Rose-Hulman Institute of Technology
Computer Science and Software Engineering
Profs. Laurence D. Merkle and Archana Chidanandan

Homework 4
(Datapath and Control Design, Computer Arithmetic)

This assignment is due Thursday, October 23, 2003 for Sections 1, 2 and 3.

Learning Objectives

In the process of completing this homework assignment, students will develop their abilities to

- Design datapaths to support machine language instruction sets by specifying the interconnections between components and the behavior of the associated control units.
- Design control units to support machine language instruction sets by applying combinational and sequential digital logic design principles.
- Design digital logic circuits for computer arithmetic.
- Predict the qualitative effect on clock cycle time of modifications to the design of digital logic circuits.

General Instructions

1. Submit your solutions on a separate sheet of paper.
2. Remember that the figures in the book are available in electronic form on the book's website.

Problems

1. [5 points each] For each of the problems below, modify the textbook's multi-cycle datapath and control (Figure 5.33 on page 383 and Figure 5.42 on page 396, respectively) to implement the indicated MIPS instruction.

   a. Modify the datapath and control to implement the MIPS `addi` instruction.

   b. Modify the datapath and control to implement the MIPS `mfc0` instruction.

   c. Modify the datapath and control to implement the MIPS `mtc0` instruction.

2. [5 points each] For each of the problems below, modify the 32-bit ripple-carry ALU developed in class as indicated. Use only inverters, AND gates, OR gates, and multiplexers. In each case, describe any new or modified control signals. Also, assuming that the ALU is currently on the critical path for your design, determine whether or not your modifications would extend the clock cycle time.

   a. Modify the ALU to support the MIPS `xor` instruction.

   b. Modify the ALU so that it detects overflow for both addition and subtraction (i.e. so that it has a new 1-bit output called Overflow which is asserted iff overflow occurs).

Rose-Hulman Institute of Technology
Computer Science and Software Engineering
Profs. Laurence D. Merkle and Archana Chidanandan

c. Modify the ALU so that it implements `slt` correctly even when overflow occurs. Assume that you have a combinational logic unit that detects overflow, as you are required to design for the previous problem.

d. Modify the ALU so that the MIPS pseudoinstruction `abs` could be implemented as an actual instruction.