NAME: ____________________________________________________________

STUDENT NUMBER: ________________________________

EE557--FALL 2000

MIDTERM 1

Closed books, closed notes
Time limit: 1 hour and 20 minutes MAX. No extension.

Q1: /14

Q2: /14

Q3: /22

TOTAL: /50

Grade: /25
**QUESTION 1. 14 points**

(6pts) The only addressing mode to memory in DLX is displacement, i.e.,

\[
\text{LW } r2, D(r1)
\]

where D is a 16 bits two's complement value added to r1 to find the memory address.

To deal with other addressing modes, we need to synthesize them with DLX instruction-

Please synthesize the following instructions using DLX instructions only.

1. Indexed:

\[
\text{LW } r4, (r1+r2) /r4 \leftarrow MEM[r1+r2]
\]

DLX sequence:

---

---

---

---

2. Memory indirect:

\[
\text{LW } r4, @(r1) /r4 \leftarrow MEM[MEM[r1]]
\]

DLX sequence:

---

---

---

---

3. Pre increment:

\[
\text{LW } r4, +(r1)
\]

DLX sequence:

---

---

---

---

2. (8pts) We consider adding the predecrement addressing mode to help with loop indexing. The instruction mix for the original machine and number of clocks per instruction are
10% of all ALU operations are used to index into array variables and can be removed by adding the new addressing mode, but we do not know the effect that the new mode will have on the cycle time. The CPI for each instruction class remains the same.

What is the upper bound on the new cycle time so that the new addressing mode improves overall performance.

Table 1:

<table>
<thead>
<tr>
<th>Instruction type</th>
<th>Frequency</th>
<th>Clock cycle count</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALU</td>
<td>45%</td>
<td>1</td>
</tr>
<tr>
<td>Loads</td>
<td>20%</td>
<td>2</td>
</tr>
<tr>
<td>Stores</td>
<td>15%</td>
<td>1</td>
</tr>
<tr>
<td>Branch</td>
<td>20%</td>
<td>2</td>
</tr>
</tbody>
</table>
QUESTION 2 (14pts)

Figure 1 shows the datapath of the single cycle processor.

The proposed datapath to implement the addressing modes given in question 1 is shown in Figure 1. As you can see I have done some modifications to the datapath. I have added one write port to the register file and removed the MemtoReg multiplexer. Now Register-to-register instructions use write port 2 and some load instructions use write port 1. I have also added one input to the ALU. Remember that in the single cycle design there is no forwarding inside the register file. You may assume that LW r1, +(r1) is an illegal instruction.

Fill in the table below with the values of the control points to support these new addressing modes. All instructions must still execute in one single clock. If it is impossible to implement an instruction, please say why it is so in the space below the table.

<table>
<thead>
<tr>
<th>Instruction</th>
<th>RegWrite1</th>
<th>RegWrite2</th>
<th>RegDst</th>
<th>MemRead</th>
<th>MemWrite</th>
<th>Branch</th>
<th>ALUSRC</th>
</tr>
</thead>
<tbody>
<tr>
<td>add rd,rs,rt</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>lw rt, D(rs)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>lw rt, +(rs)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>lw rd, (rs+rt)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>lw rt, @rs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sw rt, D(rs)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sw rt, +(rs)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sw rd, (rs+rt)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sw rt, @rs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

You may assume that LW r1, +(r1) is an illegal instruction.
QUESTION 2 (22pts)

Figure 2 shows a DLX floating point pipeline with multiple execution units. The address (integer) unit takes one clock. The FP adder/multiplier is fully pipelined and takes 3 clocks. The divider is not pipelined and takes 4 clocks.

1. (3pts) Please give the latency of some floating point operations by filling out the following table. Consider dependencies on the floating-point operand in all cases.

<table>
<thead>
<tr>
<th>TABLE 3. Latencies for Floating Point operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>LATENCY (CLOCKS)</td>
</tr>
<tr>
<td>LD followed by ADDD</td>
</tr>
<tr>
<td>LD followed by SD</td>
</tr>
<tr>
<td>ADDD followed by SD</td>
</tr>
<tr>
<td>DVD followed by STORE</td>
</tr>
<tr>
<td>DVD followed by DVD</td>
</tr>
<tr>
<td>ADDD followed by ADDD</td>
</tr>
</tbody>
</table>

2. (5pts) Consider the following program.

```
LOOP    LD F0,1000(R1)
       LD F2,2000(R1)
       DVD F4,F0,F2
       SD F4,3000(R1)
       SUBI R1,8
       BNE R1,R0,LOOP
```

The branch is executed in the ID stage and forwarding is applied whenever possible. The branch is NOT delayed, is predicted not taken and is executed in the MEM stage. In the following table please show the timing diagram for this loop by indicating the stage in each clock for each instruction in the table. Structural hazards on the write port of the register file are handled by issuing instructions only if they don’t conflict with previous instructions at the register file. This is done in the ID stage with a shift register.

<table>
<thead>
<tr>
<th>TABLE 4.</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
</tr>
<tr>
<td>LD F0,1000(R1)</td>
</tr>
<tr>
<td>LD F2,2000(R1)</td>
</tr>
<tr>
<td>DVD F4,F0,F2</td>
</tr>
<tr>
<td>SD F4,3000(R1)</td>
</tr>
<tr>
<td>SUBI R1,8</td>
</tr>
<tr>
<td>BNE R1,R0,LOOP</td>
</tr>
<tr>
<td>LD F0,1000(R1)</td>
</tr>
</tbody>
</table>
3. (2pts) Assuming that the loop is executed N times, what is the time in cycles needed to execute the loop as it is. Note that the last iteration of the loop will continue execution after the branch, and that useful instructions will be executed, so that the number of clocks associated with the last iteration is different from other iterations.

TIME= __________________Clocks.

4. (3pts) Below I have shown the previous code with the stalls for all iterations except the last one. Assume now that the branch is delayed by one and that the compiler can schedule the code to improve performance. (No loop unrolling; just code scheduling within each iteration). Please show the new code. Show the stalls on the code.

```
LOOP
  LD F0,1000(R1)
  LD F2,2000(R1)
stall
  DVD F4,F0,F2
  stall
  stall
  SD F4,3000(R1)
  SUBI R1,8
  BNE R1,R0,LOOP
  stall
  stall
  stall
```

New code:
```
----------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------
'''

6
5. (6pts) Describe 3 methods by which precise exceptions can be maintained in a pipeline such as the one in Figure 2.

Method #1:

---------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------

Method #2:

---------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------

Method #3:

---------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
6. (3pts) Imagine that the DVD is now pipelined in two stages, but each stage takes 2 clocks. Please fill in the following table for the latencies due to dependencies on FP operands:

<table>
<thead>
<tr>
<th></th>
<th>LATENCY (CLOCKS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DVD followed by LD</td>
<td></td>
</tr>
<tr>
<td>DVD followed by SD</td>
<td></td>
</tr>
<tr>
<td>DVD followed by ADDD</td>
<td></td>
</tr>
<tr>
<td>DVD followed by DVD</td>
<td></td>
</tr>
</tbody>
</table>

Additionally what is the initiation interval for DVD?

Initiation interval = ________________ cycles.
Figure 1. Simplified single cycle datapath
Figure 2. DLX pipeline with multiple execution units.