Computer Architecture and Organization

Chapter 8
Interfacing Processors and Peripherals

Henk Corporaal / Ben Juurlink
Delft University of Technology
Spring 2000
Email: h.corporaal@et.tudelft.nl
URL : http://cardit.et.tudelft.nl/~heco/courses/et2-010

Topics

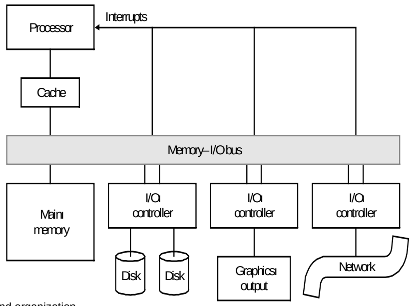
- Introduction
- I/O how?
- I/O devices
  - Examples
- Buses

Introduction

- I/O Design affected by many factors (expandability, cost, performance, standards, availability)
- Performance:
  - — access latency
  - — throughput
  - — connection between devices and the system
  - — the memory hierarchy
  - — the operating system

Input-Output (I/O): how

- Typical collection of I/O devices



I/O

- Important but neglected

  "The difficulties in assessing and designing I/O systems have often relegated I/O to second class status"

  "courses in every aspect of computing, from programming to computer architecture often ignore I/O or give it scanty coverage"

  "textbooks leave the subject to near the end, making it easier for students and instructors to skip it!"

- We won't be looking at I/O in much detail

I/O example

- Assume current application requires:
  - 90 seconds CPU time
  - 10 seconds I/O time
- What will happen in 5 years if CPUs get faster 50% each year but I/O performance stays constant
- Answer:
  - CPU time reduces to 22 seconds
  - I/O time increases from 10% to 45% !!!

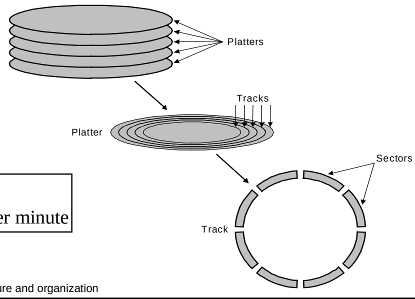
I/O Devices

- Very diverse devices
  - — behavior (i.e., input vs. output)
  - — partner (who is at the other end?)
  - — data rate

Device	Behavior	Partner	Data rate (KB/sec)
Keyboard	input	human	0.01
Mouse	input	human	0.02
Voice input	input	human	0.02
Scanner	input	human	400.00
Voice output	output	human	0.60
Line printer	output	human	1.00
Laser printer	output	human	200.00
Graphics display	output	human	60,000.00
Modem	input or output	machine	2.00-8.00
Network/LAN	input or output	machine	500.00-6000.00
Floppy disk	storage	machine	100.00
Optical disk	storage	machine	1000.00
Magnetic tape	storage	machine	2000.00
Magnetic disk	storage	machine	2000.00-10,000.00

I/O Example: Disk Drives

- To access data:
  - — seek: position head over the proper track (8 to 20 ms. avg.)
  - — rotational latency: wait for desired sector (.5 / RPM)
  - — transfer: grab the data (one or more sectors) 2 to 15 MB/sec

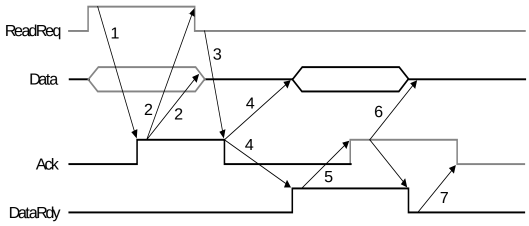


RPM=
rotations per minute

I/O Example: Buses

- Shared communication link (one or more wires)
- Difficult design:
  - — may be bottleneck
  - — length of the bus
  - — number of devices
  - — tradeoffs (buffers for higher bandwidth increases latency)
  - — support for many different devices
  - — cost
- Types of buses:
  - — processor-memory (short high speed, custom design)
  - — backplane (high speed, often standardized, e.g., PCI)
  - — I/O (lengthy, different devices, standardized, e.g., SCSI)
- Synchronous vs. Asynchronous
  - — use a clock and a synchronous protocol, fast and small but every device must operate at same rate and clock skew requires the bus to be short
  - — don't use a clock and instead use handshaking

Some Example Problems



- Look at some examples from the text

  "Performance Analysis of Synchronous vs. Asynchronous"
  "Performance Analysis of Two Bus Schemes"

Other important issues

- Bus Arbitration:
  - — daisy chain arbitration (not very fair)
  - — centralized arbitration (requires an arbiter), e.g., PCI
  - — self selection, e.g., NuBus used in Macintosh
  - — collision detection, e.g., Ethernet
- Operating system:
  - — polling
  - — interrupts
  - — DMA
- Performance Analysis techniques:
  - — queuing theory
  - — simulation
  - — analysis, i.e., find the weakest link (see "I/O System Design")
- Many new developments

Computer Architecture and Organization

Chapter 9 Multiprocessors

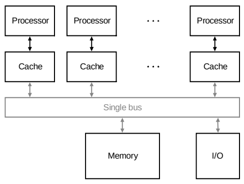
Henk Corporaal / Ben Juurlink
Delft University of Technology
Spring 2000
Email: h.corporaal@et.tudelft.nl
URL : http://cardit.et.tudelft.nl/~heco/courses/et2-010

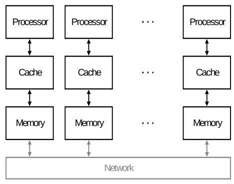
Multiprocessors

- Idea: create powerful computers by connecting many smaller ones

good news: works for timesharing (better than supercomputer)
vector processing may be coming back

bad news: its really hard to write good concurrent programs
many commercial failures



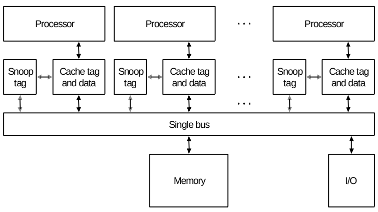


Questions

- How do parallel processors share data?
  — single address space (SMP vs. NUMA)
  — message passing
- How do parallel processors coordinate?
  — synchronization (locks, semaphores)
  — built into send / receive primitives
  — operating system protocols
- How are they implemented?
  — connected by a single bus
  — connected by a network

Some Interesting Problems

- Cache Coherency

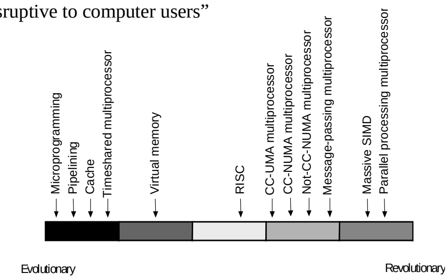


- Synchronization
  — provide special atomic instructions (test-and-set, swap, etc.)
- Network Topology

Concluding Remarks

- Evolution vs. Revolution

"More often the expense of innovation comes from being too disruptive to computer users"



Exercises

From Chapter eight and nine

- 8.1, 8.3 - 8.6
- 9.2, 9.5, 9.6