Murdoch
UNIVERSITY

Topic Title 7: Parallelism and Performance Evaluation

ICT170: Foundations of Computer Systems

Overview

- Background
- Parallel Computer Architectures
- On-chip Parallelism
- Co-Processors
- Multiprocessors
- Multicomputers
- Message-passing Multicomputers
- Cluster Computing
- Taxonomy of Parallel Computing
- Grid and Cloud Computing
- Performance Evaluation

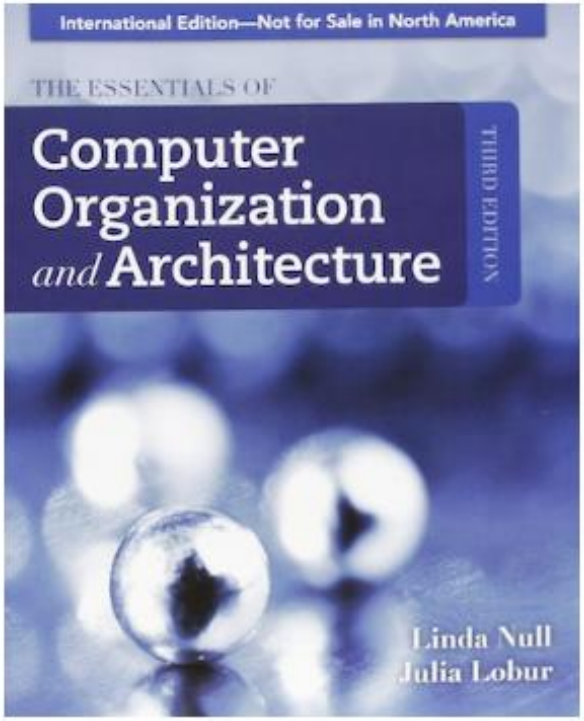
Objectives

In order to achieve the unit learning objectives, on successful completion of this topic, you should be able to:

- Explain how the components of system architecture contribute to improving its performance.
- For a given program, distinguish between its sequential and parallel execution, and the performance implications thereof.
- Explain other uses of parallelism, such as for reliability/redundancy of execution.
- Define the differences between the concepts of Instruction Parallelism, Data Parallelism, Thread Parallelism/Multitasking, Task/Request Parallelism.
- Articulate the concept of strong vs. weak scaling, i.e., how performance is affected by scale of problem vs. scale of resources to solve the problem.
- Articulate the differences between single thread vs. multiple thread, single server vs. multiple server models, motivated by real world examples

Reading

Title: **The Essentials of Computer Organization and Architecture (3rd Edition)**
Author: Linda Null, Julia Lobur,
Publisher: Jones & Bartlett Learning
Keywords: architecture, organization, computer, essentials
Pages: 880
Published: 2010-12-17
Language: English
Category: Design & Architecture, Hardware, Computers & Internet,
ISBN-10: 1449600069 ISBN-13: 9781449600068
Binding: Hardcover (3)
Reading: Chapter 11 "Performance Measurement and Analysis"

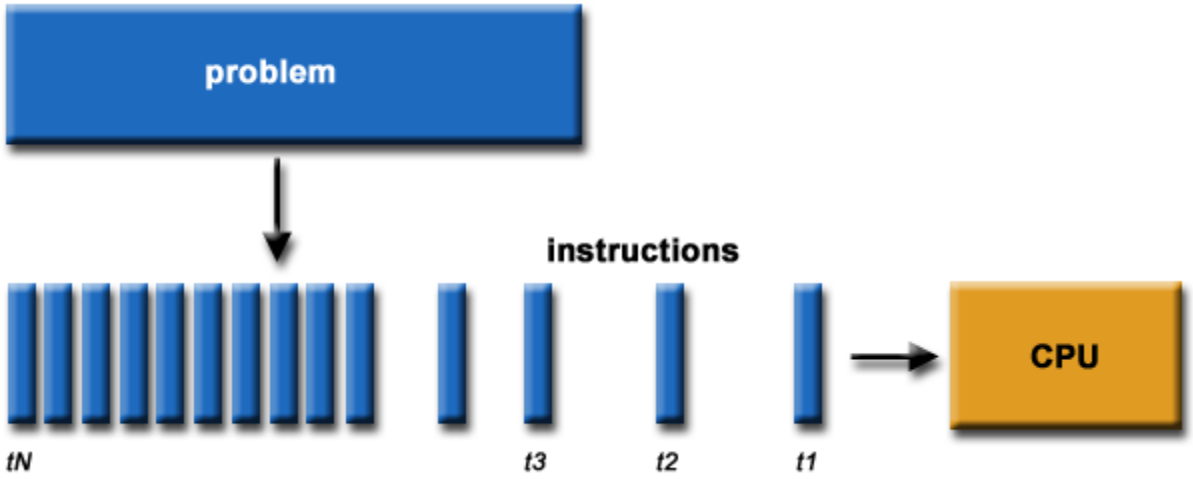


Murdoch
UNIVERSITY

Parallelism and Performance Evaluation

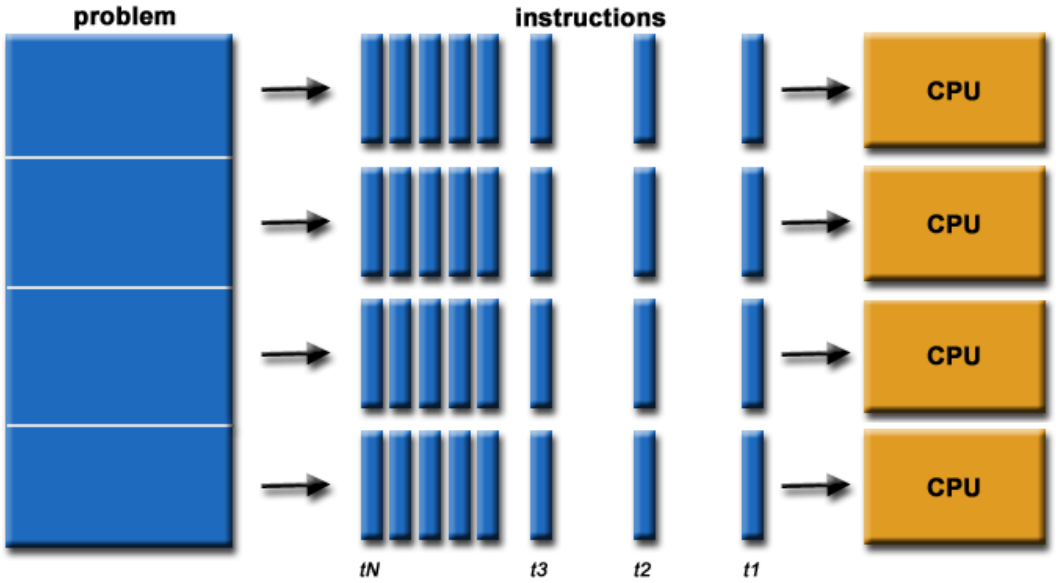
Mu
UNIV

What is Parallel Computing?



- Traditionally, software has been written for ***serial*** computation:
  - To be run on a single computer having a single Central Processing Unit (CPU);
  - A problem is broken into a discrete series of instructions.
  - Instructions are executed one after another.
  - Only one instruction may execute at any moment in time.

What is Parallel Computing?



- In the simplest sense, ***parallel computing*** is the simultaneous use of multiple compute resources to solve a computational problem.
  - To be run using multiple CPUs
  - A problem is broken into discrete parts that can be solved concurrently
  - Each part is further broken down to a series of instructions
- Instructions from each part execute simultaneously on different CPUs

Murdoch
UNIVERSITY

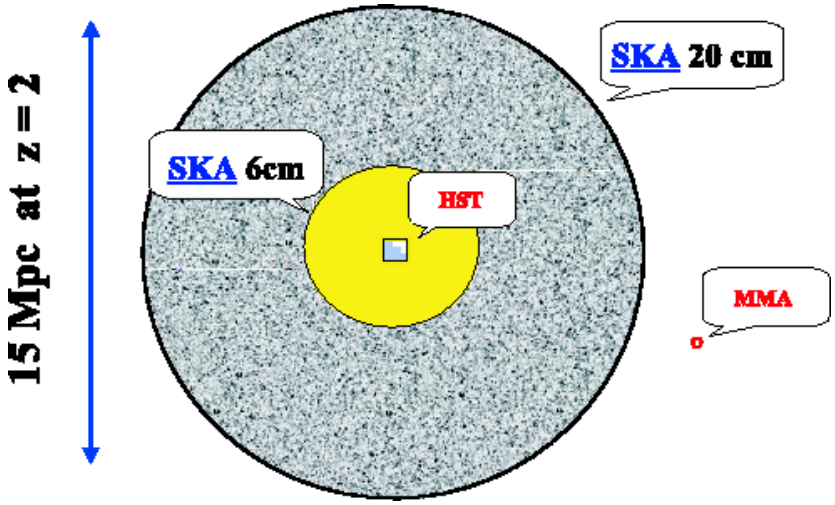
Background

Parallel Computer Architectures: Overview

- Computers are getting faster and faster
- But there will always be a demand for more power:
  - Astronomers building mosaics of the sky
  - Aircraft designers simulating designs
  - Physicists processing data in real-time – LHC
  - Biologists search for drugs
- Anything of any complexity will always push the required computing infrastructure!
- So there is always a desire for more power!

Example: SKA

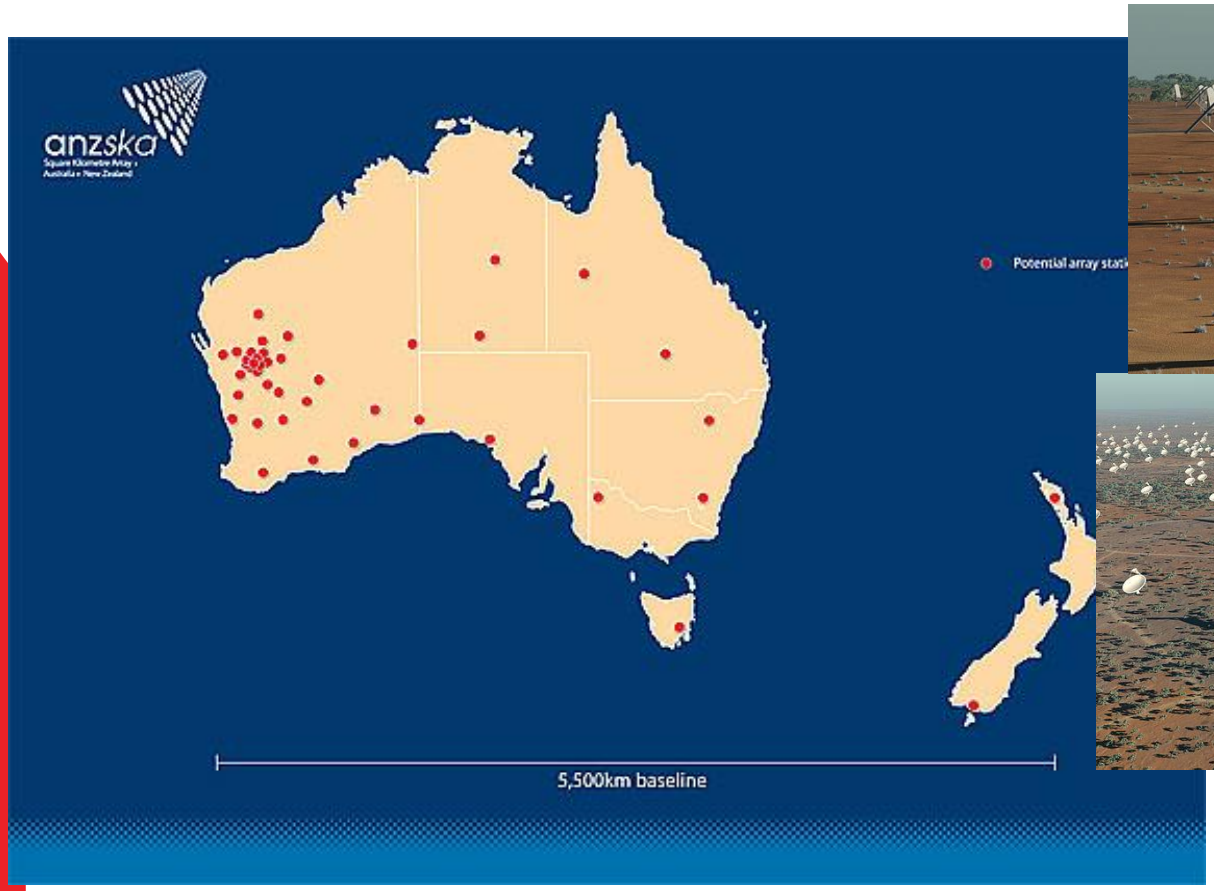
- Square Kilometer array
- Radio telescope with the collecting area of a square kilometre
- Will analyze this data and look for:
  - Galaxies
  - Dark matter
  - Dark energy
  - Life



- A ton more data than Hubble Space Telescope HST or Large Hadron Collider LHC

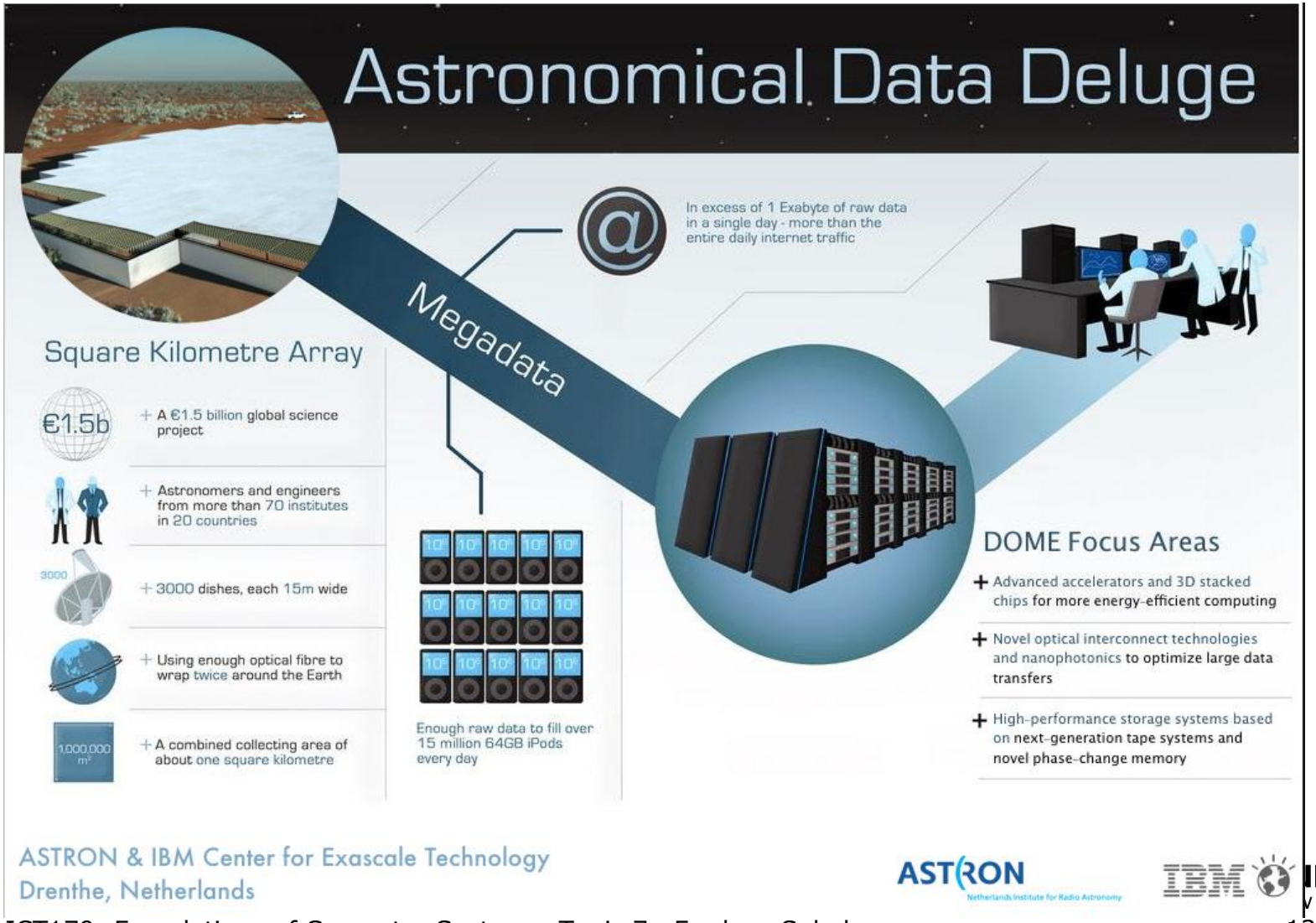
Example: SKA

- At least some of it will be built here…



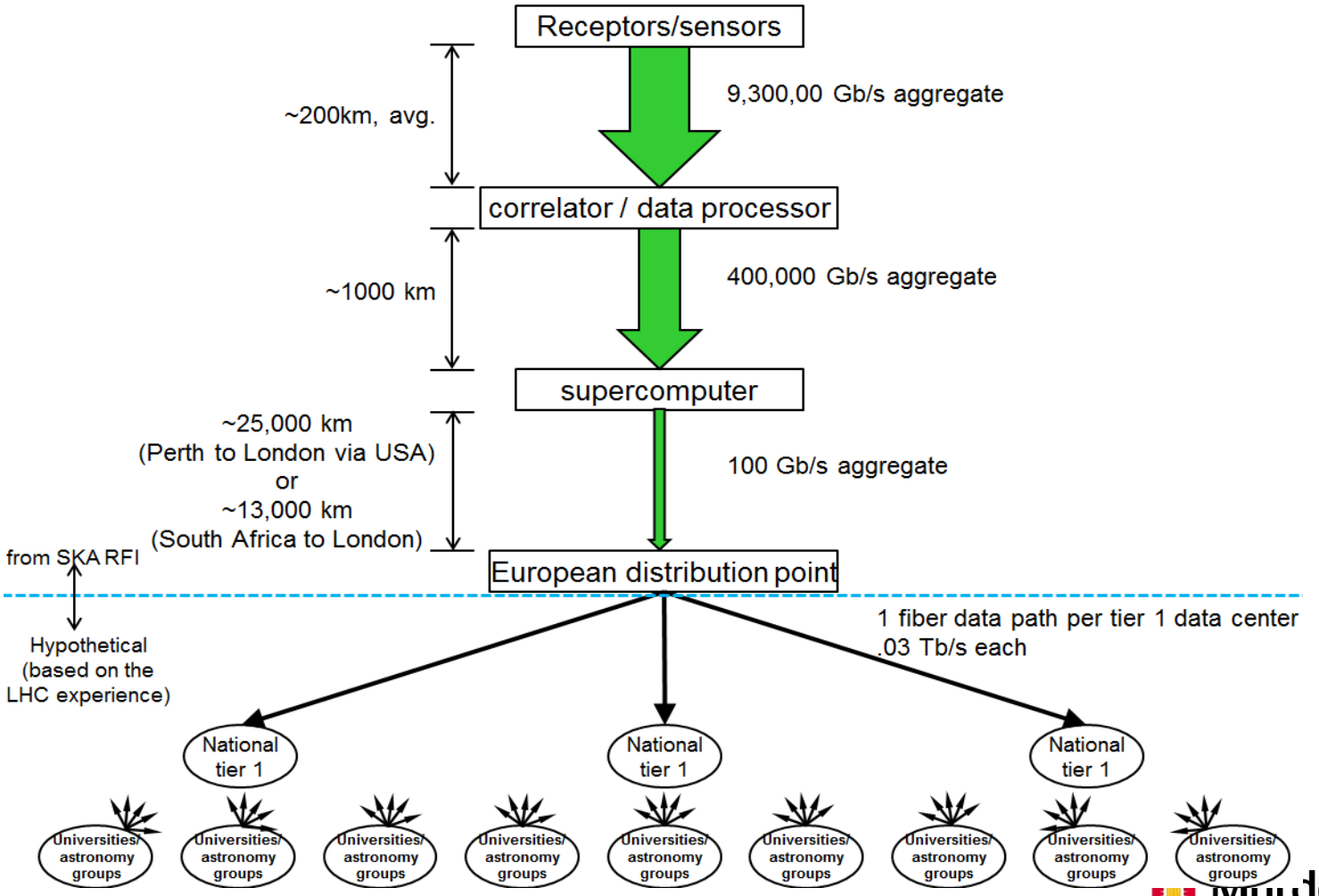
Example: SKA

Computational issues…



Example: SKA

Computational issues…



Murdoch
UNIVERSITY

Parallel Computer Architectures

Parallel Computer Architectures: Overview

- Up until recently, more power = more CPU cycles per second.
- Clock cycles are still increasing, but there are limits to this:
  - Limits to serial computing - both physical and practical reasons pose significant constraints to simply building ever faster serial computers.
  - Transmission speeds - the speed of a serial computer is directly dependent upon how fast data can move through hardware. Absolute limits are the speed of light (30 cm/nanosecond) and the transmission limit of copper wire (9 cm/nanosecond). Increasing speeds necessitate increasing proximity of processing elements.
  - Limits to miniaturization - processor technology is allowing an increasing number of transistors to be placed on a chip. However, even with molecular or atomic-level components, a limit will be reached on how small components can be.
  - Economic limitations - it is increasingly expensive to make a single processor faster. Using a larger number of moderately fast commodity processors to achieve the same (or better) performance is less expensive.
- There are future promises, quantum or organic computing
- We must deal with the problems now as these are a long way off!

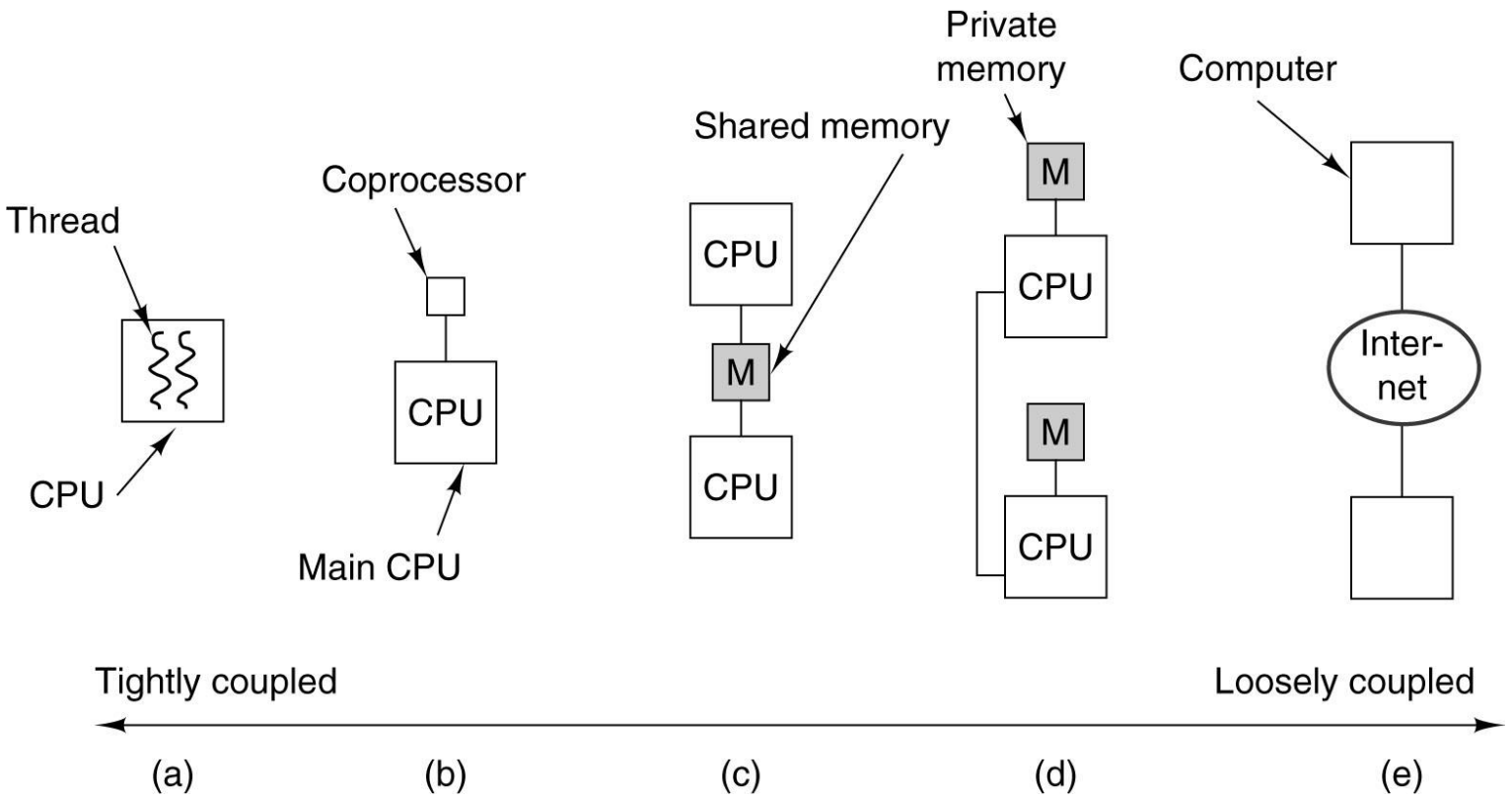
Parallel Computer Architectures: Overview

- Solution: Do computation in parallel
  - Instead of building 1 CPU with a cycle time of 0.001nsec
  - Build 1000 CPU with a cycle time of 1nsec
  - Achieves the same goal for 1000 operations!
  - Computation is theoretically the same!

Parallel Computer Architectures: Overview

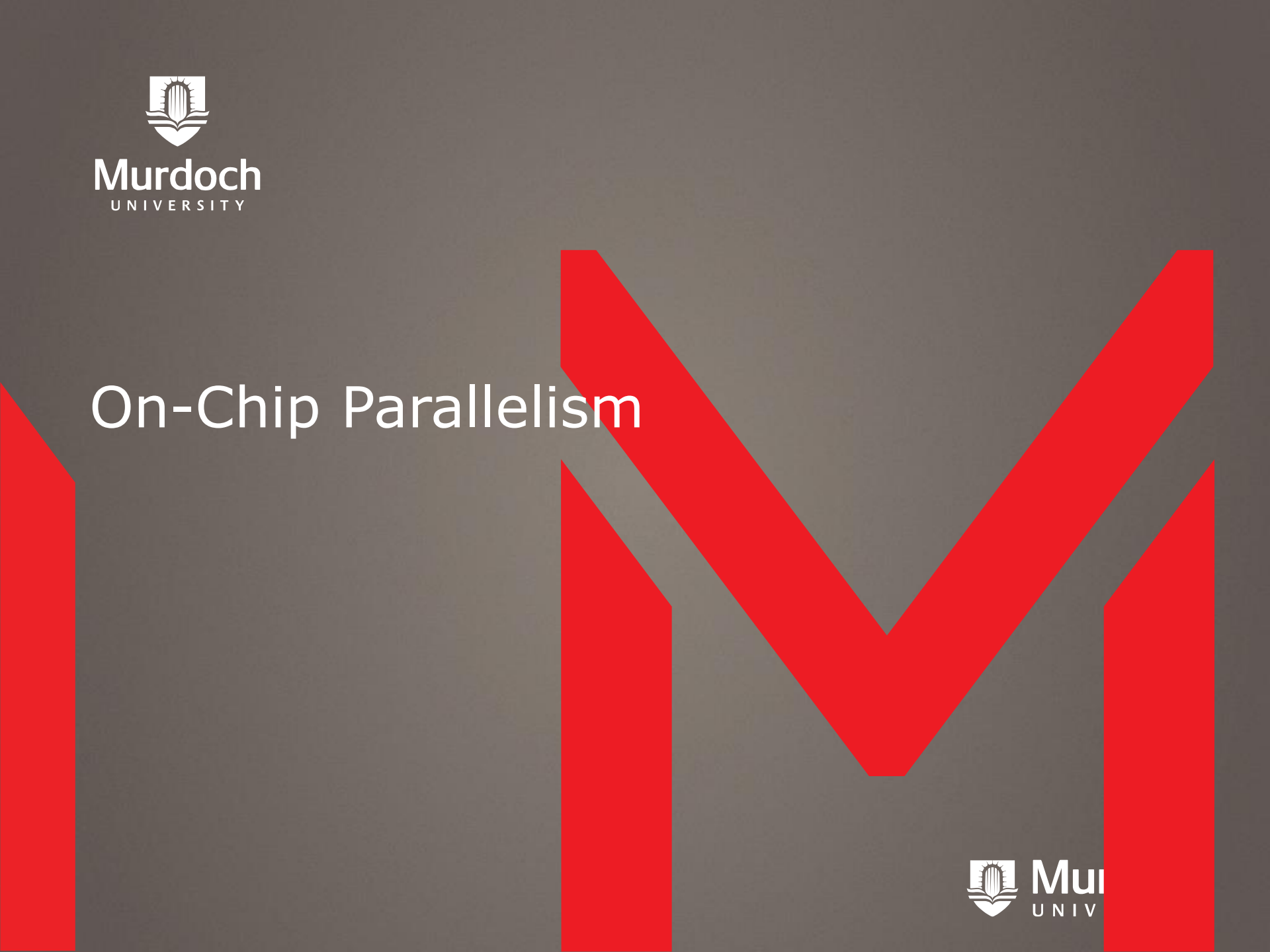
- Range of solutions to introducing parallelism
  - At the CPU Level
    - Pipelining, Superscalar designs, multiple functional units
  - Adding additional CPUs – crypto boards, graphics processors
  - Replicating entire CPUs
  - Utilising the power of many complete computers!
- When CPUs or processing elements are close together – tightly coupled
- When CPUs or processing elements are remote – loosely coupled

Parallel Computer Architectures: Range of Options



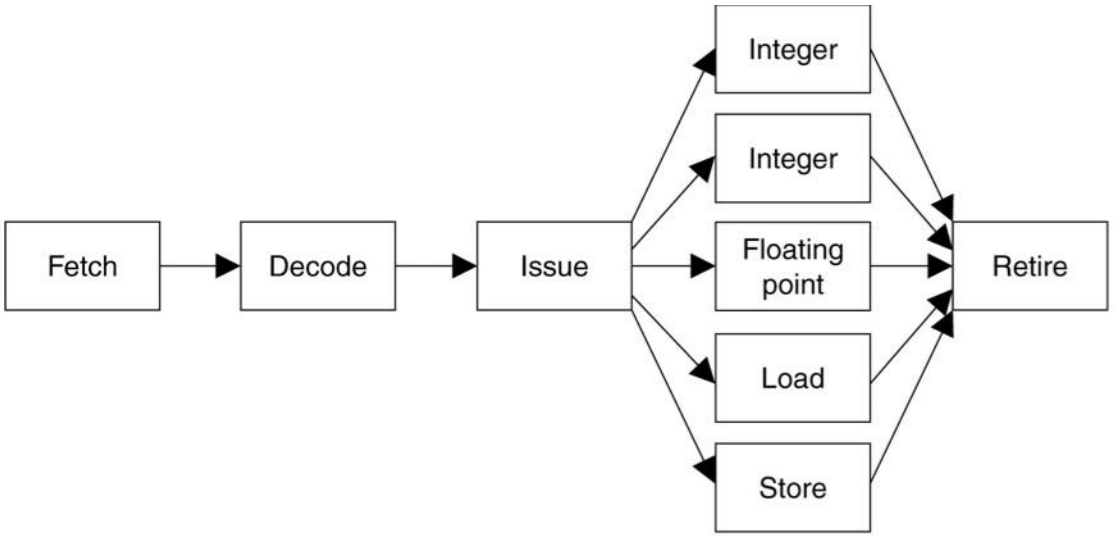
(a) On-chip parallelism. (b) A coprocessor.

(c) A multiprocessor. (d) A multicomputer. (e) A cluster/grid


Murdoch
UNIVERSITY
On-Chip Parallelism

Instruction-Level Parallelism (1)

- One way to achieve parallelism is to issue multiple instructions per clock cycle
- The hardware instruction pipeline determines the number of instructions that can be executed – in 1 clock cycle
- We've seen this previously – intelligence is in the microprogram



Instruction-Level Parallelism (2)

- Another way to manage this statically
  - Using Very Long Instruction Words (VLIW)
  - more simple microprogram
- Directly define all the parallel instructions to be executed in one word – multiple instructions
- i.e. int, float, load, store – all in one instruction
- Bundled together so can be read in sequence without lots of nops

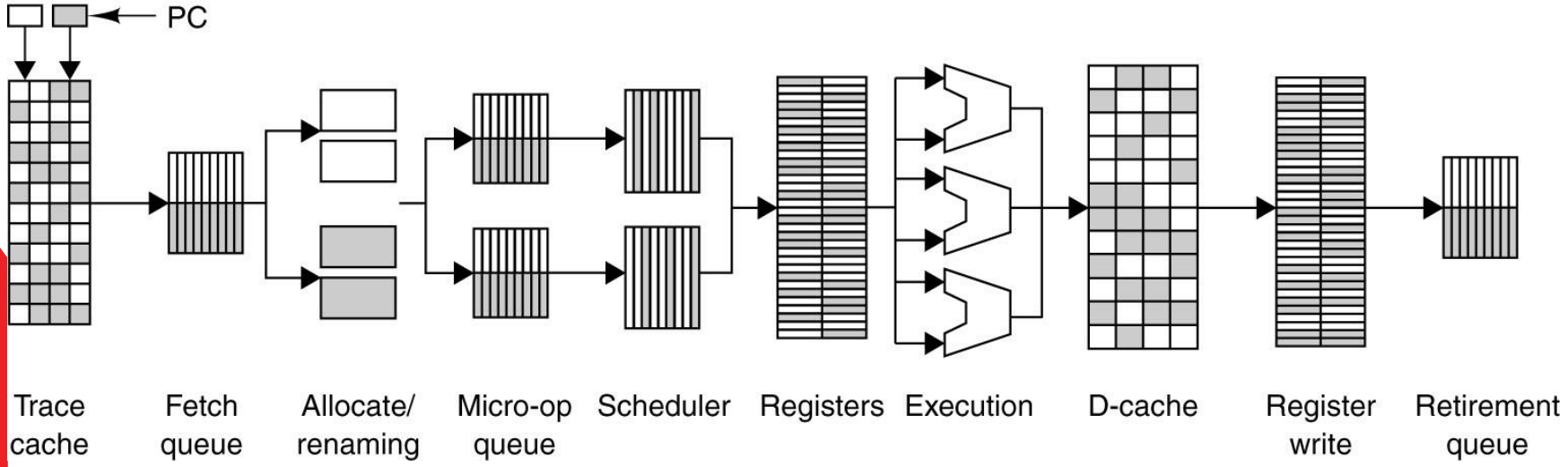
On-Chip Multithreading

- Problem with instruction-level parallelism
  - When the pipeline is refreshed or an instruction i.e. the fetch-execute cycle stalls – the whole pipeline grinds to a halt!
- So we give CPUs the ability to handle multiple threads.
- This means that if *thread 1* is blocked the CPU can continue to run *thread 2*

- *Example is hyperthreading on the Pentium 4+*
- *Core and i5/i7 NetBurst (same thing)*

On-Chip Multithreading

Hyperthreading on the Pentium 4

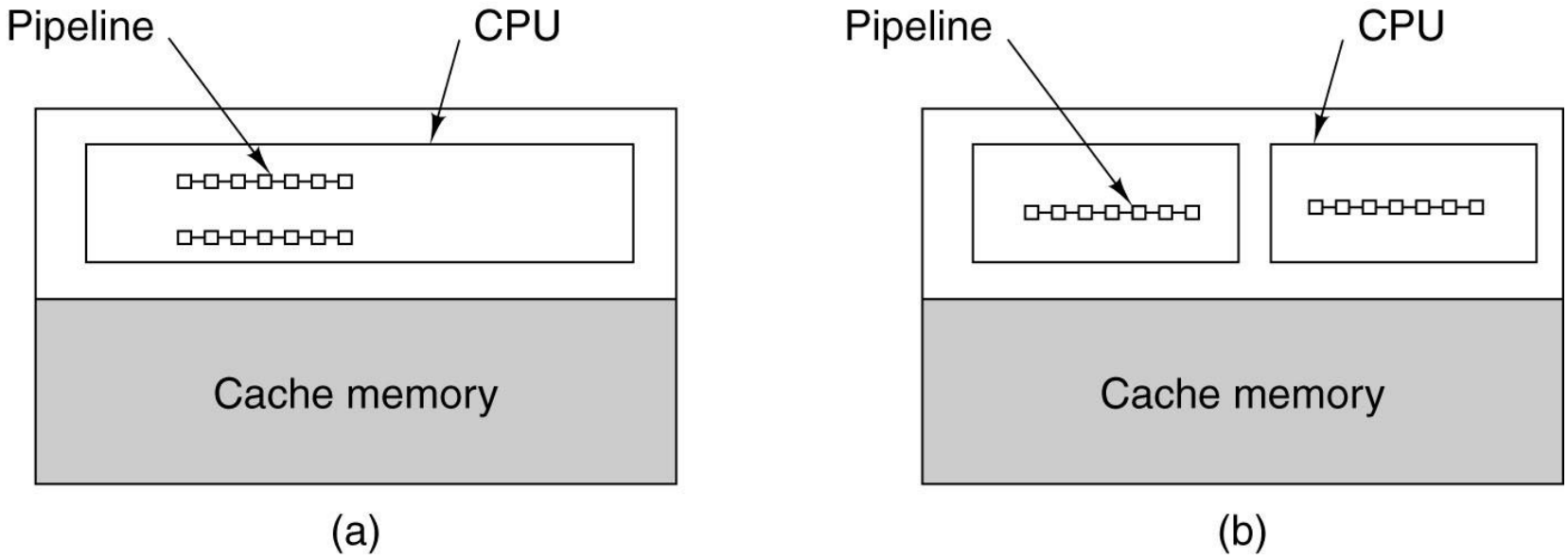


Resource sharing between threads in the Pentium 4 NetBurst microarchitecture.

Homogeneous Multiprocessors on a Chip (1/2)

- With advances in very large scale integration (VLSI) technology and chip manufacturing
  - Possible to put multiple CPUs in a chip
- Share level 1 and 2 cache, main memory.
- Share interconnects, disks, NICs.
- Share coprocessors and functional units
- Generally for custom applications!

Homogeneous Multiprocessors on a Chip (2/2)



Single-chip multiprocessors.

(a) A dual-pipeline chip. (b) A chip with two cores.

Murdoch
UNIVERSITY

Co-Processors

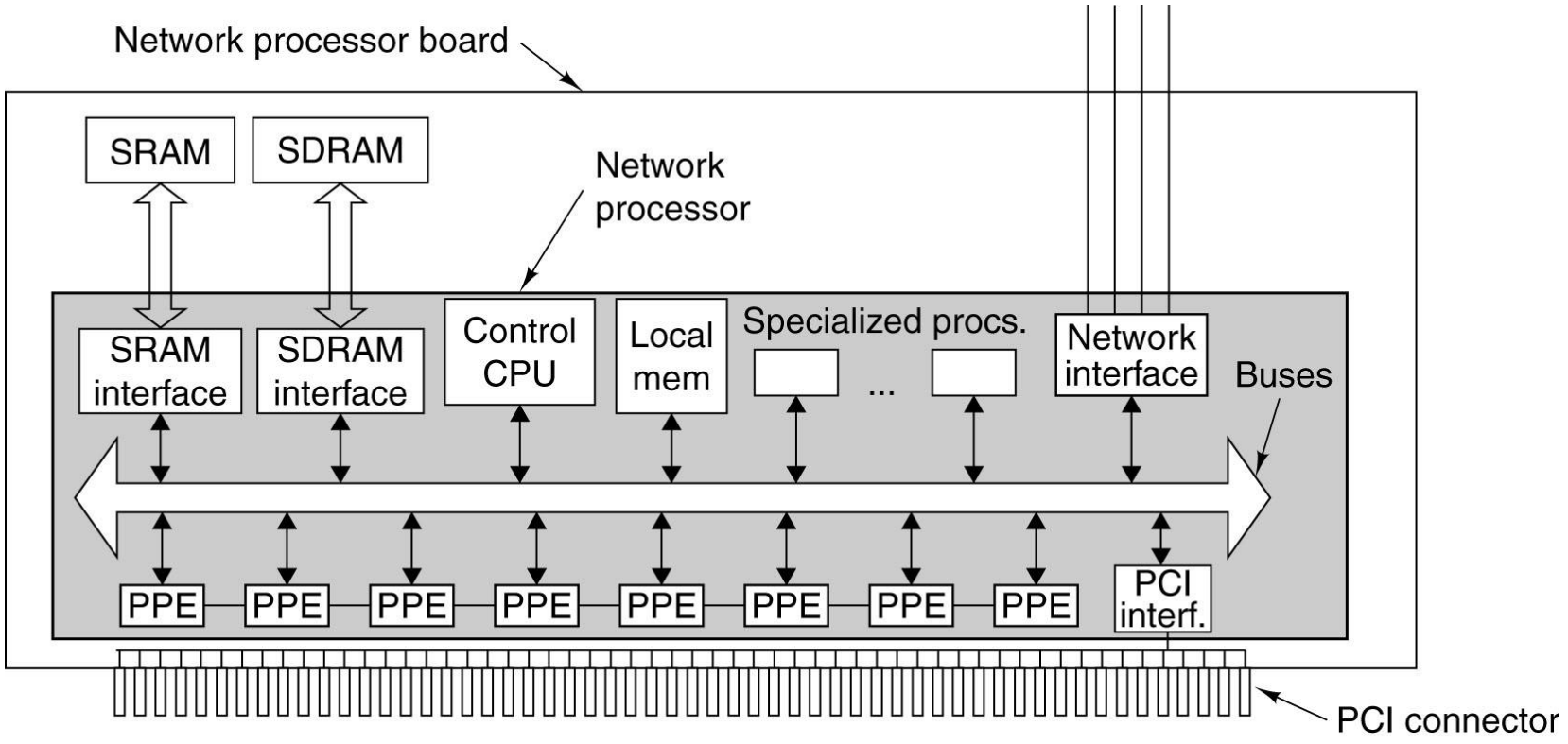
Subtitle if required

Co-Processors (1/3)

- As well as the general CPU style we concentrated on – what if we only need one task doing well
- We use a dedicated specialised Processor
  - Called a Co-Processor!
  - As they are specialised – they are often much cheaper and a lot better then general CPUS
- Examples include network processors, GPUs, media processors, FPU processors, Crypto engines

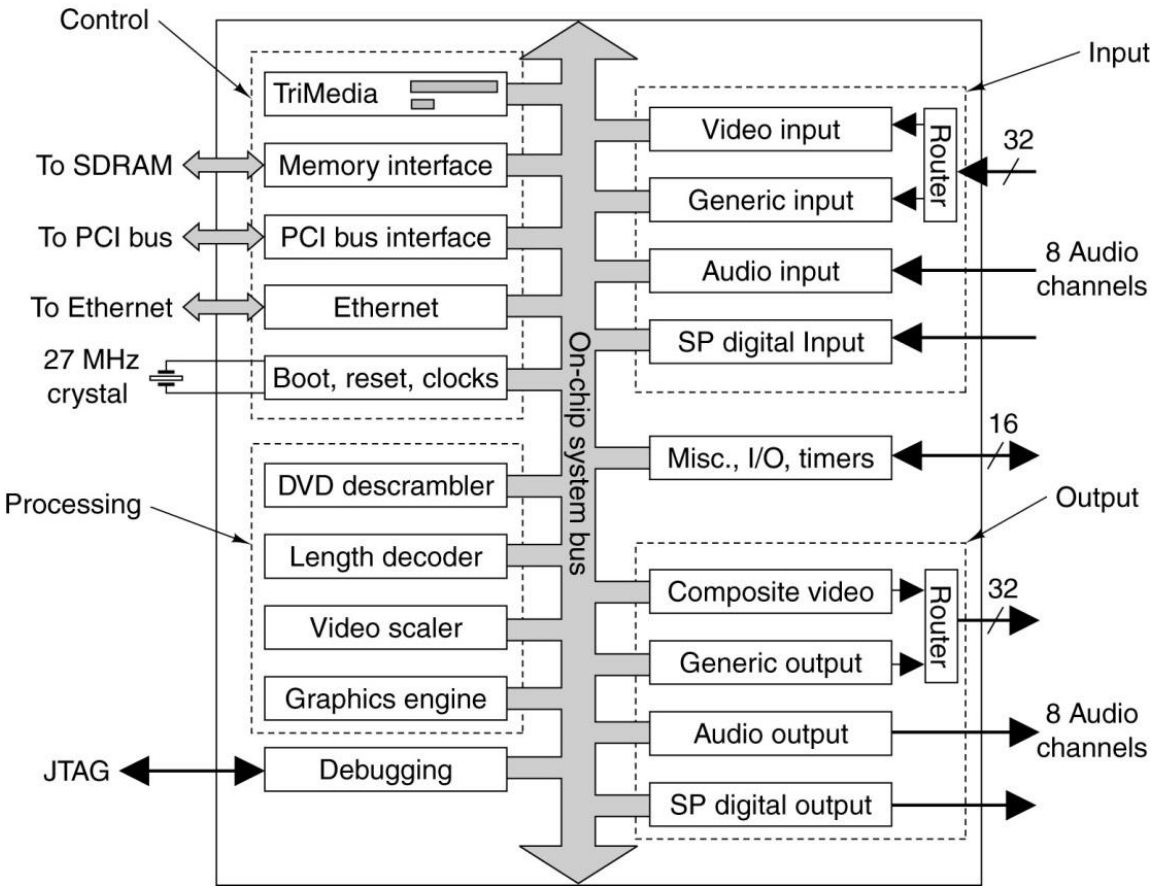
Co-Processors (2/3)

A typical network processor board and chip.



Co-Processors (3/3)

The Philips Nexperia Media Processor – for DVD players etc



The Nexperia heterogeneous multiprocessor on a chip.

Murdoch
UNIVERSITY

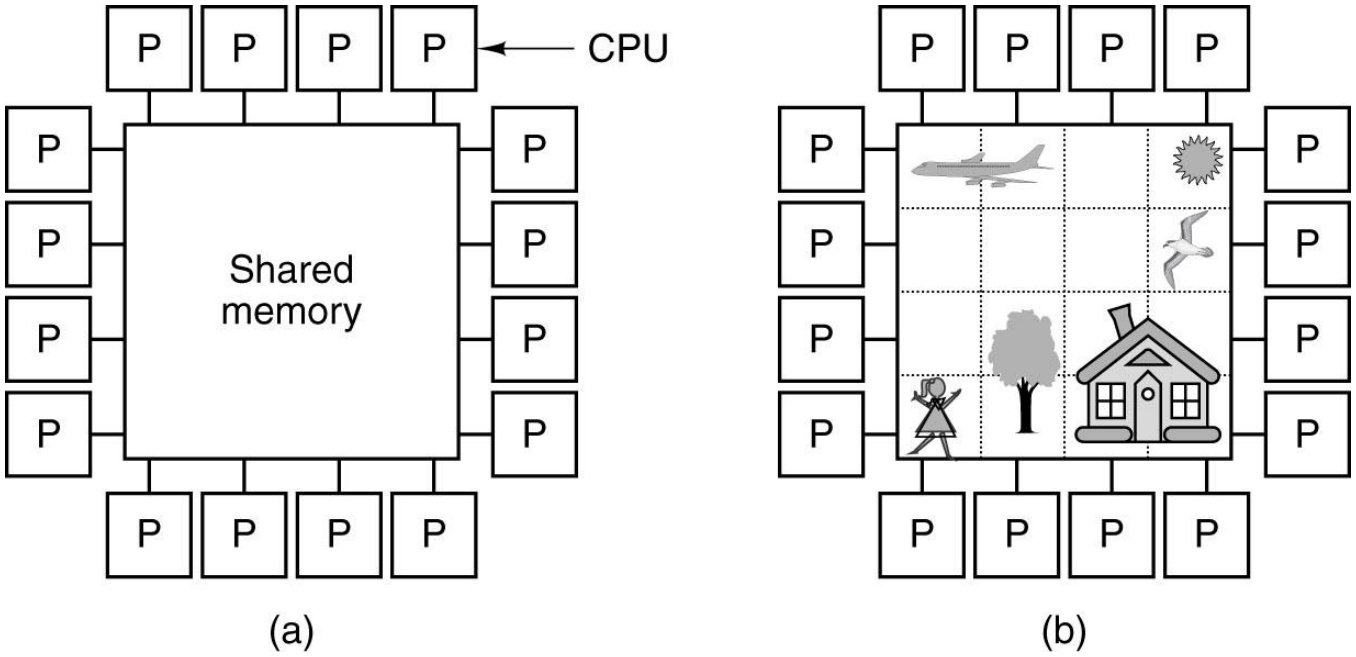
Multiprocessors

Subtitle if required

Multiprocessors

- A Parallel computer when many CPUs share a common memory
- One copy of the Operating System and paging file.
- Must have access to I/O, network, etc.
- When every CPU is treated equally, then this is called a Symmetric MultiProcessor

Multiprocessors



(a) A multiprocessor with 16 CPUs sharing a common memory.

(b) An image partitioned into 16 sections, each being analyzed by a different CPU.

Murdoch
UNIVERSITY

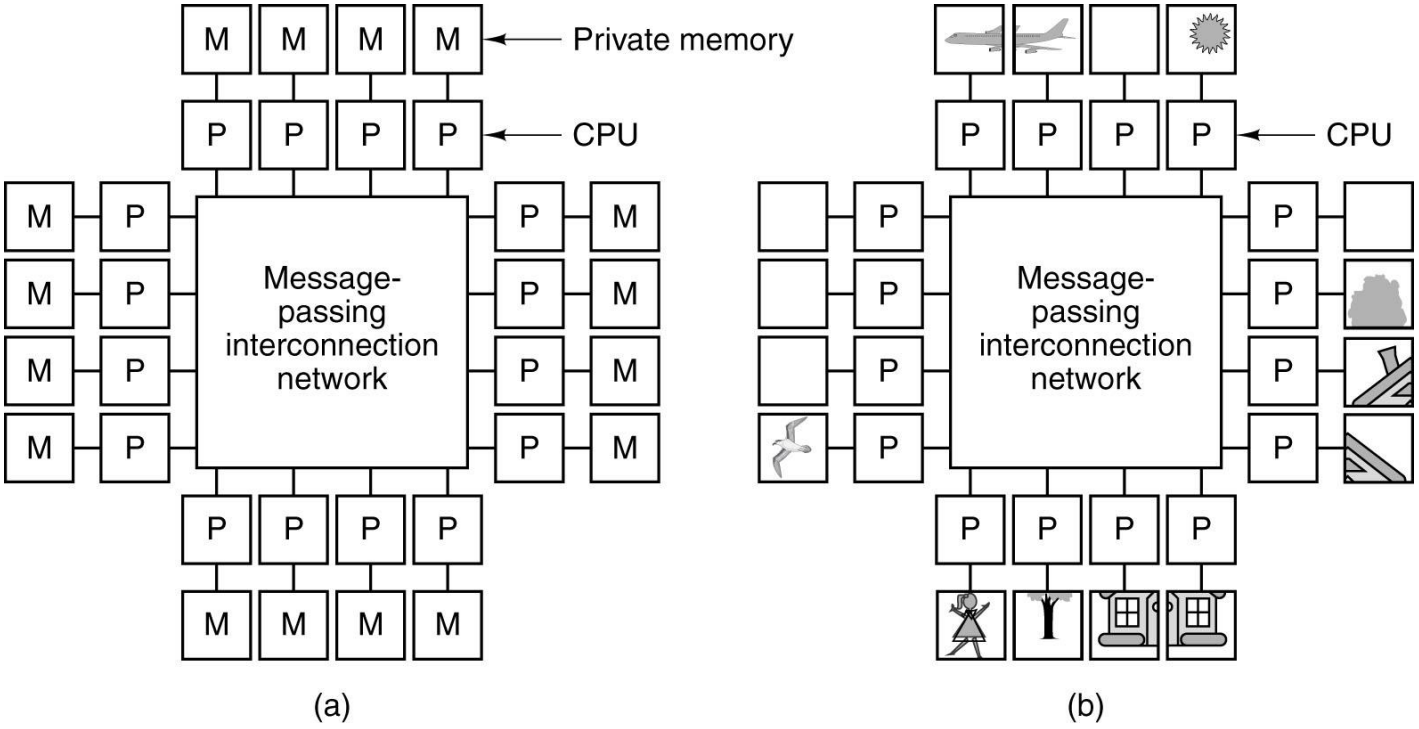
Multicomputers

Subtitle if required

Multicomputers

- A further extension from multiprocessors is multi-computers
  - Many CPUs – each has its own memory space
  - This memory is only accessible to the individual CPU
    - Called Multi-computer or Distributed Memory System
- The CPU uses load and store to access memory
- An interconnection network is required!

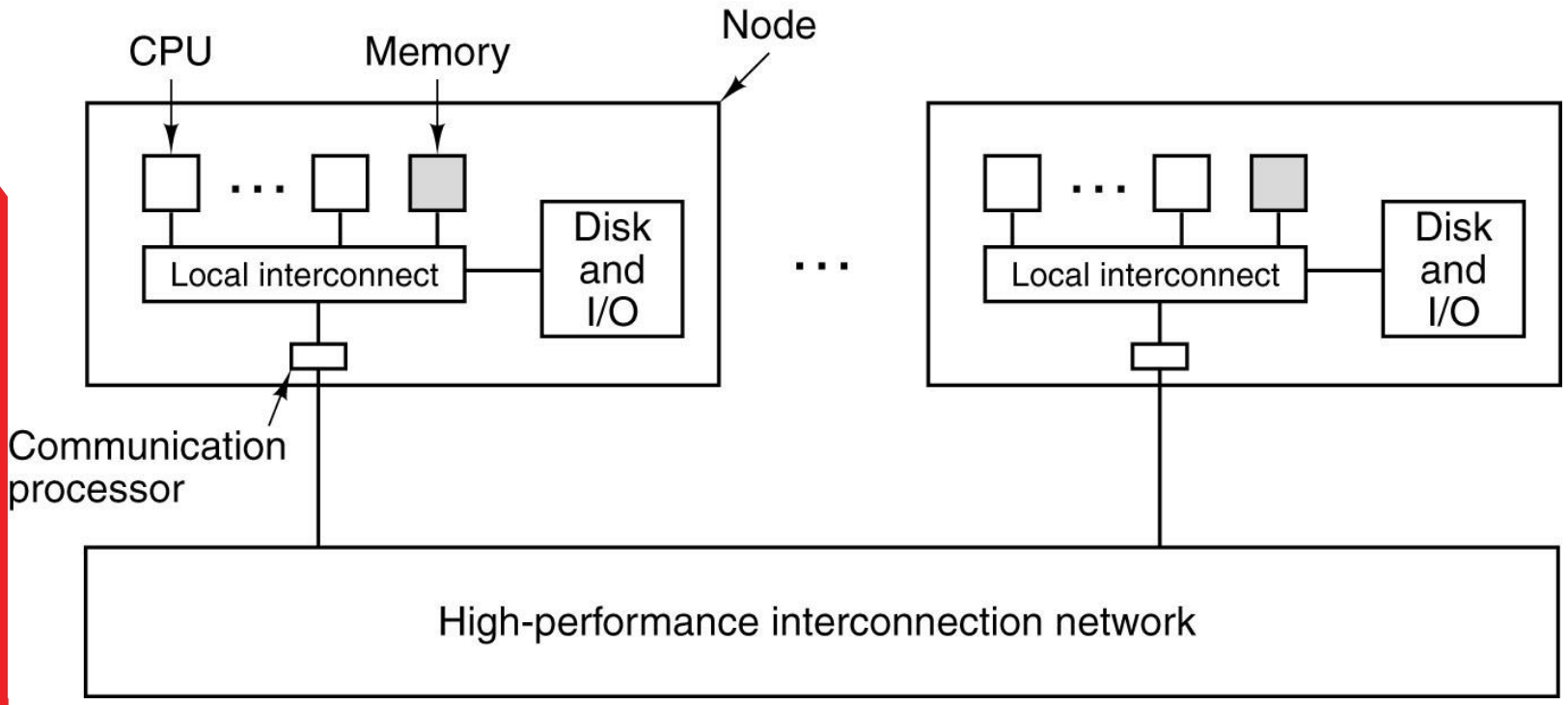
Multicomputers



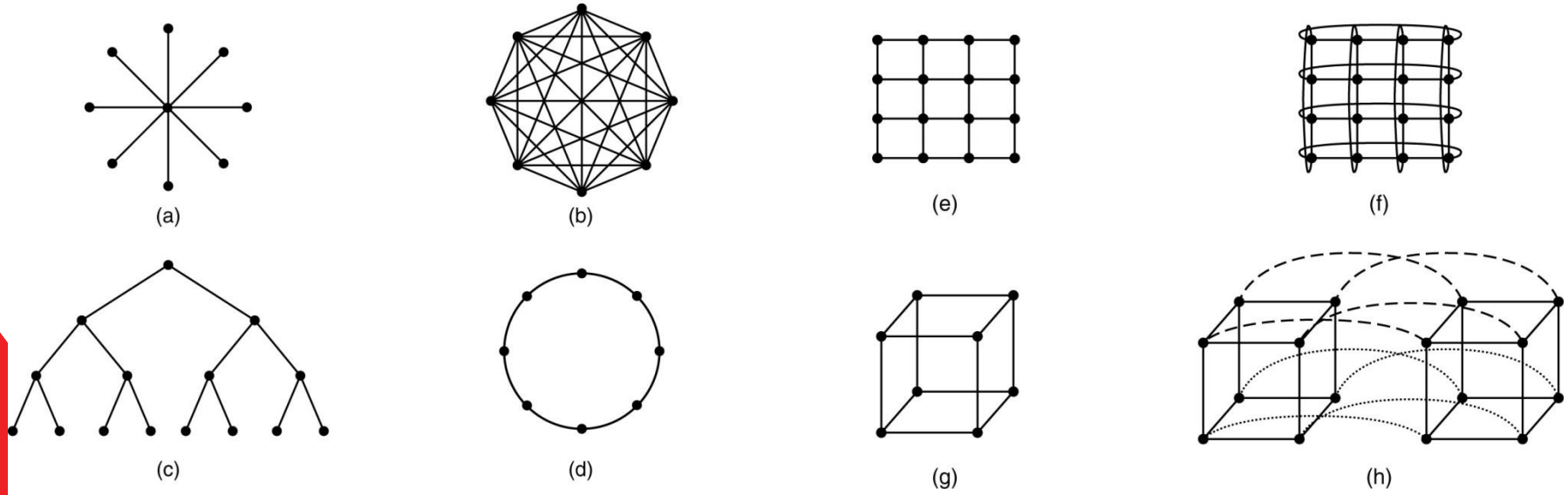
(a) A multicomputer with 16 CPUs, each with its own private memory.

(b) The bit-map image of previous figure split up among the 16 memories.

Multicomputers



Multicomputers: possible topologies



Various topologies. The heavy dots represent switches.
The CPUs and memories are not shown.
(a) A star. (b) A complete interconnect.
(c) A tree. (d) A ring. (e) A grid. (f) A double torus.
(g) A cube. (h) A 4D hypercube.

Murdoch
UNIVERSITY

Cluster Computing

Subtitle if required

Mu
UNIV

Cluster Computing

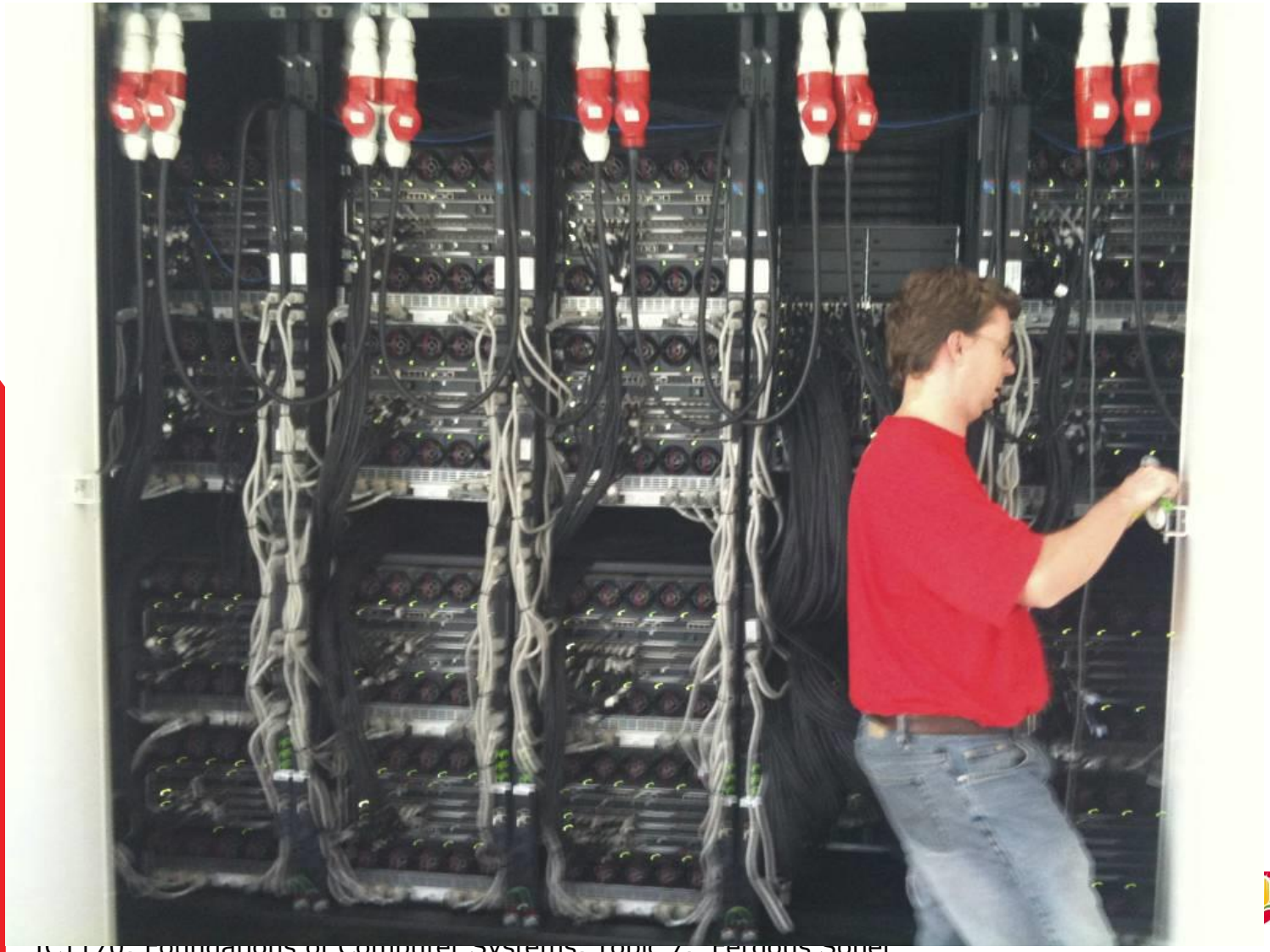
- Tightly linked group of computers, that can work on complex tasks.
- independent internal buses but specialised group buses.
- Originated from Military and Scientific computing.
- Can be local area (LAN) or wide area (WAN).
- Can be for High-availability, Load-Balancing and Computing.

- Hard to define the differences between Cluster Computing and Message-passing Multicomputers!

Cluster Computing

Cluster Computing

Cluster Computing



Cluster Computing

Murdoch
UNIVERSITY

Taxonomy of Parallel Computers

Subtitle if required

Mur
UNIV

Taxonomy of Parallel Computers (1)

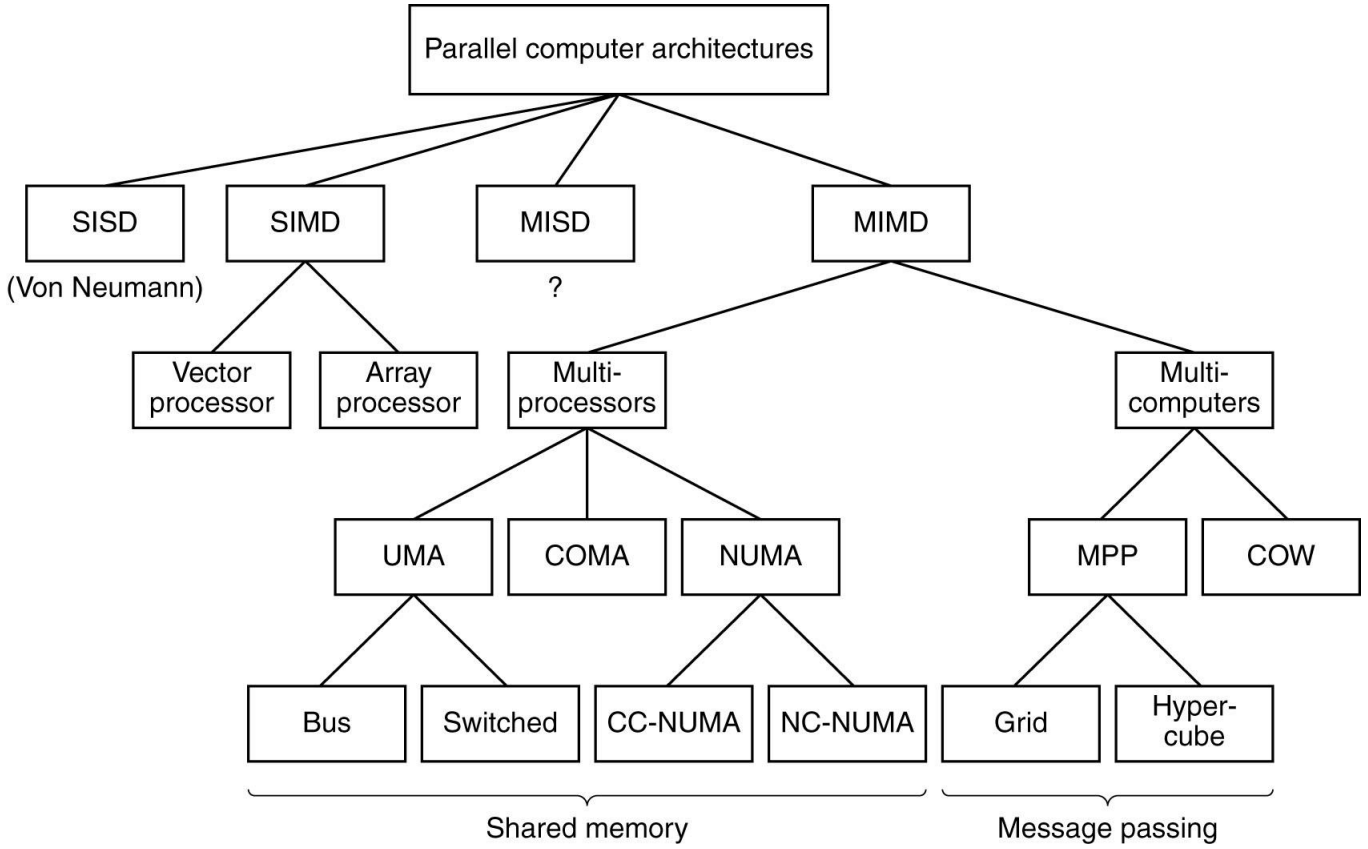
One way of thinking about the different parallel architectures is using "Flynn's taxonomy of parallel computers"

SIMD – Single Instruction Multiple Data
MIMD – Multiple Instruction Multiple Data

Instruction streams	Data streams	Name	Examples
1	1	SISD	Classical Von Neumann machine
1	Multiple	SIMD	Vector supercomputer, array processor
Multiple	1	MISD	Arguably none
Multiple	Multiple	MIMD	Multiprocessor, multicomputer

Not great – but we don't have much more!

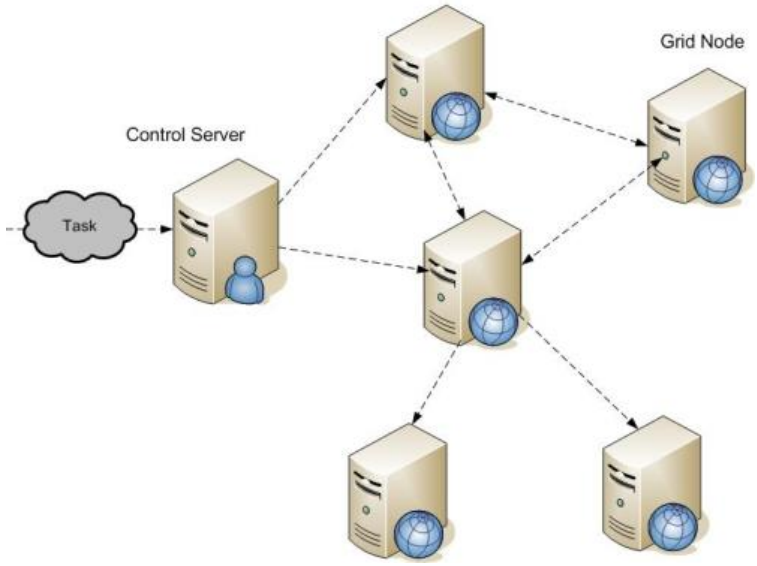
Taxonomy of Parallel Computers (2)



A taxonomy of parallel computers.

Grid and Cloud Computing

Grid Computing



- Combination of Computing resources from multiple administrative domains.
- For tasks involving large amounts of computer cycles or data to process.
- Made up of multiple clusters.
- Customised submission and security software per grid.
- Examples include Teragrid (USA), D-GRID (De), UKGRID (UK), EGEE (EU), iVEC (AU)

Cloud Computing

- A Service model that takes advantage of technologies and resources
  - The Internet
  - Fat-pipes (The internet tubes are getting bigger)
  - Ubiquitous computing
  - Virtualisation
  - Web services
- From a technical Point of view
  - Pay-per-use (no commitment, utility prices)
  - Elastic Capacity – scale up/down on demand.
  - Self-Service Interface - programmable
  - Resources are abstracted/Virtualised
- Also called Utility Computing as it treats I.T. as a utility

Murdoch
UNIVERSITY

Performance Evaluation

Performance and Cost

- Which computer is fastest?
- Not so simple
- Depends what your doing…
  - Scientific simulation – FP performance and I/O
  - Program development – Integer performance
  - Commercial workload – Memory, I/O

Measuring Performance

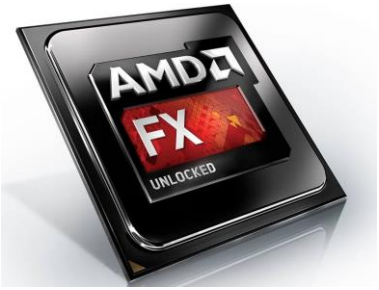
- In order to compare how fast computers can process data, we have to measure their performance.
- There are a number of measurements of performance.
- Clock speed, MIPS, FLOPS & Benchmark tests are all used. Some are a better measure than others.

Clock Speed

- The clock signal is carried by one of the lines on the control bus.
- One single pulse is called a 'clock cycle'.
- Measured in Megahertz (MHz) & Gigahertz (GHz). 1 MHz = 1 million pulses per second. 1 GHZ = 1000 MHz.

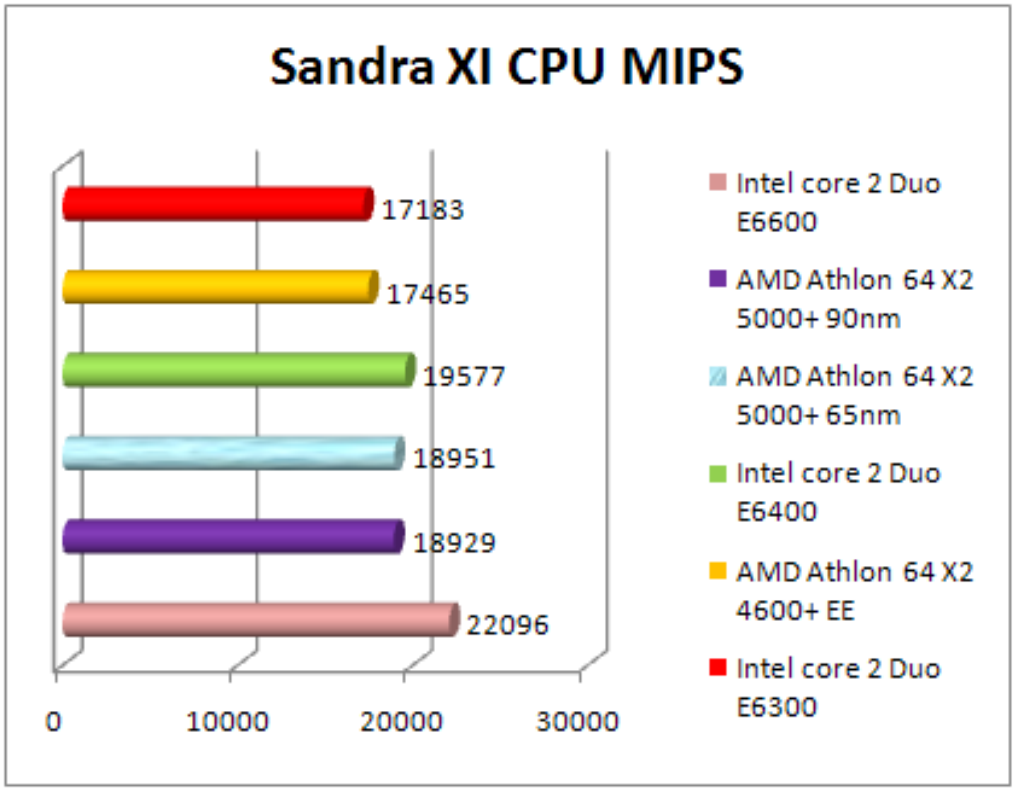
Processor Clock Speed

- CPU clock speeds are compared at http://www.cpubenchmark.net/common_cpus.html



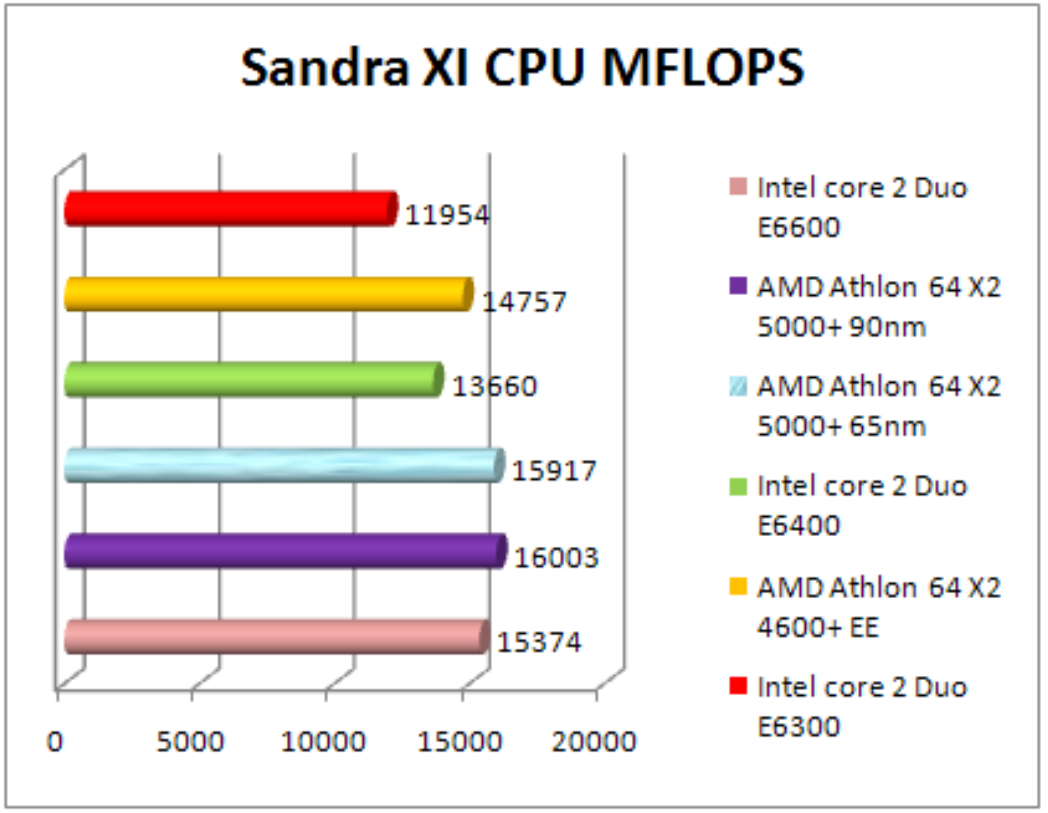
MIPS

- Stands for Millions of Instructions per second.
- Is a measure of how many machine code instructions a processor execute per second.



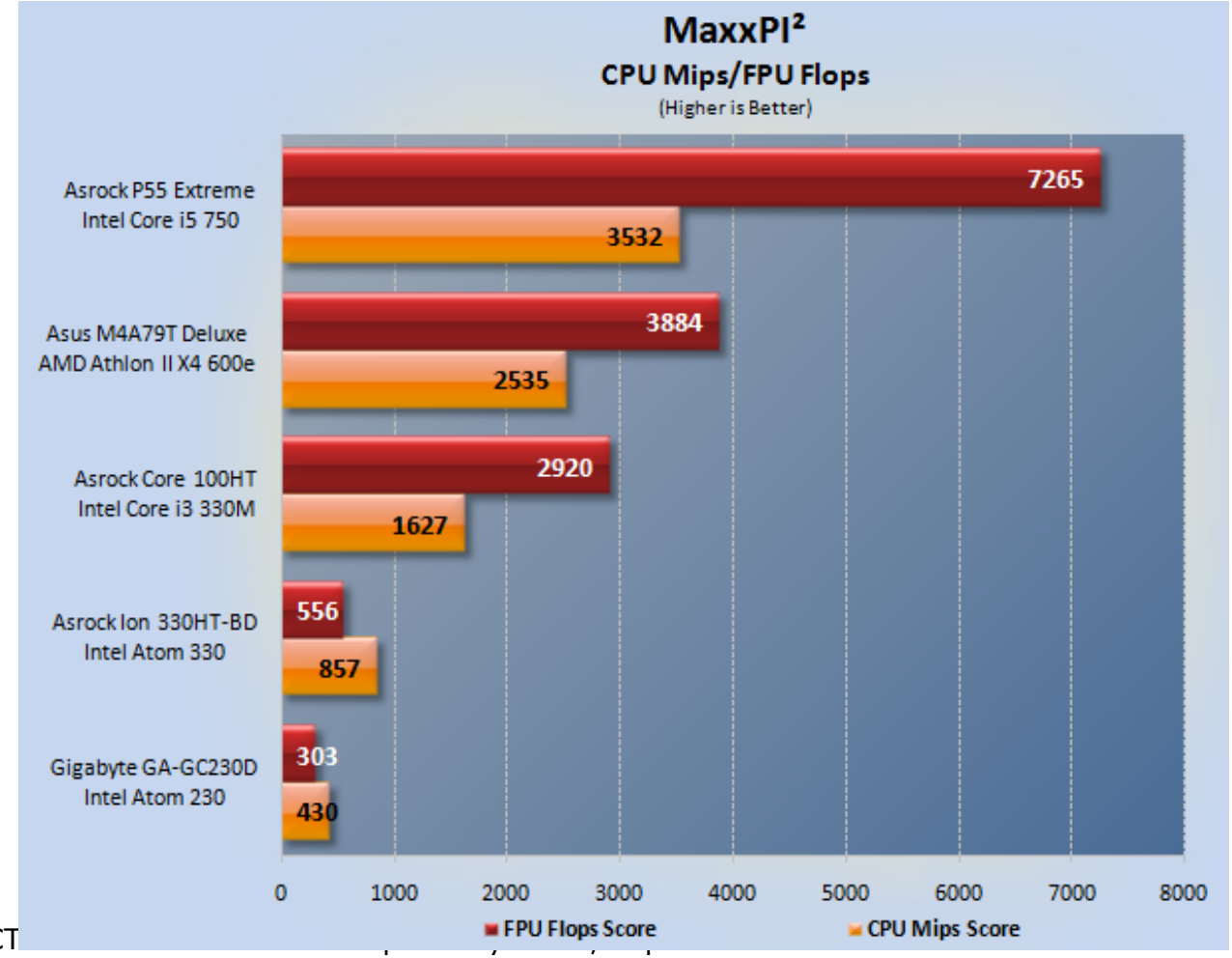
FLOPS

- Stands for 'Floating Point Operations Per Second.
- Seen as a reliable indicator of performance.
- It's a measure of the arithmetical calculating speed of a computer.



MIPS/FLOPS

- Generally used at the same time



Benchmark Tests

- Benchmark tests simply time how long a computer system takes to complete a standard set of application based tasks i.e. reformatting a 100 page 'Word' document.
- http://www.passmark.com/ is one make of benchmark test software.
- www.futuremark.com/benchmarks/pcmark is part of a collection of benchmarks
- www.**3dmark**.com/ focuses on graphics performance

Factors affecting performance

Tactic	Effect on Performance
Increase clock speed	Increase
Increase data bus width	Increase
Increase Cache memory	Increase
Increase Address Bus	None
Number of processors	Increase
Increase RAM	Slight Increase
Increase VRAM	Increase graphics performance
Increase data transfer rate	Increase

Murdoch
UNIVERSITY

Summary

Subtitle if required

Summary

- Background
- Parallel Computer Architectures
- On-chip Parallelism
- Co-Processors
- Multiprocessors
- Multicomputers
- Message-passing Multicomputers
- Cluster Computing
- Taxonomy of Parallel Computing
- Grid and Cloud Computing
- Performance Evaluation

Murdoch

UNIVERSITY