1. (20 points) Identify all the hazards (RAW, WAW, WAR) that could emerge from executing the following code on an OOO execution engine.

LD F2, 0(Rx)
MULTD F2, F0, F2
DIVD F8, F2, F0
LD F4, 0(Ry)
ADDD F4, F0, F4
ADDD F10, F8, F2
SD F4, 0(Ry)
ADDI Rx, Rx, 8
ADDI Ry, Ry, 8
SUB R20, R4, Rx

2. (30 points) Illustrate on a cycle-by-cycle basis the execution of the following program on a dynamic execution engine with a ROB to support speculative execution and precise exception. Assume a dual dispatch and a dual common data bus (CDBs). Add latency is two cycles, and multiply latency is three cycles. An instruction can begin execution in the same cycle that is dispatches assuming all dependences are satisfied. Show the content of ROB, reservation stations (RS), register file (RF) and the register alias table (RAT) for each cycle. Indicate which instruction is executing in each functional unit in each cycle. Also indicate any results forwarding across the CDBs.

\[
\begin{align*}
w & : R4 & \leftarrow & R0+R8 \\
x & : R2 & \leftarrow & R0*R4 \\
y & : R4 & \leftarrow & R4+R8 \\
z & : R8 & \leftarrow & R4*R2 \\
\end{align*}
\]

3. (30 points) Consider the execution of the following loop, which increments each element of an integer array, on a two-issue processor, once without speculation and once with speculation:

\[
\begin{align*}
\text{LOOP:} & \quad \text{LD} & \quad R2, 0(R1) & ; \text{R2 = array element} \\
& \quad \text{DADDIU} & \quad R2, R2, #1 & ; \text{increment R2} \\
& \quad \text{SD} & \quad R2, 0(R1) & ; \text{store result} \\
& \quad \text{DADDIU} & \quad R1, R1, #8 & ; \text{increment pointer} \\
& \quad \text{BNE} & \quad R2, R3, LOOP & ; \text{branch if not last statement} \\
\end{align*}
\]

Assume that there are separate integer functional units for effective address calculation, for ALU operations, and for branch condition evaluation with execution latency of 1 cycle. Create a
table for the first three iterations of this loop for both processors. Assume that up to to two instructions of any type can retire per clock cycle. I have the first few entries for both cases. Note that the execution of a load and store instruction means to compute the effective address.

<table>
<thead>
<tr>
<th>Iteration number</th>
<th>Instruction</th>
<th>Dispatches clock cycle number</th>
<th>Executes at clock cycle number</th>
<th>Memory access at clock cycle number</th>
<th>Write CDB at clock cycle number</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>LD R2, 0(R1)</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>DADDIU R2, R2, #1</td>
<td>1</td>
<td>5</td>
<td>7</td>
<td>6</td>
</tr>
<tr>
<td>1</td>
<td>SD R2, 0(R1)</td>
<td>2</td>
<td>3</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>DADDIU R1, R1 #8</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>:</td>
<td>:</td>
<td>:</td>
<td>:</td>
<td>:</td>
</tr>
</tbody>
</table>

Table 1: Non-speculative execution

<table>
<thead>
<tr>
<th>Iteration number</th>
<th>Instruction</th>
<th>Dispatches clock cycle number</th>
<th>Executes at clock cycle number</th>
<th>Memory access at clock cycle number</th>
<th>Write CDB at clock cycle number</th>
<th>retires at clock cycle number</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>LD R2, 0(R1)</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>DADDIU R2, R2, #1</td>
<td>1</td>
<td>5</td>
<td>7</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>1</td>
<td>SD R2, 0(R1)</td>
<td>2</td>
<td>3</td>
<td></td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>1</td>
<td>DADDIU R1, R1 #8</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>:</td>
<td>:</td>
<td>:</td>
<td>:</td>
<td>:</td>
<td>:</td>
</tr>
</tbody>
</table>

Table 2: speculative execution