561
High Performance Computing

David Malone

dwmalone@maths.tcd.ie

First Term 1999

What is HPC?

Solving problems with a computer in an acceptable amount of time.

- long running numerical simulations,
- quick problems — with real time constraints,
- examining large amounts of data,
- providing services at an acceptable rate.

Usually the big issue is scale.

CPU bound problems:

- Simulation of the weather, the galaxy or particles,
- Designing good turbines, safer cars or good portfolios,
- Finding primes, aliens and secret messages.

Real Time constrained problems:

- Routing network traffic and telephone calls,
- Controlling missiles and robots,
- Predicting weather and stocks.

Data Size problems:

- Searching databases of genetic data or web pages,
- Data mining,
- Usenet news and web proxying.

Service problems:

- Web, news and e-mail services,
- Managing databases of customers and stock,
- High availability and data safety.

Aims

This course will largely focus on CPU bound problems. We want to look at:

- hardware,
- operating systems,
- compilers,
- languages and programming.

Once we have looked at these hopefully you'll be able to:

- choose the correct tools for a given problem,
- take advantage of the tools which are available,
- tell your friends what buzzwords mean at parties.

What isn't HPC?

HPC isn't an alternative to common sense. Think First!

- Over optimisation.
- Optimizing the wrong bit.
- Should have done it by hand.
- Under optimization.
- Better algorithm.

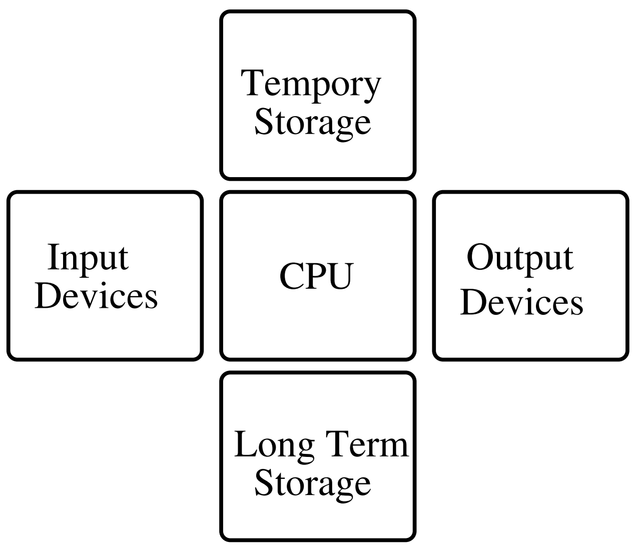
Remember Moore's Law: Computers have been doubling in speed every 18 months since the 60s.

(Should continue for next 10/20 years.)

Course Overview

1. Intro to HPC.
2. Computer Basics.
3. Processors.
4. Memory.
5. Operating systems.
6. Data representation.
7. Compilers.
8. Parallel Computing.
9. Algorithm, Data Structures and good programming.
10. Optimisation, profiling and benchmarking.

Simple View of Computer Hardware



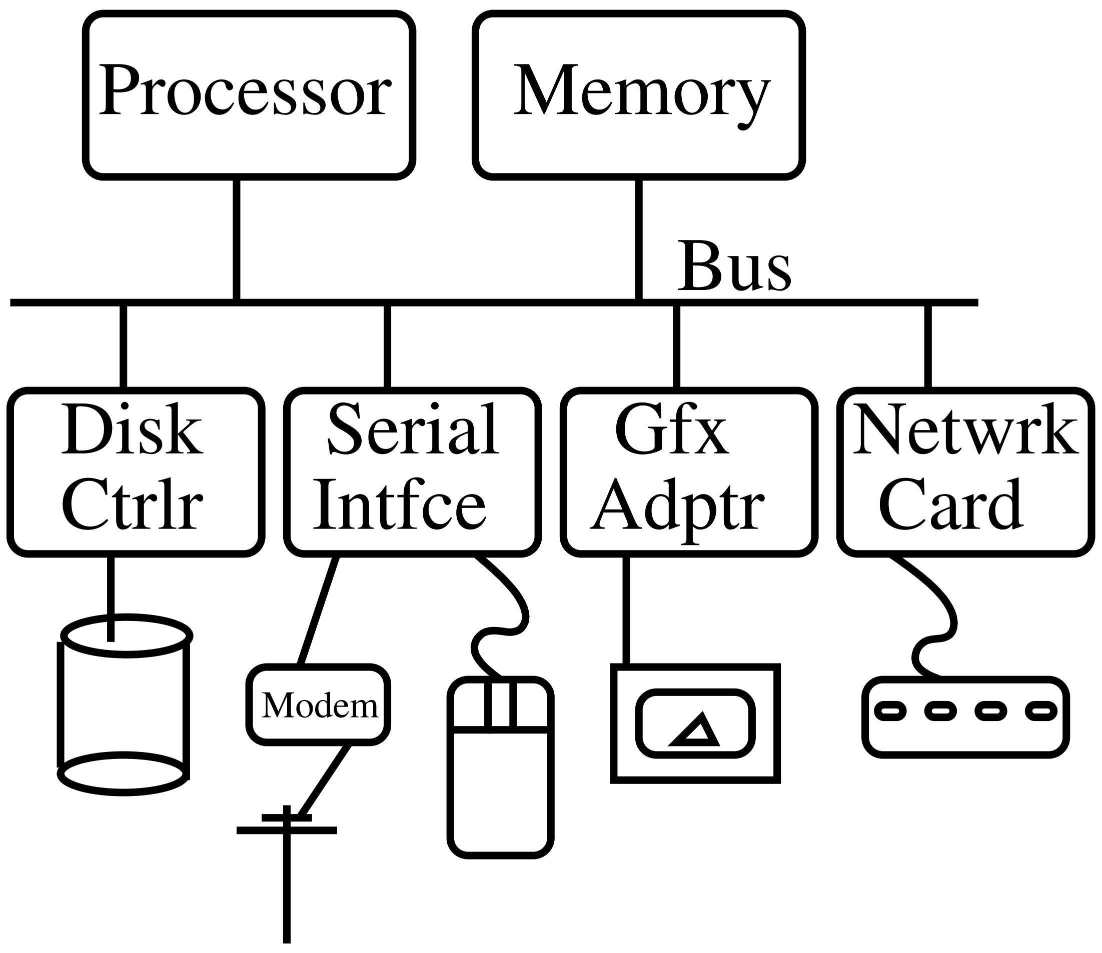
Input: Punch card, keyboard, mouse, camera, microphone

Output: Monitor, printer, speakers, robot

Tempery Storage: Ram, ‘swap’ space

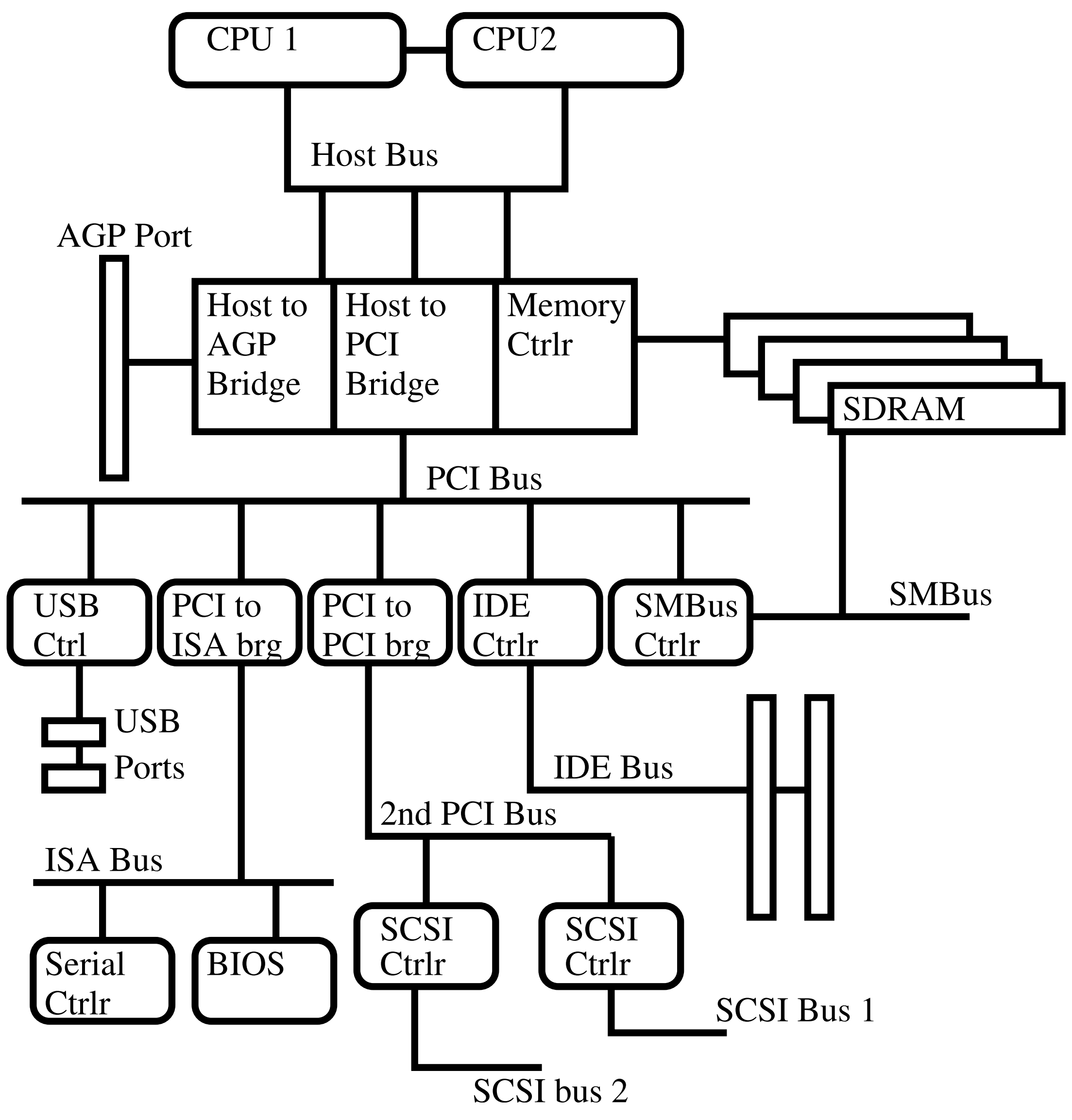
Long Term Storage: Hard disk, zip disk, CD Rom, mag tape, network

More detailed view



Usual programmers view. *Peripherals* are connected via *Bus*. CPU and peripherals may communicate via special IO instructions or memory mapping (eg. DMA). IO may be *polled* or *interrupt driven*.

Low level view



Modern PC. Many controllers will have their own processor and memory. Many busses. Several controllers on one chip.

Standard bus types

ISA Industry Standard Arch. Developed by IBM for PC. 44Mb/s.

VESA Video Electronics Standards Association. Short lived. 256Mb/s.

PCI Peripheral Component Interconnect. Developed by Intel, used in PCs and workstations 1056Mb/s.

IDE Integrated Drive Electronics. Creeping standard disks. 528Mb/s.

SCSI Small Computer Systems Interface. Connects disks, tapes, scanners and computers together. 640Mb/s.

USB Universal Serial Bus. Emerging standard for modems, scanners, removable disks etc. 12Mb/s.

Software View

Main Concerns: Procedures & Storage.

Various Levels of Procedures:

Applications: main.c poisson.f
Shared Libraries: strlen, printf, MPI_send
Top of Kernel: write, exec, getrusage
Bottom of kernel: interrupt handlers
Peripherals: disk & network controllers

Storage as an Array:

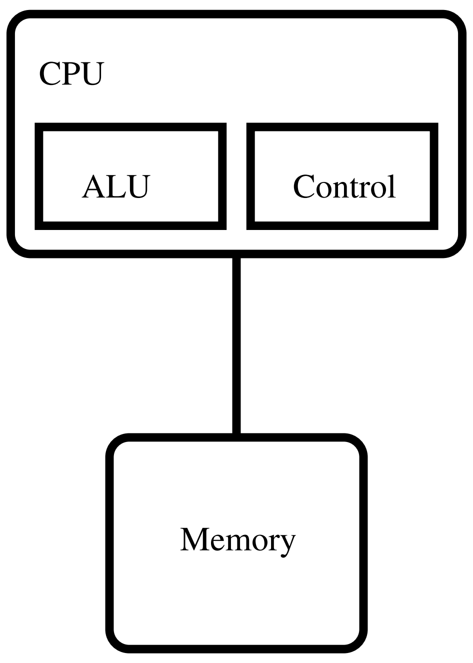
`0x14`	`0x4d`	`0xff`	`0x81`	`...`

Good for RAM, disks, tapes, . . .

Can be arranged into more complicated structures.

Storing Procedures

Most computers are *von Neuman Machines*. Program and Data stored in same memory.



Next instruction fetched from address given by program counter, which is incremented after each instruction.

Binary and Hexadecimal

Whole numbers represented as binary (base 2):

$$(1010)_2 = 1.2^3 + 0.2^2 + 1.2^1 + 0.2^0 = 10$$

Adding in binary is easy:

$$\begin{array}{r} 1010 \\ 1110 \\ \hline 11000 \end{array}$$

Long winded though: $167 = (10100111)_2$.

Hex is often used as an alternative:

2	10	16	2	10	16
0	0	0	1000	8	8
1	1	1	1001	9	9
10	2	2	1010	10	a
11	3	3	1011	11	b
100	4	4	1100	12	c
101	5	5	1101	13	d
110	6	6	1110	14	e
111	7	7	1111	15	f

Logical Operations

Not:

A	A'
0	1
1	0

And:

A	B	$A \wedge B$
0	0	0
0	1	0
1	0	0
1	1	1

Or:

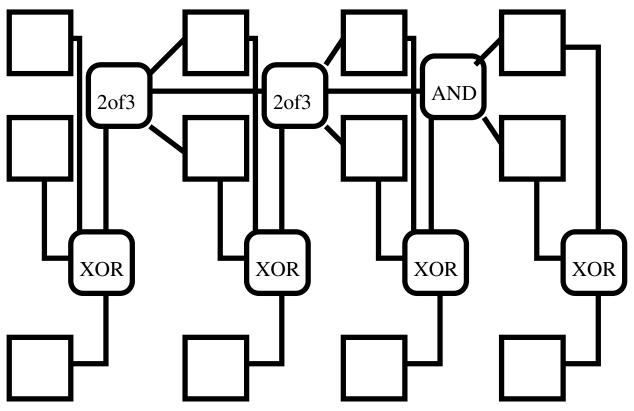
A	B	$A \vee B$
0	0	0
0	1	1
1	0	1
1	1	1

Xor:

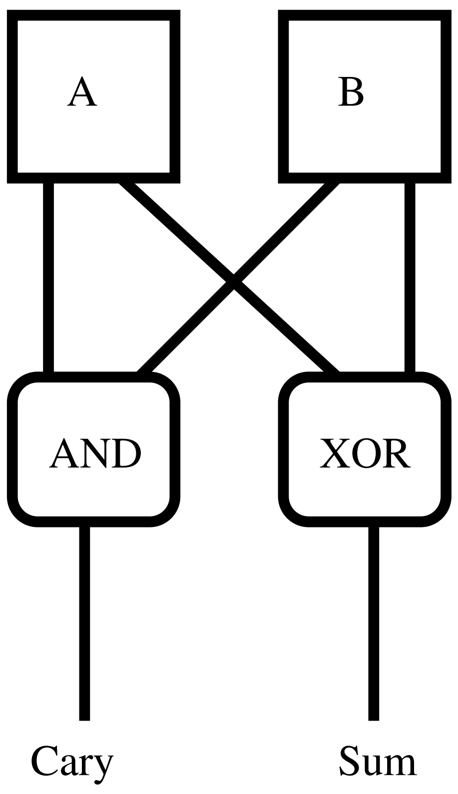
A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

Add

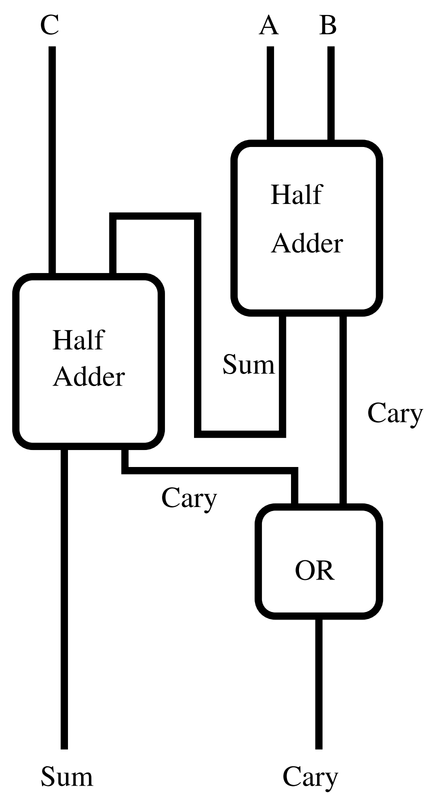
Idea:



Half adder



Full adder



Multiply

Inputs: x and y.

1. Let $r = 0$.
2. If the rightmost bit of x is set add y to r.
3. Shift x right and shift y left.
4. If x is zero stop, otherwise go to 2.

Now $r = x \times y$.

Example: $x = (101)_2$, $y = (111)_2$.

x = 101	y = 111	r = 0
x = 101	y = 111	r = 111
x = 10	y = 1110	r = 111
x = 1	y = 11100	r = 100011
x = 0	y = 111000	r = 100011

Algorithms

A finite Step by step procedure for solving a problem.

The soul of good involves choosing the right algorithms.

$O(n^2)$ means takes roughly n^2 steps if you put in n amount of data.

Libraries & languages often include implementations of good algorithms. (`qsort`, Linpac, LISP, Perl).

Bubblesort

Start with a list of n numbers.

1. Let $m = 1$.
2. Search the elements $m..n$ for the smallest number.
3. Swap the smallest number with the one in position m.
4. Add 1 to m. If $m < n - 1$ go to step 2, otherwise stop.

This algorithm is bad for large amounts of data. It takes $O(n^2/2)$ operations. Sorting can be done in $O(n \log n)$ steps.

Measuring Computers

bit Binary digIT, a single 0 or 1.

byte group of 8 bits.

second 9,192,631,770 cesium periods.

k,M,G 2^{10},2^{20},2^{30} or 10^3,10^6,10^9.

Mb/s How many megabits per second you can move.

MIPS Millions of operations per second.

FLOPS Floating point operations per second.

Other benchmarks measure other aspects of computers.

References

- "High Performance Computing, second edition", Dowd & Severance, O'Reilly & Associates.
- "An update on Moore's Law", `http://developer.intel.com/pressroom/archive/speeches/gem93097.htm`
- Play with logic gates: `http://www.brunel.ac.uk/~castjjg/hndcfund/material/logic/part5.htm`
- "The Art of Computer Programming, Vol I,II & III", Knuth, Addison-Wesley.
- "Numerical Recipies in C", Press et al,

Cambridge.

- Distributed computing on the net: `http://www.distributed.net`, `http://setiathome.ssl.berkeley.edu`, `http://www.mersenne.org`
- CPU Info center: `http://infopad.eecs.berkeley.edu/CIC/`
- Top 500 Linpack supercomputers: `http://www.netlib.org/benchmark/top500.html`

Assignment 1

Look at the list of top 500 supercomputers. Select four vendors and find out what high performance computers they offer. Look information about other benchmarks and see how these computers perform under different tests.