CS 383: Homework Assignment 3
Due: November 20, 11:55pm

Collaboration Policy. Homeworks will be done **individually**: each student must hand in their own answers. It is acceptable for students to collaborate in understanding the material but not in solving the problems or programming. Use of the Internet is allowed, but should not include searching for existing solutions.

Under absolutely no circumstances code can be exchanged between students. If some code was shown in class, it can be used, but it must be obtained from Canvas, the instructor or the TA.

Assignments from previous offerings of the course must not be re-used. Violations will be penalized.

Late Policy. No late submissions will be allowed without consent from the instructor. If urgent or unusual circumstances prohibit you from submitting a homework assignment in time, please e-mail me.

Deliverable. A single **pdf** file on Canvas.

Notes. The weight of each problem is in brackets. **Pay attention to the section of the textbook each problem is based on. The key difference in most cases is whether the processor is pipelined or not.**

Instruction Set For the problems below, unless otherwise specified, assume that the instruction set includes: (i) R-type instructions (ADD, SUB, AND, ORR), (ii) LDUR, (iii) STUR, (iv) CBZ, and (v) B.

Problem 1 (15 points) When silicon chips are fabricated, defects in materials (e.g., silicon) and manufacturing errors can result in defective circuits. A very common defect is for one signal wire to get broken and always register a logical 0. This is often called a stuck-at-0 fault. Answer the following questions based on **Section 4.4** of the textbook (see slide 19 in Chapter 4 on Canvas).

(a) [5] Which instructions fail to operate correctly if the MemToReg wire is stuck at 0?

(b) [5] Which instructions fail to operate correctly if the ALUSrc wire is stuck at 0?

(c) [5] Which instructions fail to operate correctly if the RegWrite wire is stuck at 0?

Problem 2 (15 points) Consider the addition of a multiplier to the CPU shown in Figure 4.23 (slide 19 in Chapter 4 on Canvas). This addition will add 300 ps to the latency of the ALU, but will reduce the number of instructions by 10% (because there will no longer be a need to emulate the multiply instruction). Answer the following questions based on **Section 4.4** of the textbook and the following table for the latencies of the stages of the pipeline.

IF	ID	EX	MEM	WB
250 ps	350 ps	150 ps	300 ps	200 ps

(a) [5] What is the clock cycle time with and without this improvement?

(b) [5] What is the speedup achieved by adding this improvement?

(c) [5] What is the slowest the new ALU can be and still result in improved performance?

Problem 3 (30 points) In this exercise, we examine how pipelining affects the clock cycle time of the processor. Questions in this problem assume that individual stages of the datapath have the latencies shown in Problem 2 above. Also, assume that instructions executed by the processor are broken down as follows:

ALU/Logic	Jump/Branch	LDUR	STUR
45%	20%	20%	15%

Answer the following questions based on **Section 4.5** of the textbook.

(a) [5] What is the clock cycle time in a pipelined and non-pipelined processor?

(b) [10] What is the total latency of an LDUR instruction in a pipelined and non-pipelined processor?

(c) [10] If we can split one stage of the pipelined datapath into two new stages, each with half the latency of the original stage, which stage would you split and what is the new clock cycle time of the processor?

(d) [5] Assuming there are no stalls or hazards, what is the utilization of the data memory?

Problem 4 (10 points) Consider the following loop.

```
LOOP: LDUR X10, [X1, #0]
      LDUR X11, [X1, #8]
      SUB X13, X11, X10
      ADD X12, X12, X13
      SUBI X1, X1, #16
```

```
CBNZ X12, LOOP
```

Assume that perfect branch prediction is used (no stalls due to control hazards) and that the pipeline has full forwarding support. Show a pipeline execution diagram for the first two iterations of this loop based on **Section 4.7** of the textbook.

Making a table is not required. Just align stages of difference instructions that take place concurrently.

Problem 5 (15 points) Questions in this problem refer to the following sequence of instructions, and assume that it is executed on a five-stage pipelined datapath:

```
ADD X5, X2, X1
LDUR X3, [X5, #4]
LDUR X2, [X2, #0]
ORR X3, X5, X3
STUR X3, [X5, #0]
```

Answer the following questions based on **Section 4.7** of the textbook.

(a) [5] If there is no forwarding or hazard detection, insert NOPs to ensure correct execution.

(b) [10] Now, change and/or rearrange the code to minimize the number of NOPs needed. You can assume register X7 can be used to hold temporary values in your modified code.

Problem 6 (15 points) This problem examines the accuracy of various branch predictors for the following repeating pattern of branch outcomes: T, NT, T, NT, NT. Answer the following questions based on **Section 4.8** of the textbook.

(a) [5] What is the accuracy of always-taken and always-not-taken predictors for this sequence of branch outcomes?

(b) [10] What is the accuracy of the 2-bit predictor for this pattern, assuming that the predictor starts off in the bottom left state from Figure 4.62 (predict not taken)? Show the state of the predictor at each step.