Homework 5

● Graded

Student

Ashley Leora Cain

Total Points

100 / 100 pts

Question 1

Instruction Replacement **14** / 14 pts

1.1 **Execution Time Improvement** **14** / 14 pts

✔ **– 0 pts** Correct (6.83)

– 2 pts Correct formula and numbers, but incorrect final answer

– 4 pts Incorrect old execution time (3.22N)

– 7 pts Incorrect new execution time (3N)

– 5 pts Incorrect formula for improvement ((old - new) / old)

– 2 pts Answer not in percentage (6.83)

– 5 pts Other minor mistake

– 14 pts Incorrect

1.2 **Work (Optional)** **0** / 0 pts

✔ **+ 0 pts** Graded

Question 2

Instruction Improvement **20** / 20 pts

2.1 **New Execution Time** **10** / 10 pts

- **– 0 pts** Correct

- ✔ **– 0 pts** Correct (158.5)

- **– 2 pts** Correct formula and numbers, but incorrect final answer

- **– 5 pts** Incorrect formula (should be Amdahl's Law)

- **– 5 pts** Does not consider that 34% of execution time is reduced by 1/3 (incorrect numbers in formula)

- **– 10 pts** Incorrect

- **– 7 pts** Attempted to use correct formula, but majority of values enter were incorrect

2.2 **Speedup** **10** / 10 pts

- **– 0 pts** Correct

- ✔ **– 0 pts** Correct (1.29)

- **– 2 pts** Correct formula with correct numbers, but incorrect final answer

- **– 5 pts** Incorrect formula (should be speed-up)

- **– 5 pts** Incorrect numbers in formula (allow for follow-through points)

- **– 10 pts** Incorrect

2.3 **Work (Optional)** **0** / 0 pts

- ✔ **+ 0 pts** Correct

Question 3

Pipeline Buffers **24** / 24 pts

3.1 **FBUF Contents** **6** / 6 pts

✔ **– 0 pts** Correct

– 1 pt Includes PC

– 2 pts Selected everything contained within the fetched instruction ($Rx, $Ry, Immediate Value)

Can also have "fetched instruction" selected

– 6 pts Decodes instruction (contents in $Ry, etc.)

– 6 pts Incorrect

– 1 pt Includes one extra answer choice

3.2 **DBUF Contents** **6** / 6 pts

✔ **– 0 pts** Correct

– 1 pt Does not include Contents in $Ry

– 1 pt Does not include Register number of $Rx

– 1 pt Does not include Opcode

– 1 pt Does not include Immediate Value

– 1 pt Includes one extra answer choice

– 2 pts Includes two extra answer choices

– 3 pts Includes 3 extra answer choices

– 1 pt Included PC

– 6 pts Incorrect

3.3 **EBUF Contents** **6** / 6 pts

✔ **– 0 pts** Correct

– 1 pt Does not include Register number of $Rx

– 1 pt Does not include Opcode

– 1 pt Does not include Result of ALU operation

– 1 pt Includes one extra component

– 2 pts Includes two extra components

– 3 pts Includes three extra components

– 6 pts Incorrect

3.4 MBUF Contents — 6 / 6 pts

✔ **– 0 pts** Correct

– 1 pt Does not include Register number of $Rx

– 1 pt Does not include Opcode

– 1 pt Does not include Value read from memory

– 1 pt Includes one extra component

– 2 pts Includes two extra components

– 6 pts Incorrect

Question 4

Pipeline Branching — **22** / 22 pts

4.1 Conservative Branching — 6 / 6 pts

✔ **– 0 pts** Correct

– 6 pts Incorrect

– 6 pts Incorrect/blank/no answer

4.2 Branch Prediction — 6 / 6 pts

✔ **– 0 pts** Correct (3)

– 0 pts Correct

– 6 pts Incorrect

4.3 Branch Mis-prediction — 10 / 10 pts

✔ **– 0 pts** Correct (BTB, Any heuristic that increases prediction chances)

– 5 pts Only mentions 1 correct mechanism to reduce branch mis-prediction (branching heuristic or BTB)

– 10 pts Does not mention 2 mechanism to reduce branch mis-prediction

– 10 pts Blank/no answer

Question 5

Hazards **20** / 20 pts

5.1 **Type of Hazard** **10** / 10 pts

✔ **– 0 pts** Correct

– 3 pts Not mentioning structural hazard.

– 5 pts Incorrectly explains the reason for having two ALUs on the 5-stage pipeline.

– 10 pts Incorrect or Blank

5.2 **Number of Bubbles** **5** / 5 pts

✔ **– 0 pts** Correct

– 5 pts Incorrect

5.3 **Type of Bubbles** **5** / 5 pts

✔ **– 0 pts** Correct

– 5 pts Incorrect

Q1 Instruction Replacement

14 Points

Assume that we are using the LC-2199 ISA with the following CPI's for these instructions.

Instruction	CPI
SW	5
ADDI	3
Others	3 (average including SW and ADDI)

Now, suppose a new instruction `PUSH SR1, offset(SR2)` is added with a **CPI of 6** and the following semantics:

```
SW SR1, 0(SR2)
ADDI SR2, SR2, SEXT(offset)
```

The instruction `SW` immediately followed by `ADDI`, with the goal of pushing some data into memory and then adding to the memory address, occurs in **22%** of the dynamic frequency of the program.

Q1.1 Execution Time Improvement
14 Points

What is the **percentage improvement** of the execution time of the program if every `SW`, `ADDI` sequence is replaced with the new `PUSH` instruction?
(Round your answer to **two decimal places**. Do not include the percentage sign, e.g., 3.00)

NOTE: If you choose to reference example 5-5 in the textbook, it has an error. The correct math for that question can be found under Canvas > Files > Textbook Resources > errata-example 5.5.

6.83

Q1.2 Work (Optional)
0 Points

If you would like partial credit in case of a wrong answer, include or attach your work here.

No files uploaded

Q2 Instruction Improvement
20 Points

A group of software engineers are invited to test the LC-2199 datapath. They have reported that `SKPEQ` and `SKPGT` are used very frequently in assembly code. As a student of CS 2200, you come up with a datapath improvement such that both `SKPEQ` and `SKPGT` instruction take **one-third** time to execute. In a specific program, for `SKPEQ` and `SKPGT` together, the static instruction frequency is **69**, the dynamic frequency is **510**, and it makes up **34%** of the execution time of a program. If the program takes **205.0** ns to execute before the improvement, calculate:

Q2.1 New Execution Time
10 Points

The time taken in **ns** to execute the program after the improvement. (Round your answer to **one decimal places**, e.g., 91.0)

158.5

Q2.2 Speedup
10 Points

Speedup achieved by the improvement. (Round your answer to **two decimal places**, e.g., 4.10)

1.29

Q2.3 Work (Optional)
0 Points

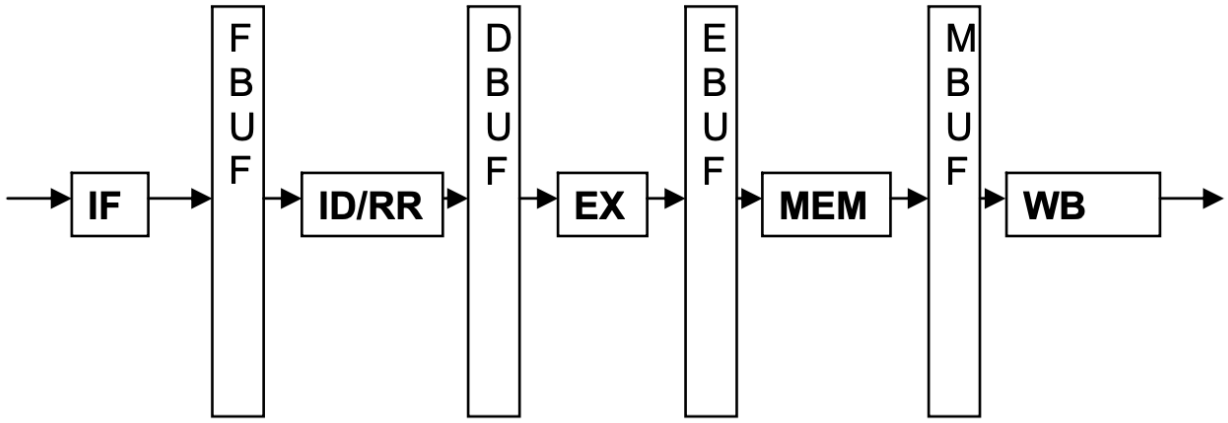
If you would like partial credit in case of a wrong answer, include or attach your work here.

No files uploaded

Q3 Pipeline Buffers

24 Points

Suppose we want to implement the LC-2199 Instruction `LW DR, BaseR, offset20`. According to the pipelined processor architecture described in the textbook, what are the **minimum** contents that must be placed in each pipeline buffer to support this instruction? Select the corresponding contents that are stored in each buffer (buffers pictured below).



Q3.1 FBUF Contents
6 Points

FBUF Contents (Select all that apply):

- [ ] PC
- [x] Fetched Instruction
- [ ] Contents in `$Rx`
- [ ] Contents in `$Ry`
- [ ] Contents in `$Rz`
- [ ] Register number of `$Rx`
- [ ] Register number of `$Ry`
- [ ] Register number of `$Rz`
- [ ] Opcode
- [ ] Result of ALU operation
- [ ] Immediate Value
- [ ] Value read from memory

Q3.2 DBUF Contents
6 Points

DBUF Contents (Select all that apply):

- [ ] PC
- [ ] Contents in `$Rx`
- [x] Contents in `$Ry`
- [ ] Contents in `$Rz`
- [x] Register number of `$Rx`
- [ ] Register number of `$Ry`
- [ ] Register number of `$Rz`
- [x] Opcode
- [ ] Result of ALU operation
- [x] Immediate Value
- [ ] Value read from memory

Q3.3 EBUF Contents
6 Points

EBUF Contents (Select all that apply):

- [ ] PC
- [ ] Contents in `$Rx`
- [ ] Contents in `$Ry`
- [ ] Contents in `$Rz`
- [x] Register number of `$Rx`
- [ ] Register number of `$Ry`
- [ ] Register number of `$Rz`
- [x] Opcode
- [x] Result of ALU operation
- [ ] Immediate Value
- [ ] Value read from memory

Q3.4 MBUF Contents
6 Points

MBUF Contents:

- [ ] PC
- [ ] Contents in `$Rx`
- [ ] Contents in `$Ry`
- [ ] Contents in `$Rz`
- [x] Register number of `$Rx`
- [ ] Register number of `$Ry`
- [ ] Register number of `$Rz`
- [x] Opcode
- [ ] Result of ALU operation
- [ ] Immediate Value
- [x] Value read from memory

Q4 Pipeline Branching

22 Points

Suppose we have a pipeline running the LC2200 ISA and a set of instructions to run. We want to find the costs of different approaches to branching.

Branch instructions can disturb the sequential execution of a program, leading to problems in the pipeline.

Consider the following instructions for 4.1 - 4.3

```
START: ADDI
    LEA
    ...
    BGT START
    BEQ START
    LW
    NAND
    SW
```

The chart below shows the progression of the instructions through the pipeline with register forwarding.

Cycle Number	IF	ID/RR	EX	MEM	WB
1	**BGT**	-	-	-	-
2	**?**	**BGT**	-	-	-
3	**?**	**?**	**BGT**	-	-
4	**?**	**?**	**?**	**BGT**	-
5	**?**	**?**	**?**	**?**	**BGT**
6	**?**	**?**	**?**	**?**	**?**
7	**?**	**?**	**?**	**?**	**?**

Q4.1 Conservative Branching
6 Points

Suppose the pipeline is implemented using the **conservative approach** and that **BGT is not taken** but **BEQ is taken**. What instruction is in the pipeline in the **ID/RR stage** in **cycle 7**?

- ○ BEQ
- ○ SW
- ◉ ADDI
- ○ LEA

Q4.2 Branch Prediction
6 Points

Now suppose that the pipeline is implemented using **branch prediction**. It predicts that **BGT will be taken** and **BEQ will not be taken**. If **BGT is not taken** and **BEQ is taken**, how many NOPs will be in the pipeline in **cycle 7**?

3

Q4.3 Branch Mis-prediction
10 Points

In the previous problem, we saw how branch mis-prediction can cost us NOPs in the pipeline. Describe **two** mechanisms to **reduce branch mis-prediction** in the pipeline.

One way is to compare the target address and current PC in order to make an educated guess about whether the branch will be taken. If target < PC, that typically means that the branch is included in the functionality of a loop since they usually branch backwards, and you should guess that the branch WILL be taken. If target > PC, this is most likely a conditional statement that is utilizing the for loop. Therefore, since this is a less likely branch to be correct, guess that the branch WONT be taken.
Another way is through the Branch Target Buffer, which bases prediction of whether a branch will be taken on the history of that branch. The BTB is a table that will contain the best guess at what the next PC will be. The new PC is predicted in Fetch and is carried through the pipeline, so it can be compared later on to the correct PC and we can determine whether the prediction was correct or incorrect. Using the PC of the branch, we will write the correct PC into the BTB, so that next time, we will get the correct prediction for the PC on the next time we encounter it.

Q5 Hazards

20 Points

Dealing with hazards in a pipeline leads to bubbles depending on the implementation.

Q5.1 Type of Hazard

10 Points

Explain the problem that arises due to having one ALU in the EX stage of the 5-stage pipeline while executing the BEQ instruction. Also, name the hazard associated with it.

A structural hazard can arise due to there being only 1 ALU in the EX stage, when we are completing the BEQ instruction. Two arithmetic operations need to occur in the EX stage, including A-B (always needed in order to determine equality between values) and PC + offset (sometimes needed for branching if equal).

Q5.2 Number of Bubbles
5 Points

Consider the following LC2200 code.

```
ADDI $t0, $zero, 1
ADDI $t1, $zero, 2
BEQ $t1, $t0, halt
ADDI $t1, $t1, 1
ADD $t0, $t0, $t1
halt: HALT
```

Suppose the 5-stage pipelined version of the LC 2200 from the textbook implements **data forwarding**, and it handles control hazards using **branch prediction**. Assume there's a table entry for line 3 (BEQ instruction) in the Branch Target Buffer (BTB) which states 'branch not taken.'
What is the total number of bubbles generated in the pipeline after executing this code?

- (●) 0
- ( ) 1
- ( ) 2
- ( ) 3

Q5.3 Type of Bubbles
5 Points

Consider the following LC2200 code.

```
ADDI $t0, $zero, 1
ADDI $t1, $t1, 1
ADD $s1, $s0, $s0
ADDI $s2, $zero, 1
BEQ $zero, $t0, Label
ADD $t2, $s2, $s1
...
Label ADD $t0, $zero, 1
...
```

Suppose the pipeline **doesn't implement** data forwarding, and it handles control hazards using the **conservative approach**. The **first bubble** generated will be a result of which of the following hazards?

- ○ RAW(Data) Hazard
- ○ WAR(Data) Hazard
- ○ WAW(Data) Hazard
- ◉ Control Hazard
- ○ There is no hazard.