Systems Architecture I CS 281 Final Exam Fall 2020-21
Professor: C. Katsinis

Name:_____

Instructions:

- **The exam must be completed in 2 hours.**
- **The exam has TWO problems and is worth 100 points.**
- **The exam is closed-book but you may use the MIPS Reference sheets, the pseudoinstructions sheet and one page of your notes.**
- **Your answers must fit in ONE pdf file.**
  - **If you need to draw something, you can copy a page from the exam pdf file, edit it and include it in your pdf file with your answers, OR**
  - **Print the page with the diagram, write on it clearly, take a photo with your cell phone and include it in your pdf file.**
  - **Remember, the BBLearn exam page accepts only one pdf file.**
- **Make the name of this file equal to your last name followed by your first name.**

Problem 1 (50 points).

Start with the figure of the **non-pipelined (single-cycle) processor** and add the necessary hardware to execute the instruction

jalr rs

the operation of which is

ra = pc + 4; pc = rs

- Do not be concerned with the fact that the reference sheet shows pc+8 and assume pc+4.
- Describe any new, updated or deleted hardware, data lines and control signals. CLEARLY place your updates directly on the figure and describe your changes.
- Make sure you do not change the earlier functionality of the processor (the processor should still be able to correctly execute the instructions we studied in class).

Problem 2 (50 points).

As you have noticed in our study of pipelines, hazard occur frequently. For example, compiling the following C code (which uses an array of pointers to store values in memory)

```
test1 ()
 {
  int i, c, *p, *data[10];
  c = 13;
  i = 0;
  while (i < 10)
   {
    p = data[i];
    *p = c;
    i++;
   }
 }
```

produces a code fragment like this

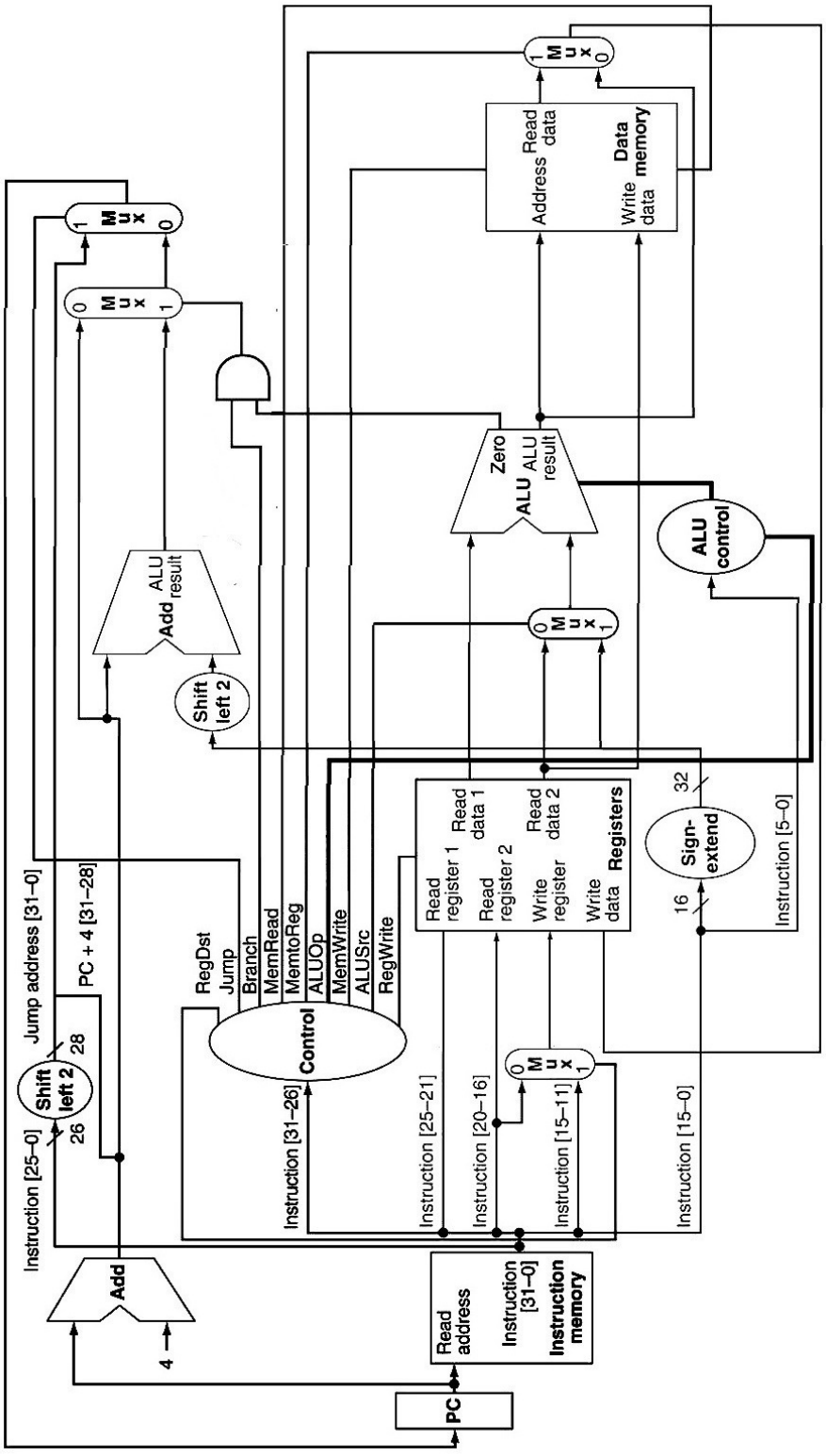
```
$L2:
        lw      $s0,0($v0)     # p = data[i];
        addiu   $v0,$v0,4      #
        sw      $v1,0($s0)     # *p = c;
        bne     $v0,$a1,$L2    #
```

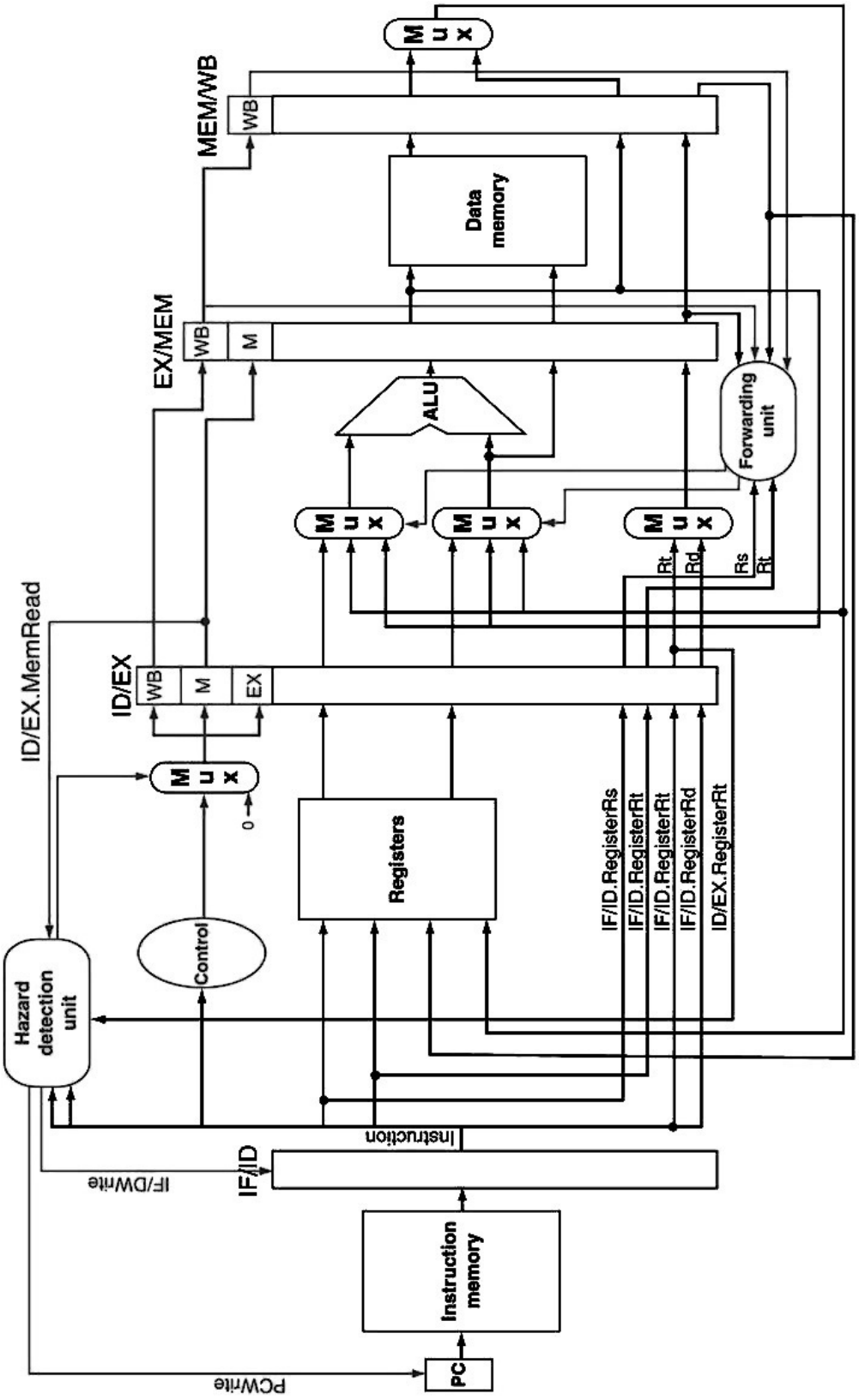
Notes:

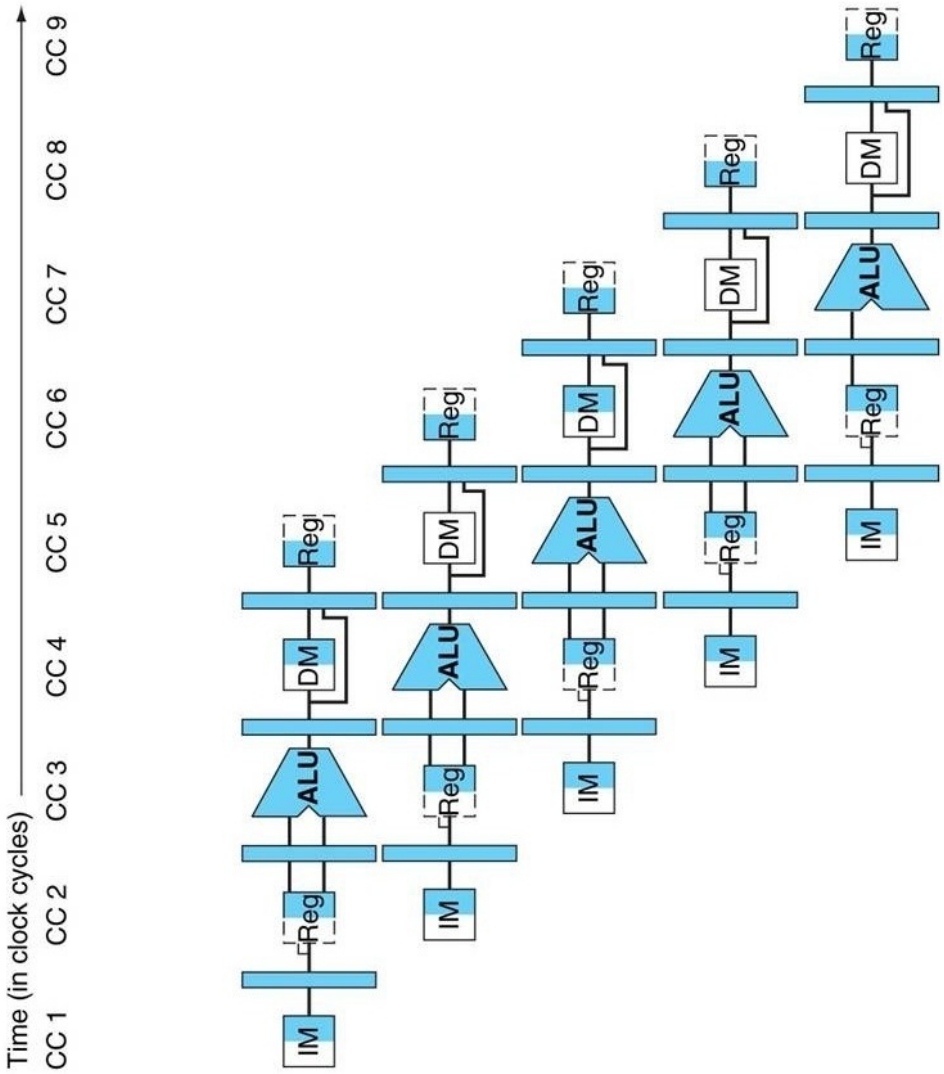
- This is the code that represents the loop
- register v0 points to individual elements of array data[]
- the loop terminates when v0 points to the element after the last one in data[] (register a1 points to the element after the last one in data[])

Questions:

1. Explain the progress of the lw and sw instructions in the pipeline.
2. Normally, the addiu instruction should appear after the sw instruction, since the i++ follows the *p=c. Why does the compiler place the addiu instruction between the lw and sw instructions?
3. Does this loop execute correctly on our latest pipelined processor design (Figure 4-60)? Explain in detail.
4. If not, fix the hardware of the pipelined processor design starting with Figure 4-60, so that the lw and sw instructions in the loop are executed correctly. Show additional hardware in Figure 4-60 and explain the logical operations in detail. Make sure not to break earlier functionality.


Jump address [31–0]
Instruction [25–0]
Shift left 2
26
28
PC + 4 [31–28]
Add
4
PC
Read address
Instruction [31–0]
Instruction memory
RegDst
Jump
Branch
MemRead
MemtoReg
ALUOp
MemWrite
ALUSrc
RegWrite
Control
Instruction [31–26]
Instruction [25–21]
Instruction [20–16]
Instruction [15–11]
Instruction [15–0]
Instruction [5–0]
Read register 1
Read register 2
Write register
Write data
Read data 1
Read data 2
Registers
16
Sign-extend
32
ALU
Add
ALU result
Shift left 2
Zero
ALU
ALU result
ALU control
Address
Read data
Write data
Data memory
M u x
0
1


M u x
MEM/WB
WB
Data memory
EX/MEM
WB
M
ALU
Forwarding unit
M u x
M u x
M u x
Rt
Rd
Rs
Rt
ID/EX.MemRead
ID/EX
WB
M
EX
M u x
0
Registers
IF/ID.RegisterRs
IF/ID.RegisterRt
IF/ID.RegisterRt
IF/ID.RegisterRd
ID/EX.RegisterRt
Control
Hazard detection unit
Instruction
IF/DWrite
IF/ID
Instruction memory
PC
PCWrite


Time (in clock cycles)
CC 1
CC 2
CC 3
CC 4
CC 5
CC 6
CC 7
CC 8
CC 9
IM
Reg
ALU
DM
Reg