Problem 1

Assume—as in the Amdahl’s Law example on the top of page 47—that we make an enhancement to a computer that improves some mode of execution by a factor of 10. Enhanced mode is used 50% of the time, measured as a percentage of the execution time when the enhanced mode is in use. Recall that Amdahl’s Law depends on the fraction of the original, unenhanced execution time that could make use of enhanced mode. Thus, we cannot directly use this 50% measurement to compute speedup with Amdahl’s Law.

Part A

What is the speedup we have obtained from fast mode?

Part B

What percentage of the original execution time has been converted to fast mode?

Problem 2

Suppose we are considering a change to an instruction set. The base machine initially has only loads and stores to memory, and all operations work on the registers. Such machines are called load-store machines. Measurements of the load-store machine showing the instruction mix and clock cycle counts per instruction are given as follows:

<table>
<thead>
<tr>
<th>Instruction type</th>
<th>Frequency</th>
<th>Clock cycle count</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALU ops</td>
<td>43%</td>
<td>1</td>
</tr>
<tr>
<td>Loads</td>
<td>21%</td>
<td>2</td>
</tr>
<tr>
<td>Stores</td>
<td>12%</td>
<td>2</td>
</tr>
<tr>
<td>Branches</td>
<td>24%</td>
<td>2</td>
</tr>
</tbody>
</table>

Let’s assume that 25% of the arithmetic logic unit (ALU) operations directly use a loaded operand that is not used again.
We propose adding ALU instructions that have one source operand in memory. These new register-memory instructions have a clock cycle count of 2. Suppose that the extended instruction set increases the clock cycle count for branches by 1, but it does not affect the clock cycle time. Would this change improve CPU performance?

**Problem 3**

In lecture, we reviewed a 5-stage MIPS pipeline that provides bypassing to the execute stage (handout #5, page 10). However, interlocks are needed since the bypass wires cannot forward between a load instruction and an immediately following instruction that depends on the load (requiring a 1-cycle stall). The stall condition for this interlock is:

\[
\text{Stall}_{\text{all-loads}} = ((RS1_R == RD_E) \times r1) + ((RS2_R == RD_E) \times r2) \times (IR.E.op == LW)
\]

(Assume \(r1\) and \(r2\) are asserted when an instruction reads the first or second source register specifier, and that these signals are the same as the ones given in lecture).

It is possible to bypass a load followed by a dependent store since the store doesn’t need the result of the load until the beginning of its memory stage. For instance:

- LW R1, 0(R2) \[RS1 = R2, RD = R1\]
- SW R1, 0(R3) \[RS1 = R3, RS2 = R1\]

Modify the pipeline in Figure 1 below to eliminate the interlock in the above instruction sequence by adding a 2-to-1 MUX and making some wiring changes.
Give the logic equations for controlling the new 2-to-1 MUX, and the new stall signal:

\[ MUX_{select} = \]

\[ Stall_{new} = \]

**Problem 4**

Consider adding a new index addressing mode to MIPS. The addressing mode adds two registers and an 11-bit signed offset to get the effective address.

Our compiler will be changed so that code sequences of the form

```
ADD R1, R1, R2
LW Rd, 100(R1)
```

will be replaced with a load (or store) using the new addressing mode. Use the overall average instruction frequencies from Figure A.27 in evaluating this addition.

Part A
Assume that the addressing mode can be used for 10% of the displacement loads and stores (accounting for both the frequency of this type of address calculation and the shorter offset). What is the ratio of instruction count on the enhanced MIPS compared to the original MIPS?

Part B

If the new addressing mode lengthens the clock cycle by 5%, which machine will be faster and by how much?