• Plan of the Course.

Slide 1
• Recommended Text-Book
• Introduction to the labs

The course consists of 3 main parts.

Slide 2
• Assembly language programming
• Logic, Processor Implementation, Caches and Virtual Memory
• I/O
• 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.

• 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.

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.

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

Key Idea:-

Slide 7
• 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

<table>
<thead>
<tr>
<th>Level 0</th>
<th>Digital logic level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Level 1</td>
<td>Microarchitecture level</td>
</tr>
<tr>
<td>Level 2</td>
<td>Instruction set architecture level</td>
</tr>
<tr>
<td>Level 3</td>
<td>Operating system machine level</td>
</tr>
<tr>
<td>Level 4</td>
<td>Assembly language level</td>
</tr>
<tr>
<td>Level 5</td>
<td>Problem-oriented language level</td>
</tr>
</tbody>
</table>

Translation (compiler)
Translation (assembler)
Partial interpretation (operating system)
Interpretation (microprogram) or direct execution
Moore’s Law

Slide 9

Bigger memories

- 1965: 1K
- 1970: 4K
- 1975: 16K
- 1980: 64K
- 1985: 256K
- 1990: 1M
- 1995: 4M
- 2000: 16M
- 2005: 64M

Year of introduction

Slide 10

More complex processors

- 1970: 8004
- 1972: 8080
- 1974: 8086
- 1976: 80286
- 1978: 80386
- 1980: 80486
- 1992: Pentium
- 1995: Pentium Pro
- 1996: Pentium II

Year of introduction

Lecture 1
Introduction to Computer Architecture – Justin Pearson
Page 11
RISC versus CISC

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
- 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 13

- 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 program runs faster.

- Not so simple any more, the boundaries get fuzzy.

Slide 14
Recommended Reading

Recommend Text Book

- *Structured Computer Organisation*  Andrew S. Tanenbaum,
  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.

Slide 16

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

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

Slide 17  

- Tanenbaum Chapter 1.