UNIVERSITY OF SWAZILAND

Faculty of Science

Department of Computer Science

SUPPLEMENTARY EXAMINATION 2012

Title of paper: COMPUTER ORGANISATION II

Course number: CS341

Time allowed: 3 hours

Instructions to candidates:

This question paper consists of FIVE (5) questions. Answer any FOUR (4) questions. Marks are indicated in the square brackets.
All questions carry equal marks.

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

QUESTION 1

a) What is the main purpose of the control store? Communication with memory can be done in 2 different ways. State and briefly describe the 2 ways in which memory access can be achieved. [7]

b) With the aid of appropriate diagrams, describe the following allocation algorithms:
   i) First fit
   ii) Worst fit [8]

c) With the aid of suitable diagrams compare the following:
   i) Cache miss and page fault
   ii) Conditional and unconditional branching [8]

d) Why does the Intel have segment registers and SPARC not? [2]

QUESTION 2

a) Assume you have an expanding ***opcode*** that supports the following formats, with a 3 bit register:

   4 Instruction with 3 registers
   255 Instructions with one register
   16 Instructions with zero registers

   i) How many ***opcodes***, in total, does the preceding require? [4]
   ii) How many ***bits*** does the ***opcode*** require to support the 3 formats? [4]

b) What is a latch? Give one example of a latch and describe how it works. [6]

c) What is Microprogramming? What are the advantages and drawbacks of microprogrammed control compared to hardwired control? [7]

d) State and describe any 4 datapath registers [4]

QUESTION 3

a) Briefly explain how the concept of speculative execution is useful in improving performance. [5]

b) Evaluate the following arithmetic expression into Decimal.

   $121_{16} + 122_{10} - 123_{8}$ [5]

c) Evaluate the following reverse polish expression, where each number is a (decimal) digit. $ABCDE * F / + G - H / * +$ [5]

d) Convert into reverse polish notation the following infix expression (where operators have their usual/normal precedence) $(2\times 3+4)-(4/2+1)$. Generate IJVM code to evaluate it. Show values in the Stack during the evaluation. [10] ***(See the IJVM instruction set on last page)***

QUESTION 4

a) Why do interrupt service routines have priorities associated with them whereas normal procedures do not have priorities? [7]

b) Using correct terminology and illustrations, in short paragraphs, describe the following terms:

i)	Segmentation	[7]
ii)	Paging	[7]
iii)	Paged segmentation	[4]

QUESTION 5

a) Syntax and Semantic error messages refer to source code line numbers. Illustrate how these numbers are affected by Macro Expansion. [6]

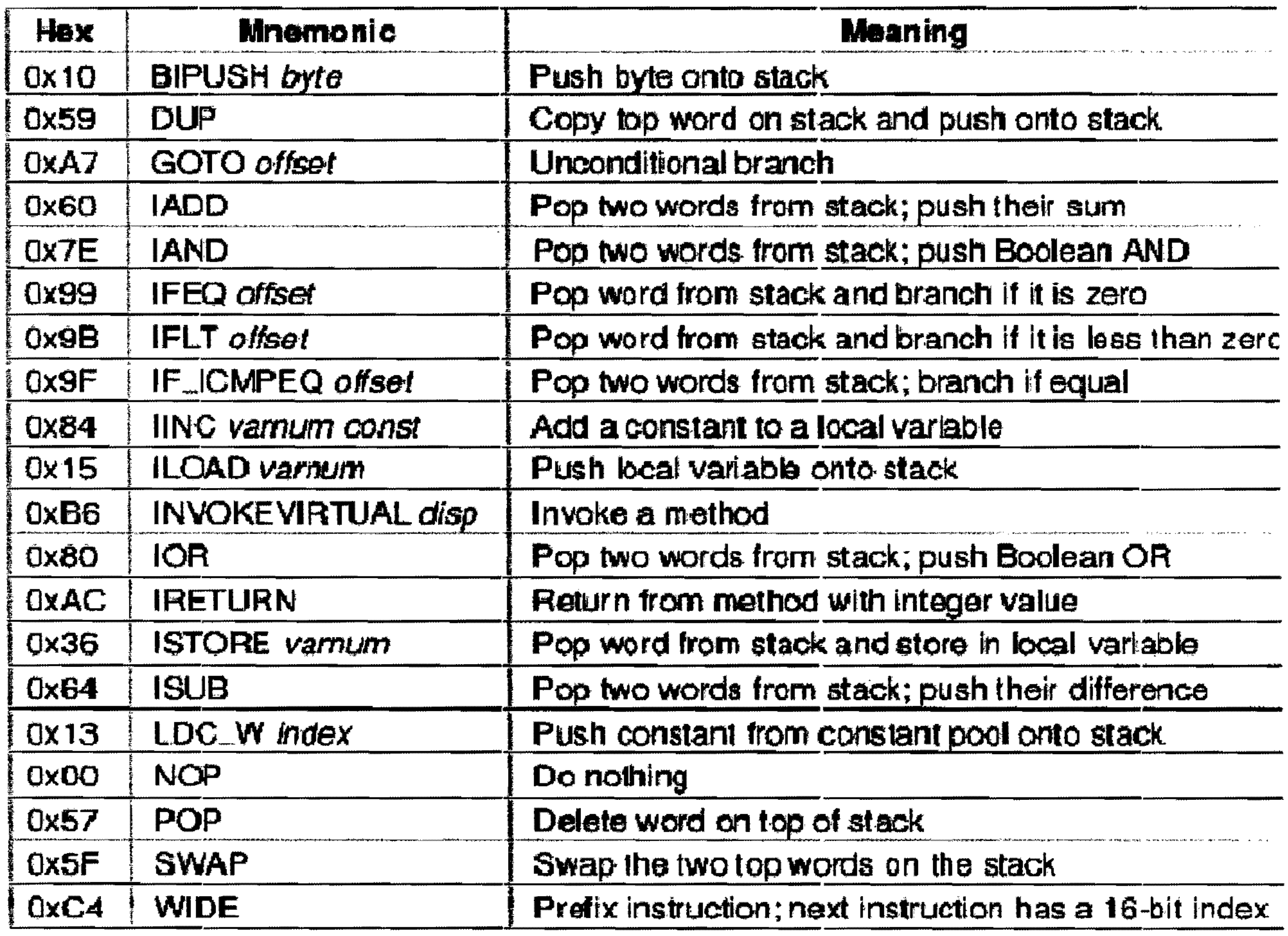
b) An Instruction set has 4 bits for opcode and 32 bits for addresses. What percentage change in instructions and memory resolution results if the opcode is increased by 2 bits without altering the instruction length (by taking bits from address portion)? [6]

c) Distinguish between the following pairs of concepts.

i.	Shared and distributed memory systems	[3]
ii.	Big Endian and Little Endian	[3]
iii.	Immediate and Indirect addressing modes	[3]

d) Distinguish between the relocation and external reference problem with respect to linker functions. [4]

The IJVM Instruction Set



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 zerc
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