CONFIDENTIAL

Paper 3.32(MEng3 test) Parallel Architectures

Department of Computing
Imperial College

Examination paper **March 27, 1996**

First examiner: ..... **Second examiner:** .....

Paper 3.32(MEng3 test) Parallel Architectures

This examination is of TWO HOURS' duration.

Answer ALL THREE questions.

You MAY refer to notes, textbooks etc. during the examination, providing you do so quietly and without communicating with other candidates.

1 This question concerns the performance of the following loop on two advanced static pipeline implementations of DLX:

```
for i = 1 to 100 do
  A[i] = B[i] + S * B[i-1]
```

Consider the following implementation for the DLX machine:

```
// Use R1 as ptr to B, R2 as ptr to A,
//     R3 as iteration count, initially 100, and
//     F0 as S
loop:    LD    F4, -8(R1)
         LD    F2, 0(R1)
         SUB   R3, R3, #1
         MULD  F4, F4, F0
         ADDI  R1, R1, #8
         ADDI  R2, R2, #8
         ADDD  F2, F2, F4
         SD    F2, -8(R2)
         BNEQZ R3, loop
```

a For this part of the question, assume a static pipelined implementation of the DLX processor with one non-pipelined, 2-cycle floating-point add/subtract unit, one non-pipelined, 4-cycle floating-point multiply unit, and an eight stage integer pipeline as follows:

IF	IS	RF	EX	DF	DS	TC	WB
Initiate I-cache access	Complete I-cache access	Decode instr'n and fetch registers	Execution	Initiate fetch from D-cache	Complete data fetch, and forward result	Tag check: check for cache hit	Write results to registers

(The MIPS R4000 processor has such a structure; the idea is called "superpipelining", and allows the clock period to be smaller than the cache access time).

Draw a diagram showing the execution of one iteration of this loop on this machine, assuming no cache misses, and assuming no stalls due to control hazards. Explain carefully where any forwarding or stalls are required. Explain any assumptions you have to make.

b For this part of the question, assume a similar pipeline, but modified so that each stage can potentially handle a pair of adjacent instructions. In such a "superscalar" design, two adjacent instructions can be issued per clock cycle provided they have no dependence, and the first updates only floating-point registers, while the second updates only integer registers. If these conditions cannot be met, the instructions are issued one at a time as in part (a).

(The DEC Alpha is an example of such a superscalar, superpipelined design).

Modify the assembly code given above to avoid as many stalls as possible. Explain carefully what transformations you apply, and why.

(The two parts carry, respectively, 50% and 50% of the marks).

2 In this question you are invited to consider the performance of a range of similar machines, each with a 4KB data cache, sufficient to hold 512 eight-byte double-precision floating-point words. The machines differ in the details of the cache design.

Consider the following example program fragment:

```
declare float A[1024,1024], B[1024,1024];
for i = 0 to N-1 do
  for j = 0 to M-1 do
    B[j,i] := A[i,j] + 1;
```

Assume that all two-dimensional arrays are addressed from 0, and are arranged in memory in row order (ie. row-by-row so that A[i,j] is immediately followed by A[i,j+1]).

a Explain briefly the three causes of cache misses in a single-processor system.

b Assume $M = 1024$, and 16-byte cache lines. Estimate the *read* miss rate, and explain your answer.

c Assume $M = 1024$, and 16-byte cache lines. Estimate the *write* miss rate, and explain your answer.

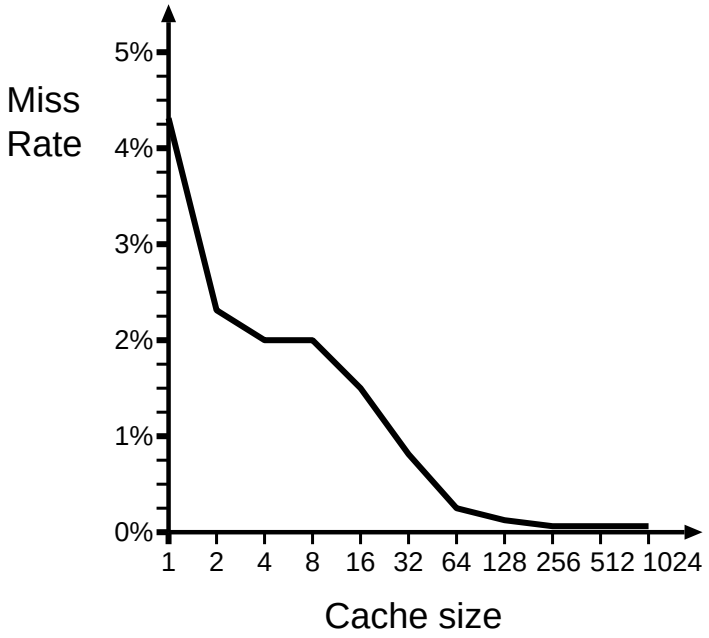
d Assume $M = 32$, and 16-byte cache lines. Estimate the write miss rate, and explain your answer. What additional assumptions must you make?

e Assume $N = M = 1024$, 16-byte cache lines and a write-back policy. Show how this loop could be restructured to achieve a high overall cache hit rate, and explain the resulting behaviour.

f Assume $M = 32$, 16-byte cache lines, and the cache is direct-mapped. Under what circumstances would a high rate of *associativity* misses occur when running this example?

(The six parts carry, respectively, 15%, 15%, 15%, 15%, 20% and 20% of the marks).

3 The graph below shows the approximate cache miss rate for a certain application, for increasing cache size:



Suppose the job is running on *one* processor of a two-processor shared-memory machine which uses a straightforward bus-based invalidation cache coherency protocol, with a cache for each processor, and a cache line size of 16 bytes. Assume the following:

1 25% of instructions are loads or stores.

2 The CPI assuming no cache misses is 1.5

3 The clock rate is 300MHz (clock period 3.33ns)

4 A cache miss takes 40 cycles (133ns)

a Calculate the MIPS rate of this application assuming no cache misses.

b Explain how it might be possible to improve performance while reducing the MIPS rate.

c Estimate the CPI the system will achieve on this application for cache sizes of 4KB, 16KB and 64KB.

d Suppose that the operating system's scheduler decides to stop the job, and restart it on the other CPU. To do this involves a management overhead of 40μs (to save the registers of the process, and to restart the job on the new CPU by setting up the registers).

Suppose that the new CPU's cache contains no useful data, so the restarted job will suffer cache misses which would not have occured if the job had not been migrated. *Estimate* the total delay attributable to additional cache misses each time the job is migrated, for the three cases when cache size is 4KB, 16KB and 64KB.

(The four parts carry, respectively, 10%, 20%, 30% and 40% of the marks).