
**Question 1**

The CPU of a computer can be partitioned into 8 stages, with 2ns, 2ns, 3ns, 2ns, 4ns, 2ns, 3ns, and 2ns delays, respectively. (We are assuming an architecture similar to the 5-stage architecture for MIPS. Except, here we have 8 stages).

(a) What is the maximum number of instructions that can be executed in one second (**non-pipelined operation**)?

(b) What is the maximum clock frequency for this CPU if a 8-stage pipeline is used? Assume interstage registers add 1ns to each stage.

(c) What is the maximum number of instructions that can be executed in one second in the pipelined CPU?

**Question 2**

Consider the **pipelined** 5-stage architecture for MIPS. The (binary machine code of) following sequence of instructions are in memory, starting at memory address 1000. Assume the PC is loaded with this address (1000) when the CPU starts execution.

ADD R1, R2, R3
ADD R4, R5, R6
SUB R7, R4, R1
ADD R9, R8, R7

ADD is an (integer) add instruction, and SUB is an (integer) subtract instruction. **ASSUME EACH REGISTER HOLDS A NUMBER EQUAL TO THE REGISTER NUMBER AT THE BEGINNING**, that is, R1 holds 1, R2 holds 2, etc...

In a table similar to the following, describe the actions that are performed in the CPU for each clock cycle. Include details such as any forwarding that takes place. Also show the numerical contents of the registers (internal and general purpose registers) that change during a clock cycle. **BE PRECISE.** Note that all stages of the CPU may be active processing up to five instructions during each clock cycle.

<table>
<thead>
<tr>
<th>action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1: IF stage for instruction 1: IF/ID.IR &lt;- Mem[PC]; IF/ID.NPC &lt;- PC+4; PC &lt;- PC+4 (IF/ID.IR is loaded with instruction 1)</td>
</tr>
<tr>
<td>2: ID stage for instruction 1: ID/EX.A &lt;- R2; ID/EX.B &lt;- R3; IMM &lt;- ? (A=2; B=3) Transition: ID/EX.IR &lt;- IF/ID.IR (IF/ID.IR is moved to next inter-stage latches) IF stage for instruction 2: IF/ID.IR &lt;- Mem[PC]; NPC &lt;- PC+4; PC &lt;- PC+4 (IF/ID.IR is loaded with instruction 2)</td>
</tr>
<tr>
<td>3: EX stage for instruction 1: ... Transition: EX/MEM.IR &lt;- ID/EX.IR; EX/MEM.B &lt;- ID/EX.B (ID/EX.IR and ID/EX.B are moved to next inter-stage latches) ID stage for instruction 2: ... Transition: ID/EX.IR &lt;- IF/ID.IR (IF/ID.IR is moved to next inter-stage latches) IF stage for instruction 3: ...</td>
</tr>
</tbody>
</table>

Cont’d on back of the page
Question 3

Use the following code fragment. Use a pipeline timing chart like Figure A.10 (instead of A.6). For part (a), assume the architecture of Figure A.18. For parts (b) and (c), assume the architecture of Figure A.24. Assume both architectures are enhanced to handle Branch if not equal (BNEQ) instruction.

```
loop: LD  R1,0(R2)
  LD  R4,0(R5)
  DADD R1,R1,R4
  SD  R1,0(R2)
  DADDI R5,R5,#4
  DADDI R2,R2,#4
  BNEQ R2,R3,loop
```

Assume the initial value of R3 is R2+396.

(a) Show the timing of this instruction sequence for MIPS standard five-stage integer pipeline without any forwarding and bypassing hardware. How many cycles does this loop take to execute?

(b) Show the timing of this instruction sequence for MIPS standard five-stage integer pipeline with normal forwarding and bypassing hardware. Assume the branch condition and target address are computed in the second stage (ID), and branch is handled by freezing the pipeline. How many cycles does this loop take to execute?

(c) Show the timing of this instruction sequence for MIPS standard five-stage integer pipeline with normal forwarding and bypassing hardware. Assume a delayed branch, and assume the branch condition and target address are computed in the second stage (ID). Schedule the code including the branch delay slot to minimize the total number of cycles for this loop. Write the scheduled code, and its timing diagram. How many cycles does this loop take to execute in this case?

Additional practice problems. You do not need to solve these for the homework.

Question 4

Whenever a load instruction is followed by another instruction that uses the loaded value in the execute stage, the MIPS pipeline (with forwarding) should be stalled by one cycle. Give examples (assembly language code) that need the stall for the following cases:

**Example:** A load instruction followed by a register-register ALU instruction using the loaded value as its source operand A.

```
LD  R5, 4(R1)
DADD R30, R5, R4
```

(a) A load instruction followed by a register-register ALU instruction using the loaded value as its source operand B.

(b) A load instruction followed by a register-immediate ALU instruction using the loaded value as its source operand.

(c) A load instruction followed by a store instruction.

(d) A load instruction followed by another load instruction.

(e) A load instruction followed by a branch instruction.

Question 5

If 20% of the instructions are loads, and 60% of the time the instruction following load depends on the result of the load (hence resulting in a one cycle stall). What is the CPI on the average? Assume the ideal case (no stalls) has a CPI of 1.