Advanced Computer Architecture
MEng3 Test

Monday 17th March 2003

Answer two *questions*
You have one-and-half hours

1 This question concerns branch prediction in the IBM Power4 processor, as described in the paper "POWER4 System Architecture" (Tendler et al, IBM Journal of Research and Development, V46 No.1, Jan 2002), which you should have available to you in the examination. **See, in particular, pages 8 and 9.** Where the paper is incomplete, you are invited to speculate using your understanding of the underlying architectural principles.

a Give an example (in a suitable high-level language) in which the POWER4's local predictor would mispredict frequently, but the global history table would achieve better branch prediction. Briefly, explain why.

b How many bits of the instruction address would influence the prediction outcome in a 16K-entry *gselect(11,1)* predictor?

c In the POWER4, the 16K-entry global history table is indexed using the exclusive-or of the (low-order bits of the) branch instruction's address, and the 11-bit global history vector (this is called a *gshare* predictor).

(i) Explain why *gshare* sometimes leads to better branch prediction accuracy than the *gselect(11,1)* scheme, assuming the same number of entries.

(ii) Could *gshare* sometimes lead to worse branch prediction accuracy than *gselect*? Justify your answer carefully.

Why might this be so? Why did the POWER4 designers choose just one bit per entry?

d The POWER4's 32-entry Branch Target Buffer is called the "count cache". It is used for indirect branch and procedure call instructions (where the branch destination is given in a register).

The C/C++/Java "switch" statement allows control flow to be selected from a number of alternatives. A common use might be in a bytecode interpreter:

```
switch (opcode) {
case 1: /* code for when opcode==1 */
case 2: /* code for when opcode==2 */
...
case 64: /* code for when opcode==64 */
}
```

This can be implemented either using a sequence of conditional branches, or using an indirect branch. What determines whether it is better to use an indirect branch?

(The four parts carry, respectively, 10%, 15%, 25%, and 50% of the marks).

2 This question concerns dynamic instruction scheduling and speculative execution in the IBM Power4 processor, as described in the paper "POWER4 System Architecture" (Tendler et al, IBM Journal of Research and Development, V46 No.1, Jan 2002), which you should have available to you in the examination. **See, in particular, pages 11-14.** Where the paper is incomplete, you are invited to speculate using your understanding of the underlying architectural principles.

a POWER4 IOPs (internal instructions) are dispatched and committed in groups.

(i) How might this lead to an unnecessary stall at the Commit stage of the pipeline?

(ii) How might this lead to an unnecessary stall at the Dispatch stage of the pipeline?

(iii) When might this lead to unnecessarily executing an instruction multiple times?

b The following loop is shown in MIPS assembly code for your convenience. It computes the sum of a vector of double-precision (8-byte) floating point numbers:

```
$L5:
        l.d     $f4,0($4)     # vector base address in reg $4
        addu    $5,$5,-1      # no of elements in reg $5
        add.d   $f2,$f2,$f4   # sum into $f2
        addu    $4,$4,8
        bne     $5,$0,$L5     # (POWER4 branches are not delayed)
                              # on exit result is in $f2
```

Suppose this loop were executed on the POWER4, and that each MIPS instruction corresponds to one POWER4 IOP.

(i) Assuming no memory delays, estimate the number of cycles per 1000 iterations for this loop (note that the floating point units are fully-pipelined but take 6 cycles to complete).

(ii) Estimate the number of instructions per clock cycle (IPC).

(iii) Briefly explain how the loop could be modified to improve the performance (you do not need to show detailed code - just explain the principle).

(iv) Estimate the number of cycles per 1000 iterations that could be achieved using your restructured loop.

(v) Estimate the number of instructions per clock cycle (IPC).

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

3 This question concerns memory access in the IBM Power4 processor, as described in the paper "POWER4 System Architecture" (Tendler et al, IBM Journal of Research and Development, V46 No.1, Jan 2002), which you should have available to you in the examination. **See, in particular, pages 12-14.** Where the paper is incomplete, you are invited to speculate using your understanding of the underlying architectural principles.

a The POWER4 architecture supports two pages sizes – 4KB and 16MB (see page 14)

(i) Under what circumstances might an application benefit from using a page size of 16MB? Why?

(ii) Under what circumstances might an application benefit from using a page size of 4KB? Why?

b As in most modern designs, the POWER4's L1 cache is indexed by the virtual address (IBM call this the Effective Address). The L2 and L3 caches are indexed by the physical address (IBM call this the Real Address).

Each processor's L1 data cache is two-way set-associative with capacity 64KBytes. Draw a diagram which shows an example of how one physical memory location might be cached twice in the same L1 data cache, and show how this could lead to inconsistent results.

c Each speculatively-executed store instruction is delayed in the POWER4's Store Re-order Queue (SRQ) and Store Data Queue (SDQ) (see page 12). If a later load instruction refers to the same address, it must collect the data from the delayed store.

This mechanism relies on comparing virtual addresses. What problem does this raise? How could it be solved?

(The three main parts carry, respectively, 30%, 30% and 40% of the marks).

End of Paper