THE UNIVERSITY OF WARWICK

High Performance Computing

Course Notes

HPC Fundamentals

Contacts details

Dr. Ligang He

Home page: http://www.dcs.warwick.ac.uk/~liganghe

Email: ligang.he@warwick.ac.uk

Office: Room 205

Course Administration

Course Format

Monday: 1100-1200 lecture in CS104,

1200-1300 lab session in CS001 and CS003: 1) Practice the knowledge learned in lectures; 2) Gain foundation skills for completing assignments; 3) Using the Tinis cluster; 4) troubleshoot the assignments

Thursday: 1000-1100 Lecture in CS104

Assessment:

15 CATs

70% examined, 30% Assignments

2-hour final exam in Term 3

Learning Objectives

- Commonly used models (e.g., OpenMP, MPI, GPU) to write HPC applications (mainly parallel programs)

- Commonly used HPC platforms (e.g., cluster)

- The means by which to measure, analyse and predict the performance of HPC applications running on their supporting HPC platforms

- The role of administration, scheduling and data management in an HPC management software

Materials

- The slides will be made available online after each lecture

- Relevant reference books, papers and online resources will be announced throughout the course

Lab sessions

- Practising C/C++ programming
- OpenMP programming
- MPI programming

- GPU programming

- Using the Tinis Cluster
- Troubleshooting

Assignments

- Two assignments counts 30% of the final mark
  - The first assignment counts 10%
  - The second assignment counts 20%

- The first assignment involves using OpenMP to write a parallel program

- The second assignment involve the development of a parallel application using the Message Passing Interface (MPI)

- **Deadlines:**
  - Assignment 1: 12pm, Feb 5th, 2018; Assignment 2: 12pm, Mar 14th, 2018

Introduction

- **What is High Performance Computing (HPC)?**

- **Difficult to answer – it's a moving target.**
  - **Later 1980s, a supercomputer performs 100m FLOPs**
  - **Today, a typical desktop/laptop performs tens of giga Flops (e.g., i7 core is about 70 giga Flops)**
  - **Today, a supercomputer typically performs hundreds of Tera Flops**
    - Sunway Taihulight, No. 1 in Top 500 list, 93 Peta Flops - China
    - TianHe-2: No.2, 33.8 Peta Flops – China
    - Titan: No.3 in Top 500 list, 17.6 Peta Flops – US (No. 1, 2012)
    - The entry level in the Top 500 list is 548.7 Tera Flops
    - The entry level last year is 349.3 Tera Flops
    - The entry level in Nov 2012 is 76.5 Tera Flops

Note: Mega (10^6), giga (10^9), tera (10^{12}), peta (10^{15}), exa (10^{18})

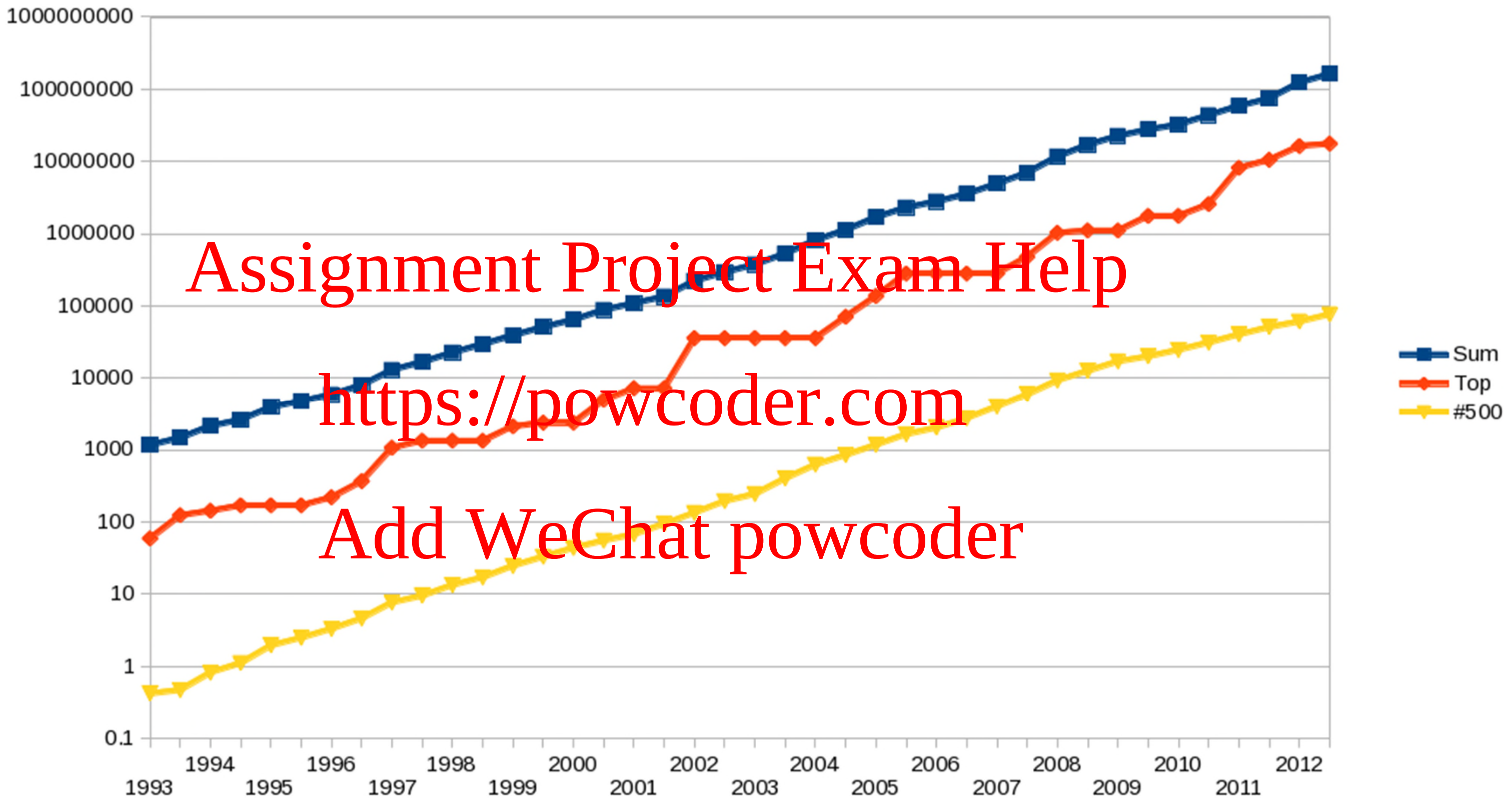
What is High Performance Computing (HPC)?

O(1000) more powerful than the latest desktops

If using i7 core as the baseline, which is about 70 giga Flops

A HPC system should have the performance of 70 tera Flops

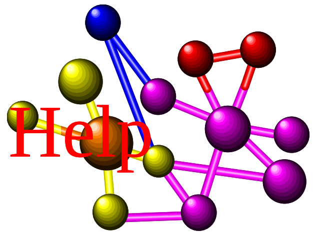
Growth of performance in Top500

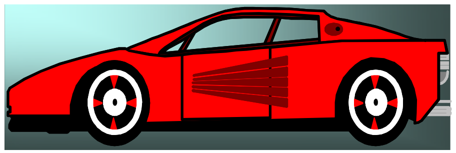


- Performance increases by ten folds every four years
- Moore’s law (double every 18 months): better or worse?

Applications of HPC

- HPC is driven by demand of computation-intensive applications from various areas
  - Weather forecast
    - Weather model captures the relation among weather parameters





Governing Equation of Weather Forecast

Momentum equations

$$\frac{\partial u}{\partial t} = -u\frac{\partial u}{\partial x} - v\frac{\partial u}{\partial y} - w\frac{\partial u}{\partial z} - \frac{1}{\rho}\frac{\partial p}{\partial x} + fv$$

$$\frac{\partial v}{\partial t} = -u\frac{\partial \rho}{\partial x} - v\frac{\partial v}{\partial y} - w\frac{\partial v}{\partial z} - \frac{1}{\rho}\frac{\partial p}{\partial y} - fu$$

$$\frac{\partial w}{\partial t} = -u\frac{\partial w}{\partial x} - v\frac{\partial w}{\partial y} - w\frac{\partial w}{\partial z} - \frac{1}{\rho}\frac{\partial p}{\partial z} - g$$

Thermodynamic equation

$$\frac{\partial \theta}{\partial t} = -u\frac{\partial \theta}{\partial x} - v\frac{\partial \theta}{\partial y} - w\frac{\partial \theta}{\partial z} + \dot{Q}$$

Mass continuity equation

$$\frac{\partial \rho}{\partial t} = -u\frac{\partial \rho}{\partial x} - v\frac{\partial \rho}{\partial y} - w\frac{\partial \rho}{\partial z} - \rho\nabla\cdot\vec{V}$$

Moisture equation

$$\frac{\partial q}{\partial t} = -u\frac{\partial q}{\partial x} - v\frac{\partial q}{\partial y} - w\frac{\partial q}{\partial z} + \mathrm{micro}(q)$$

Ideal gas law

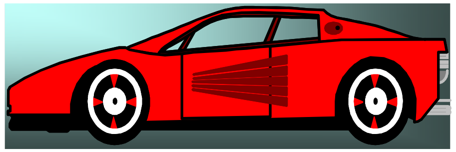
$$p = \rho RT$$

- Impossible to use math derivation to solve the equation;
- Use the numerical method

Applications of HPC

- **HPC is driven by demand of computation-intensive applications from various areas**

  - **Weather forecast**
  - **Finance (e.g. predict the trend of the stock market)**
  - **Biology, neuroscience (e.g. simulation of brains)**



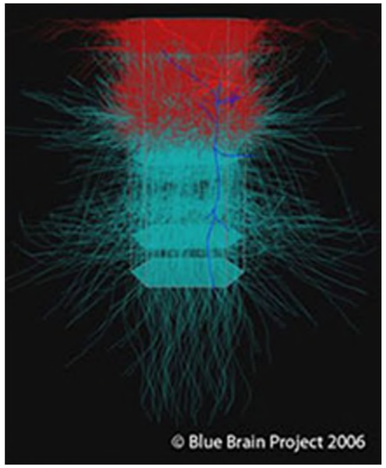
An HPC application in neuroscience

- **Project: Blue Brain**
- **Aim: construct a virtual brain**
- **Building blocks of a brain are neurocortical columns**

  - **A column consists of about 60,000 neurons, interacting with each other**

- **First step: simulate a single column (each processor acting as one neuron)**

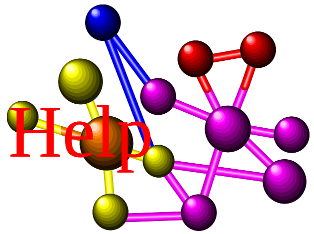
- **Then: simulate a small network of columns**
- **Ultimate goal: simulate the whole human brain**
- **Scale of the problem:**
  - **Human brain contains millions of such columns**

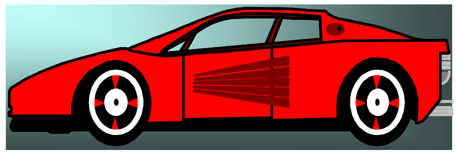


Applications of HPC

- HPC is driven by demand of computation-intensive applications from various areas

  - Weather forecast
  - Finance (e.g. modelling the trend of the stock market)
  - Biology, neuroscience (e.g. simulation of brains)
  - Engineering (e.g. simulations of a car crash)



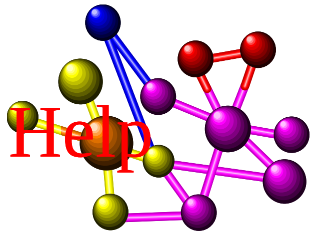



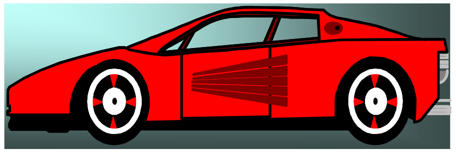
Simulation of Car Crash

http://www.youtube.com/watch?v=NrvuFiDqn5A

Applications of HPC

- HPC is driven by demand of computation-intensive applications from various areas
  - Weather forecast
  - Finance (e.g. modelling the trend of the stock market)
  - Biology, neuroscience (e.g. simulation of brains)
  - Engineering (e.g. simulations of a car crash)
  - Military and Defence (e.g. modelling explosion of nuclear bombs)



Related Technologies

- **HPC covers a wide range of technologies:**
  - **Computer architecture**
    - **CPU, memory,**
    - **VLSI: transistors**
      - increasingly difficult (density and heat)
    - **multicore**

volume 48 number 1 north american

ieee spectrum 1.11

COVER: MICHAEL SOLITA

#1 Smartphones
Finally, all pocketable gadgets have converged in a single device that goes everywhere and does everything. *By Joshua J. Romero* p. 28

#2 Social Networking
Eavesdropping on friends' private lives has never been so easy. *By Ariel Bleicher* p. 31

#3 Vo[illegible]
Say good-bye to switching circuits, hello to digital telephony. *By James Middle[illegible]* p. [illegible]

#4 LED Lighting
Solid-state lighti[illegible] only when engineers mastered the blue arts. *By Richard Stevenson* p. 38

#5 Multicore CPUs
Processors have gone from having a single core to dozens. Where will it end? *By Samuel K. Moore* p. 40

#6 Cloud Computing
Your data can now wander the globe without you. *By Sandra Upson* p. 43

#7 Drone Aircraft
Unmanned aerial vehicles have given war fighters remote eyes—and arms. *By David Schneider* p. 47

#8 Planetary Rovers
Robotic rovers are expanding our knowledge of the universe by exploring strange new worlds. *By Erico Guizzo* p. 51

#9 Flexible AC Transmission
At last, engineers can make alternating current go exactly where they want it. *By Peter Fairley* p. 55

#10 Digital Photography
When cameras abandoned film for pixels, they changed the way we communicate. *By Tekla S. Perry* p. 59

#11 Class-D Audio
Now you can annoy your neighbors at higher fidelity—and with stunning efficiency. *By Glenn Zorpette* p. 62

Related Technologies

- HPC covers a wide range of technologies:
  - Computer architecture
    - CPU, memory,
    - VLSI: transistors
      - increasingly difficult (density and heat)
    - multicore
    - GPU

Related Technologies

- **HPC covers a wide range of technologies:**
  - **Computer architecture**
  - **Networking**
    - **bandwidth, latency,**
    - **communication protocols,**
    - **Network topology**

Related Technologies

- HPC covers a wide range of technologies:
  - Computer architecture
  - Networking
  - Compilers
    - Identify inefficient implementations
    - Make use of the characteristics of the computer architecture
    - Choose suitable compiler for a certain architecture

Related Technologies

- HPC covers a wide range of technologies:
  - Computer architecture
  - Networking
  - Compilers
  - Algorithms
    - Design algorithm -> choose the language and write the program to implement it
    - Design parallel algorithm: partition the task into sub-tasks, collaboration among multiple CPUs
    - Choose the parallel programming paradigm and implement the algorithm

Related Technologies

- HPC covers a wide range of technologies:
  - Computer architecture
  - Networking
  - Compilers
  - Algorithms
  - Workload and resource manager
    - A big HPC system handles many parallel programs from different users
    - Task scheduling and resource allocation
    - metrics: system throughput, resource utilization, mean response time

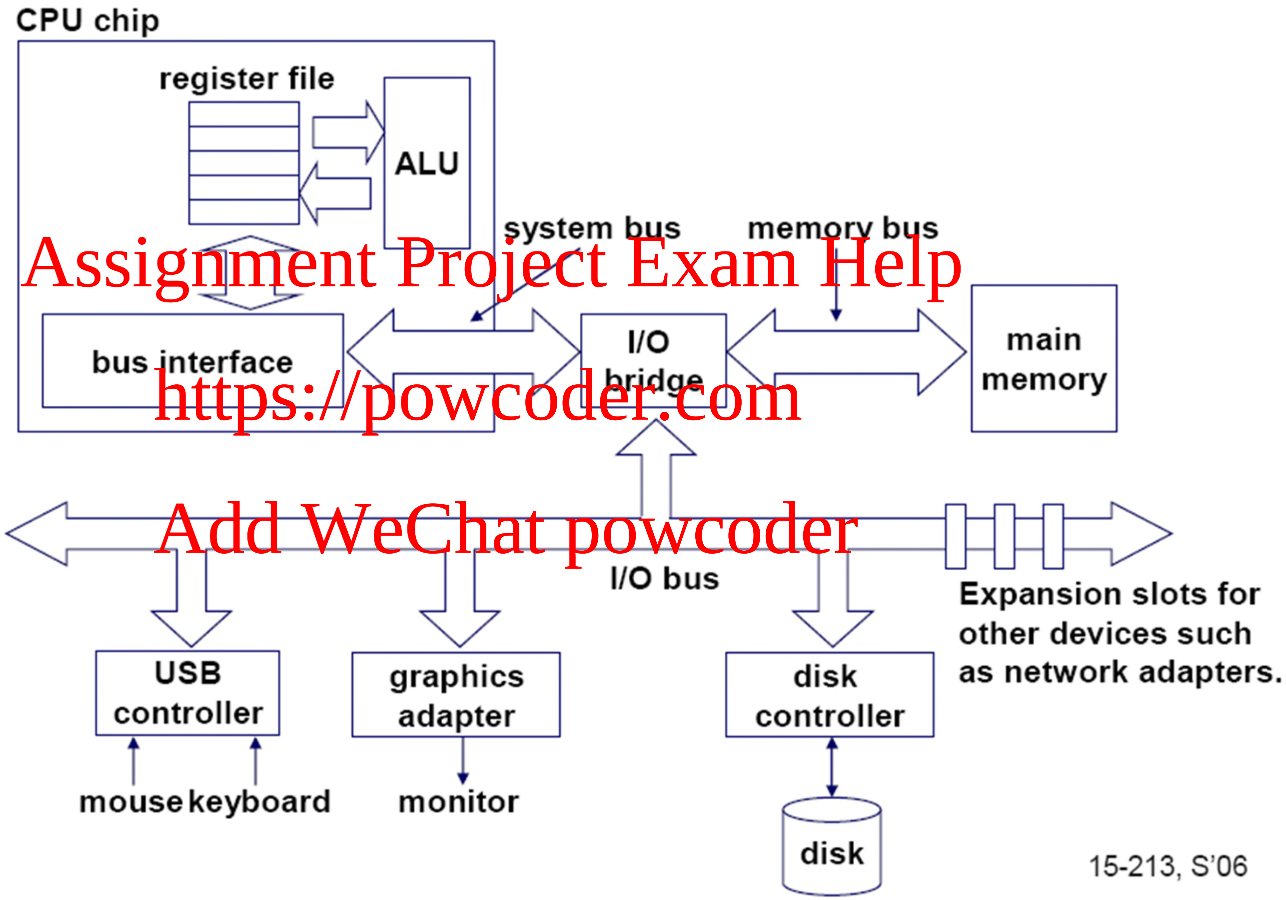
Related Technologies

- **HPC covers a wide range of technologies:**
  - **Computer architecture**
  - **Networking**
  - **Compilers**
  - **Algorithms**
  - **Workload and resource manager**

History and Evolution of HPC Systems

- 1960s: Scalar processor
  - Process one data item at a time

Scalar processor



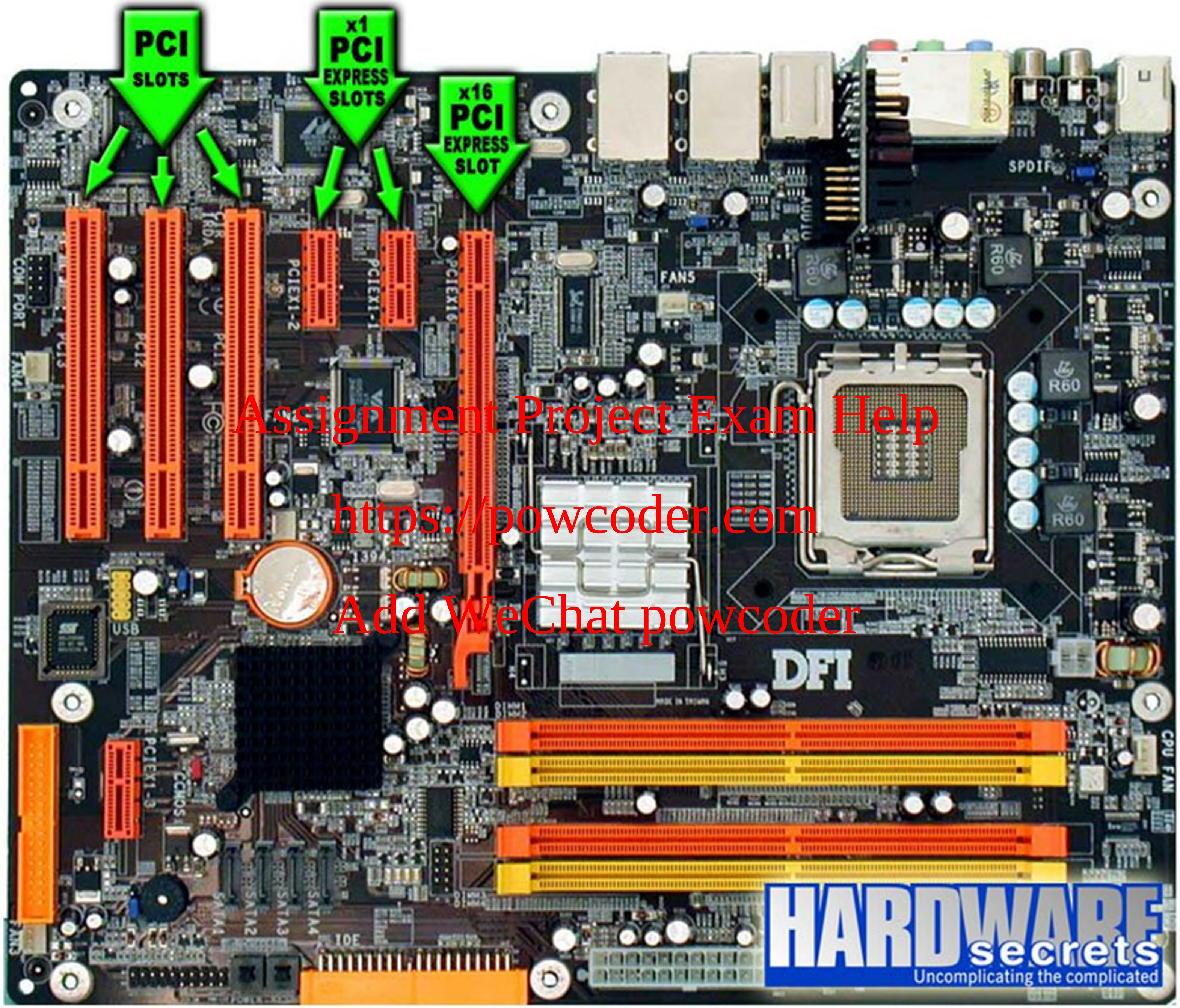
History and Evolution of HPC Systems

- 1960s: Scalar processor
- 1970s: Vector processor
  - Can process an array of data items in one go
  - Architecture: one master processor and many math co-processors (ALU)
  - Each time the master processor fetches an instruction and a vector of data items and feed them to ALUs
  - Overhead: more complicated address decoding and data fetching procedure
  - Difference between vector processor and scalar processor

GPU (Vector processor)

GPU: Graphical Processing Unit

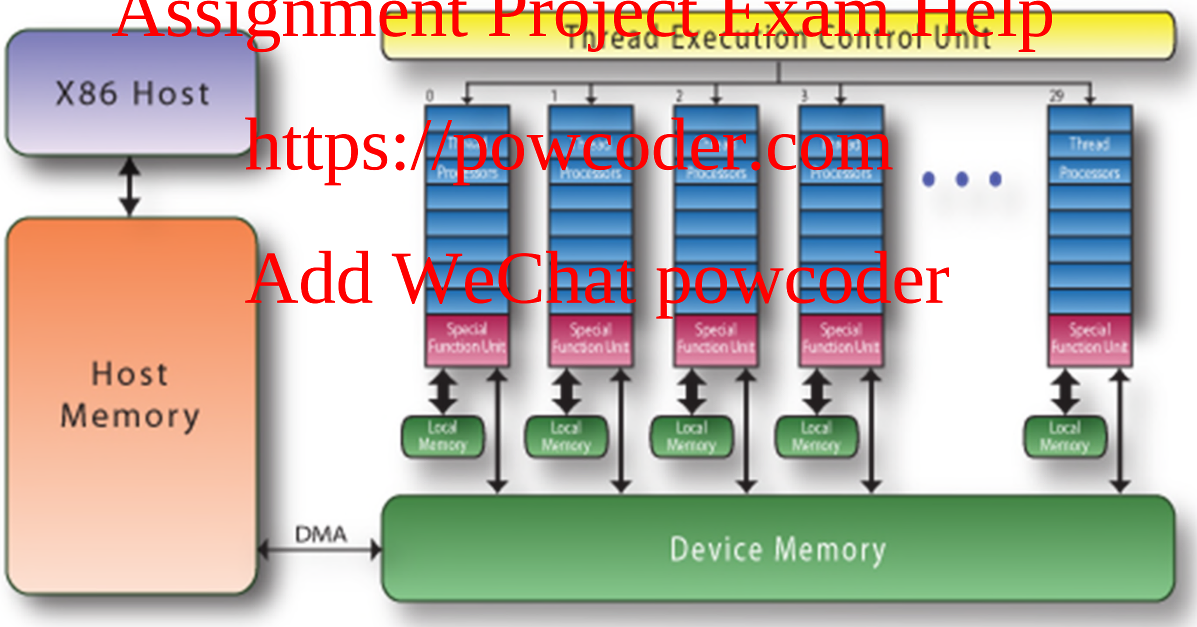
GPU is treated as a PCIe device by the main CPU


PCI
SLOTS
x1
PCI
EXPRESS
SLOTS
x16
PCI
EXPRESS
SLOT
DFI
HARDWARE
secrets
Uncomplicating the complicated

GPU (Vector processor)

GPU: Graphical Processing Unit

GPU is treated as a PCIe device by the main CPU

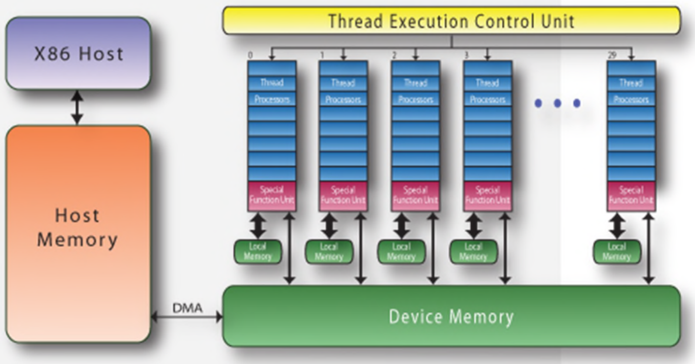


Data processing on GPU

- CUDA: programming on GPU
- Get the array A and B in one memory access operation
- Different threads process different data items
- If no much parallel processing, slower on GPU due to overhead


```
// Kernel definition
__global__ void VecAdd(float* A, float* B, float* C)
{
    int i = threadIdx.x;
    C[i] = A[i] + B[i];
}

int main()
{
    ...
    // Kernel invocation with N threads
    VecAdd<<<1, N>>>(A, B, C);
    ...
}
```

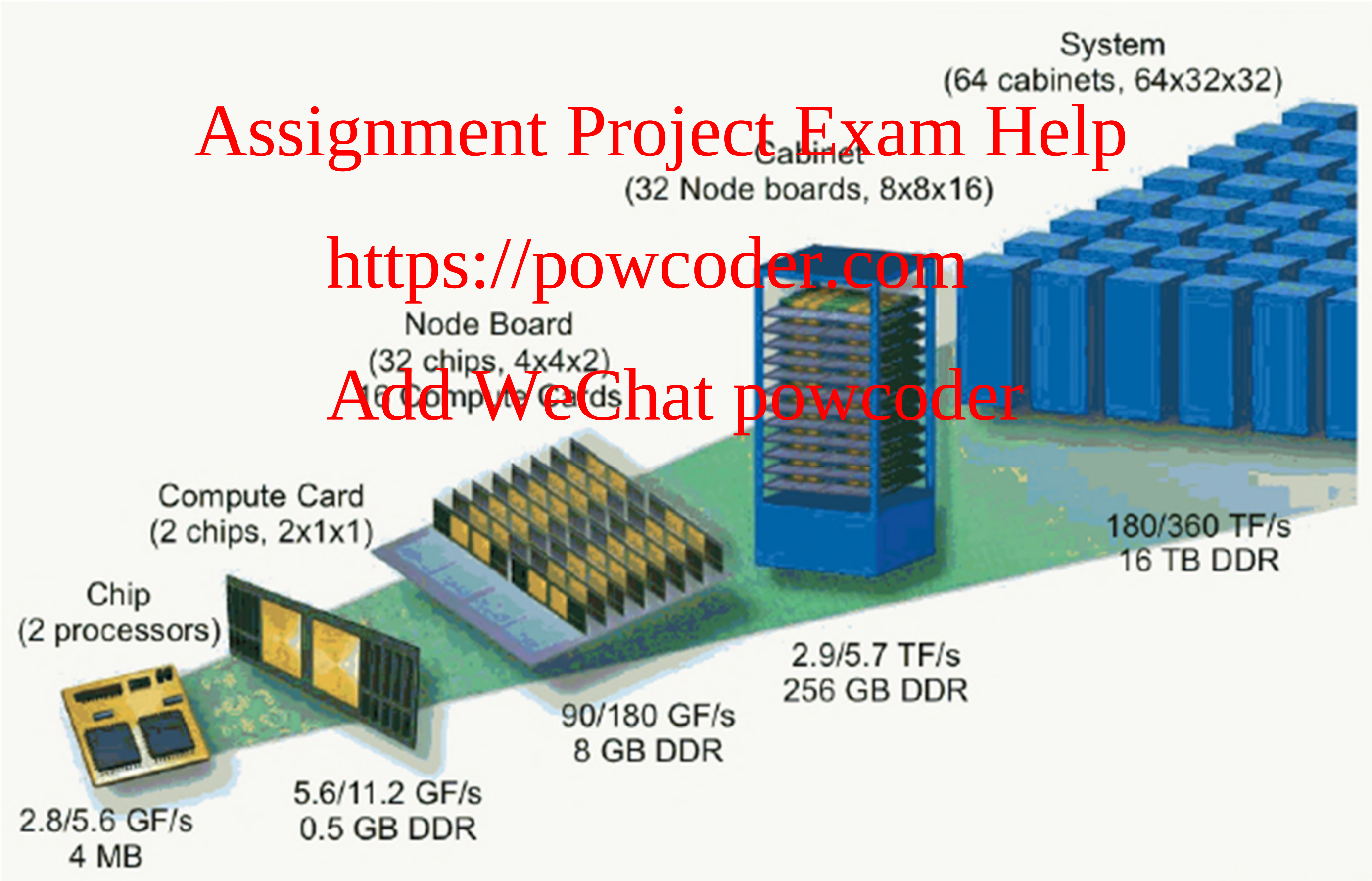


History and Evolution of HPC Systems

- 1960s: Scalar processor
- 1970s: Vector processor
- Later 1980s: Massively Parallel Processing (MPP)
  - Up to thousands of processors, each with its own memory
  - Processors can fetch and run instructions in parallel
  - Break down the workload in a parallel program
    - Workload balance and processor communications
  - Difference between MPP and vector processor

Architecture of BlueGene/L (MPP)

- Create a philosophy of using a massive number of low performance processors to construct supercomputers



Source: IBM

History and Evolution of HPC Systems

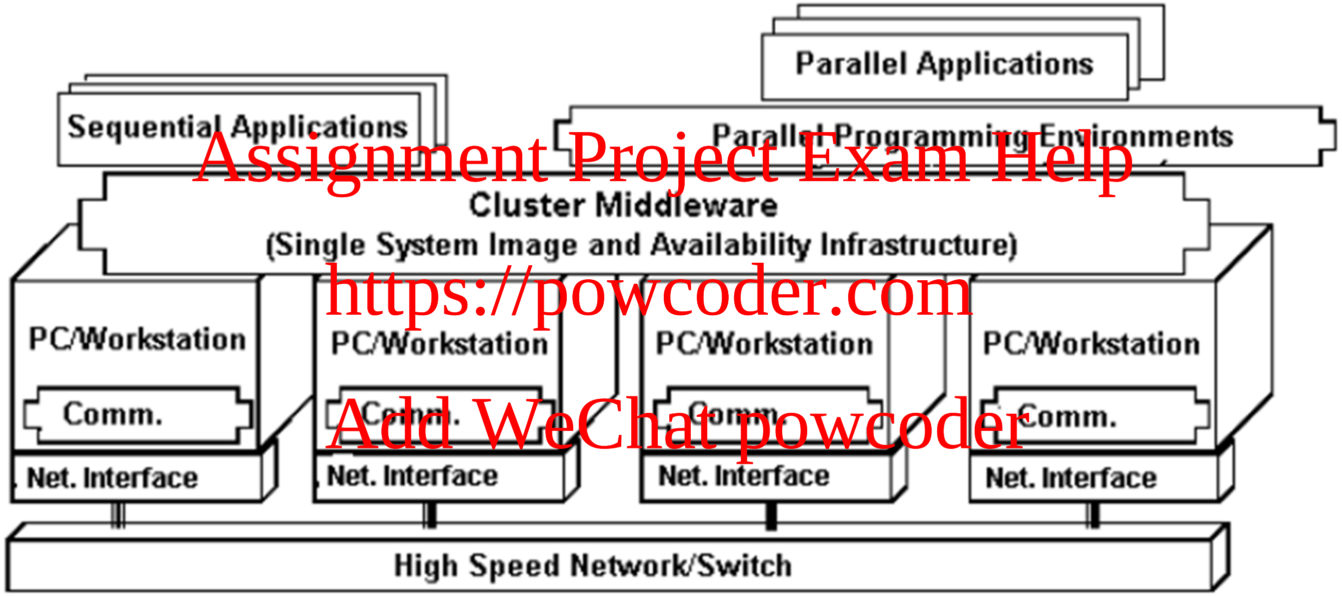
- 1960s: Scalar processor
- 1970s: Vector processor
- Later 1980s: Massively Parallel Processing (MPP)

- Later 1990s: Cluster
  - Connecting stand-alone computers with high-speed network (over-cable networks)

    - Commodity off the shelve computers

    - high-speed network: Gigabit Ethernet, infiniband
    - Over-cable network vs. on-board network
  - Not a new term itself, but renewed interests
    - Performance improvement in CPU and networking
    - Advantage over custom-designed mainframe computers: Good portability

Cluster Arechitecture



History and Evolution of HPC Systems

- 1960s: Scalar processor
- 1970s: Vector processor
- Later 1980s: Massively Parallel Processing (MPP)

- Later 1990s: Cluster

- Later 1990s: Grid
  - Integrate geographically distributed resources
  - Further evolution of cluster computing
  - Draw an analogue from Power grid

History and Evolution of HPC Systems

- 1960s: Scalar processor
- 1970s: Vector processor
- Later 1980s: Massively Parallel Processing (MPP)

- Later 1990s: Cluster

- Later 1990s: Grid

- Since 2000s: Multicore computing
  - Release the pressure of further increasing the transistor density
  - Multiple cores reside on one CPU chip (processor)
  - There can be multiple CPU chips (processors) in one computer
  - Multicore computers are often interconnected to form a cluster
    - On-board communication and over-cable communication

Architecture Types

- All previous HPC systems can be divided into two architecture types
  - Shared memory system
  - Distributed memory system

Architecture Types

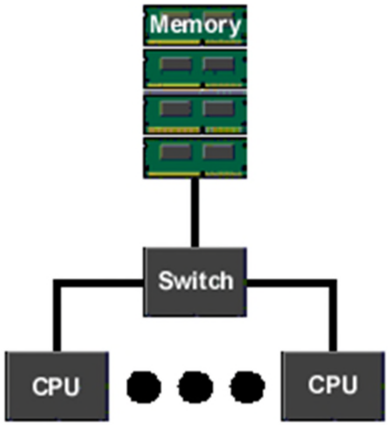
- Shared memory (uniform memory access - SMP)
  - Multiple CPU cores, single memory, shared I/O (Multicore CPU)
  - All resources in a SMP machine are equally available to each core
  - Due to resource contention, uniform memory access systems do not scale well
  - CPU cores share access to a common memory space.
    - Implemented over a shared system bus or switch
  - Support for critical sections is required
  - Local cache is critical:
    - If not, bus/switch contention (or network traffic) reduces the systems efficiency.
    - Cache introduces problems of coherency (ensuring that stale cache lines are invalidated when other processors alter shared memory).

Architecture Types

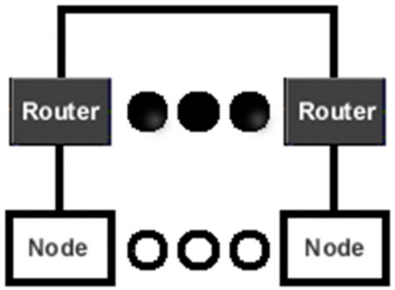
- **Shared memory (Non-Uniform Memory Access: NUMA)**
  - Multiple CPUs
  - Each CPU has fast access to its local area of the memory, but slower access to other areas

  - Scale well to a large number of processors due to the hierarchical memory access

  - Complicated memory access pattern: local and remote memory address
  - Global address space.



Node of a NUMA machine



NUMA machine

Architecture Types

- **Distributed Memory (MPP, cluster)**
  - Each processor has it's own independent memory
  - Interconnected through over-cable networks
  - When processors need to exchange (or share data), they must do this through an explicit communication
    - Message passing (MPI language)
  - Typically larger latencies between processors
  - Scalability is good if the task to be computed can be divided properly

Parallel computing vs. distributed computing

Parallel Computing

- Breaking the problem to be computed into parts that can be run simultaneously in different processors
- Example: an MPI program to perform matrix multiplication
- Solve tightly coupled problems

Distributed Computing

- Parts of the work to be computed are computed in different places (Note: does not necessarily imply simultaneous processing)
- An example: running a workflow in a Grid
- Solve loosely-coupled problems (no much communication)

Lab session today – Practising C/C++

- Write a "Hello World" program
- Calculate factorials
- Work with pointers
- Allocating memory
- Classes in C++
- Use gdb for debugging

Download the lab session sheet today from this link:
https://warwick.ac.uk/fac/sci/dcs/teaching/material/cs402/cs402_seminar1-C.pdf

Lets move down to Lab CS001 and CS003 now!