Uva Wellassa University, Sri Lanka
End Semester Examination – July 2010
CST 104-2 Microcomputer Architecture & Logic Design
Time: Two (02) hours

Total 04 Questions.
Answer **all** questions.

---

01) I. Explain what is a combinational circuit. (03 marks)

II. Decoder is a combinational circuit. Give another 04 examples for combinational circuits. (04 marks)

III. Design a 3-8 line decoder from 2-4 line decoders and draw the logic diagram. (08 marks)

IV. Fig. 1 shows a Two-to-One binary multiplexer (MUX). The output X is, A or B depending on the value of C.

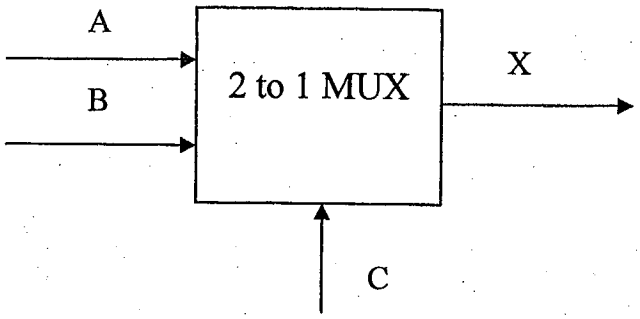

Fig. 1

i. Construct a truth table for the operation of this multiplexer.

ii. Derive the logic expression for the multiplexer output.

iii. Simplify the logic expression.

iv. Draw a logic circuit for the multiplexer. (10 marks)

02) I. Draw a block diagram of a sequential circuit. (03 marks)

II. Explain the difference between a Latch and Flip Flop. (04 marks)

III. Explain what is meant by "Latches are transparent". (03 marks)

IV. Draw logic circuits for S-R Latch and $\bar{S}$-$\bar{R}$ Latch. (05 marks)

V. Describe the two ways that latches are combined to form a flip-flop. (06 marks)

VI. Explain the difference between S-R Flip Flop and J-K Flip Flop. (04 marks)

03) I. Name and explain four types of addressing modes in computer instructions. (08 marks)

II. Using an example illustrate how for a given task, stack based addressing would reduce the size of the assembly language program. (06 marks)

III. List five generic functions of an I/O module. (05 marks)

IV. Compare "Programmed I/O" and "Direct Memory Access" as related to I/O data transfer. (06 marks)

04) I. Illustrate the translation hierarchy by describing the four steps in transforming a C program in a file on disk into a program running on a computer. (07 marks)

II. Compare registers, instruction operands, memory and addressing modes, function calls and instruction formats of MIPS architecture and 8086 architecture. (06 marks)

III. Convert the following JAVA code segment to 8086 instruction set, to find the maximum of two given numbers. Choose the suitable instructions from the given 'Instruction set' on page 03.

```
public class max {
      public int findMax(int i, int j) {
            int max = i;
            if(j>i){
                max = j;
            }
            return max;
      }
      public static void main(String[] args) {
            max  m1 = new max();
            System.out.println(m1.findMax(7, 8));
      }
}
```

(12 marks)

(2) a. The internal structure of the Intel 8086 microprocessor can be divided into two parts as Bus Interface Unit (BIU) and Execution Unit (EU). What are the major operations perform inside these two units? (4marks)

b. What is the main different between two execution modes, Pipelining and Single Step? (3marks)

c. What are the uses of following registers in the Intel 8086 microprocessor? (3marks)

i. Segment registers

ii. Data registers

iii. Instruction Pointer

d. To develop the logical address concept the memory is dividing into several segments. The maximum amount of memory that a segment can have in 8086 microprocessor is 64KB. Briefly explain the reason for this limitation. (5marks)

e. i. The physical address of a memory location is 4B653 in hexadecimal format. Find a possible logical address for this memory location. (10marks)

ii. During the execution of a program, the SS contains 3B21H, IP contains 0085H and SP contains 0045H. Calculate the physical address of the next memory location to be access in the stack segment.

----------END----------

20

Instruction set

Registers to store values : AX, BX, CX
SI, DI

Syntax for a variable declaration : *name* DB *value*
DB - stays for Define Byte.

Copies the second operand (source) to the first operand (destination) :
MOV REG, memory
MOV memory, REG
MOV REG, REG
MOV memory, immediate
MOV REG, immediate

Defining a Label : *Example*: SUM:

Defining a procedure : *Example*:
```
name PROC
        ; here goes the code
        ; of the procedure ...
RET
name ENDP
```

To call a procedure : call PROC

JZ, JE	→	jump if Zero(Equal)
JC , JB, JNAE	→	Jump if Carry (Below, Not Above Equal)
JNZ , JNE	→	Jump if Not Zero (Not Equal)
JMP	→	always jump
LOOP	→	decrease cx, jump to label if cx not zero.
LOOPE	→	decrease cx, jump to label if cx not zero and equal (zf = 1).
LOOPNE	→	decrease cx, jump to label if cx not zero and not equal (zf = 0).
INC	→	operand = operand + 1

Example:
```
MOV AL, 4
INC AL     ; AL = 5
RET
```

LEA → the address of an indexed variables

Example :
```
MOV   AL, VAR1; check value of VAR1 by moving it to AL.
LEA   BX, VAR1     ; get address of VAR1 in BX.
```

DEFINE_PRINT_NUM → required for print_num
procedure that prints a signed number in AX register.