Parallel Computing

Jorge Barbosa
University of Porto
www.fe.up.pt/~jbarbosa

Introduction

Before: CPU Gflop/s increased by increasing frequency

"the more ticks you have per second, the more work will get done"

Why not push the clock faster?

Speed/power tradeoff
It's no longer worth the cost in terms of power consumed and heat dissipated.

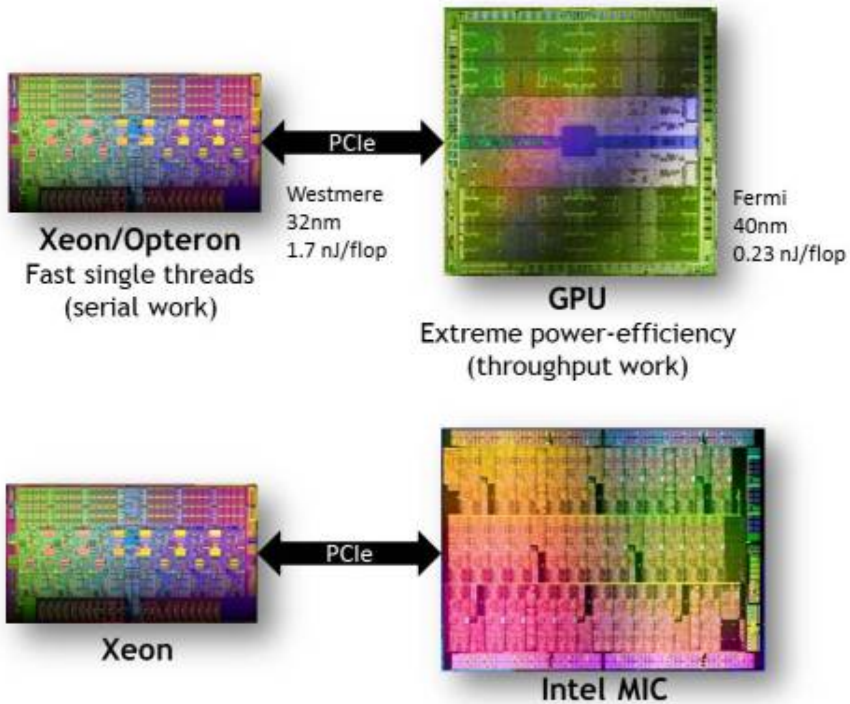
Underclocking a single core by **20%** saves **50% of the power** while sacrificing just **13%** of the performance.

Dividing the work between **two cores** running at an **80%** clock rate, we get **43% more** performance for the **same power**.

Source: "Why CPU Frequency Stalled" By Philip E. Ross, IEEE Spectrum April 2008

Heterogeneous Computing

- Evolution of computing systems:
  highly parallel & heterogeneous !
  - new computing units: gpGPU/MIC/…



HPC systems in Top500: #1,2,6,10 with Intel Xeon MIC & NVidia GPU
…
Tianhe-2: 3,120,000 cores 16,000 nodes
…
NVidia K20x: 2,880 arith cores

Intel Xeon Phi (2013)

Intel® Xeon Phi™ coprocessor 5110P:
Ideal for high density environments

- Highly parallel applications using over 100 threads
- Memory bandwidth-bound applications
- Applications with extensive vector use

Buy the Intel® Xeon Phi™ coprocessor 5110P today >

xeon-phi-serverblade-feature-320x160.jpgKey specifications:

- 60 cores/1.053 GHz/240 threads
- 8 GB memory and 320 GB/s bandwidth
- Standard PCIe* x16 form factor, passively cooled
- Linux* operating system, IP addressable
- 512-bit single instruction, multiple data instructions
- Supported by the latest Intel® software development products
- Built using Intel's 22nm process technology—Intel's most energy efficient process yet—featuring the world's first 3-D tri-gate transistors.

60 Intel cores in a desktop

Manycore GPUs (attached processors)

- **GeForceGTX 280**
  - 240 scalar cores
    - Organized in blocks of 8 scalar cores
      - 16K 32-bit registers (64KB)
      - usual ops: float, int, branch, …
    - Shared double precision unit
    - …
- **TESLA**
  - Up to 2880 scalar cores

- **Manycore programming**
  - CUDA -- NVIDIA only
  - OpenCL -- integration of CPU and GPU
  - OpenACC

Mobile Computing

Quad-Core 1.4GHz

Programming multicore processors

- Will compilers do the job?
  - Probably they won't
  - Even for sequential programming we need to do explicitly memory management to get performance and scalable programs (data size and data locality).

```
for (i=1; i<n; i++)
  for (j=1; j<n; j++)
     for (k=1; k<n; k++)
        c[i,j]+= a[i,k]*b[k,j]
```

```
for (i=1; i<n; i++)
  for (k=1; k<n; k++)
     for (j=1; j<n; j++)
        c[i,j]+= a[i,k]*b[k,j]
```

```
a,b,c are matrices nxn
```

Equivalent programs in terms of results

Substantially different performance

Programming multicore processors

- APIs for Multicore programming:
  - OpenMP (Open Multi-Processing)
  - Intel Parallel Studio (TBB - Threading Building Blocks)
  - OpenCL, OpenACC
  - MPI

- Main challenge
  - To write **scalable** programs that:
    - Keep efficiency level as Data increases
    - Keep efficiency level as more Cores are available

Main goal of PCOM

- Scalable (resource-aware) computing
- Resources in computing
  - sets of (processor + memory + interconnection)
  - understand the trend past-present-future
  - be prepared for heterogeneity: general-purpose & attached devices
- Performance evaluation
  - **Performance** and **Efficiency** measures
  - **Scalability** analysis

Course Contents

- Introduction to Parallel Computing
- Cache memory effect on processor performance
- Shared Memory model
- Distributed Memory model
- Data Parallel model
- Parallel machines
- Computational Models
- Performance measures and Scalability analysis

Course Evaluation

Course work:

- Two assignments (60%)

Written test (40%)