Rose-Hulman Institute of Technology
Computer Science and Software Engineering
Profs. Archana Chidanandan and J.P. Mellor

Homework 3
(Control unit design and Recursive procedures)
Maximum points: 60

Directions
This assignment is due by 5:00PM on Wednesday, Oct. 27th for Sections 1, 2 and 3.

Learning Objectives

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

- Design control units to support machine language instruction sets by applying combinational and sequential digital logic design principles.
- Use stacks to implement procedure calls in assembly language.
- Implement recursive procedures in an assembly language.

General Instructions

- For the micro-programming problems, you do not have to re-produce the micro-program in Figure 5.7.3. Assume that that part of the micro-program already exists and write down only the new micro-instructions that are required. Also, if you create any new values for the fields or new fields, clearly specify what each new value/field denotes.
- Submit your answers on a separate sheet of paper.
- The state diagram for the FSM representation of the control unit (Figure 5.38) is provided on the class website.
- Figures 5.7.2 and 5.7.3 that deal with micro-programming are provided on the class website.

Problems

1. [25 points] Figure 5.7.3 in Hennessy and Patterson (the CD actually), shows the microprogram for the control unit of the MIPS architecture that can handle `lw, sw, beq, j` and R-type instructions. Add microcode to this program to implement the following MIPS instructions:
   a. `bne`
   b. `mfc0`
   c. `mtc0`
   d. `jal`
   e. `lwr` - The description is provided below.

   `lwr rd, rs, rt`
   The instruction must read a value from memory and store it in register `rd`. The address, of the value in memory, is determined by adding the contents of registers `rs` and `rt`.

   Figure 5.7.2 on the CD lists the control fields and their values that the microprogram uses.

Rose-Hulman Institute of Technology
Computer Science and Software Engineering
Profs. Archana Chidanandan and J.P. Mellor

2. [25 points] For each of the problems below, modify the textbook's multi-cycle control (Figure 5.38 on page 345) to implement the indicated MIPS instruction. Use a separate printout for each of the instructions. A copy of Figure 5.38 is provided on the class website.

   a. `bne`
   b. `mfc0`
   c. `mtc0`
   d. `jal`
   e. `lwr` - The description is provided with problem 1.

   Look for problem 3 on the next page.

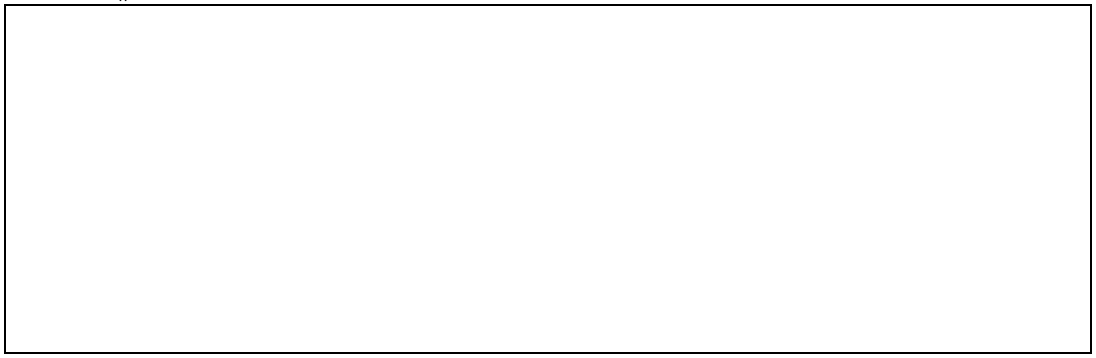
3. [10 points] The incomplete MIPS procedure below computes the recursive function

	{	*b*, if *a = 0*
foo(a,b) =		
		*foo(a-1,b-a) * b*, otherwise

Finish the procedure by adding the code for the procedure entrance, the recursive procedure call, and the procedure exit. Do not modify any of the given code, and be sure to follow the MIPS convention for register usage.
Hint: `$t0` and `$s0` hold the local values of *a* and *b*, respectively.

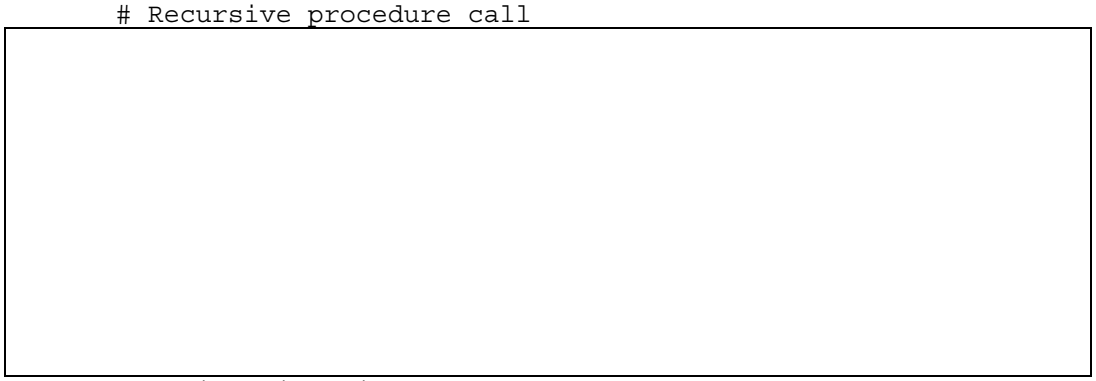
You may write the answer to this question in the boxes provided and attach the sheets with the rest of your solutions.

```
foo:
      # Procedure entrance
```



```
      # if (a == 0) return b
      bne  $t0, $zero, Else
      move $t1, $s0
      j    Exit

      # otherwise, return foo(a-1,b-a)*b
Else: sub  $t2, $t0, 1
      sub  $t3, $s0, $t0
      # Recursive procedure call
```



```
      mul  $t1, $t1, $s0

Exit: move $v0, $t1
      #Procedure exit
```

Rose-Hulman Institute of Technology
Computer Science and Software Engineering
Profs. Archana Chidanandan and J.P. Mellor