E2.13

Corrected Copy

UNIVERSITY OF LONDON
IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY AND MEDICINE

EXAMINATIONS 2005

BEng Honours Degree in Computing Part II
MEng Honours Degrees in Computing Part II
BEng Honours Degree in Information Systems Engineering Part II
MEng Honours Degree in Information Systems Engineering Part II
for Internal Students of the Imperial College of Science, Technology and Medicine

This paper is also taken for the relevant examinations for the Associateship of the City and Guilds of London Institute

PAPER C210=E2.13

ARCHITECTURE II

Thursday 5 May 2005, 14:30
Duration: 120 minutes

Answer THREE questions

Paper contains 4 questions
Calculators required

1 a i) Provide a formula showing how the execution time of a program can be computed based on information such as the number of instructions for that program.

ii) Suggest three ways of improving the execution time for a given program.

b Two machines, M and N, implement the same instruction set with two classes of instructions, X and Y. M takes x cycles to run each Class X instruction and y cycles to run each Class Y instruction, while N takes αx cycles to run each Class X instruction and βy cycles to run each Class Y instruction. M runs at z Hz, while N runs at γz Hz. α, β and γ are constants.

i) State the peak performance, in number of instructions per second, of M and N.

ii) State the execution time of M and N for programs with m Class X instructions and n Class Y instructions.

iii) For M, a program P1 has $u1$ Class X instructions and $v1$ Class Y instructions, while another program P2 has $u2$ Class X instructions and $v2$ Class Y instructions. The corresponding parameters for N are respectively $r1$, $s1$, $r2$ and $s2$.

If P1 is run k times more frequently than P2, derive a value for k such that the total execution time for M will be less than that for N.

iv) For the two programs in part iii, what is the ratio of the clock speed for the two machines if P1 is run as frequently as P2, and the total execution time is the same for both machines?

The two parts carry, respectively, 30%, and 70% of the marks.

2a Consider a 4-bit Arithmetic Logic Unit (ALU) which can perform addition and bit-wise logical OR for numbers in two's complement representation.

i) Provide a circuit diagram showing the internal structure of a component A which can be replicated 4 times to form the ALU. Do not show the internal structure of multiplexors and fulladders.

ii) Provide a circuit diagram showing how the ALU can be built from 4 copies of A. Label the signal values on all the wires in the diagram when the ALU adds the two numbers 0011 (3 in base ten) and 1111 (-1 in base ten).

b i) Show how A (in part a) can be modified to become B by adding a single 2-input logic gate, so that the appropriate connection of multiple copies of B can compute either bit-wise OR, addition or subtraction depending on a control signal.

ii) Provide a circuit diagram of this modified ALU which contains 4 copies of B.

c i) Show how B (in part b) can be modified to become C by adding a single 2-input logic gate and an inverter, so that the appropriate connection of multiple copies of C, together with additional components, can compute either bit-wise OR, addition or the absolute value of one of its inputs.

ii) Provide a circuit diagram of this modified ALU which contains 4 copies of C.

The three parts carry, respectively, 25%, 35%, and 40% of the marks.

3 a A datapath containing m control signals has been developed for a set of instructions with n-bit opcode. A microsequencer, which supports k-bit horizontal microinstructions, has been developed as the control unit for this datapath. The microsequencer contains an incrementer, a ROM (Read-Only Memory), a register, and address select logic. Provide a diagram for this microsequencer, and label the size of components and wires in the diagram where possible.

b Describe how the address select logic in part a can be implemented using ROMs and multiplexors. Explain how the number of ROMs in the address select logic relates to the state diagram for the control unit.

c Given that there are α identical ROMs in the address select logic, what is the total ROM size (in bits) that this microsequencer contains?

d Calculate the total ROM size (in bits) if the control unit is implemented as a finite state machine containing a single ROM for both the output logic and the next state logic.

The four parts carry, respectively, 30%, 30%, 20%, and 20% of the marks.

4 a What is the total number of bits of storage required for a direct-mapped cache with 2^m bytes of data and one-word blocks, given that the address size is n bits and each word contains 2^k bytes?

b Give one advantage and one disadvantage of a direct-mapped cache with multi-word blocks.

c What is the total number of bits of storage required for a direct-mapped cache with 2^m bytes of data and multi-word blocks, given that the address size is n bits, each block contains 2^α words and each word contains 2^k bytes?

d Give one advantage and one disadvantage of a fully-associative cache.

e What is the total number of bits of storage required for a fully-associative cache with 2^m bytes of data, given that the address size is n bits and each word contains 2^k bytes?

f Compare your answer to part a and part e, and explain whether there are additional factors that can affect the size of the two caches.

The six parts carry, respectively, 20%, 10%, 25%, 10%, 20%, and 15% of the marks.

IE2 - Computer Architecture

SOLUTIONS 2005

Department of Computing Examinations — 2004–2005 Session	*Confidential*
MODEL ANSWER and MARKING SCHEME	
First Examiner wl	Paper Code C210 = E2.13
Second Examiner om	Question 1 Page 1 *out of* 4
Question labels in left margin	Mark allocations in right margin

1a i) $T = nct$, T: exec. time of program, n: the instruction count, c: cycles per instruction, t: cycle time of machine running the program — 3

ii) To improve exec time:
- use better compiler to reduce n
- use machine with lower c
- " " " reduced t — 3

b i) instructions per second = $\frac{\text{cycles per second}}{\text{cycles per instr}}$

Peak performance of $M = \frac{z}{\min(x,y)}$ Peak performance of $N = \frac{\gamma z}{\min(\alpha x, \beta y)}$ — 3

ii) exec time for M = exec time for x instr. + exec time for y instr. $= \frac{mx}{z} + \frac{ny}{z} = \frac{(mx+ny)}{z}$

exec time for $N = \frac{m\alpha x}{\gamma z} + \frac{n\beta y}{\gamma z} = \frac{\alpha m x + \beta n y}{\gamma z}$ — 3

iii) exec time for M < exec time for N

$k \cdot$ exec time for P1 on M + exec time for P2 on M < $k \cdot$ exec time for P1 for N + exec time for P2 for N

$$k\left(\frac{u_1 x + v_1 y}{z}\right) + \left(\frac{u_2 x + v_2 y}{z}\right) < k\,\frac{r_1 \alpha x + s_1 \beta y}{\gamma z} + \frac{r_2 \alpha x + s_2 \beta y}{\gamma z}$$

$$k\left(\frac{u_1 x + v_1 y}{z} - \frac{r_1 \alpha x + s_1 \beta y}{\gamma z}\right) < \frac{r_2 \alpha x + s_2 \beta y}{\gamma z} - \frac{u_2 x + v_2 y}{z}$$

$$k < \frac{r_2 \alpha x + s_2 \beta y - u_2 \gamma x - v_2 \gamma y}{u_1 \gamma x + v_1 \gamma y - r_1 \alpha x - s_1 \beta y}$$ — 5

iv) exec time for M = exec time for N and $k = 1$,

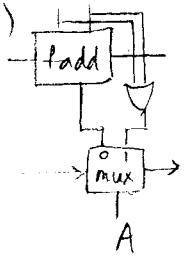
$$u_1 \gamma x + v_1 \gamma y - r_1 \alpha x - s_1 \beta y = r_2 \alpha x + s_2 \beta y - u_2 \gamma x - v_2 \gamma y$$

$$\gamma(u_1 x + v_1 y + u_2 x + v_2 y) = r_2 \alpha x + s_2 \beta y + r_1 \alpha x + s_1 \beta y$$

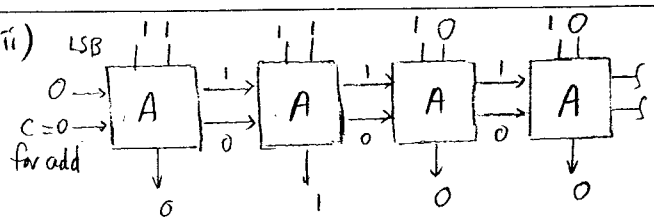
$$\gamma = \frac{(r_1 + r_2)\alpha x + (s_1 + s_2)\beta y}{(u_1 + u_2)x + (v_1 + v_2)y}$$ — 3

2 a

i)



ii)

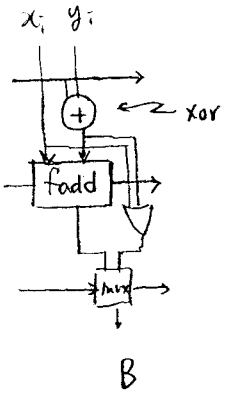


C = 0 : Add, C = 1 : or

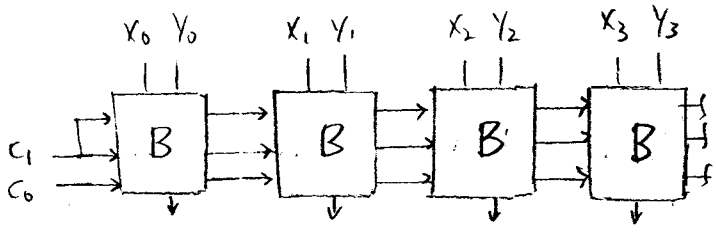
5

b

i)



ii)

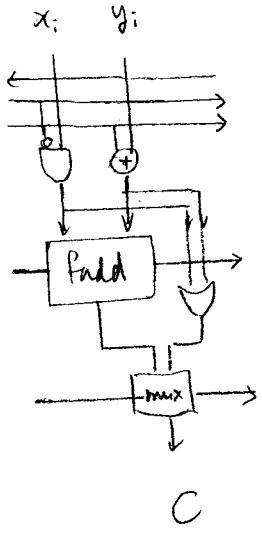


C_0	C_1	fn
0	0	add
0	1	sub
1	0	or
1	1	or

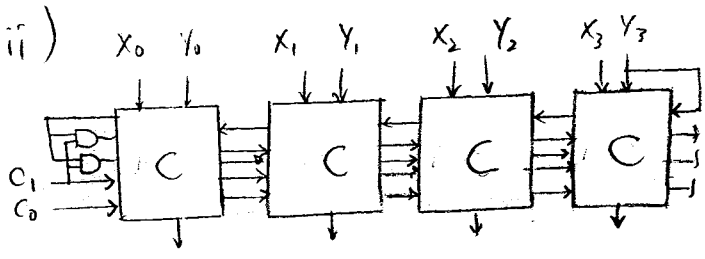
7

c

i)



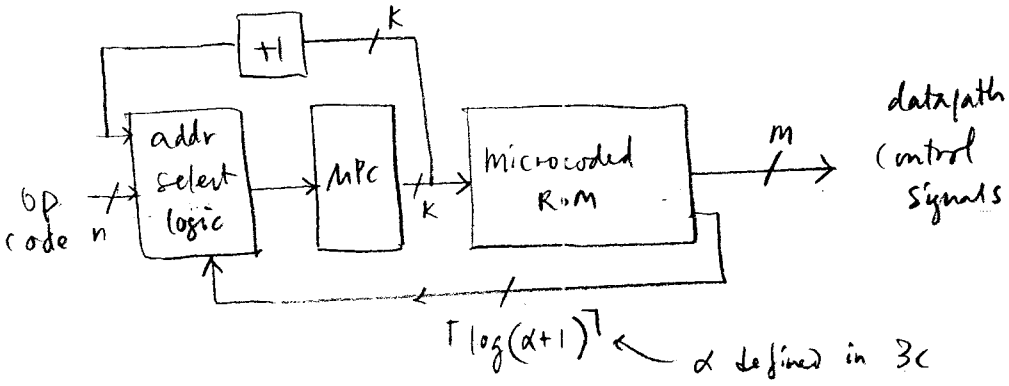
ii)



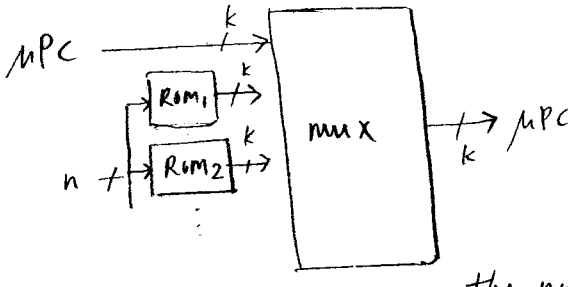
C_0	C_1	function	
0	0	add	(x, y)
0	1	abs	(Y)
1	0	or	(x, y)
1	1	or	(x, y)

8

3a



b. addr select logic

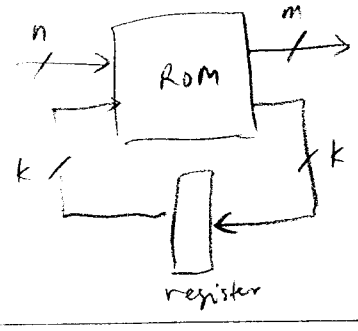


The number of ROMs is the same as the number of states which have more than one possible next states.

c. total ROM size for microsequencer = α (ROM size for addr decode) + μcoded ROM size

$= \alpha\,(2^n \cdot k) + 2^k\,(m + \lceil \log_2(\alpha+1) \rceil)$

d. ROM size for FSM = $2^{n+k}\,(m+k)$



4a total number of words = $(2^m)/(2^k) = 2^{m-k}$ = total number of blocks

tag size = $n - (m-k) - k = n-m$

total size = (num of words) x (addr size + tag size + valid bit)

$= 2^{m-k}(2^{k+3} + (n-m) + 1)$

$= 2^{m+3} + 2^{m-k}(n-m+1)$ — 4

b adv: fetch multiple words per block: explicit spatial locality esp. for instructions

disadv: more complex & slower, more complex control for write hit/miss (cf single word) to avoid false sharing — 2

c total number of blocks $= 2^{m-k-\alpha}$

tag size = $n-m$

total size = $2^{m-k-\alpha}(2^{\alpha+k+3} + n - 3 + 1)$

$= 2^{m+3} + 2^{m-k-\alpha}(n-m+1)$ — 5

d adv: efficient use of cache, since no miss unless cache full

disadv: need storage for tags, more complex control — 2

e same as 4a — 4

f direct mapped cache has 1 comparator

fully associative cache has 2^{m-k} comparators — 3