ISE1/EE2 Course on Introduction Computer Systems

- Course materials based on two books:
  - “Computer Organization & Design” 2nd edition, Patterson & Hennessy 1998 (£28.50)
    - Must buy - covers most topics on this course
    - Very well written and pages/£
    - Based mainly on MIPS R3000 architecture
  - “ARM System-on-Chip Architecture”, Steve Furber, 2000 (around £30)
    - Strongly recommended - best book on ARM processor
    - Very reasonably priced & will be needed towards the second half of term.
- Administration issues:
  - Around 20 lectures on this part of the course (may be less)
  - For EE2 students: two hours lab per week
  - For ISE1 students: two hours lab per week (shared with programming) & personal tutorial & one hour per week study group
  - Lab exercises starts in week 4 - one assessed assignment

All you need to know about computing (as an Electronic Engineer)

What Learning Computing is NOT?

- Computing is ***not just*** about programming in C or any specific language
- Computing is not about learning machine code or a specific microprocessor and its instruction set
- Computing is not about learning how to use Office 97

It is:

- Having a model (at the appropriate level) of how a computer works
- About designing and engineering a good software solution to a problem
- Able to handle complexity in a system
- Able to use computer/microprocessor to solve an engineering problem

Computing as a discipline

- Computing is not just about a PC or a Mac
- Most microprocessors are used in embedded applications
- These range from a few Kbyte programme on a simple 4-bit micro in a washing machine to a few Mbyte of code on a 32-bit processor on a mobile phone
- As an engineer, we are more interested in embedded applications then, say, database programming on a IBM mainframe (in a bank?)
- Our view of computing should be linked to subjects such as chip design, telecommunication, signal processing, neural networks, ….
- Computing should be a tool to help us in practicing these disciplines, not an end in itself
- As an engineer, you need to learn different things relating to computer HARDWARE and SOFTWARE

Skill & Knowledge expected of our students

Skills

- Able to design and write a reasonably large program in a procedural language to perform a useful non-numerical task.
- Able to use **Matlab** to solve a non-trivial numerical problem.
- Able to publish an article or to create a home page on the Web.

Knowledge

- Understand the basic principles of software engineering such as:
  - Top-down design methodology
  - Modularity, information hiding, cohesion
  - Software must not just be written but designed and “engineered”
  - Testing and documentation

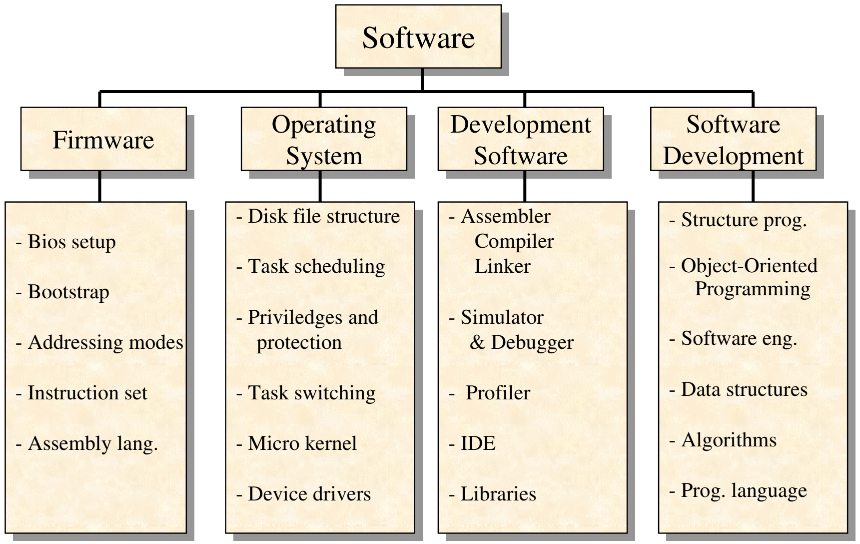
Knowledge (con't)

- Understand data-structures, algorithm and object-oriented concepts such as:
  - Basic data structures – record, linked lists, trees, queues, tables etc.
  - How to design data structure to help both the efficiency of algorithms and maintainability of the code
  - Some commonly used algorithms
  - Ideas of efficiency and complexity of algorithms
  - Concepts of encapsulation in the context of object-oriented design
- Understand how a computer works. This should include:
  - Some basic understanding of computer architecture and low-level instruction set
  - The major building blocks inside a PC
  - The relationship between hardware, firmware (e.g. BIOS), operating system and application software
  - Some aspects of interfacing such as i/o, why need drivers, interrupts and DMA.

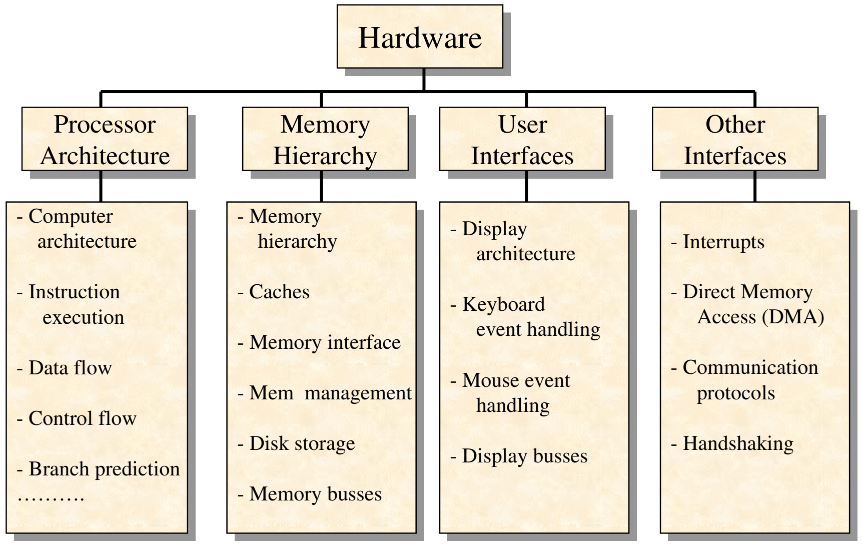
What will you be learning?

- What is a computer?
- What is **I**nstruction **S**et **A**rchitecture (ISA)
- How is a ISA implemented in hardware? (in abstract level)
- Computer arithmetic
- Processor architecture - datapath & control
- Simple pipelining in processors
- Basic memory hierarchy
- Computer interfacing with peripherals
- Embedded processors
- Assembly language programming - large part of the course

Computer Software



Computer Hardware



Structure of ISE1 Computing Courses

- Around 50 lectures spread over two terms

Autumn Term

Wk 2-10: Programming
(*Pitt & Madden*)

Wk 2-10: Computer architecture
ARM architecture
Assembly language
(*Cheung & Constantinides*)

Spring Term

Wk 1-2: Visual Programming
(*Pitt & Madden*)

Wk 3-10: Data structures & Algorithms (*Shanahan*)

Wk 1-6 Introd. to Operating Systems (*Demiris*)

Structure of EE2 Computing Courses

- Around 40 lectures spread over two terms

Autumn Term

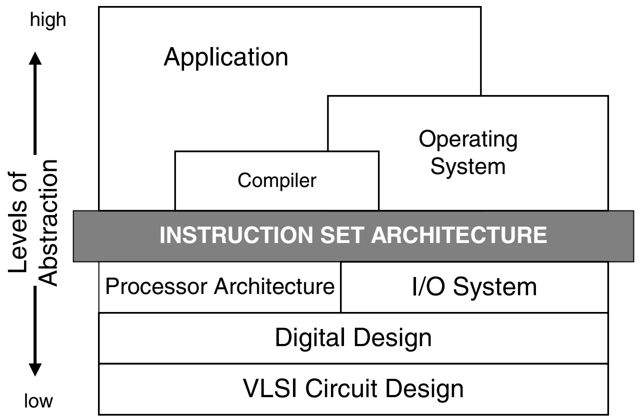
Wk 2-10: Computer architecture
ARM architecture
Assembly language
(*Cheung & Constantinides*)

Spring Term

Wk 1-2: Matlab
(*Cheung*)

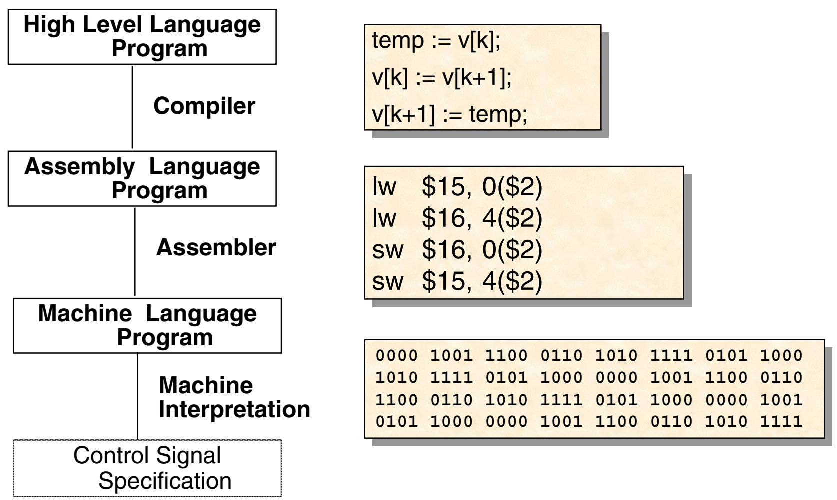
Wk 3-10: Data structures & Algorithms (*Shanahan*)

What is "Computer Architecture" ?



- Key: Instruction Set Architecture (ISA)
- Different levels of abstraction

Levels of representation in computers



What is “Instruction Set Architecture (ISA)”?

- “. . . the attributes of a [computing] system as seen by the programmer, i.e. the conceptual structure and functional behavior, as distinct from the organization of the data flows and controls the logic design, and the physical implementation.”
    - ➤ Amdahl, Blaaw, and Brooks, 1964

ISA includes:-

- Organization of Programmable Storage
- Data Types & Data Structures: Encodings & Representations
- Instruction Formats
- Instruction (or Operation Code) Set
- Modes of Addressing and Accessing Data Items and Instructions
- Exceptional Conditions

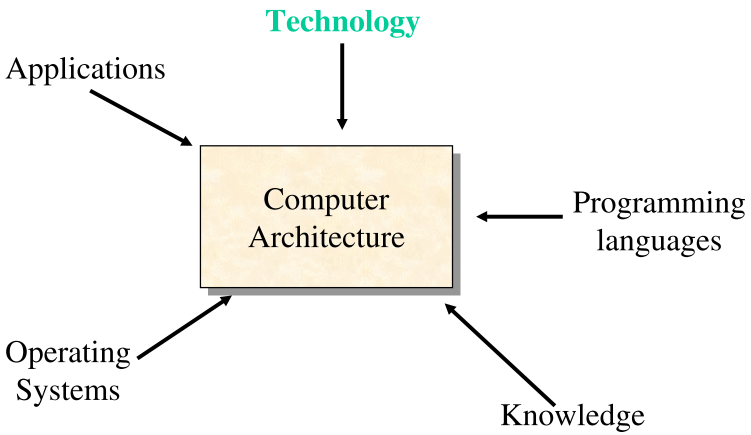
Instruction Set Architecture (ISA)

- A very important abstraction
    - interface between hardware and low-level software
    - standardizes instructions, machine language bit patterns, etc.
    - advantage: *different implementations of the same architecture*
    - disadvantage: *sometimes prevents using new innovations*

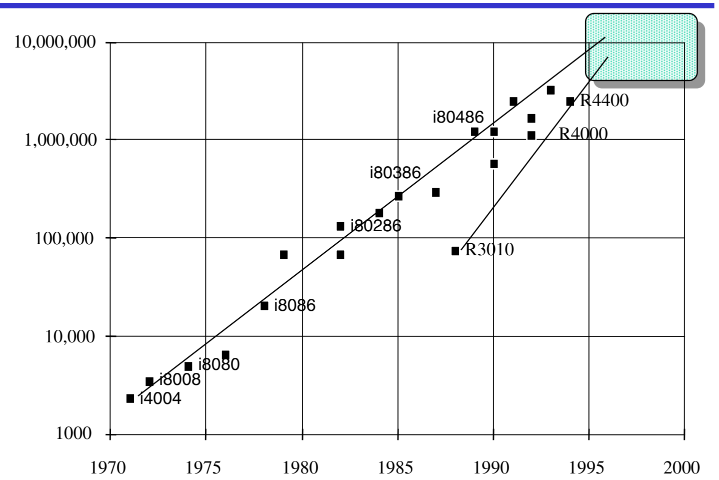
True or False: Binary compatibility is extraordinarily important?

- Modern instruction set architectures:
    - ARM, 80x86/Pentium/K6, PowerPC, DEC Alpha, MIPS, SPARC, HP

Factors influencing computer architectures

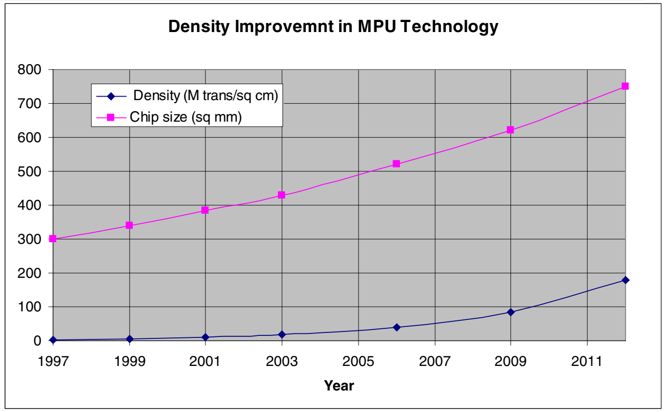


Technology: Logic Density (processors)



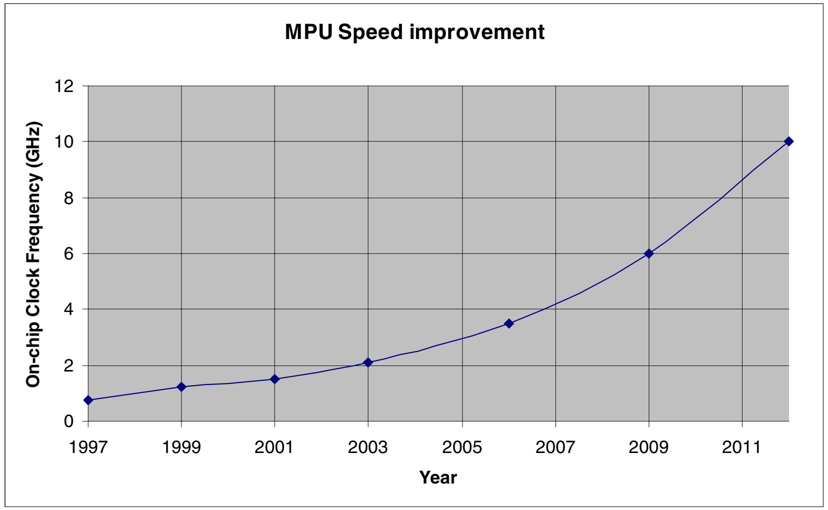
- “No. of transistors double around every 18 months”

Improvements in Digital IC Technology



	Capacity	Speed
Logic	2x in 3 years	2x in 3 years
DRAM	4x in 3 years	1.4x in 10 years
disk	4x in 3 years	1.4x in 10 years

Processor Speed Improvements



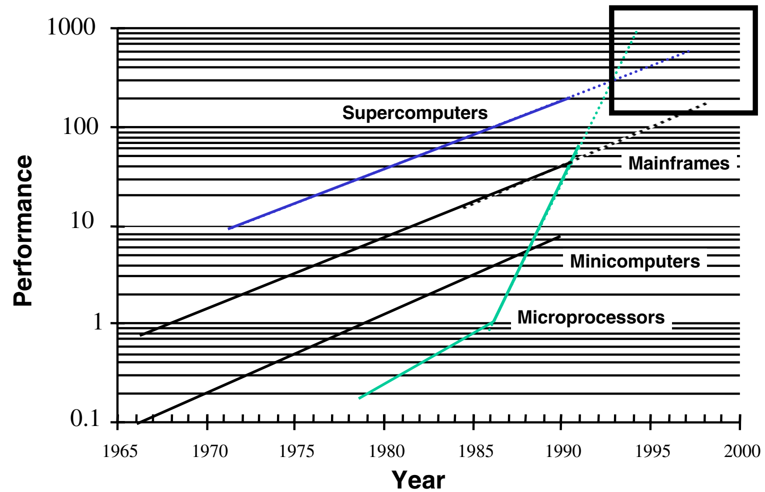
Cost of a processor

IC cost = (Die cost + Testing cost + Packaging cost) / Final test yield

Packaging Cost: depends on pins, heat dissipation

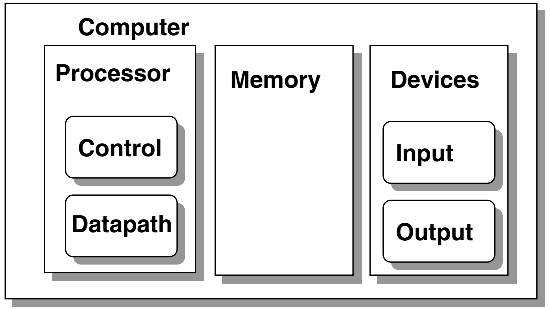
Chip	Die cost	Package pins	Package type	Package cost	Test & Assembly	Total
386DX	$4	132	QFP	$1	$4	$9
486DX2	$12	168	PGA	$11	$12	$35
PowerPC 601	$53	304	QFP	$3	$21	$77
HP PA 7100	$73	504	PGA	$35	$16	$124
DEC Alpha	$149	431	PGA	$30	$23	$202
SuperSPARC	$272	293	PGA	$20	$34	$326
Pentium	$417	273	PGA	$19	$37	$473

Performance improvement of microprocessors Vs others



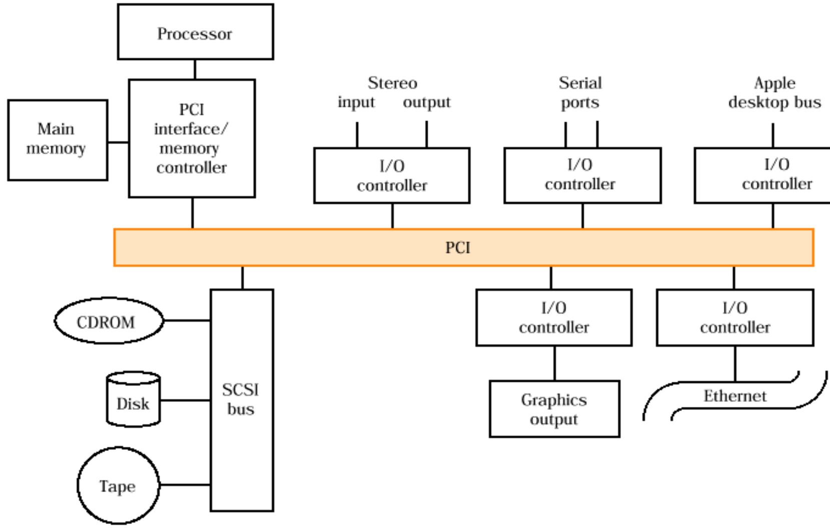
- Microprocessors take over (true or false?)

Internal Organisation



- Major components of Typical Computer System

A Typical Computer System with I/O



Summary

- All computers consist of five components
  - Processor: (1) datapath and (2) control
  - (3) Memory
  - (4) Input devices and (5) Output devices
- Not all “memory” are created equally
  - Cache: fast (expensive) memory are placed closer to the processor
  - Main memory: less expensive memory--we can have more
- Input and output (I/O) devices has the least regular organization
  - Wide range of speed: graphics vs. keyboard
  - Wide range of requirements: speed, standard, cost ... etc.
  - Least amount of research (so far)