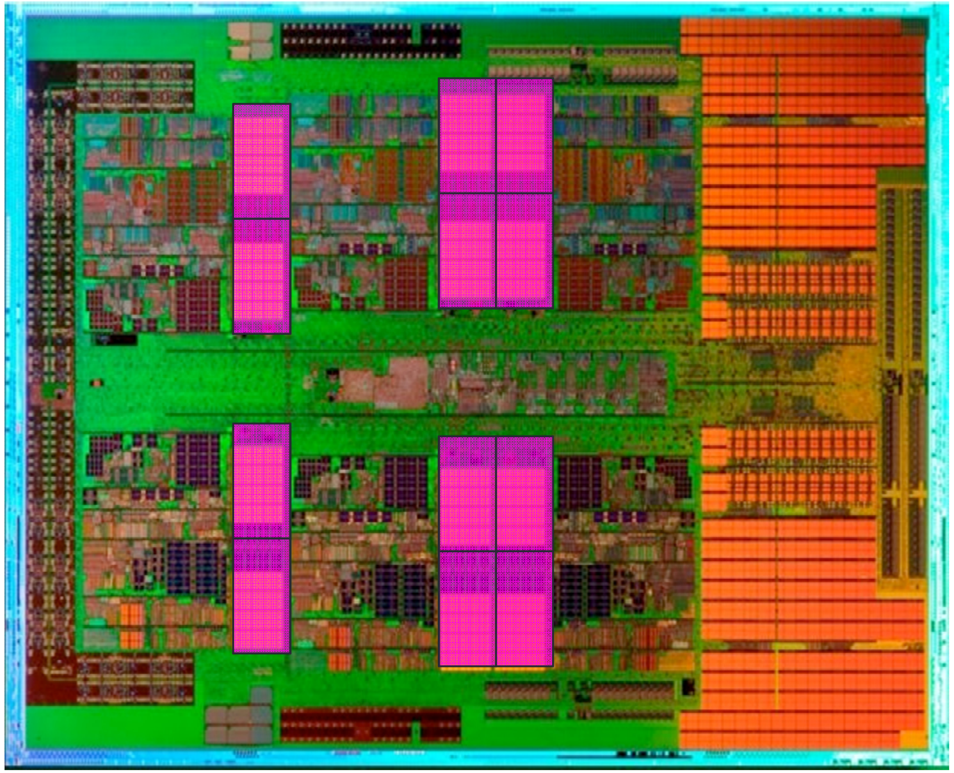
- Need a chip which can perform many parallel operations every clock cycle
  - Many cores and/or many operations per core
  - Floating Point operations (FLOPS) what is generally crucial for computational simulation
- Want to keep power/core as low as possible
- Much of the power expended by CPU cores is on functionality not generally that useful for HPC
  - Branch prediction, out-of-order execution etc

Accelerators

- So, for HPC, we want chips with simple, low power, number-crunching cores
- But we need our machine to do other things as well as the number crunching
  - Run an operating system, perform I/O, set up calculation etc
- Solution: “Hybrid” system containing both CPU and “accelerator” chips

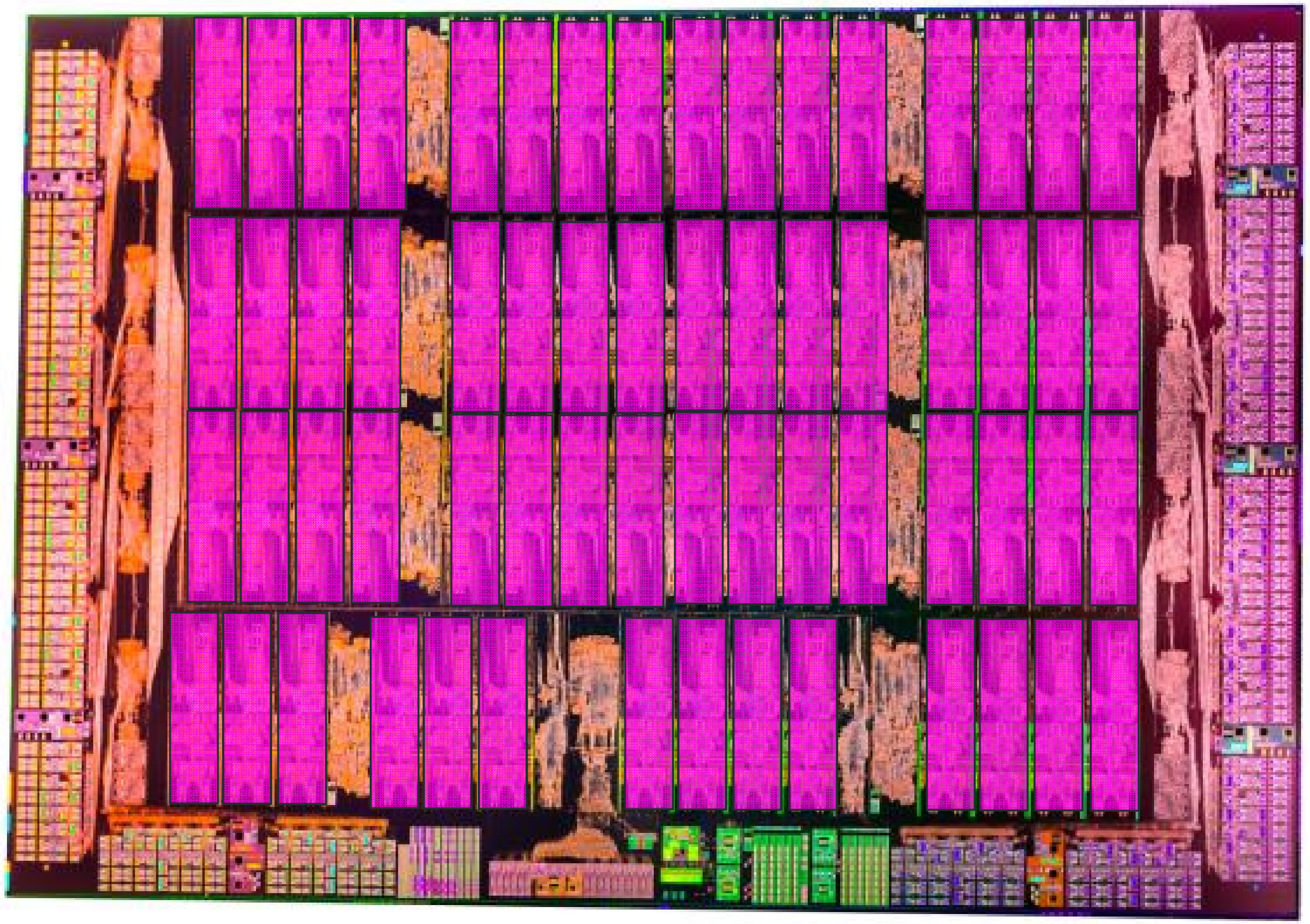
AMD 12-core CPU

- Not much space on CPU is dedicated to compute



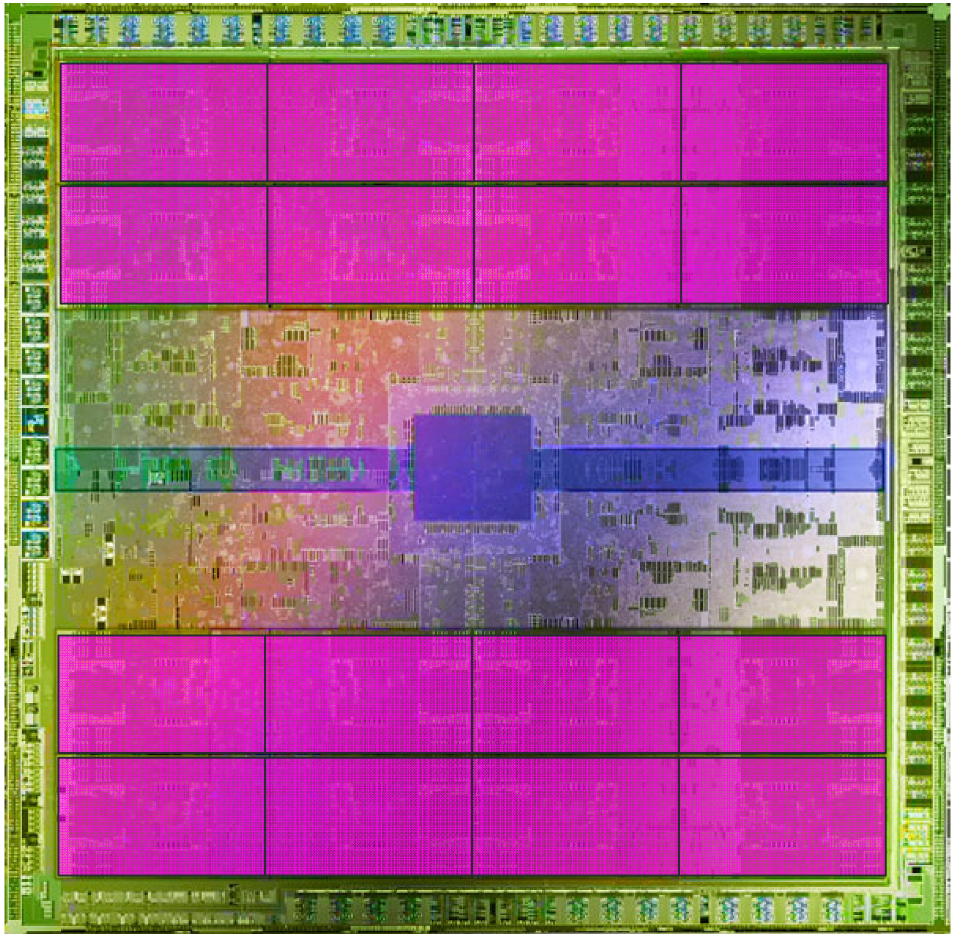
🟪 = compute unit
(= core)

Intel Xeon Phi



🟪 = compute unit (= core)

NVIDIA Fermi GPU



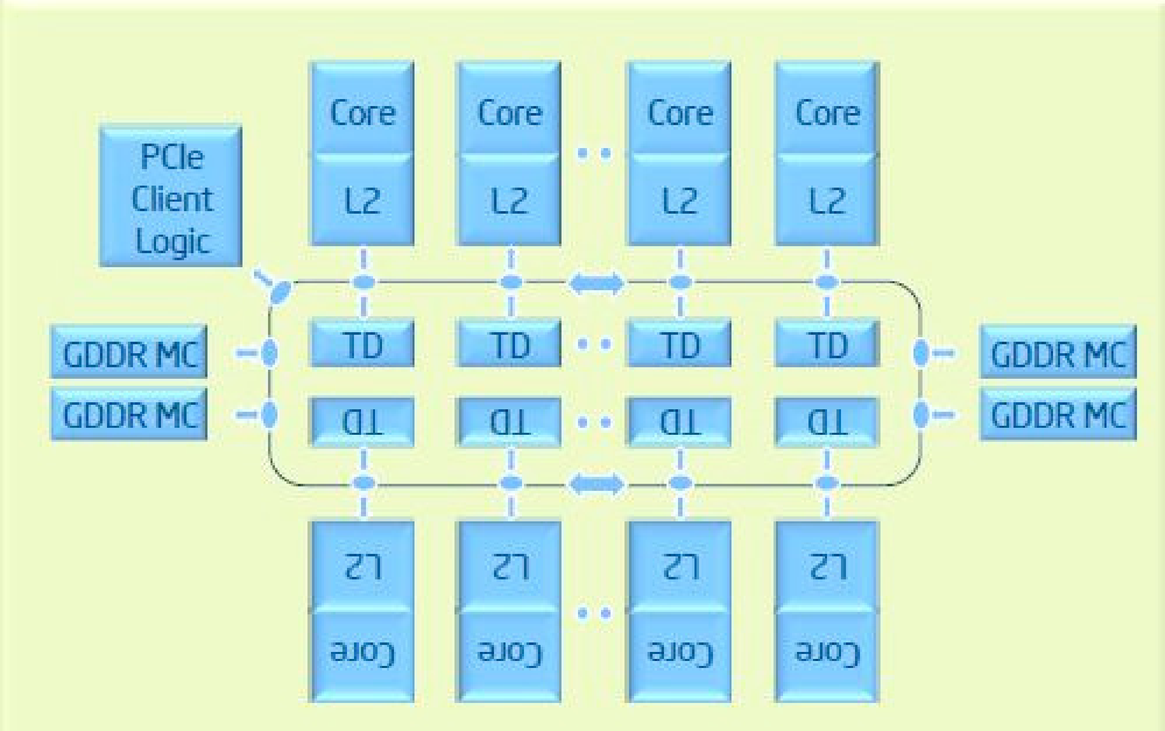
🟪 = compute unit
(= SM
 = 32 CUDA cores)

Intel Xeon Phi

- Intel Larrabee: “A Many-Core x86 Architecture for Visual Computing”
  - Release delayed such that the chip missed competitive window of opportunity.
  - Larrabee was not released as a competitive product, but instead a platform for research and development (Knight’s Ferry).
- Knights Corner derivative chip
  - Intel Xeon Phi – co-processor
  - Many Integrated Cores (MIC) architecture. No longer aimed at graphics market
    - Instead “Accelerating Science and Discovery”
  - PCIe Card
  - 60 cores/240 threads/1.054 GHz
  - 8 GB/320 GB/s
  - 512-bit SIMD instructions
- Hybrid between GPU and many-core CPU

Intel Xeon Phi

- Each core has a private L2 cache
- “ring” interconnect connects components together
- Chip is cache coherent



Intel Xeon Phi

- Intel Pentium P54C cores were originally used in CPUs in 1993
  - Simplistic and low-power compared to today's high-end CPUs

- Philosophy behind Phi is to dedicate large fraction of silicone to many of these cores

- And, similar to GPUs, Phi uses Graphics GDDR Memory
  - Higher memory bandwidth that standard DDR memory used by CPUs

Intel Xeon Phi

- Each core has been augmented with a wide 512-bit vector unit

- For each clock cycle, each core can operate vectors of size 8 (in double precision)
  - Twice the width of 256-bit “AVX” instructions supported by current CPUs

- Multiple cores, each performing multiple operations per cycle

Intel Xeon Phi

	3100 series	5100 series	7100 series
cores	57	60	61
Clock frequency	1.100 GHz	1.053 GHz	1.238 GHz
DP Performance	1 Tflops	1.01 TFlops	1.2 TFlops
Memory Bandwidth	240 GB/s	320 GB/s	352 GB/s
Memory	6 GB	8 GB	16 GB

Xeon Phi Systems

- Unlike GPUs, Each Xeon Phi runs an operating system
- User can log directly into Xeon Phi and run code
  - “native mode”
  - But any serial parts of the application will be very slow relative to running on modern CPU
- Typically, each node in a system will contain at least one regular CPU in addition to one (or more) Phis.
- Phi Acts as an “accelerator”, in exactly the same way as already described for GPU systems.
- “Offload mode”: run most source code on main CPU, and offload computationally intensive parts to Phi

Summary

- Xeon Phi is an accelerator card
- ~60 cores, each can run 4 threads
  - Slow, simple cores
  - Wide vector units
- Can run standard code