UNIVERSITY OF SWAZILAND

Faculty of Science

Department of Computer Science

Main Examination, May 2010

Title of paper: **COMPUTER ORGANISATION - I**

Course numbers: **CS241**

Time allowed: 3 hours

Instructions:
- Answer any 5 out of the 6 questions. Each question carries 20 marks.
- **The use of electronic calculators is forbidden.**

THIS EXAMINATION PAPER SHOULD NOT BE OPENED UNTIL PERMISSION HAS BEEN GRANTED BY THE INVIGILATOR

Question 1

Write short notes on any 3 of the following topics:

a) Fetch-decode-execute cycle
b) CISC vs RISC
c) Instruction pipelining
d) Memory hierarchy
e) Synchronous bus timing [20]

Question 2

a)

i. Write the 12 bit integer 001001101101_2 in hexadecimal and octal notation. [2]

ii. On a byte addressed machine, the integer $A1B2C3_{16}$ is stored in main memory starting at address 1000. At what addresses are the constituent bytes stored? In addition, what values are stored at each address, assuming that little endian byte ordering is used? [4]

b)

i. Distinguish between IEEE single- and double-precision floating point formats. [4]

ii. In hexadecimal notation, write -4.125 in IEEE single-precision format. [4]

iii. Explain the following terms in relation to floating point:

- Overflow and underflow errors
- Rounding
- Normalization [6]

Question 3

a) Draw a circuit showing 2 transistors connected together to form a NAND gate. [3]

b) Draw the following logic circuits:

i. 3-bit comparator. [3]

ii. 2-to-4 decoder. [6]

iii. Left-right shifter with a single bit control input (labelled C). The 4-bit data input (labelled D) must be shifted right by 1 bit when C=0, otherwise D must be shifted left by 1 bit. [8]

Question 4

a)

i. Draw the D-flip-flop's action table. [4]

ii. Draw a logic circuit of a clocked D-flip-flop that changes state on the falling edge of the clock. [7]

b)

i. Why are the following needed in memory chips?
- WE and CS inputs
- Decoder [4]

ii. Draw a pin diagram of a byte addressed memory chip having a capacity of 16 kilobits. [5]

Question 5

a) Explain the purpose of the following main-memory areas of the IJVM virtual machine:

i. Operand stack
ii. Local variable frame
iii. Method area [6]

b) The main loop of the Mic-1 microarchitecture is driven by a single microinstruction:

```
Main1          PC = PC + 1; fetch; goto (MBR)
```

i. What are in PC and MBR registers just before this microinstruction is executed?

ii. What is fetched from main memory? When does the data arrive at the CPU and where is it stored on arrival?

iii. What microinstruction is executed immediately after this one? [6]

c) Comment in detail on Mic-1's implementation (given below) of IJVM's ILOAD instruction: [8]

```
iload1         H = LV
iload2         MAR = MBRU + H; rd
iload3         MAR = SP = SP + 1
iload4         PC = PC + 1; fetch; wr
iload5         TOS = MDR; goto Main1
```

Question 6

a) Translate the following Pascal code into IJVM assembly language (assume that a and b have already been declared as integers): [13]

```
a := 3;
b := a;
WHILE a > 0
  BEGIN
    b := b + a;
    a := a - 1
  END
```

b) Translate your assembly language program from question a) into IJVM machine code (refer to the table of IJVM opcodes below). [7]

Hex	Mnemonic	Meaning
0x10	BIPUSH *byte*	Push byte onto stack
0x59	DUP	Copy top word on stack and push onto stack
0xA7	GOTO *offset*	Unconditional branch
0x60	IADD	Pop two words from stack; push their sum
0x7E	IAND	Pop two words from stack; push Boolean AND
0x99	IFEQ *offset*	Pop word from stack and branch if it is zero
0x9B	IFLT *offset*	Pop word from stack and branch if it is less than zero
0x9F	IF_ICMPEQ *offset*	Pop two words from stack; branch if equal
0x84	IINC *varnum const*	Add a constant to a local variable
0x15	ILOAD *varnum*	Push local variable onto stack
0xB6	INVOKEVIRTUAL *disp*	Invoke a method
0x80	IOR	Pop two words from stack; push Boolean OR
0xAC	IRETURN	Return from method with integer value
0x36	ISTORE *varnum*	Pop word from stack and store in local variable
0x64	ISUB	Pop two words from stack; push their difference
0x13	LDC_W *index*	Push constant from constant pool onto stack
0x00	NOP	Do nothing
0x57	POP	Delete word on top of stack
0x5F	SWAP	Swap the two top words on the stack
0xC4	WIDE	Prefix instruction; next instruction has a 16-bit index