Rose-Hulman Institute of Technology
Computer Science and Software Engineering
Prof. Archana Chidanandan

Homework 4
RTL, Datapath and Control Design
Total points: 25 points + 10 points (extra credit)

This assignment is due Wednesday, 28th January 2003 for Sections 1 and 2 at the beginning of class.

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.
- Describe the implementation of machine language instructions using Register Transfer Language.

General Instructions

- The pdf files for Figures 5.33, 5.29 and 5.42 are available on the class web-site(Homework).
- Submit your solutions on a separate sheet of paper.

Problems

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

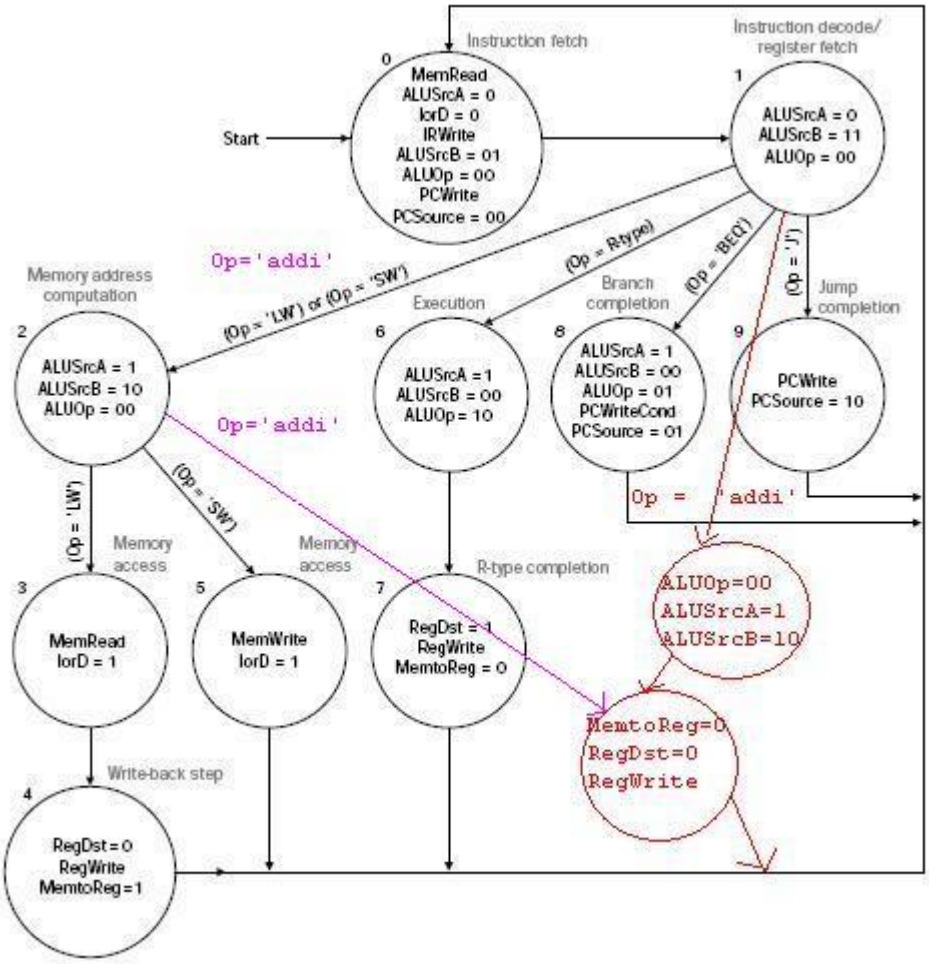
    a. Modify the control to implement the MIPS `addi` instruction.

    The RTL description is being provided here for clarity:

```
1. IR = Mem[PC]; PC = PC + 4
2. A = Reg[IR[25:21]] ; B = Reg[IR[20:16]]
   ALUOut = PC + [SE[IR[15:0]] << 2]
   If (IR[31:26] == 8) then
3. Sum = A + SE[15:0]
4. Reg[IR[20:16]] = Sum
```

    There are no changes required to the datapath on Figure 5.33.

The modified control unit state diagram is shown below, with 2 possible solutions:



b. Modify the control to implement the MIPS `mfc0` instruction.
c. Modify the control to implement the MIPS `mtc0` instruction.

The RTL for the mfc0 and mtc0 instructions are provided for clarity:

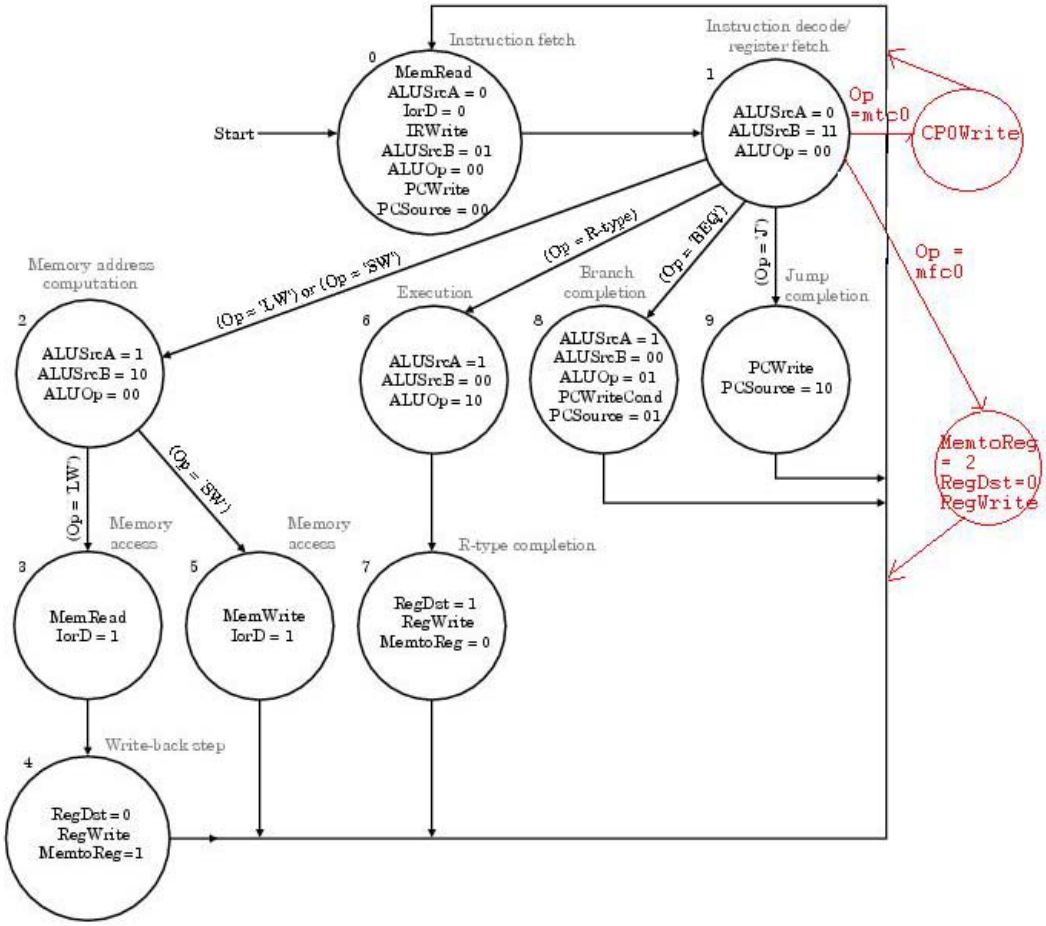
i. `mfc0 rt rd` - Move the contents of co-processor 0's register rd to register rt.

```
1. PC = PC + 4; IR = Mem[PC]
2. A = Reg[IR[25:21]];   B = Reg[IR[20:16]];
   ALUOut = PC + (SE[IR[15:0]] << 2);
   D = CoP0Reg[IR[15:11]];
   If((IR[31:26]==16)and(IR[25:21]==0)) then
3. Reg[IR[20:16]] = D;
```

ii. `mtc0 rt rd` – Move the contents of register rd to co-processor 0's register rt.

```
1. PC = PC + 4; IR = Mem[PC];
2. A = Reg[IR[25:12]];   B = Reg[IR[20:16]];
   ALUOut = PC + (SE[IR[15:0]] << 2);
   D = CoP0Reg[IR[15:11]];
   If((IR[31:26]==16)and(IR[25:21]==4))then
3. CoP0Reg[IR[15:11]] = B;
```

The modified datapath diagrams are available in HW 3. The modified state diagram for the control unit is shown below:



Control modified to implement the mfc0 and mtc0 instructions

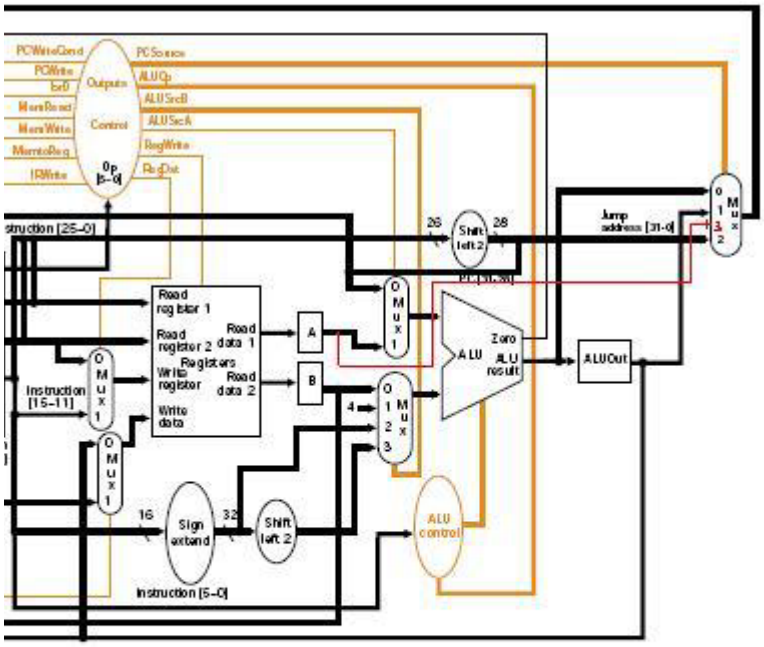
2. Figure 5.33 of Patterson & Hennessy, shows a complete datapath for a multicycle implementation of most R-Type MIPS assembly language instructions, as well as `lw`, `sw`, `beq`, and `j`. For this question, you are required to modify the datapath to include the multicycle implementation of the `jr` instruction. Specifically, you must:

   b. [4 points] Write a multicycle RTL description of an implementation of the `jr` instruction that uses as few cycles as possible without extending the clock cycle of your design.

   RTL description for the `jr` instruction:

```
1. PC = PC + 4; IR = Mem[PC]
2. A = Reg[IR[25:21]];   B = Reg[IR[20:16]];
   ALUOut = PC + (SE[IR[15:0]] << 2);
   D = CoP0Reg[IR[15:11]];
   If((IR[31:26]==0)and(IR[5:0]==8)) then
3. PC = A
```

   c. [6 points] Add any necessary datapath and control signals to the multicycle datapath to implement the `jr` instruction.



Datapth modified to include the "jr" instruction. No changes are required for any of the control signals.

3. [Extra credit: 10 points]
   a. [2.5 x 2 = 5 points]Write the single-cycle RTL descriptions for the for the MIPS instructions `jal` and `lui`

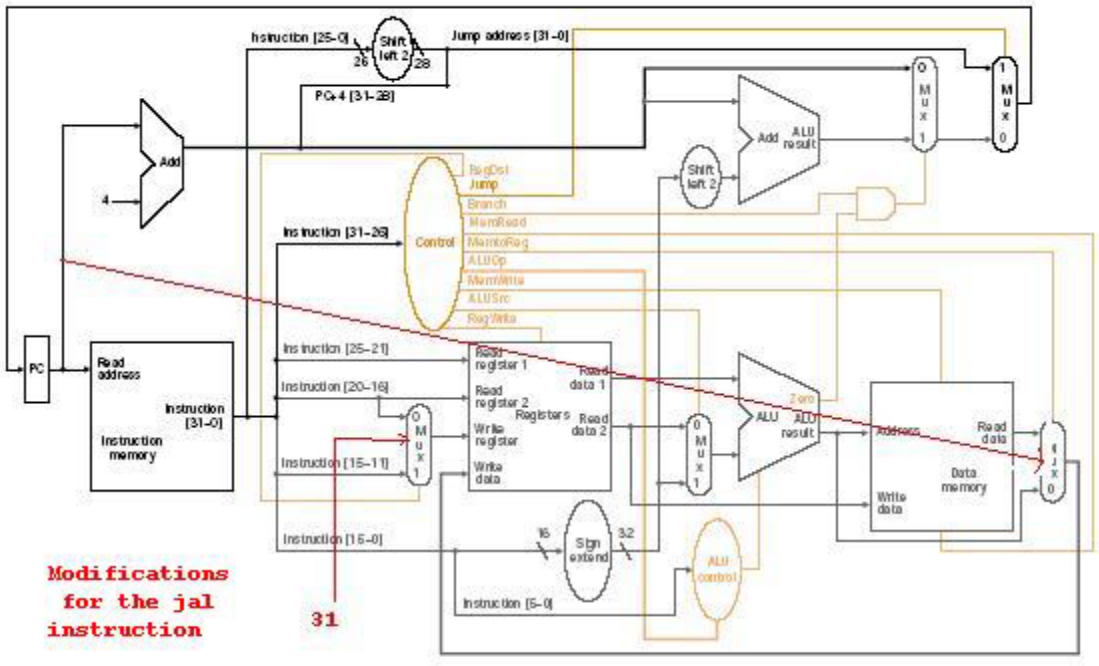
RTL for `jal`:

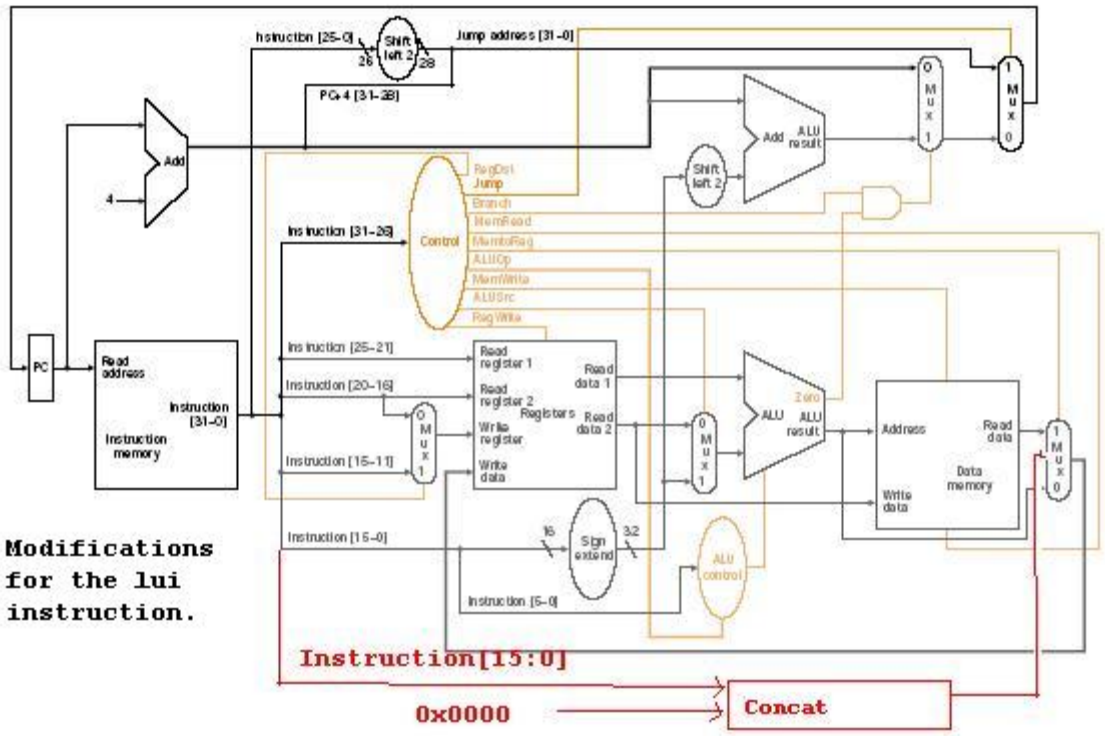
```
1.  if[(Mem[PC])[31:26] == 3) then
      PC = PC[31:28]|| (Mem[PC])[25:0]||00
      Reg[31] = PC
```

RTL for `lui`:

```
1.  if((Mem[PC])[31:26] == 15) then
        Reg[(Mem[PC])[20:16]] =
                (Mem[PC])[15:0]||0000000000000000
            PC = PC + 4
```

   b. [2.5 x 2 = 5 points] Modify the datapath in Figure 5.29 on page 372 of Hennessy and Patterson, to implement the `jal` and `lui` instructions. Use two different copies of the datapath to make the changes for each of the two instructions.





Control signal MemtoReg has to be expanded to 2 bits.