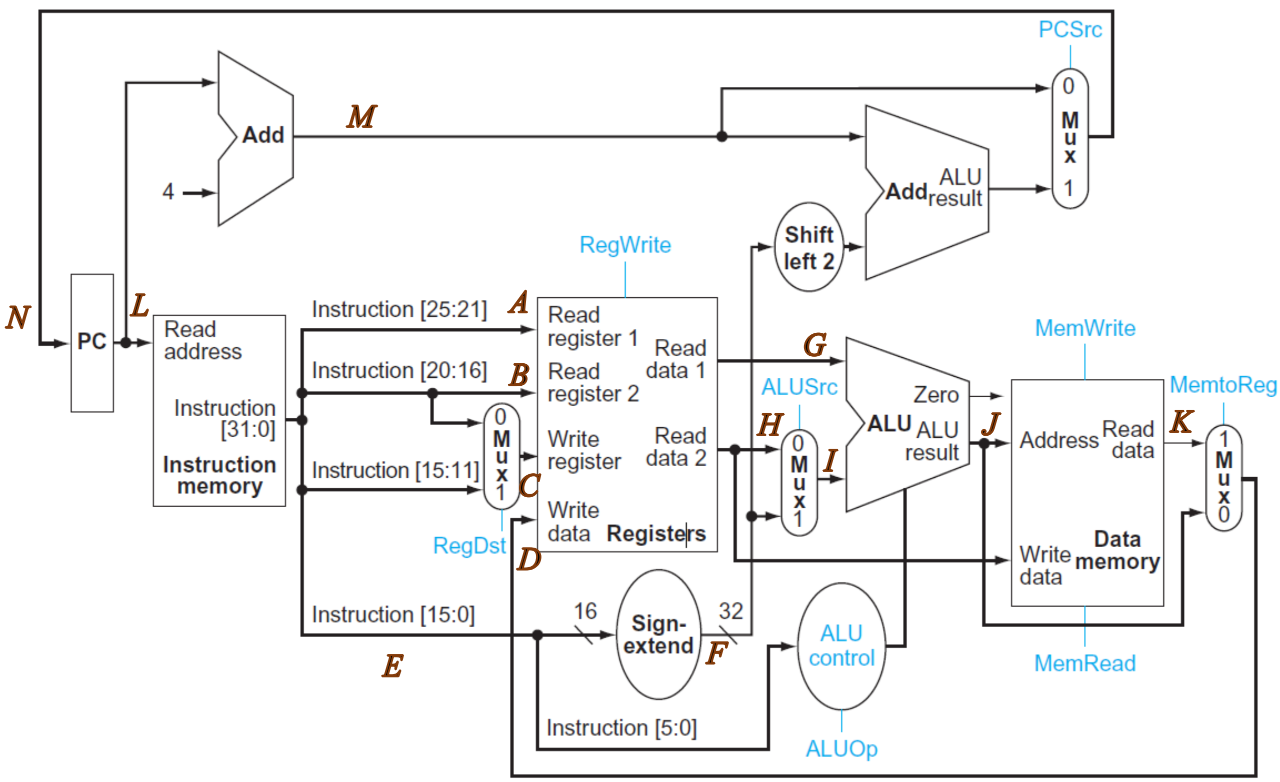
Practice Questions (and Answers) for Final Exam
CS 154, Winter 2020, Matni

IMPORTANT NOTE: These questions are NOT representative of EVERYTHING you need to study for the final exam! First of all, this is focused on post-midterm material. You should also review your midterm practice exam questions, all lab assignments questions and all class materials (lecture slides), including the examples and demos starting from the start of the quarter. The textbook is an important source of material and extra questions.

Given the following basic single-cycle CPU model, answer the following questions:



1. Given that: (a) registers **$a0 = -128** and **$s0 = 130**, and (b) the instruction at the input is **add $t0, $s0, $a0**, and (c) the instruction is in memory address **0x00404400**,
    a. what are the values for wires/buses **A** thru **N**? Give all your answers in an appropriately-sized hexadecimal.
    b. what are the values of the controls **RegDst**, **RegWrite**, **ALUSrc**, **Zero**, **MemRead**, **MemWrite**, **MemtoReg**, **PCSrc**?
    c. what operation is the ALU performing (not the code, but the function)?
2. Same as (1) for the next instruction in memory, which is **addi $v0, $a3, 510**, where **$v0** = 3, **$a3** = 2.
3. Same as (2) for the next instruction in memory, which is **sw $t3, 44($t4)**, where **$t3** = 7, **$t4** = 0x70000004.

4. Assume that logic blocks needed to implement a processor's datapath, like the one in the figure above in Question 1, have the following latencies:
   Inst.Mem = 200 ps
   Add = 70 ps
   Mux = 20 ps
   ALU = 90 ps
   Regs = 90 ps
   DataMem. = 250 ps
   Sign-Ext. = 15 ps
   Shift-Left-2 = 10 ps

   a. If the only thing we need to do in a processor is fetch consecutive instructions, what would the cycle time be?
   b. What would the cycle time be for an **add** instruction? for an **andi** instruction? for a **sw** instruction?
   c. Could we utilize a clock whose speed is 2.0 GHz for this design? Why or why not?

5. Assume a single-cycle design CPU fetches the following instruction word: **10101100011000100000000000010100**, which resides in memory address **0x00511234**. Assume that data memory is all zeros and that the processor's registers (as referenced by their MIPS register numbers) have the following values at the beginning of the cycle in which the above instruction word
   is fetched:
   r0 = 0, r1 = -1, r2 = 2, r3 = 3, r4 = -4, r5 = 10, r6 = 6, r8 = 8, r12 = 2 r31 = -16

   a. What is the MIPS instruction that was fetched?
   b. What function, if any, is the ALU performing for this instruction?
   c. What is the new PC address after this instruction is executed?
   d. What are the values of all 4 data inputs for the "Registers" unit? Give your answer in appropriate-length hexadecimals and use **X** for don't care values.

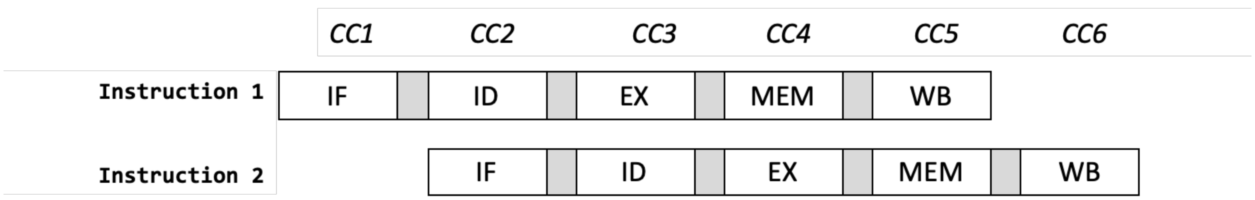
6. Assume that the individual stages of a MIPS CPU datapath have the following latencies:
   IF = 250 ps, **ID** = 350 ps, **EX** = 150 ps, **MEM** = 300 ps, **WB** = 200 ps

   a. What is the clock cycle time in a pipelined and non-pipelined processor?
   b. What is the approximate clock cycle speed-up from non-pipelined to pipelined?

7. Assume a 5-stage pipelined MIPS datapath that takes the following instructions:

```
add $t5, $t2, $t1
lw $t3, 4($t5)
lw $t2, 0($t2)
or $t3, $t5, $t3
sw $t3, 0($t5)
```

   a. Identify all hazards (all types).
   b. If the hardware does not have hazard or forwarding detection, (manually) insert **nop** instructions to ensure correct execution.
   c. Draw a multiple-clock-cycle diagram like the generalized one shown here (or something like **Figure 4.52**, on **Page 305** in the book) and show all the forwards and stalls, if any, needed to have in order to correctly execute these instructions.



ANSWERS:

See table for answers for Questions 1, 2, and 3:

Wire/Bus Control	**Instruction 1** `add $t0, $s0, $a0`	**Instruction 2** `addi $v0, $a3, 510`	**Instruction 3** `sw $t3, 44($t4)`
A	0x10	0x07	0x0C
B	0x04	0x02	0x0B
C	0x08	X	X
D	0x00000002	0x00000201	X
E	X	510 = 0x01FE	44 = 0x002C
F	X	0x000001FE	0x0000002C
G	130 = 0x00000082	0x00000002	0x70000004
H	-128 = 0xFFFFFF80	0x00000003	0x00000007
I	0xFFFFFF80	0x000001FE	0x0000002C
J	0x00000002	0x00000201	0x700000030
K	X	X	X
L	PC = 0x00404400	PC = 0x00404404	PC = 0x00404408
M	PC + 4 = 0x00404404	PC + 4 = 0x00404408	PC + 4 = 0x0040440C
N	PC + 4 = 0x00404404	PC + 4 = 0x00404408	PC + 4 = 0x0040440C
RegDst	1	0	0
RegWrite	1	1	0
ALUSrc	0	1	1
Zero	0	0	0
MemRead	0	0	0
MemWrite	0	0	1
MemtoReg	0	0	X
PCSrc	0	0	0
ALU Op type	Add	Add	Add

1.
    a. Add + Mux = 90 ps
    b. For **add**: IM + Reg + Mux + ALU + Mux = 420 ps
       For **andi**: IM + Reg + SignExt + Mux + ALU + Mux = 435 ps
       For **sw**: IM + Reg + SignExt + Mux + ALU + DM + Mux = 685 ps
    c. No, because a 2.0 GHz clock has a clock cycle of 500 ps which is too short for all instructions to work with.
2.
    a. `sw $v0, 20($v1)`
    b. Add
    c. PC + 4 = 0x00511238
    d. For "Read Reg1" (rs), the value is 3 in 5-bits = 0x03.
       For "Read Reg2" (rt), the value is 2 in 5-bits = 0x02.
       For "Write Reg" (rt), the value is 2 in 5-bits = 0x02.
       For "Write Data", there is no value that we care about, so X.
3.
    a. Non-pipelined is 1250 ps (sum of all stages' latencies) and pipelined is 350 ps (same as the longest stage latency).
    b. About 3.6
4.
    a. The $t5 reg between add and the 1st lw.
       The $t3 reg between the 1st lw and the or.
       The $t3 reg between the or and the sw.

```
add $t5, $t2, $t1
lw $t3, 4($t5)
lw $t2, 0($t2)
or $t3, $t5, $t3
sw $t3, 0($t5)
```

    b. If there is NO forwarding, then we have to insert bubble delays (**nop** instructions):
       3 **nop**s between **add** and 1st **lw**
       2 **nop**s between 2nd **lw** and **or**
       3 **nop**s between **or** and **sw**

c. 3 forwards done, no stalling:
forward \$t5 from EX/MEM pipeline reg to EX input (**add** to 1st **lw**)
forward \$t3 from MEM/WB pipeline reg to EX input (1st **lw** to **or**)
forward \$t3 from EX/MEM pipeline reg to EX input (**or** to **sw**)