(All the figures and tables are given as an attachment after all the questions)

Question 1

We want to add four new instructions to the single-cycle datapath in Figure 1. For these instructions, add any necessary datapath and control signals to the single-cycle datapath in Figure 1, and show the control signals in Table 1 (add extra control signals if necessary). Give concise explanations if necessary. Show the answers in separate pages for each instruction added.

(a) Add the instruction sll (shift left logical).

(b) Add the instruction bne (branch if not equal).

(c) Add the instruction jr (jump register)

(d) Add a variant of the sw (store word) instruction, which sums two registers to obtain the address where the data will be stored and uses the R-format:

\[
\text{sw } \text{rd}, \text{rs}, \text{rt}
\]

This instruction differs from the current store because it adds two registers (rs+rt) to form the effective address, rather than a register and an immediate.

(e) Remove the ability to specify an offset for memory access instructions. Now, all load-store instructions with non-zero offsets would become pseudo-instructions implemented using two instructions. For example: lw $rt, offset(rs) would become:

\[
\text{addi } \text{at}, \text{rs}, \text{offset}
\]
\[
\text{lw } \text{rt}, \text{at}
\]
Question 2

Consider the single-cycle datapath in Figure 1 (see attachment). One processor designer proposes to modify this single-cycle datapath by removing the control signal MemtoReg. The multiplexer that has MemtoReg as an input will instead use either the ALUSrc or the MemRead control signal.

(a) Will this designer’s modification work? Consider replacing MemtoReg with ALUSrc or MemRead respectively and show the concerned control signals in the table.

(b) Assuming we could add the instruction addi (add immediate) to Figure 1 by only adjusting the setting of the control signals (without adding any MUX, function units etc.), will this designer’s modification work?

Question 3

Some processors like ARM allow only two operands in arithmetic instructions. For example, an “add” with two operands has the “syntax” and “semantics” as:

\[
\text{add } rt, rs \quad \text{where } rt = rt + rs
\]

Considering the single-cycle datapath in Figure 1 (refer to the attachment), is there any change necessary to implement this “add” instruction? Also show the control signals for it in Table 1 (refer to the attachment).

Question 4

We wish to add a variant of the \( \text{lw} \) (load word) instruction, which increments the index register after loading the word from memory. This instruction \( \text{l_inc} \) corresponds to the following two instructions:

\[
\text{lw } rt, \text{ offset}(rs) \\
\text{addi } rs, rs, 4
\]

(a) Is it possible to modify the single-cycle datapath in Figure 1 to implement this new instruction \( \text{l_inc} \) without modifying the register file? Explain why?

(b) Is it possible to implement this new instruction \( \text{l_inc} \) without modifying the register file in the multicycle implementation? Explain why?
Question 5

Consider a program with the following instruction mix:

<table>
<thead>
<tr>
<th>Instruction Class</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Loads</td>
<td>25%</td>
</tr>
<tr>
<td>Stores</td>
<td>15%</td>
</tr>
<tr>
<td>R-type</td>
<td>40%</td>
</tr>
<tr>
<td>Branches</td>
<td>20%</td>
</tr>
</tbody>
</table>

And the following three computers:

- C1: The multi-cycle datapath (Figures 2 and 3) with a 2 GHz clock.
- C2: A computer similar to C1, except that register updates are done in the same clock cycle as a memory read or ALU operation. Thus, in Figure 5, states 3 and 4 and states 6 and 7 are combined. This machine has a 1.5 GHz clock, since the register update increases the length of the critical path.
- C3: The single-cycle datapath (Figure 1) with a 1 Ghz clock.

Quantitatively compare the performance of these three computers on the given program.
Currently only four types of instructions have been implemented: R-format, lw, sw, and beq.
Figure 2: The multi-cycle datapath (from Figure 5.28 in the textbook).
**Figure 3:** The finite state-machine control for the multi-cycle datapath (from Figure 5.37 in the textbook).