NAME: ________________________________

STUDENT NUMBER: ________________________________
Remote: YES/NO

EE557--FALL 2001

FINAL

Open books and notes
Time limit: 2Hours MAX. No extension.

Q1: /6
Q2: /6
Q3: /10
Q4: /12
Q5: /9
Q6: /14

TOTAL: /57

Grade: /40
QUESTION 1 6 points

Consider a parallel computation executed on P processors connected by a single bus. The computation is a loop. In each iteration of the loop all processors compute in parallel for a total time of C/P. Then every processor broadcasts its results on the bus one by one. The bus transfer time for each processor is T/P.

Let’s R = T/C be the communication-to-computation ratio. We only consider cases where R<1.

1. (3pts) Give the speedup as a function of P and R.

2. (1pt) What is the maximum value of this speedup as P increases?

3. (1pt) Can this speedup be “superlinear”? YES/No

4. (1pt) What is the minimum number of processors needed to reach a speedup greater than 1?

-----------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------
QUESTION 2 6 points

Suppose we have an application that runs in three modes: all processors used, half the processors in use, and serial mode. Let \( f \) be the fraction of time that is serial, and let \( P \) be the total number of processors. Assume that \( P=100 \) and \( f=2\% \).

1. (2pts) Find the maximum time that can be spent in the mode where half the processors are used in order to reach a speedup of at least 30.

2. (2pts) Find the maximum time that can be spent in the mode where half the processors are used in order to reach a speedup of at least 25.

3. (2pts) Find the maximum time that can be spent in the mode where half the processors are used in order to reach a speedup of at least 25.
QUESTION 3 10 points

Create a table for the R4000 integer pipeline shown in Fig. 3.50 (page 201) that is similar to the DLX table shown in Fig. 3.19 (page 160). Consider only the forwarding for register-to-register ALU instructions, ALU-Immediate instructions, and Load instructions. To limit the amount of writing, consider the forwarding to ALU Top only. We omit the comparison column.

Complete filling the table below.

<table>
<thead>
<tr>
<th>Source Pipeline Register</th>
<th>Source Opcode</th>
<th>DestinationPipeline Register</th>
<th>Destination Opcode</th>
<th>Destination of Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>RF/EX</td>
<td>Any</td>
<td>RF/EX</td>
<td>Any</td>
<td>ALU Top</td>
</tr>
<tr>
<td>RF/EX</td>
<td>Any</td>
<td>RF/EX</td>
<td>Any</td>
<td>ALU Top</td>
</tr>
<tr>
<td>RF/EX</td>
<td>Any</td>
<td>RF/EX</td>
<td>Any</td>
<td>ALU Top</td>
</tr>
<tr>
<td>RF/EX</td>
<td>Any</td>
<td>RF/EX</td>
<td>Any</td>
<td>ALU Top</td>
</tr>
<tr>
<td>RF/EX</td>
<td>Any</td>
<td>RF/EX</td>
<td>Any</td>
<td>ALU Top</td>
</tr>
<tr>
<td>RF/EX</td>
<td>Any</td>
<td>RF/EX</td>
<td>Any</td>
<td>ALU Top</td>
</tr>
<tr>
<td>RF/EX</td>
<td>Any</td>
<td>RF/EX</td>
<td>Any</td>
<td>ALU Top</td>
</tr>
<tr>
<td>RF/EX</td>
<td>Any</td>
<td>RF/EX</td>
<td>Any</td>
<td>ALU Top</td>
</tr>
<tr>
<td>RF/EX</td>
<td>Any</td>
<td>RF/EX</td>
<td>Any</td>
<td>ALU Top</td>
</tr>
<tr>
<td>RF/EX</td>
<td>Any</td>
<td>RF/EX</td>
<td>Any</td>
<td>ALU Top</td>
</tr>
<tr>
<td>RF/EX</td>
<td>Any</td>
<td>RF/EX</td>
<td>Any</td>
<td>ALU Top</td>
</tr>
<tr>
<td>RF/EX</td>
<td>Any</td>
<td>RF/EX</td>
<td>Any</td>
<td>ALU Top</td>
</tr>
<tr>
<td>RF/EX</td>
<td>Any</td>
<td>RF/EX</td>
<td>Any</td>
<td>ALU Top</td>
</tr>
</tbody>
</table>
QUESTION 4(Tomasulo) 12 points

Consider the following loop. This program is the same as in the 2nd midterm.

```
LOOP    LD F0,0(R1)
       LD F2,-8(R1)
       LED F0,F2
       BFPT LESS
       SD 0(R1), F2
       SD -8(R1), F0
LESS    SUBI R1,R1, 8
       BNEZ R1, LOOP
```

LED is a DLX instruction which compares FP values in F0 and F2. If F0 is less than or equal to F2, a FP status register is set to 1, otherwise it is set to 0. BFPT is also a DLX instruction which reads the value of the the FP status register; if it is 1, then it branches. FP operands have size 8 bytes (double precision).

Concerning hardware behavior, please use the assumptions in the example in the class notes, with the following modifications:

1. One single LD/ST unit. The LD/ST unit takes 2 clocks. The LD/ST unit (and RS) is busy until (not including) the cycle after write_result (LD) or exec_complete (SD). It’s assumed that the cache always hits.

2. One single integer/branch unit, which executes BFPT, SUBI and BNEZ in the program. The unit (and RS) is busy until (not including) the cycle after exec_complete (branches) or after write_result (SUBI). Execution takes one clock.

3. One single FP unit. The unit (and RS) is busy until (not including) the cycle after write_result (which updates the FP status register). The execution of the compare takes 4 cycles.

In all cases, we issue one instruction at a time whenever we can (always in program order). The integer/branch unit has 4 reservation stations. The FP unit has 2 reservation stations. There are also 2 store buffers and 2 load buffers. There is a single CDB for both integer and FP operands. If two instructions conflict for the CDB, the oldest instruction in program order has priority. The reservation stations and load/store buffers are busy starting at issue, whereas the execution units are busy starting at exec_start.

Branches are predicted when they are issued and the order of instruction issues is kept in a ROB which is also used for register renaming. BFPT and BNEZ are always predicted not taken, and we assume that the predictions are correct.

If two instructions are ready to execute in the same cycle in an execution unit, then the older instruction in the program order executes first.
The schedule is shown for clock 1 through clock 10 in the table below. Complete the schedule for clock 11 through clock 22 by filling relevant entries in the instruction status table below. Each entry in the table shows the clock cycle when a given instruction went through a specific phase of processing.

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Issue</th>
<th>Exec start</th>
<th>Exec. complete</th>
<th>Write Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>LD F0,0(R1)</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>LD F2,-8(R1)</td>
<td>2</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>LED F0,F2</td>
<td>3</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BFPT LESS</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SD 0(R1), F2</td>
<td>5</td>
<td>8</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>SD -8(R1), F0</td>
<td>6</td>
<td>10</td>
<td></td>
<td>-</td>
</tr>
<tr>
<td>SUBI R1,R1,#8</td>
<td>7</td>
<td>8</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>BNEZ R1, LOOP</td>
<td>8</td>
<td>10</td>
<td>10</td>
<td>-</td>
</tr>
<tr>
<td>LD F0,0(R1)</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LD F2,-8(R1)</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LED F0,F2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BFPT LESS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SD 0(R1), F2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SD -8(R1), F0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBI R1,R1,#8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BNEZ R1, LOOP</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LD F0,0(R1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LD F2,-8(R1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LED F0,F2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BFPT LESS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SD 0(R1), F2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SD -8(R1), F0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBI R1,R1,#8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BNEZ R1, LOOP</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
QUESTION 5 9points
Solve Problem 5.5 in your book. Do the actual calculations for the LI program only.

In this problem, the write-through buffer to memory is large enough that it never blocks the processor. Assume that the write-through cache allocates a block on each write miss so that the contents of the write-back and write-through caches are the same at all times.

On a cache miss the write-through cache blocks the processor for just the time it takes to load the block from memory. Besides loading the new block from memory the write-back cache needs to write back the dirty block (and 50% of all blocks are dirty.) There is no write back buffer in parts a and b

Finally assume that instructions other than load/store take one clock when the instruction cache hits.

Part a. 3pts

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

Part b. 3pts

-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
Part c. 3pts

Answer as in Part 2, but now there is a write-back buffer for the written back block. When a miss occurs in the WB cache, the miss request is sent immediately to memory. During that time the victim block (if modified) is stored in the writeback buffer. The victim block is then written to memory after the miss is complete. Assume that there are enough entries in the write-back buffer so that the processor is never affected by write backs.
QUESTION 6 14points Cache design question.

You are in charge of designing the on chip data cache for the new Beta microprocessor. The processor will be a 64-bit superpipelined 4-issue machine capable of issuing at most 1 float, 1 integer, 1 branch, and a data memory reference on each instruction. The target clock frequency is 500 MHz. The final decision will be based on a set of characterization codes which have been simulated to have the following characteristics with a perfect memory system (100% hit rate and a memory latency of 2 cycles):

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Value</th>
<th>CPI (if applicable)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total executed instructions</td>
<td>200,000,000,000</td>
<td>NA</td>
</tr>
<tr>
<td>Floats</td>
<td>11%</td>
<td>2.1</td>
</tr>
<tr>
<td>Integer Operations</td>
<td>45%</td>
<td>.75</td>
</tr>
<tr>
<td>Loads</td>
<td>19%</td>
<td>.50</td>
</tr>
<tr>
<td>Stores</td>
<td>9%</td>
<td>.25</td>
</tr>
<tr>
<td>Branches</td>
<td>16%</td>
<td>.80</td>
</tr>
</tbody>
</table>

The 5 options that have not already been eliminated and their simulated miss rates on the characterization codes and their hit and miss timings are:

<table>
<thead>
<tr>
<th>Cache Type</th>
<th>Total Size</th>
<th>Line Size</th>
<th>Miss Rates</th>
<th>Timing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Direct Mapped</td>
<td>2MB</td>
<td>32B</td>
<td>Compulsory 1.2%</td>
<td>Hit 2 cycles</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Conflict 2.3%</td>
<td>Miss 30 cycles</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Capacity .5%</td>
<td></td>
</tr>
<tr>
<td>Direct Mapped with 16 line</td>
<td>2MB</td>
<td>32B</td>
<td>Compulsory 1.2%</td>
<td>Hit 2 cycles</td>
</tr>
<tr>
<td>victim cache</td>
<td></td>
<td></td>
<td>Conflict 1.6%</td>
<td>Miss 30 cycles</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Capacity .5%</td>
<td></td>
</tr>
<tr>
<td>Direct Mapped</td>
<td>2MB</td>
<td>64B</td>
<td>Compulsory .9%</td>
<td>Hit 2 cycles</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Conflict 2.1%</td>
<td>Miss 46 cycles</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Capacity .7%</td>
<td></td>
</tr>
<tr>
<td>2-way Set Associative</td>
<td>2MB</td>
<td>32B</td>
<td>Compulsory 1.2%</td>
<td>Hit 3 cycles</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Conflict 1.6%</td>
<td>Miss 30 cycles</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Capacity .5%</td>
<td></td>
</tr>
<tr>
<td>4-way Set Associative</td>
<td>2MB</td>
<td>32B</td>
<td>Compulsory 1.2%</td>
<td>Hit 3 cycles</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Conflict 1.6%</td>
<td>Miss 30 cycles</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Capacity .5%</td>
<td></td>
</tr>
</tbody>
</table>

Assume that the caches are all write-back, write allocate and adding the 3 miss rate components gives the total miss-rate. THE MISS PENALTY INCLUDES
1. Given that the processor core has been initially designed for a 2 cycle hit latency what difficulties do you see in using either of the set associative options, as the core has been set up for a 2 stage cache pipeline and the set associative options are both 3 stage cache pipelines.

Hardware complexity: (2pts)

Performance: (2pts)
2. Provide a justification for why the miss rate components and timings are different for the various following options?

Adding a victim cache: (2pts)

Increasing the line size: (2pts)
3. (4pts) Assume for simplicity that none of the cache options will cause a change in the given CPI values on a hit. Which option do you choose and why? How much worse is your choice from the ideal memory system. Also assume that the miss rates are uniformly distributed over loads and stores. What’s odd about your findings?