

# NATIONAL UNIVERSITY OF SINGAPORE

## SCHOOL OF COMPUTING

### EXAMINATION FOR

Semester 2: 2004/5

### CS1104 – COMPUTER ORGANISATION

April 2005

Time allowed: 2 hours

---

#### **INSTRUCTIONS TO CANDIDATES**

1. This examination paper consists of **FIVE (5)** questions and comprises **TWENTY FOUR (24)** printed pages. The first question comprises 20 Multiple-Choice Questions (MCQs).
2. Pages 17 – 24 contain several figures and tables related to questions 4 and 5.
3. This is an **OPEN BOOK** examination.
4. Answer all questions. You may use pencil to draw diagrams.
5. Answer the MCQs on the MCQ Answer Sheet provided.
6. Answer questions 2 – 5 on the Answer Book provided.
7. For questions 4 and 5, you may, in addition, use the pages 17 – 24 of this examination booklet to draw figures and fill out the tables. Irrespective, of whether you use these pages or not, please attach them to your Answer Book. Do not forget to write your matriculation number on these pages.

**Section I: Multiple Choice Questions [20 marks]**

- Write as well as shade your Matriculation Number clearly on the MCQ Answer Sheet provided. Do not write your name.
- There are 20 Multiple Choice Questions. Each question has one correct answer. Shade your answers clearly on the MCQ Answer Sheet.
- Each correct answer will earn you one mark. No penalty will be given for incorrect answers.
- Do not bend, fold or soil your MCQ Answer Sheet.

1.1 Which of the following statements is true?

- A. The DeMorgan's Theorem is just another name for duality.
- B. For some Boolean functions, their sum-of-products expression and product-of-sums expression might be identical.
- C. The fan-out of a logic gate is the number of outputs of that gate.
- D. The exclusive-OR (XOR) logic gate is a universal gate.
- E. None of the above.

1.2 A certain four-variable Boolean function contains 9 minterms. At least how many product terms and at most how many product terms are there in the minimal sum-of-products expression for this function?

- A. At least zero and at most 9.
- B. At least 1 and at most 9.
- C. At least 2 and at most 8.
- D. At least 2 and at most 9.
- E. None of the above.

1.3 A certain 4-bit self-complementary code is used to represent the ten decimal digits. Which of the following could be some of its code values?

- A. 3 = 1011; 5 = 1111; 6 = 0101.
- B. 3 = 0101; 5 = 1001; 6 = 0100.
- C. 3 = 1100; 5 = 0011; 6 = 0011.
- D. 3 = 0010; 5 = 1111; 6 = 1101.
- E. None of the above.

1.4 Given the following functions:

$$P(A,B,C) = \Sigma m(1, 3, 5)$$

$$Q(A,B,C) = \Sigma m(0, 1, 2, 3)$$

What is the function  $R(A,B,C)$  if  $R(A,B,C) = P(A,B,C) \oplus Q(A,B,C)$ ?

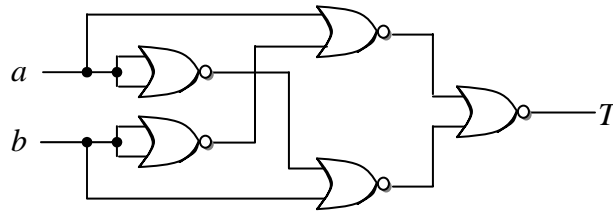
- A.  $\Sigma m(1, 3, 5)$
- B.  $\Sigma m(0, 1, 2, 3)$
- C.  $\Sigma m(0, 1, 2, 3, 5)$
- D.  $\Sigma m(1, 3)$
- E.  $\Sigma m(0, 2, 5)$

For questions 1.5 – 1.8, refer to the Boolean function  $F(A,B,C,D)$  as given below.

$$F(A,B,C,D) = \Sigma m(6, 7, 8, 10) + d(2, 3, 4, 5, 12, 14)$$

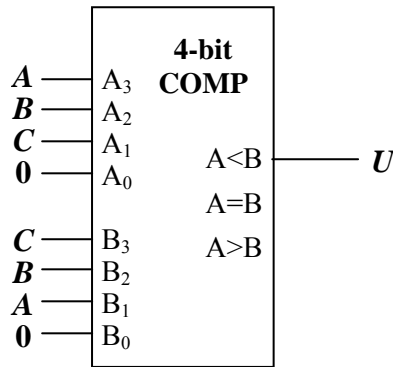
- 1.5 How many **prime implicants (PIs)** and **essential prime implicants (EPIs)** are there in the K-map of  $F$ ?
- A. 2 PIs and 2 EPIs
  - B. 3 PIs and 2 EPIs
  - C. 5 PIs and 2 EPIs
  - D. 5 PIs and 1 EPI
  - E. 5 PIs and 0 EPI
- 1.6 How many possible **minimal SOP expressions** for  $F$  are there?
- A. 1
  - B. 2
  - C. 3
  - D. 4
  - E. 5
- 1.7 Out of the six don't cares given in the original function  $F$ , how many of them are included into the list of minterms in the minimal SOP expression?
- A. 4
  - B. 3
  - C. 2
  - D. 1
  - E. 0
- 1.8 Which of the following is a **minimal POS expression** for  $F$ ?
- i.  $(A' + B').(B' + D).(A + B)$
  - ii.  $(A' + D').(A + B)$
  - iii.  $(A' + D').(A + C)$
  - iv.  $(A + D).(A' + B')$
  - v.  $(A + D).(A' + C')$
- A. (i) only
  - B. (ii) only
  - C. (iv) only
  - D. (ii) and (iii)
  - E. (iv) and (v)

1.9 Given the logic circuit below, what is  $T$ ?



- A.  $a$  XNOR  $b$
- B.  $a$  XOR  $b$
- C.  $a$  OR  $b$
- D.  $a$  AND  $b$
- E.  $a$  NOR  $b$

1.10 Given the logic circuit below, what is  $U(A,B,C)$ ?



- A. 0
- B.  $\Sigma m(1, 3)$
- C.  $\Sigma m(0, 2, 5, 7)$
- D.  $\Sigma m(0, 1, 2, 3, 5, 7)$
- E.  $\Sigma m(0, 2, 4, 5, 6, 7)$

Recall the single-cycle and the multicycle implementations we studied in this module. Given the functional unit latencies shown on the right, answer the following two questions (questions 1.11 and 1.12)

Func. Unit	Latency
Memory	4 ns
ALU	3 ns
Register File	2 ns

- 1.11 What is the minimum cycle time for the single cycle implementation?
- A. 11ns
  - B. 15ns
  - C. 13ns
  - D. 12ns
  - E. 17ns
- 1.12 What is the minimum cycle time for the multicycle implementation?
- A. 2ns
  - B. 3ns
  - C. 4ns
  - D. 5ns
  - E. Can not be determined from this information
- 1.13 Pipelining improves instruction
- A. Throughput
  - B. Latency
  - C. Cache miss rate
  - D. Width
  - E. Density
- 1.14 An ideal pipelined datapath at the steady state has a CPI of 1. However, several reasons increase the CPI to more than 1. Which among the following is NOT responsible for increasing the CPI?
- A. Register width
  - B. Structural and control hazards
  - C. Cache misses
  - D. Branch misprediction
  - E. Instruction sequences such as:
    - LW        \$t0, 0(\$t1)
    - ADD       \$t3, \$t0, \$t0

For question 1.15, please refer to the description below.

SingComputers currently manufactures four different processors—Kallang, Sentosa, Tampines and Jurong—with the following specification:  $CPI(Kallang) = 1$ ,  $CPI(Sentosa) = 30$ ,  $CPI(Tampines) = 1$  and  $CPI(Jurong) = 2.5$ . All of these processors have the same ISA, but can be run at different clock speeds.

- 1.15 Configuration X = 10 GHz Sentosa, Y = 300 MHz Kallang, Z = 30 MHz Tampines
- From the above information, it cannot be determined whether Y or Z is faster
  - It is not possible that different processors have the same ISA but are clocked at different speeds
  - X is faster than Y
  - X and Z are of the same speed
  - If someone has to decide between configuration Y and some configuration involving Jurong, then Jurong should be clocked at least at 120 MHz for it to be faster.
- 1.16 Which of the following statements is NOT true about MIPS addressing modes?
- Register addressing gets its operands from registers
  - In base addressing, an offset is added to the base address
  - In immediate addressing, the lower 16 bits of the instruction are used
  - For PC-relative addressing, used in branches, the upper 16 bits are obtained from the PC
  - In pseudo-direct addressing, used in jumps, the upper 4 bits are obtained from the PC
- 1.17 Which of the following is NOT true?
- Accessing data from registers is faster than accessing data from memory
  - Registers speed up execution because typically a program accesses the same set of variables within a small time window
  - Some of the architectures we studied in this module did not have registers
  - Having more registers in an architecture makes the compiler writer's job difficult because now he/she will have to manage more registers
  - Having more registers in an architecture will typically increase the size of the instructions

- 1.18 Which of the following is NOT true? The size of a program in machine instructions, measured in terms of number of bits, depends on:
- A. The compiler used
  - B. The number of registers in the architecture
  - C. Whether the datapath is single-cycle, multicycle or pipelined
  - D. The size of the memory
  - E. Whether the instruction set architecture is a stack or a register load-store architecture
- 1.19 Branch prediction is used
- A. in a single-cycle datapath
  - B. to improve the parallelism in a program
  - C. to minimize the effects of data hazards
  - D. to minimize the effects of control hazards
  - E. to minimize the effects of structural hazards
- 1.20 A condition in the hazard detection and forwarding unit is  $(EX/MEM.RegisterRd \neq 0)$ . Which of the following reasons behind using this condition is NOT true?
- A. It gives the programmer more flexibility
  - B. This condition is concerned with the register \$0
  - C. Even without this condition, the following code will work correctly  
add \$3, \$2, \$1  
add \$4, \$3, \$2
  - D. Even without this condition, the following code will work correctly  
add \$0, \$2, \$1  
add \$4, \$3, \$0
  - E. The reasons for using the condition  $(EX/MEM.RegisterRd \neq 0)$  and the condition  $(MEM/WB.RegisterRd \neq 0)$  are the same

Q2. [20 marks]

- a) Convert the quinary (base-5) value  $2101.4_5$  to its ternary (base-3) equivalent, correct to 4 ternary places.

Then, convert the value  $2101.4_5 \div 3^4$  to ternary base, correct to 3 ternary places.

[5 marks]

- b) A certain floating-point representation has 1-bit sign, 5-bit normalised mantissa, followed by 4-bit two's complement exponent. Given the following representations of two values  $A$  and  $B$ :

$$A = 0, 10100, 0010$$

$$B = 1, 10000, 1111$$

Compute the sum of  $A$  and  $B$ . Write out this sum in decimal form, and then in the above floating-point representation. [5 marks]

- c) Given the following function table of a certain logic device that takes in 6-bit input  $ABCDEF$  and produces 3-bit output  $PQR$ , write out the minimal SOP expressions for  $P$ ,  $Q$  and  $R$ . [8 marks]

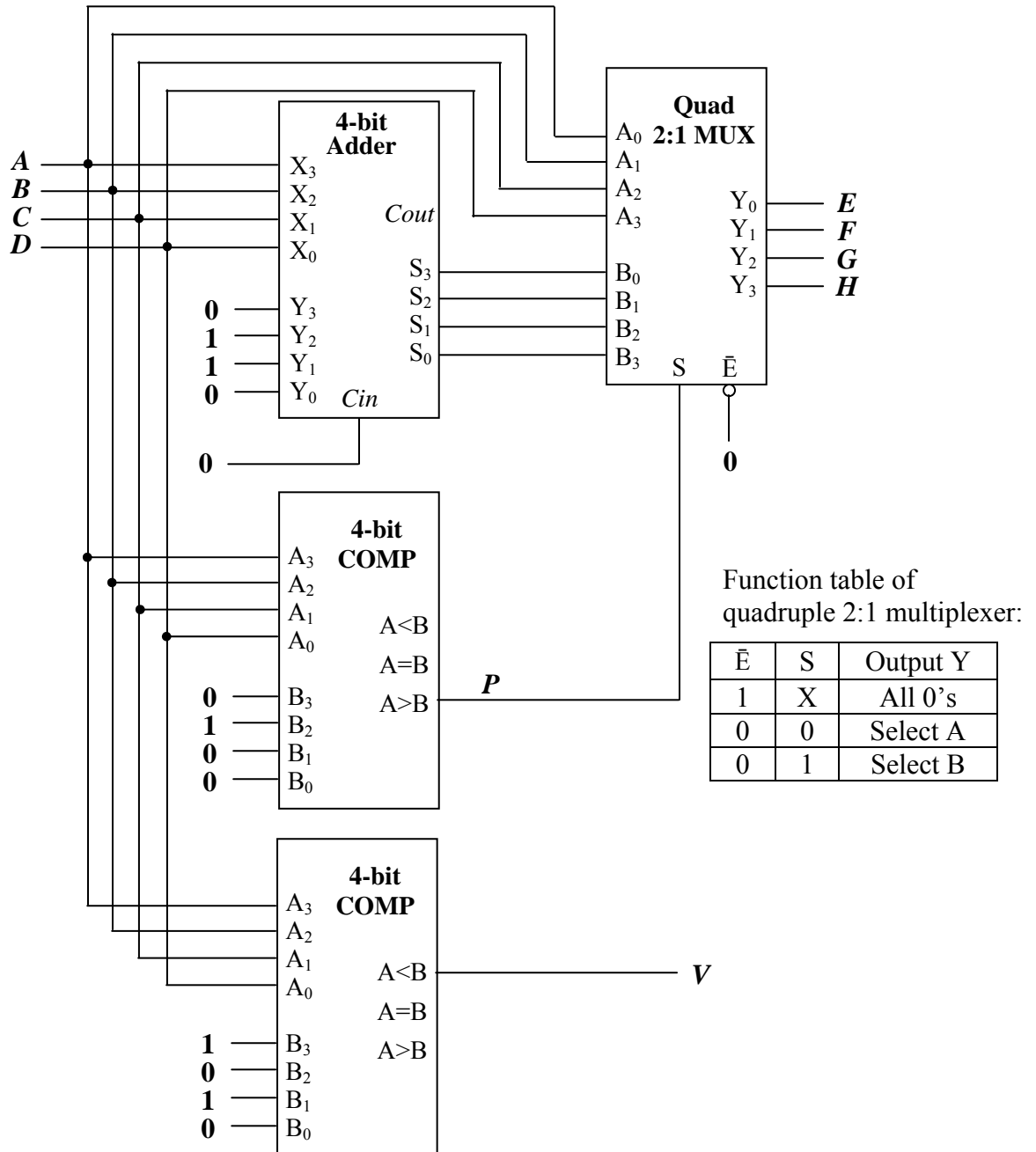
$A$	$B$	$C$	$D$	$E$	$F$	$P$	$Q$	$R$
X	X	X	X	X	0	0	0	0
X	X	X	X	0	1	0	0	1
X	X	X	0	1	1	0	1	0
X	X	0	1	1	1	0	1	1
0	0	1	1	1	1	1	0	0
1	0	1	1	1	1	1	0	1
0	1	1	1	1	1	1	1	0
1	1	1	1	1	1	1	1	1

- d) You are given only 2-input AND gates and 2-input OR gates. Implement  $R$  in part (c) above using the fewest number of these gates, assuming that primed literals are available. Draw the logic circuit. [2 marks]

Q3. [20 marks]

a) The logic circuit below shows a code converter that takes in a 4-bit code  $ABCD$  and converts it into a 4-bit code  $EFGH$ . It consists of two 4-bit magnitude comparators, a 4-bit parallel adder, and a quadruple 2:1 multiplexer devices. The output  $V$  is 1 if  $ABCD$  is a valid code, or 0 otherwise.

Analyse the circuit and complete the given truth table. From the truth table, deduce what kind of code converter it is. [10 marks]



<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>	<i>P</i>	<i>V</i>	<i>E</i>	<i>F</i>	<i>G</i>	<i>H</i>
0	0	0	0						
0	0	0	1						
0	0	1	0						
0	0	1	1						
0	1	0	0						
0	1	0	1						
0	1	1	0						
0	1	1	1						
1	0	0	0						
1	0	0	1						
1	0	1	0						
1	0	1	1						
1	1	0	0						
1	1	0	1						
1	1	1	0						
1	1	1	1						

Q3. (continued)

- b) Implement a full adder with inputs  $X$ ,  $Y$ , and  $Z$  and outputs  $C$  (carry) and  $S$  (sum) using two 4:1 multiplexers with no additional gate, assuming that primed literals are available. [5 marks]
- c) Implement the function  $F(A,B,C,D) = \Sigma m(3, 10, 11)$  using a single 2×4 decoder with zero-enable and at most one of these logic gates: inverter, AND, OR, NAND, NOR, XOR, XNOR. Primed literals are not available. [5 marks]

Q4. [20 marks]

a) Consider the following function in MIPS assembly language:

```

mycode:  li      $v0, 0
         li      $t0, 0
         li      $t1, 0
L1:      bge     $t1, $a1, L3
         mul     $t2, $t1, 4
         add     $t2, $t2, $a0
         lw      $t3, 0($t2)
         ble     $t3, $t0, L2
         move    $v0, $t1
         move    $t0, $t3
L2:      add     $t1, $t1, 1
         j       L1
L3:      jr      $ra

```

- (i) Translate the above function `mycode` into a high-level language like C or Java. Please include the types of any arguments and return values. Your code should be as concise as possible, without any `gotos` or pointer arithmetic. Marks will not be deducted for minor syntax errors (unless they are significant enough to alter the meaning of your code). [4 marks]
- (ii) Describe what the function does. Be as precise as possible (restrict your answer to **no more than 3 sentences**). [1 mark]

Q4. (continued)

b) Some of the following instructions cannot be carried out in the datapath shown on page 17 of this question paper. For each such instruction, explain why it cannot be implemented.

Restrict your answer to **only 1 sentence for every instruction**.

(i) `add rd, rs, rt`

0	rs	rt	rd	0	0x20
6	5	5	5	5	6

(ii) `lw rt, offset(rs)`

0x23	rs	rt	offset
6	5	5	16

(iii) `j target`

2	target
6	26

(iv) `lui rt, imm`

0xf	0	rt	imm
6	5	5	16

(v) `bne rs, rt, label`

5	rs	rt	offset
6	5	5	16

[5×2 = 10 marks]

Q4. (continued)

- c) The multicycle datapath that we studied in the class is reproduced on page 18 of this question paper. You are required to make suitable changes to this datapath to implement a new instruction called **xchng2**. This instruction takes 4 registers and exchanges/swaps pairs of values, as shown below

**xchng2 \$t1, \$t2, \$t3, \$t4;**    \$t1 ↔ \$t2, \$t3 ↔ \$t4

The coding of this instruction is as follows: \$t1 is in the rs field, \$t2 is in the rt field, \$t3 is in the rd field and \$t4 is in the bits [0-4] of the instruction (i.e. the bottom 5 bits). You are not allowed to add extra read or write ports to the register file. In the datapath shown on page 18, we have added labels 1, 2, 3, 4, 5 and 6. The necessary changes are to be made at these points only

- (i) Explain what these changes are. You **need not** show the necessary changes in the control unit. Restrict your answers to **at most 3 sentences per label** and **draw a clear figure** to explain each change. [4 marks]
- (ii) How many cycles will this instruction require to execute? Explain. [1 mark]

Q5. [20 marks]

a) Consider the following C code:

```
for (int i = 0 ; i < max ; ++ i)
{
    ++ A[i];
}
```

The assembly language translation of the inner loop of this code is:

```
loop:      sub      $t3,      $t3,      1
           add      $t0,      $t0,      4
           add      $t1,      $a0,      $t0
           lw       $t2,      0($t1)
           add      $t2,      $t2,      1
           sw       $t2,      0($t1)
           bgt     $t3,      $zero,     loop
```

- (i) Indicate all dependences within one iteration of this code (not just the ones that will require forwarding). “One iteration” is defined as the instructions between the `sub` and `bgt` (both inclusive). This code has been reproduced for you on page 19 of this question paper. [2 marks]
- (ii) Assume that the loop has executed many times and the pipeline has reached the steady state of execution. Now, assume that the pipeline **supports forwarding** (you can assume that a register can be read in the same cycle it is written) but **does not implement branch prediction**. For your reference, this pipelined architecture is shown on page 20 of this question paper. Using the grid given in page 21 of this question paper, indicate the pipeline stage each instruction is in for each cycle (**stalls can be marked with an S or --**). Some of the cells in this grid have been filled up for you. [3 marks]
- (iii) Label each instruction of the above assembly language program with the forwarding path it requires (1, 2, 3, 4: as labeled in the figure in page 20). The program has again been reproduced for you once again on page 22.

[2 marks]

Q5. (continued)

b) The pipelined datapath with branch prediction that we studied in the class is shown on page 23 of the question paper. Consider the following instruction sequence:

```
sub $2, $4, $5
```

```
beq $2, $3, Label1
```

- (i) Why doesn't this instruction sequence work properly in the datapath shown on page 23? [3 marks]
- (ii) What are the necessary changes in the datapath such that this instruction sequence works correctly? Please be precise and restrict your answer to no more than 4 sentences. If required, draw a clear figure or make suitable changes in the figure on page 23. [4 marks]

Q5. (continued)

- c) Given below is a series of memory read references. The cache associated with the memory is of size 128 bytes. It has 2-word blocks (i.e. 64 bits), is 2-way set associative, and uses a least-recently-used (LRU) replacement policy. Assume that the cache is initially empty.

Classify each memory reference as a hit or a miss, and classify each cache miss as either “compulsory”, “conflict”, or “capacity”. To answer this question, simply fill out the following table. For your convenience, this table has been reproduced on page 24 of the question paper as well. [6 marks]

Address	Hit/Miss?	Miss Type? compulsory/conflict/capacity
0x7		
0x4D		
0x2A		
0x79		
0xAB		
0xCE		
0x2E		
0x4B		
0x6D		
0x8A		
0xAF		
0x29		
0xC8		
0xCE		
0x6A		

—End of Questions—

(Pages 17 – 24 contain figures and tables for Q4 and Q5)

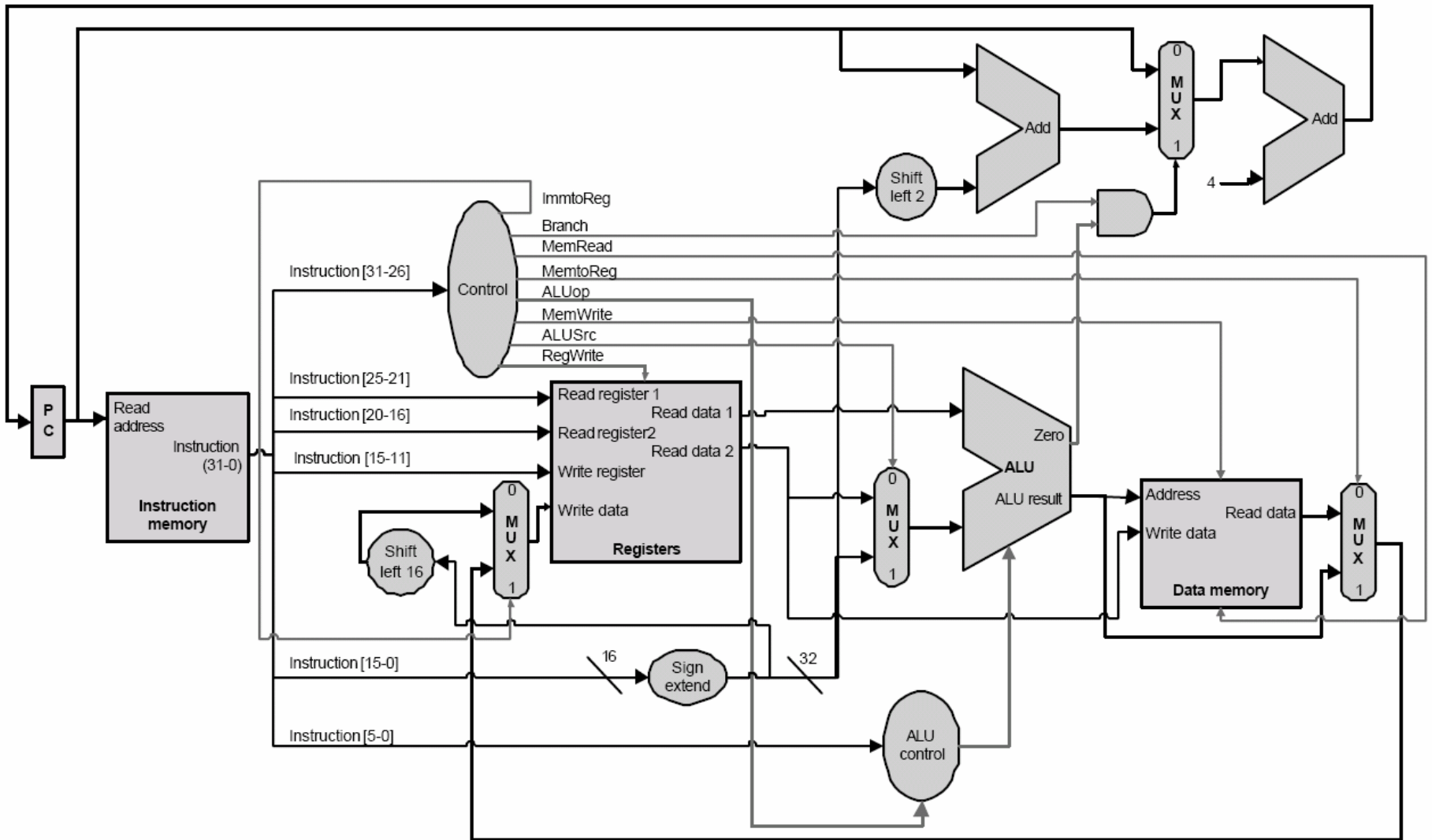
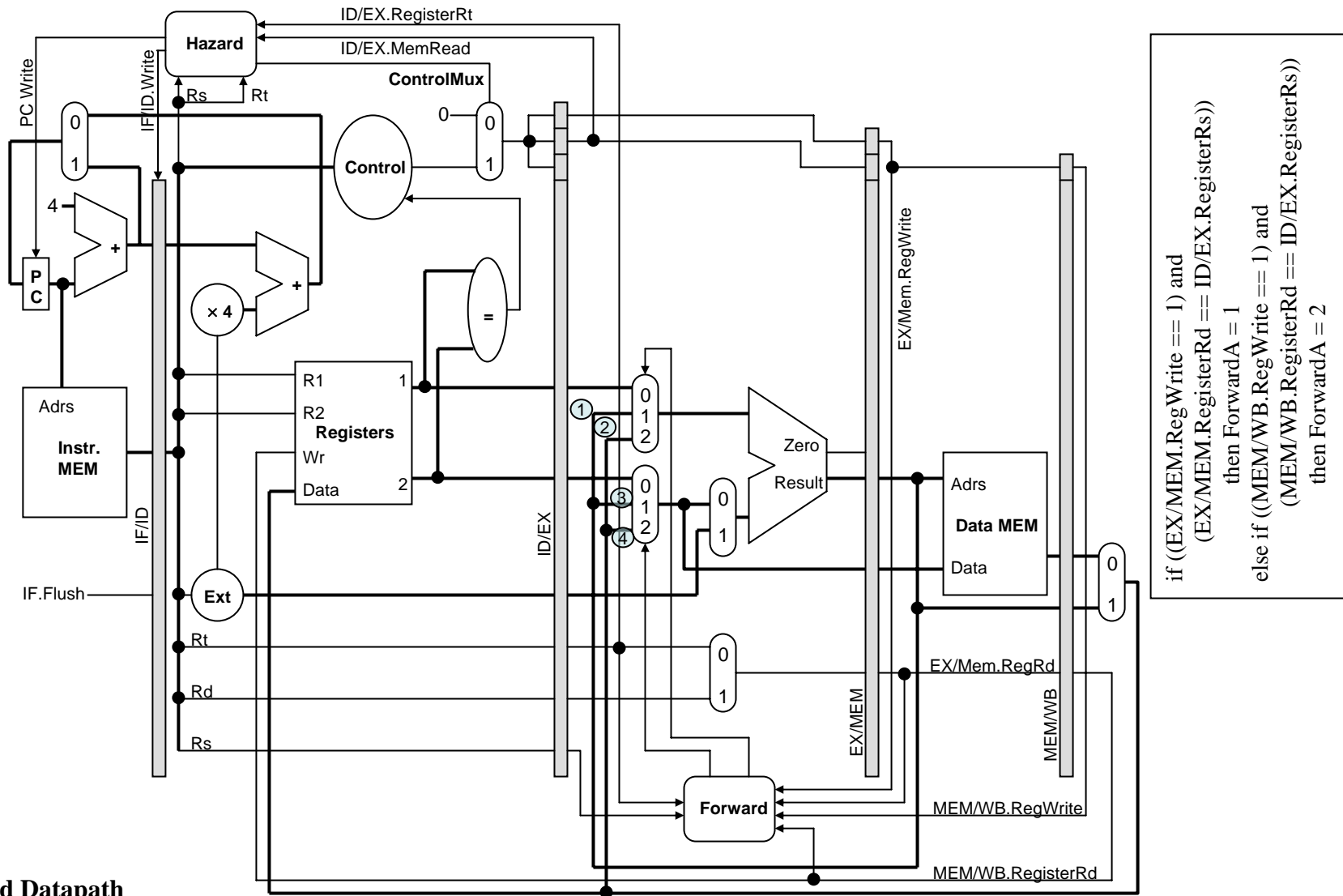


Figure for Q4b)



```
loop:  sub    $t3,    $t3,    1
      add    $t0,    $t0,    4
      add    $t1,    $a0,    $t0
      lw     $t2,    0($t1)
      add    $t2,    $t2,    1
      sw     $t2,    0($t1)
      bgt   $t3,    $zero,  loop
```

**Figure for Q5a) (i)**



**Pipelined Datapath**

- The control equation for the Rs register input to the ALU is shown on the right. The Rt input is similar.
- Branch resolution takes place in the Decode stage.
- A hazard unit inserts stalls for lw and branch instructions

**Figure for Q5a)**

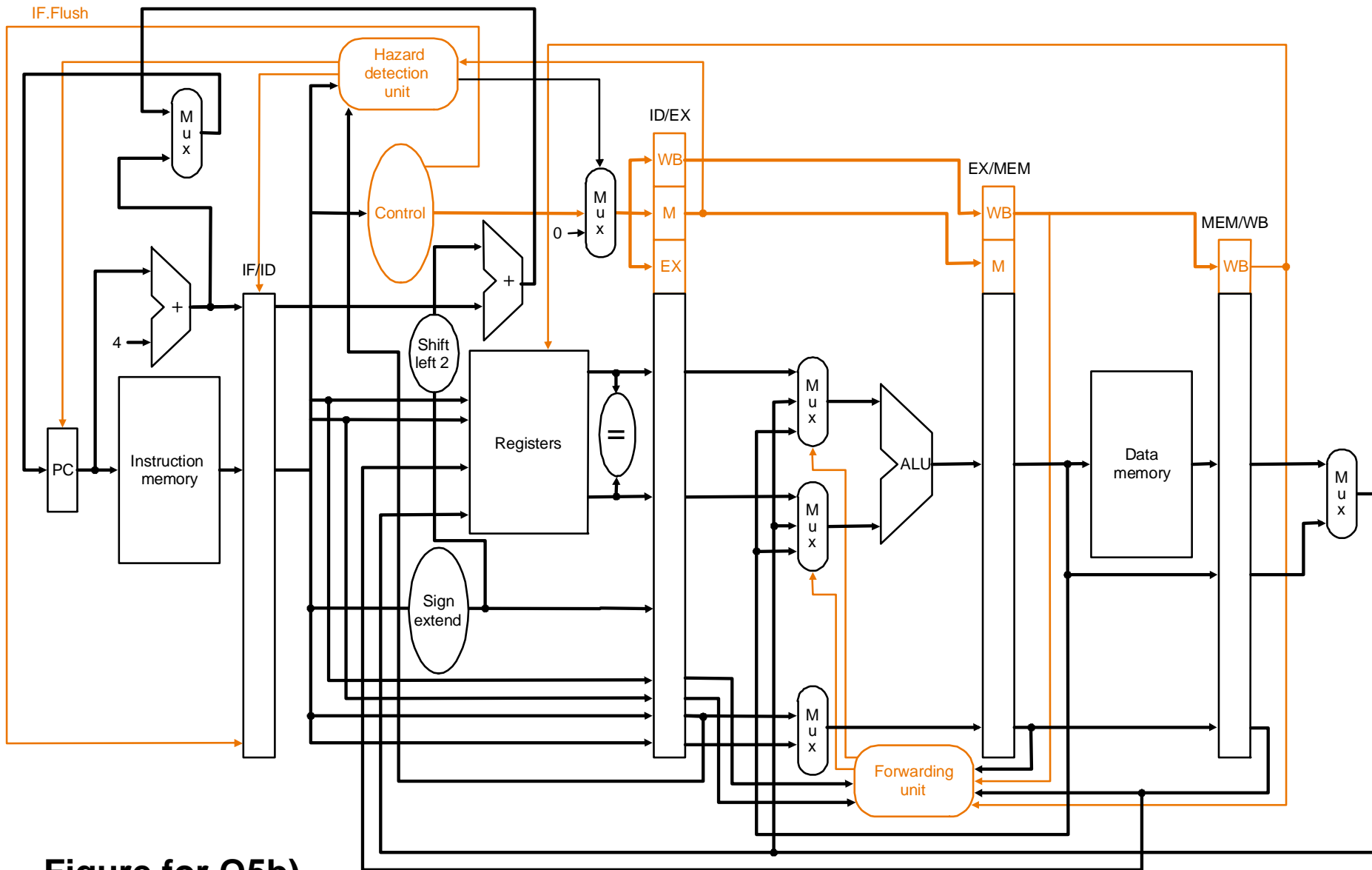
Inst	iter	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	
sub	N	F	D	E	M	W																		
add	N		F																					
add	N																							
lw	N																							
add	N																							
sw	N																							
bgt	N																							
sub	N+1																							

**Figure for Q5a) (ii)**

Matriculation No. \_\_\_\_\_

					Forwarding path (1, 2, 3 or 4)
loop:	sub	\$t3,	\$t3,	1	_____
	add	\$t0,	\$t0,	4	_____
	add	\$t1,	\$a0,	\$t0	_____
	lw	\$t2,	0(\$t1)		_____
	add	\$t2,	\$t2,	1	_____
	sw	\$t2,	0(\$t1)		_____
	bgt	\$t3,	\$zero,	loop	_____

**Figure for Q5a) (iii)**



**Figure for Q5b)**

<b>Address</b>	<b>Hit/Miss?</b>	<b>Miss Type? compulsory/conflict/capacity</b>
<b>0x7</b>		
<b>0x4D</b>		
<b>0x2A</b>		
<b>0x79</b>		
<b>0xAB</b>		
<b>0xCE</b>		
<b>0x2E</b>		
<b>0x4B</b>		
<b>0x6D</b>		
<b>0x8A</b>		
<b>0xAF</b>		
<b>0x29</b>		
<b>0xC8</b>		
<b>0xCE</b>		
<b>0x6A</b>		

**Table for Q5c)**

Matriculation No. \_\_\_\_\_