6.004 Tutorial Problems
L20 – Control Hazards in Pipelined Processors

Problem 1. ★

The loop on the right has been executing for a while on our standard 5-stage pipelined RISC-V processor with branch annulment and full bypassing.

```
    …
L1: addi x10, x10, -4
    slti x11, x10, 10
    beqz x11, L2
    lw x12, 0x200(x10)
    j L3
L2: lw x12, 0x300(x10)
L3: sw x12, 0x400(x0)
    bnez x10, L1
    addi x12, x12, 1
    xor x12, x12, x0
    …
```

Cycle #	*300*	*301*	*302*	*303*	*304*	*305*	*306*	*307*	*308*	*309*
IF	addi									
DEC										
EXE										
MEM										
WB										

(A) **Fill in the pipeline diagram** for cycles 301-309 assuming that at cycle 300 the instruction at L1 is fetched. Also, assume that the branch to L2 is taken, as well as the final branch back to L1. Finally, assume that the value for x10 is available in the register file prior to cycle 300. **Indicate which bypass/forwarding paths are active in each cycle by drawing a vertical arrow in the pipeline diagram** from pipeline stage X in a column to the RF stage in the same column if an operand would be bypassed from stage X back to the RF stage that cycle. Note that there may be more than one vertical arrow in a column.

Fill in pipeline diagram including bypass arrows in pipeline diagram above

(B) Assume that the previous iteration of the loop executed the same instructions as the iteration shown in part (A). Please complete the pipeline diagram for cycle 300 by filling in the OPCODEs for the instructions in the DEC, EXE, MEM, and WB stages.

Fill in OPCODEs for Cycle 300

(C) Indicate which branches are taken by providing the cycle in which the taken branch instruction enters the IF stage.

Cycle number(s) or NONE: _______________

(D) During which cycle(s), if any, do we have stalled instructions?

Cycle number(s) or NONE: _______________

Now consider a modified processor, P2, which has extra hardware in the decode stage (DEC) to resolve simple branches one cycle earlier: the decode stage includes both a circuit to check whether a register is equal to zero, and an extra adder to compute the branch target for taken branches. This processor can thus compute nextPC for `beqz` and `bnez` in DEC instead of EXE.

(E) Redo part A using processor P2 assuming the same path is taken through the code.

Cycle #	*300*	*301*	*302*	*303*	*304*	*305*	*306*	*307*	*308*	*309*
IF	`addi`									
DEC										
EXE										
MEM										
WB										

(F) Compare the number of cycles per loop iteration using the original processor and the modified processor.

Cycles per loop in original processor: _______________

Cycles per loop in processor P2: _______________

Problem 2. ★

You've discovered a secret room in the basement of the Stata center full of discarded 5-stage pipelined RISC-V processors. Unfortunately, many have certain defects. You discover that they fall into four categories:

C1: A modified, completely functional 5-stage RISC-V processor with working bypass paths, annulment, and other components, as well as extra hardware support that allows the nextPC to be calculated in the Decode stage.
C2: A defective version of **C1** with a bad register file: all data read from the register file is zero.
C3: A defective version of **C1** with broken bypass muxes: all source operands come from the register file, even if they should be read from bypassed paths.
C4: A defective version of **C1** without annulment of instructions following branches.

To help sort the processors into the above classes, you write the following small test program:

```
. = 0x0
// Start at 0x0, with ZERO in all registers…
      addi x10, x0, 4
      jal x12, X
      slli x12, x12, 1
X:    addi x12, x12, -4
      add x13, x12, x10
      jr x13
```

Your plan is to single-step through the program using each processor, carefully noting the address the final `jr` loads into the PC. Your goal is to determine which of the above four classes a chip falls into by this `jr` address.

For each class of RISC-V processor described above, specify the value that will be loaded into the PC by the final `jr` instruction.

Pipeline diagram showing first 7 cycles of test program executing on C1:

cycle	*0*	*1*	*2*	*3*	*4*	*5*	*6*
IF	addi	jal	slli	addi	add	jr	
DEC		addi	jal	NOP	addi	add	jr
EXE			addi	jal	NOP	addi	add
MEM				addi	jal	NOP	addi
WB					addi	jal	NOP

C1: jr goes to address: __________

C2: jr goes to address: __________

C3: jr goes to address: __________

C4: jr goes to address: __________

Problem 3. ★

(A) How many cycles does it take to run each iteration of the following loop (assuming the beqz is always taken) on a standard 5-stage pipelined RISC-V processor?

```
loop: lw x10, 0x100(x0)
      beqz x10, loop
      add x12, x10, x11
      sub x13, x12, x1
```

Number of cycles per loop iteration: __________

cycle	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
IF											
DEC											
EXE											
MEM											
WB											

(B) Assuming a defective 5-stage pipelined RISC-V processor where the instructions following a taken branch are not annulled, which of the following statements would be true?

1. The add instruction would be executed each time through the loop.
2. The loop would take 5 cycles to execute
3. The value of the register x10 that is tested by the beqz instruction comes from a bypass path.
4. The value of register x10 that is accessed by the add instruction comes from the register file.

cycle	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
IF											
DEC											
EXE											
MEM											
WB											

(C) Consider a modified processor, P2, which has extra hardware for the special case of checking if a register is equal to zero or not in the decode stage. What would be the number of cycles per loop iteration in this case?

Number of cycles per loop iteration on processor P2: __________

cycle	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
IF											
DEC											
EXE											
MEM											
WB											

(D) Now consider a third processor, P3, whose instruction and data memories are pipelined and take 2 clock cycles to respond (instead of a single cycle as usual). Assume that P3 also has the extra hardware for checking if a register is equal to zero or not in the decode stage. What would be the number of cycles per loop iteration using P3?

Number of cycles per loop iteration on processor P3: __________

Cycle	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**
IF														
DEC														
EXE														
MEM														
WB														

Problem 4.

You've been given a 5-stage pipelined RISC-V processor. Unfortunately, the processor you've been given is defective: it has no bypass paths, annulment of instructions in branch delay slots, or pipeline stalls.

```
            nop
            nop
            nop
            nop

      loop:
            lw x10, 0x0(x10)
      AA:
            sll x14, x10, x11
      BB:
            bnez x10, loop
      CC:
            add x13, x10, x13

            nop
            nop
            nop
            nop
```

You undertake to convert some existing code, designed to run on an unpipelined RISC-V, to run on your defective pipelined processor. The fragment of code above is a sample of the program to be converted. Add the **minimum** number of **NOP** instructions at the various tagged points in this code to make it give the same results on your defective pipelined RISC-V as it gives on a normal, unpipelined RISC-V.

Note that the code fragment begins and ends with sequences of **NOPs**; thus, you don't need to worry about pipeline hazards involving interactions with instructions outside of the region shown.

(A) Specify the minimal number of **NOP** instructions (*defined as addi x0, x0, 0*) to be added at each of the labeled points in the above program.

NOPs at Loop: ______

NOPs at AA: ______

NOPs at BB: ______

NOPs at CC: ______

cycle	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
IF									
DEC									
EXE									
MEM									
WB									

(B) On a **fully functional** 5-stage pipeline (with working bypass, annul, and stall logic), the above code will run fine with no added **NOP**s. How many clock cycles of execution time are required by the fully functional 5-stage pipelined RISC-V **for each iteration** through the loop?

Clocks per loop iteration: ________

cycle	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*
IF								
DEC								
EXE								
MEM								
WB								

Problem 5

You are designing a four stage RISC-V processor (IF, DEC, EXE, WB). Currently you are trying to decide whether to include bypassing from the write-back stage to the decode stage. As part of this evaluation, you construct two processors:

Processor A: No bypassing from WB to DEC.
Processor B: Bypassing from WB to DEC.

You are using the following loop of an important program to evaluate the performance of the processor:

```
L1:   lw t0, 0(a0)
      add a1, a1, t0
      addi a0, a0, 4
      blt a0, a2, L1
```

For the following questions, assume this loop has been running for a long time.

(A) (4 points) How many cycles per loop iteration does the decode stage stall due to read after write hazards in the following cases?

Processor A decode stall cycles per iteration: _______________

Processor B decode stall cycles per iteration: _______________

(B) (2 points) How many cycles does this loop take to execute in the following cases?

Processor A cycles per iteration: _______________

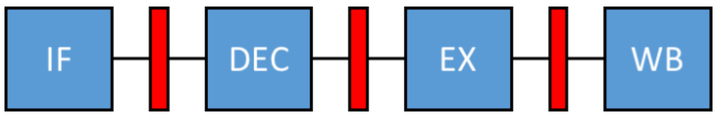
Processor B cycles per iteration: _______________

A	IF											
	DEC											
	EXE											
	WB											

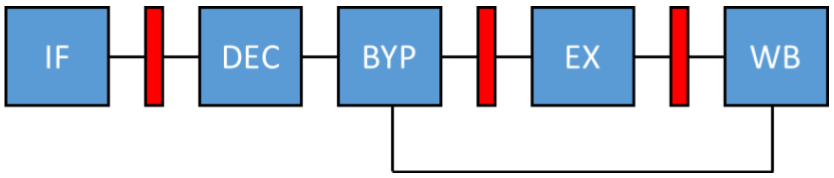
B	IF											
	DEC											
	EXE											
	WB											

Processor A has the following propagation delays for each of the pipeline stages:

IF: 2 ns
DEC: 3.0 ns
EX: 3.5 ns
WB: 1.0 ns



The logic for the bypassing path of processor B can be viewed as taking the output from the DEC and WB stages of processor A and adding an additional bypass logic (BYP) as shown in the picture below.



Assuming the BYP logic has a propagation delay of 1 ns and that all registers are ideal.

(C) (4 points) What is the minimum clock period for each processor?

Clock period for processor A: ________________

Clock period for processor B: ________________

(D) (4 points) For the loop shown above, what is the average cycles per instruction for the two processors:

Average cycles per instruction for processor A: ________________

Average cycles per instruction for processor B: ________________

(E) (4 points) For the loop shown above, what is the average number of instructions per second for the two processors:

Average number of instructions per second for processor A: ________________

Average number of instructions per second for processor B: ________________

Problem 6

Ben Bitdiddle and Alyssa P. Hacker are building a five stage RISC-V processor to help grade 6.004 exams. Their pipeline has the standard stages and functionality presented in lecture (IF, DEC, EXE, MEM, and WB). You may assume that instruction and data memories are single-cycle memories with clocked reads and writes.

The following code segment simulates counting the number of correct answers on a question. It loads a student answer, increments the count if its correct, and then loops back to the next student.

```
...
grade_question:
    lw t0, 0(a2)              // load a student's answer
    bne t0, a1, next_student  // check answer; assume bne is NOT TAKEN
    addi t1, t1, 1            // increment num correct
next_student:
    addi a2, a2, 4
    blt a2, a3, grade_question  // next student; assume blt is ALWAYS TAKEN
next_question:
    xor a4, a0, a4
    slli a4, a4, 2
...
```

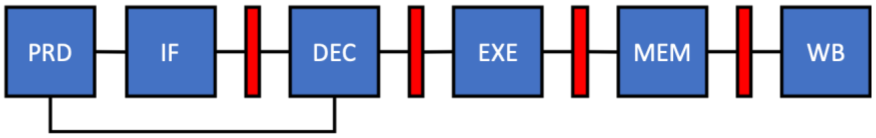
6.004 students are doing well on the exam and are all getting the questions right, so **the bne branch is never taken**. Also, the processor is in the middle of grading exams, so the **blt branch is always taken.**

Ben starts out with a fully functional 5-stage RISC-V processor with **full bypassing and annulment hardware**. Assume branch decisions are made in the EXE stage. Ben's processor always speculates that nextPC will be PC + 4.

Alyssa thinks she can achieve better performance with a smarter branch predictor to better handle loops. Her processor includes additional hardware to speculate that the branch is not taken (i.e. nextPC = PC + 4) on all forward branches (i.e. the branch target address is greater than the current PC) and that the branch is taken on all backwards branches. More formally:

$$nextPC = \begin{cases} PC + immB, & immB \le 0 \\ PC + 4, & immB > 0 \end{cases}$$

When the branch is detected in the DEC stage, the hardware will make its prediction based on immB and impact the instruction fetched **in that same cycle**. The following schematic shows the position of the prediction logic (PRD). The red bars represent the pipeline registers.



The following pipeline diagrams may be useful. You are not required to fill them out.
Ben's Pipeline Diagram

	0	1	2	3	4	5	6	7	8	9	10
IF											
DEC											
EXE											
MEM											
WB											

Alyssa's Pipeline Diagram

	0	1	2	3	4	5	6	7	8	9	10
IF											
DEC											
EXE											
MEM											
WB											

(A) (4 points) How many cycles does this loop take to execute in each of the processors?

(label: 5A_Ben) Ben's processor cycles per iteration: _________________

(label: 5A_Alyssa) Alyssa's processor cycles per iteration: _________________

(B) (4 points) Within one iteration of this loop, how many instructions need to be annulled due to incorrect speculation?

(label: 5B_Ben) Number of annulled instructions in Ben's proc: ________________

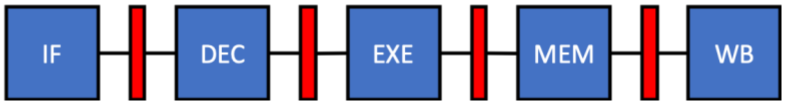
(label: 5B_Alyssa) Number of annulled instructions in Alyssa's proc: ________________

(C) (4 points) For this loop, what is the average cycles per instruction (CPI) for each of the processors?

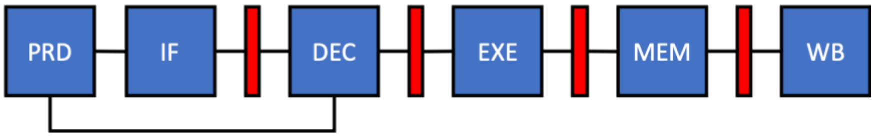
(label: 5C_Ben) Average CPI for Ben's processor: ________________

(label: 5C_Alyssa) Average CPI for Alyssa's processor: ________________

The logic for Ben's processor is shown below with the red bars representing pipeline registers.



The logic for the additional hardware that Alyssa added can be viewed as taking values from the DEC stage and adding additional prediction logic (PRD) as shown below (diagram copied from above).



Assume that all registers are ideal ($t_{SETUP} = t_{HOLD} = t_{PD} = t_{CD} = 0\ ns$) and each pipeline stage/piece of combinational logic has the following propagation delays.

PRD	1 ns
IF	3 ns
DEC	4 ns
EXE	7 ns
MEM	5 ns
WB	2 ns

(D) (4 points) What is the minimum clock period for each processor?

(label: 5D_Ben) Minimum clock period for Ben's processor (ns): ______________

(label: 5D_Alyssa) Minimum clock period for Alyssa's processor (ns): ______________

(E) (4 points) Which processor takes less time to execute one iteration of this loop, and how much faster is it?

(label: 5E_1) Processor that executes loop in less time (Ben's or Alyssa's):

____

(label: 5E_2) Number of nanoseconds difference (ns): ______________________