332 Parallel Architectures

Final assessment test for MEng3 students

To be answered in **two hours** (you may refer freely to textbooks, handouts and your own notes).

ANSWER THREE QUESTIONS

Question 1

Cache consistency (sometimes called cache coherency) concerns the problem of ensuring that the correct value is always used, when caching leads to multiple copies of a value.

Describe as briefly as possible the cache consistency issues in the following situations (some of them do overlap—explain):

- a Separate caches for instructions and data.
- b Direct memory access for peripherals.
- c Multiple processors.
- d Virtual memory (suppose that the cache is indexed using a virtual address, and uses virtual addresses as tags).

Where possible, suggest how the consistency problem might be solved.

(Please turn over)

Question 2

When a write miss occurs, two policies are possible: a cache line could be allocated and updated with the value written (called *allocate-on-write*), or alternatively the cache could be left unchanged and the value could be written direct to main memory *(no-allocate-on-write)*.

a Explain as briefly as possible the meaning of the terms *write-back* and *write-through* when describing a cache design.

b Give a small example program in which a no-allocate-on-write policy is more efficient than allocate-on-write.

c Explain why allocate-on-write is problematic when large cache lines are in use.

d Explain carefully and as briefly as possible how sub-block placement can lead to reduced write-miss time.

e Some machines (e.g. the Intel i860) have two kinds of load instructions, one which allocates on load, the other which doesn't. Give an example and explain when this might be useful.

Question 3

Consider the following example program:

```
for i = 1 to 4 do
  for j = 1 to 4 do
    A[i,j] := A[i,j+1]+A[i-1,j]
```

a What dependences are present in this program? Explain.

b Explain how the *inner* loop can be vectorised.

c Can these loops be interchanged? Explain using a diagram.

d Suppose the 4×4 array `A` is distributed across a 4×4-processor SIMD machine. *Sketch* how this loop could be implemented, using a typical SIMD instruction set (pseudo-code is satisfactory).

You may find it useful to use the following instruction:

```
ROW m,K ≡ m[1:N,1:N] := 0
          m[1:N,K] := 1
```

The mask array `m` is set to 0 except for row `K` where it is set to 1.

(Please turn over)

Question 4

a Explain as briefly as possible the circumstances under which instruction issue must block in a dynamically-scheduled CPU using scoreboarding. Explain why.

b Explain as briefly as possible the circumstances under which instruction issue must block in a dynamically-scheduled CPU using Tomasulo's scheme. Explain why.

c What are the costs and benefits of loop unrolling?

Question 5

Suppose you have a P^2-processor MIMD machine with message-passing implemented on a $P \times P$ 2-dimensional torus (i.e. wraparound) mesh.

a Describe very briefly an efficient parallel implementation of a matrix multiplication

C := A $\times$ B;

Assume that A and B are large $N \times N$ matrices, whose initial placement can be chosen freely.

You may find it useful to consider the following sequential implementation of matrix multiply as a starting point:

```
for i = 1 to N
  for j = 1 to N
    for k = 1 to N
      C[i,j] := C[i,j] + A[i,k] * B[k,j];
```

b Suppose that A, B, C and D are large $N \times N$ matrices. Consider the the following sequence:

C := A $\times$ B;
D := C $\times$ A;

What problem arises?

(The marks for this question are distributed as follows: a: 80%, b: 20%).