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

Homework 3 - Solutions
(RTL, Datapath and Exception Handling)

Problems

1. Figure 5.33 of Patterson & Hennessy, shows the complete datapath for the multicycle implementation of MIPS assembly language instructions `add, lw, sw, beq` and `j`. For this question, you are required to modify and augment the datapath, to include the multicycle implementation of the `lui` instruction. Specifically, you must

   a. [5 points] Write a multicycle RTL description of an implementation of the `lui` instruction that uses as few cycles as possible without extending the clock cycle of your design.

```
1. PC = PC + 4  ;  IR = Mem[PC]
2. A = Reg[IR[25:21]];   B = Reg[IR[20:16]];
   Sum = PC + (SE[IR[15:0]] << 2);
   If (IR[31:26] = = 15) then
3. Reg[IR[20:16]] = IR[15:0] || 0x0000
```

(|| = Concatentate – not OR)

   b. [5 points] List all new and modified components required for the implementation of the `lui` instruction. Also, list the inputs and outputs for each component along with any control signals that are required.

There are no new components added. No components are modified.

   c. [5 points] Add any necessary datapaths and control signals to the multicycle datapath of Figure 5.33 of Patterson & Hennessy. You can photocopy existing figures or download figures from www.mkp.com/cod2e.htm to make it easier to show your modifications.

2. [15 points] Repeat Steps 1a, 1b and 1c for the `mfc0` and `mft0` MIPS instruction (combine the component lists and the datapath modifications).

   a. RTL description for `mfc0` and `mtc0`

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

      2. `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[20:26]];
   Sum = PC + (SE[IR[15:0]] << 2);
   If ((IR[31:26] = = 0) and (IR[25:21] = = 4))
   then
3. CoP0Reg[IR[15:11]] = B;
```

   b. New and modified components list

Component	Inputs	Outputs	Control signals
Co-processor 0 Register file	$CP0WriteData_{31:0}$, $CP0ReadAddr_{4:0}$, $CP0WriteAddr_{4:0}$	$CP0DataOut_{31:0}$	CP0Write
D	$DIn_{31:0}$	$DOut_{31:0}$	-

3. [15 points] Repeats Steps 1a, 1b and 1c for an "undefined" instruction in MIPS. *Hint:* Save PC-4 in EPC, modify the PC and update the Status Register. Read through pages 410-413 of Patterson and Hennessy for more information.
   a. RTL description for an "undefined" instruction

```
1. PC = PC + 4; IR = Mem[PC];
2. A = Reg[IR[25:21]];   B = Reg[IR[20:16]];
   Sum = PC + (SE[IR[15:0]] << 2);
   If (IR[31:26] != …) then
3. If(Status[0] == 1)
   Sum = PC – 4; PC = 0x80000080;
   Cause = Cause[31:6] || xxxx || Cause[1:0]
   Status = Status[31:6] || Status [3:0] || 00
4. EPC = Sum
```

   b. New and modified components list

Component	Inputs	Outputs	Control signals
Cause Register	$CauseIn_{31:0}$	$CauseOut_{31:0}$	CauseWrite
EPC	$EPCIn_{31:0}$	$EPCOut_{31:0}$	EPCWrite
Status Register	$SRIn_{31:0}$	$SROut_{31:0}$	SRWrite

4. [5 points] In question 1 of this assignment, you wrote the RTL description for the `lui` instruction. For question 4, you must modify the RTL description for `lui`, to handle an interrupt i.e. an unexpected event that occurs outside the processor, and causes a change in the flow of execution of the program. *Note:* You do not need to list the new and modified components, nor do you need to show the modified datapath.

   RTL description for `lui` instruction, with interrupt handling:

```
1. If((Cause[15:11] AND Status[15:11])  != 0) and
   Status[0] == 1)
   PC = 0x80000080;
   Cause = Cause[31:6] || 0000 || Cause[1:0]
   Status = Status[31:6] || Status [3:0] || 00
   EPC = PC

2. Same as steps 1-3 of Q1a.
```