Computer Architecture Course Outline and Plan

Justin Pearson

October 11, 2002

1. *Course Registration and introduction to the course.*
2. *MIPS Assembly language.* ,
   - The Von Neumann machine
   - Instructions
   - registers,
   - arithmetic, `add , addi` translating high level arithmetic into assembly.

   Web resources, the SPIM manual can be found at `http://www.cs.wisc.edu/~larus/spim.html` . Reading material, Tanenbaum Chapter 1, Chapter 2.1. Read the description of the R2000 ion the SPIM manual. The lecture continues with :-
3. *More on Mips Assembly*
   - Memory and Registers, `lw` and `sw`. Addressing modes.
   - Arrays. Some example programs with arrays.

   Remember that 4 is the magic number.
4. *More on Mips Assembly*
   - Subroutines
   - Stacks and the answer to recursive subroutines

   Practice some examples with stacks and recursive functions. Remember to use functions in your programs.
5. *Revision of Digital logic.*
   (a) logic gates, truth tables, implementing truth tables.
   (b) Longest path.

   Reading Tanenbaum Chapter 3, 3.1,3,2,3.3.
6. *Arithmetic*

(a) Binary numbers, hex-numbers.

(b) The concept of representation.

(c) Negative Numbers and Two's complement

(d) The evils of floating point.

i. All that you needed to know about floating point.

ii. Floating point addition is not associative

Reading Tanenbaum Appendix A and Appendix B

7. *Implementing Addition*, Ripple adder, Faster Adder. Reading Tanenbaum 3.2.3. Handout on the ripple carry adder.

8. *Implementing the MIPS processor*

   - Latches and memory Reading Tanenbaum 3.3.
   - The single cycle approach. Nothing much in Tanenbaum. Look at the slides. The basic idea, use the opcode of the instruction to specify what functional units have to be switched on. Do every thing in a single cycle.
   - Problems with the single cycle approach. Slowest instruction gives the cycle time, a functional unit can only be used for one thing at a time so have to repeat functional units, separate adders.

9. *Multicycle implementations*

   - The concept of a cycle.
   - Finite state machines.
   - Balancing the work into single cycles.
   - Controlling the work done with a finte statemachine.
   - The five cycles of the MIPS.

   Reading material the slide from the lecture.

10. *Implementing Finite State machines*

    - Roms, PLAs
    - Microcode
    - RISC/CISC
    - Microcode and some modern processors.

    Reading Material: Tanenbaum 2.1 ,4.1 and Slides.

11. *Pipelines*

    - Doing more than one thing at once.

- Problems with pipelines, stalls, branch delay slots.
- Making programs faster by avoiding stalls.

Reading material slides, plus read all the entries to do with pipelines in the index of the Tanenbaum book.

12. *Caches.*

    - Principle of locality
    - Direct Mapped Caches
    - Set Associative Caches
    - LRU
    - write through, write back
    - Cache line, length

    Tanenbaum Section 4.2

13. *Virtual Memory.*

    - Sections 6.1.1,6.1.2. TLBs.
    - The material on the slides.

14. *I/O* Polling and Interrupts

    - Memory Mapped I/O (pages 195-197)
    - Interrupts, Polled I/O.