SOLUTION NOTES

Computer Design 2001 Paper 5 Question 2 (SWM)

(a) What is a pipeline bubble and why might a branch instruction introduce one or more bubbles?

[4 marks]

Ans: A pipeline bubble is introduced when an instruction is removed from a pipeline or the pipeline is forced to stall. A branch instruction may cause a pipeline bubble because instructions proceeding it may be fetched in error and so they have to be flushed from the pipeline, thereby introducing one or more bubbles.

(b) Explain, with the aid of an example, how conditional instructions may be used to reduce the number of bubbles in a pipeline.

[4 marks]

Ans: Conditional instructions may be used to remove short branches. Consider the following code fragment:

```
  if(a>b) a=b
```

with branches this would be written as (assuming a and b are held in registers):

```
  cmp a,b
  ble skip
  mov a,b
skip:
```

with conditional instructions this becomes:

```
  cmp a,b
  movgt a,b
```

if the ARM7 3 stage pipeline was in use then the first code excerpt would introduce 2 bubbles if the branch was taken where as the second excerpt would introduce at most 1 bubble (if the movgt instruction is skipped).

(c) What is the difference between branches, interrupts, software interrupts (initiated by a SWI instruction on the ARM) and exceptions?

[8 marks]

Ans: Branches are instructions inserted into the code by the programmer/compiler. They results in control flow changes within one application (usually within one protection domain). The destination

address is usually part of the branch instruction encoding.

Interrupts result in a jump to an interrupt handler whenever an external interrupt signal is sent to the processor. This is independent of the instructions being executed and typically results in the operating system being called and the protection domain changing to "supervisor" or the equivalent. The control flow target is typically specified via some interrupt vector table which usually contains a pointer into the operating system.

Software interrupts are like branches in that they are instructions inserted into the code by the programmer/compiler. However, like interrupts, their control flow target is specified by some interrupt vector table and the protection domain is also changed.

Exceptions are the result of some error in an instruction being executed. This initiates a change of control flow like an interrupt or software interrupt.

(d) What is an imprecise exception and why might a processor designer prefer it to a precise exception mechanism?

[4 marks]

A precise exception is one where the exception handler is able to determine which instruction caused the exception and is able to correct what ever caused the exception and restart the application as if the exception had not occurred. Thus data produced by any instructions after the one causing the exception must not have overwritten current data in the register file.

An imprecise exception is one where it is not possible for the exception handler to determine exactly which instruction caused the exception. Furthermore, intermediate results of instructions after the one causing the exception may have corrupted the register file.

Since less state has to be preserved with imprecise exceptions, it is easier to implement and so a processor designer may prefer it.