Computer Architecture Examination Paper with Sample Solutions and Marking Scheme

CS3 June 1995

Nigel Topham (January 7, 1996)

Instructions to Candidates

Answer any *two* questions. Each question carries 25 marks. Where marks are shown against a section of a question, they indicate the number of marks available for that section. Candidates have 90 minutes to complete the paper.

Question 1

a. Give a precise definition of hit ratio, as applied to cache memory. [2 marks]

b. Explain the difference between local and global hit ratios in the context of a multi-level cache hierarchy. [4 marks]

c. Explain why cache hit ratios are *not* an objective measurement of cache effectiveness. [6 marks]

d. A certain memory system has a single level of cache, connected to a main memory. The time to access memory at an address which is present in the cache is t_h, whereas the time for the same access when the address is not present in the cache is t_m.

Derive an equation for E, the effectiveness of the cache, defined as the ratio t_h/t_a, where t_a is the mean access time for all hits and all misses over the measurement interval. [8 marks]

e. Briefly discuss the similarities between your equation for part (d) and Amdahl's Law. [5 marks]

Question 2

a. The CPU time of a program is defined as the product of the CPI (cycles per instruction) for the processor on which it runs, the total number of instructions executed (I), and processor clock period (ϕ).

Desribe the major factors which influence CPI, I and ϕ. [8 marks]

b. For a new architecture to be worth developing it must have a commercial lifespan of at least 10 years.

What long-term factors must designers of a new architecture take into consideration during the design process? [8 marks]

c. Microprocessor core speeds increase at a rate of 40–60% per annum, compared with speed increases of 30% every ten years for DRAM devices.

In the light of this increasing discrepancy between CPU and main memory speeds, what can architects, system designers and memory chip designers do to reduce the harmful effects of high memory latency in future computer systems? [9 marks]

Question 3

A certain RISC microprocessor has a five-stage pipeline operating according to the register-transfer specification given in the table supplied overleaf. The pipeline hardware detects all data hazard conditions and stalls the pipeline when necessary for correct program behavour. You are asked to consider how such a pipeline would execute the following loop:

```
loop:  lw   $t0, ($s1)
       add  $s2, $s2, $t0
       sub  $s3, $s3, $s2
       add  $s4, $s4, $s2
       sw   $s4, ($s8)
       addi $s1, $s1, 4
       addi $s8, $s8, 4
       slt  $t1, $t5, $s2
       bne  $t1, $0, loop
```

a. (i) Draw a space-time graph showing the progression of the instructions for one iteration of the loop through this pipeline.

[Hint: your diagram should show time as processor cycles progressing vertically downwards, and space as pipeline stages progressing left-to-right across the page.] [7 marks]

(ii) How many cycles elapse between the initiation of successive iterations? [2 marks]

b. (i) Describe briefly the meaning of the term register bypass in the context of pipeline design. [5 marks]

(ii) What would be the revised initiation interval for successive loop iterations if the pipeline had unlimited bypassing capability? (explain your reasoning) [3 marks]

c. (i) Show how the instructions of the above loop can be legally re-ordered to fill the load delay slot and the branch delay slot of the load and the branch instructions respectively. [5 marks]

(ii) What is the resulting initiation interval for successive loop iterations? [3 marks]

Pipeline specification for Question 3

Stage	ALU instructions	Load or Store instructions	Branch instructions
IF	`IR ← Mem[PC];` `PC ← PC + 4;`	`IR ← Mem[PC];` `PC ← PC + 4;`	`IR ← Mem[PC];` `PC ← PC + 4;`

ID	`A ← Rs1;`
	`B ← Rs2;`
	`IR1 ← IR;`
		`A ← Rs1;`
		`B ← Rs2;`
		`IR1 ← IR;`
			`BTA ← PC+offset;`
			`if (Rs1` *op*[1] `0)`
			`PC ← BTA;`

EX	`ALUout ← A` *op*[2] `[B | literal];`	`DMAR ← A+offset;` `SMDR ← B;`	

MEM	`ALUout1 ← ALUout`	`LMDR ← Mem[DMAR];`
		or
		`Mem[DMAR] ← SMDR;`	

WB	`Rd ← ALUout1;`	`Rd← LMDR;`	

Notation

`PC`	program counter
`IR`	instruction register
`IR1`	copy of `IR` at ID stage
`Rs1`	source register 1
`Rs2`	source register 2
`A,B`	copies of `Rs1` and `Rs2`
`Rd`	destination register
`BTA`	branch target address
`ALUout`	ALU result register
`ALUout1`	copy of `ALUout` at MEM stage
`DMAR`	data memory address register
`SMDR`	store memory data register
`LMDR`	load memory data register
`offset`	16-bit sign-extended offset from $\texttt{IR}_{16\ldots31}$
`literal`	16-bit sign-extended offset from $\texttt{IR}_{16\ldots31}$
`Mem[`a`]`	contents of memory at address a

[1] $op \in \{=, \neq\}$

[2] op can be any integer or logical operation

Marking Scheme and Outline Solutions