CIS 314 Computer Organization - Fall 2015
Problem Set 3

General guidelines: Always state your assumptions and clearly explain your answers. Please upload your solution document (PDF or TXT) to Canvas.

100/100 points possible – Due Wednesday, November 18th by 11:59 PM through Canvas.

Notes between the square brackets are simply to help you find the equivalent of the problem or question in the textbook.

Part 1: Pipelining and Hazards
Exercise 1 [4.9]
In this exercise, we examine how data dependences affect execution in the basic 5-stage pipeline MIPS processor. Questions in this exercise refer to the following sequence of instructions:

or r1,r2,r3
or r2,r1,r4
or r1,r1,r2

Also, assume the following cycle times for each of the options related to forwarding:

<table>
<thead>
<tr>
<th>Without Forwarding</th>
<th>With Full Forwarding</th>
<th>With ALU-ALU Forwarding Only</th>
</tr>
</thead>
<tbody>
<tr>
<td>250ps</td>
<td>300ps</td>
<td>290ps</td>
</tr>
</tbody>
</table>

Question 1: Indicate dependences and their type.

Question 2: Assume there is no forwarding in this pipelined processor. Indicate hazards and add nop instructions to eliminate them.

Question 3: Assume there is full forwarding. Indicate hazards and add nop instructions to eliminate them.

Question 4: What is the total execution time of this instruction sequence without forwarding and with full forwarding? What is the speedup achieved by adding full forwarding to a pipeline that had no forwarding?

Question 5: Add nop instructions to this code to eliminate hazards if there is ALU-ALU forwarding only (no forwarding from the MEM to the EX stage).

Question 6: What is the total execution time of this instruction sequence with only ALU-ALU forwarding? What is the speedup over a no-forwarding pipeline?
Exercise 2 [4.10]
Problems in this exercise refer to the following fragment of MIPS code:

```mips
sw r16,12(r6)
lw r16,8(r6)
beq r5,r4,Labe l #Assume r5!=r4
add r5,r1,r4
slt r5,r15,r4
```

Assume that individual pipeline stages have the following latencies:

<table>
<thead>
<tr>
<th></th>
<th>IF</th>
<th>ID</th>
<th>EX</th>
<th>MEM</th>
<th>WB</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>200ps</td>
<td>120ps</td>
<td>150ps</td>
<td>190ps</td>
<td>100ps</td>
</tr>
</tbody>
</table>

**Question 1:** If we only have one memory (for both instructions and data), there is a structural hazard every time we need to fetch an instruction in the same cycle in which another instruction accesses data. To guarantee forward progress, this hazard must always be resolved in favor of the instruction that accesses data. What is the total execution time of this instruction sequence in the 5-stage pipeline that only has one memory? We have seen that data hazards can be eliminated by adding nops to the code. Can you do the same with this structural hazard? Why?

**Question 2:** If we change load/store instructions to use a register (without an offset) as the address, these instructions no longer need to use the ALU. As a result, MEM and EX stages can be overlapped and the pipeline has only 4 stages. Change this code to accommodate this changed ISA. When EX and MEM are done in a single stage, most of their work can be done in parallel. As a result, the resulting EX/MEM stage has a latency that is the larger of the original two, plus 20 ps needed for the work that could not be done in parallel. What speedup is achieved in this instruction sequence?

**Part 2: Caching and Cache Structures**

**Exercise 1 [5.6]**
In this exercise, we will look at the different ways capacity affects overall performance. In general, cache access time is proportional to capacity. Assume that main memory accesses take 70 ns and that memory accesses are 36% of all instructions. The following table shows data for LI caches attached to each of two processors, P1 and P2.

<table>
<thead>
<tr>
<th></th>
<th>L1 Size</th>
<th>L1 Miss Rate</th>
<th>L1 Hit time</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>2 KiB</td>
<td>8.0%</td>
<td>0.66ns</td>
</tr>
<tr>
<td>P2</td>
<td>4 KiB</td>
<td>6.0%</td>
<td>0.90ns</td>
</tr>
</tbody>
</table>

**Question 1:** Assuming that the LI hit time determines the cycle times for P1 and P2, what are their respective clock rates?

**Question 2:** What is the Average Memory Access Time for P1 and P2?
Question 3: Assuming a base CPI of 1.0 without any memory stalls, what is the total CPI for P1 and P2? Which processor is faster?

**Exercise 2 [5.3]**
For a direct-mapped cache design with a 32-bit address, the following bits of the address are used to access the cache.

<table>
<thead>
<tr>
<th>Tag</th>
<th>Index</th>
<th>Offset</th>
</tr>
</thead>
<tbody>
<tr>
<td>31-10</td>
<td>9-5</td>
<td>4-0</td>
</tr>
</tbody>
</table>

Question 1: What is the cache block size (in words)?

Question 1.a: How many entries does the cache have?

Question 1.b: What is the ratio between total bits required for such a cache implementation over the data storage bits?

Question 1.c: Starting from power on, the following byte-addressed cache references are recorded.

<table>
<thead>
<tr>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
</tr>
</tbody>
</table>

Question 2.a How many blocks are replaced?

Question 2.b What is the hit ratio?

Question 2.c List the final state of the cache, with each valid entry represented as a record of <index, tag, data>.