Slide 1

- Plan of the Course.
- Recommended Text-Book
- Introduction to the labs

Slide 2

The course consists of 3 main parts.

- Assembly language programming
- Logic, Processor Implementation, Caches and Virtual Memory
- I/O

Slide 3

- In the course we will be using the MIPS processor, this is related to the ARM processor and various versions of the MIPS are used in set-top boxes, Playstations and embedded cores.
- The MIPS processor is very different from the Intel x86 series. The MIPS processor has many registers and very few forms of instructions.
- We will see how it is possible to implement the MIPS processor in hardware and various tradeoffs that can be achieved.
- We will be using the SPIM simulator for the exercises and labs.

Slide 4

- We will look briefly at digital electronics, not with a view to designing circuits, but understanding how design relates to performance.
- We various possible implementation strategies and how implementation relates to performance.
- We will look at pipelines, a way of increasing the instruction throughput of a processor and the problems related to pipelines.

Caches :-

- Short story, memory is slow, processors are fast how do we reorganise things so as to make things faster.

Slide 5

Virtual Memory :-

- There is never enough memory, use some of the techniques developed in Cache memory to give the programmer the illusion of more memory by using the disk.

- Key idea :- It is possible to achieve better performance by reorganisation. There is a limit to how fast hardware can go.

Slide 6

- By the end of the course you should know that the clock-speed of a processor has little to do with how fast your programs run, but more to do with how much you pay for the machine.

Where is computer architecture heading?

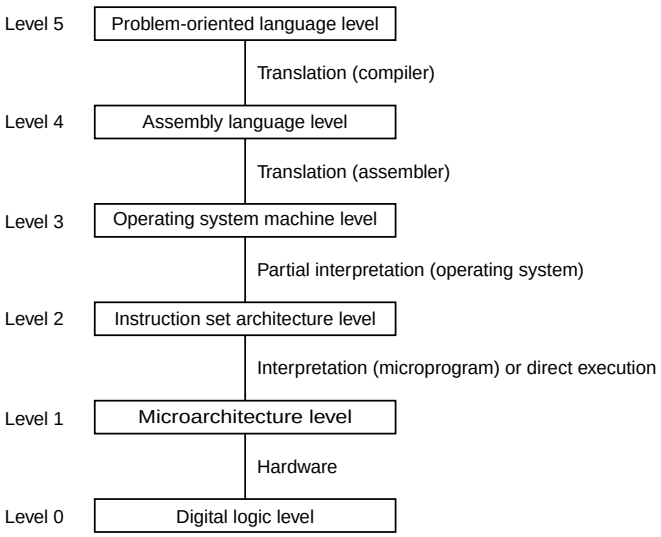
- Finally we will look how you connect the computer to the outside world.

Slide 7

Key Idea:-

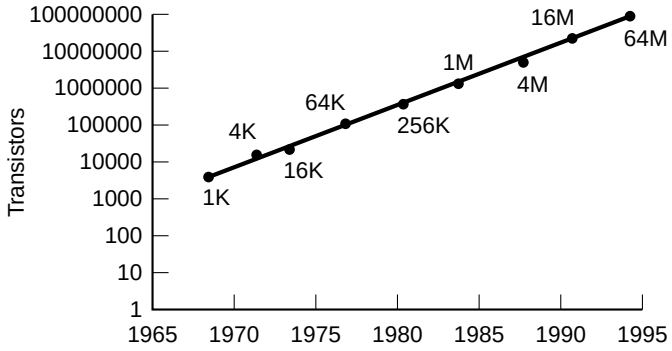
- I/O devices are relatively slow, we have to organise things carefully so not to slow down things too much.
- Interrupts and Polled I/O.

Slide 8



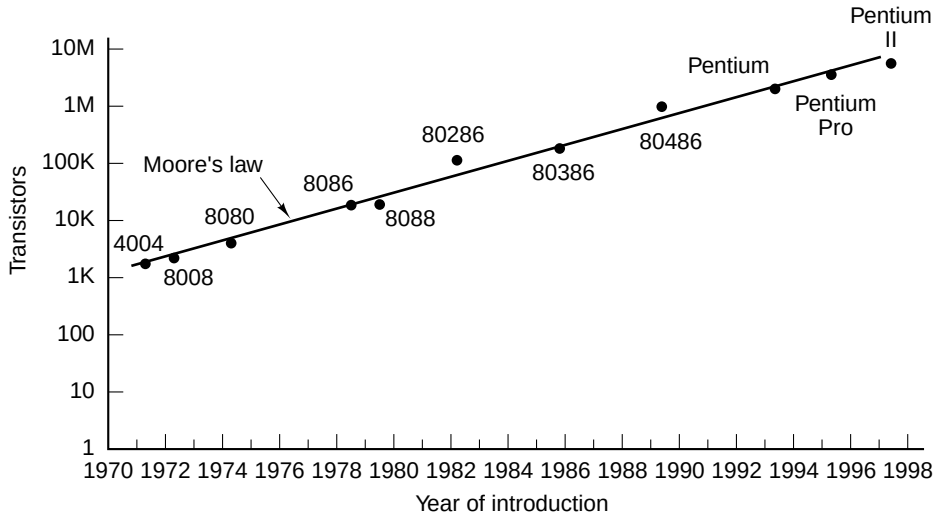
Slide 9

Bigger memories



Slide 10

More complex processors



Slide 11

- Computers get twice as fast, twice as much memory ... every 18 months.
- Been true since the 60's
- There is a limit (quantum physics and all that). What do we do when we hit that limit? Be more clever, parallel processing?

Slide 12

- Instruction set - Lowest level you can program at, interface between hardware and software (the pineal gland)
- Instruction like `add`, `lw` load data in from memory, `sw` store data in memory

Slide 13

- CISC - Complex instruction set computer (Pentiums)
  - Provide a large number of basic instructions
- RISC - Reduced instruction set computer (ARMs)
  - Provide only very basic set of instructions

Slide 14

- RISC - simple chip, easy to design and optimise. Memory is cheap.
- Hard to design, but provides a rich set of instructions for the programmer.
- Idea design a RISC chip so that the equivalent programm runs faster.
- Not so simple any more, the boundaries get fuzzy.

Recommended Reading

Recommend Text Book

- *Structured Computer Organisation* Andrew S. Tanenbaum, Prentice Hall Fourth Edition. You need the Fourth Edition, since it has been updated and one of the labs is based on a chapter in the Book.

Slide 15

The Tanenbaum book does not contain much information on the MIPS processor. There will be material in the lectures some people might find the following book useful.

- *Introduction to RISC Assembly Language Programming* John Waldron, Addison-Wesley.

- The course web-page can be found (eventually) via my home-page at `http://user.it.uu.se/~justin/`

Slide 16

Slides will be put there as we go along. As well as information about the labs.

Recommended Reading

Slide 17

- Tanenbaum Chapter 1.