OLLSCOIL NA hÉIREANN, CORCAIGH
THE NATIONAL UNIVERSITY OF IRELAND, CORK

COLÁISTE NA hOLLSCOILE, CORCAIGH
UNIVERSITY COLLEGE, CORK

SUMMER EXAMINATION 2004

First Year Computer Science

CS1101: Systems Organisation

Professor Muffy Calder
Professor Cormac J. Sreenan
Dr. Barry O'Sullivan

Instructions

Answer all questions.

All questions carry equal marks.
This examination is worth 160 marks.
Coursework submitted during term is worth 40 marks.

Calculators may be used.
Please indicate the make and model of your calculator at the start of your exam script.

Duration

3 Hours

1. a) Explain any 3 of the following, making use of suitable examples:

   i. In the context of shell scripts, what are positional parameters?
   ii. Explain how simple arithmetic computations can be performed in a shell script. What limitations are there on the types of the arithmetic expressions that can be processed?
   iii. Explain how access permissions on files in UNIX can be modified.
   iv. In the context of shell scripts, what is a subshell?

(*8 marks*)

b) In the laboratory sessions for this course students were asked to develop simple shell scripts as exercises. The `test` command can be used to distinguish between files and directories. For example, `test -f filename` is true if `filename` is a file, while `test -f filename` is true if `filename` is a directory.

Write a shell script for the Bourne Shell which fulfills the following specification.

Name: `myls` – list the contents of a directory labelling elements as either files or directories.

Syntax: `myls <path>`

Description: This is the outline for `myls`:

- Select `sh`
- Use a `for` loop to examine each file/directory in the directory denoted by path (Hint: You should use `ls` to get the list of files/directories). Use an `if` statement based on test on each file to distinguish between files and directories.

(*16 marks*)

c) Explain *precisely* the effects of the following UNIX commands. Note that `<return>` means pressing the Return or Enter key on the keyboard; `file1` and `file2` are files; `www` and `var` are directories;

   i. `mkdir ~/www <return>`
   ii. `cd ../.. <return>`
   iii. `cp file1 ~john/file2 <return>`
   iv. `mv file1 ~/john <return>`
   v. `chmod ugo=r file1 <return>`

(*8 marks*)

2. a) Explain any 3 of the following, making use of suitable examples:

   i. Show how to represent a 4-bit negative number in the appropriate *excess notation*
   ii. Explain the *fetch-execute-cycle*?
   iii. Explain the relationship between the binary radix system and both octal and hexadecimal systems.
   iv. What is the difference between a signed an unsigned binary number? For 8-bit binary, what is the maximum number representable as a signed and as an unsigned number?

(*8 marks*)

b) Answer all of the following:

   i. Convert the following numbers to binary using both the *successive halving method* and the *powers of two method*:
      - 17

- 26

ii. Convert the both of the above numbers into octal and hexadecimal.

iii. Convert the following numbers into 8-bit *signed-magnitude*, *one's complement*, *two's complement* and *excess notation*:

- -11
- +11

iv. What is the largest number that can be represented in each of the following:

A. 8-bit signed magnitude;

B. 8-bit one's complement;

C. 8-bit two's complement;

D. 8-bit excess notation;

(*16 marks*)

c) Explain the differences between a *carry-in* bit and a *carry-out* bit when adding two numbers. Illustrate how binary addition works using 2 1-bit numbers to illustrate.

(*8 marks*)

3. a) Explain any 3 of the following, making use of suitable examples:

i. Explain, using De Morgan's Law (see Figure 1 on Page 5), the relationship between a NAND Gate and an OR-Gate, and between an NOR Gate and an AND-Gate.

ii. Illustrate with the aid of a diagram, how two 1-bit full adders can be used to add 2 2-bit numbers. Assume that the carry-out is to be handled according to the two-complement convention.

iii. Explain how an SR Latch works. In particular, making use of diagrams, explain what is meant by State 0 and State 1 of the latch.

iv. How many Boolean functions involving 2 inputs are there?

(*8 marks*)

b) Consider the following truth-table – having 2 inputs (A,B) and 2 outputs (Carry-out and Sum):

A	B	Carry-out	Sum
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

i. Derive Sum-of-Products expressions for each of the outputs in the truth-table;

ii. Draw a logic circuit of the Sum-of-Products expressions you have derived.

(*16 marks*)

c) Algebraicly (using De Morgan's Laws), derive an equivalent NAND-only implementation of the XOR-Gate. (*8 marks*)

4. a) Explain any 3 of the following, making use of suitable examples:

   i. Explain how the range and precision of a number are controlled in scientific notation.

   ii. How are zero, infinite and 'not-a-number' represented in the IEEE 754 Floating Point Standard?

   iii. Give examples of 3 instruction types used at the Instruction-Set Architecture Level.

   iv. What is a microprogram?

(*8 marks*)

b) i. Convert the following decimal numbers into IEEE 754 format single precision numbers. Give your answer in hexadecimal.

   - 10.5
   - -2.25

   ii. Convert the following IEEE 754 format single precision numbers into decimal.

   - 3F880000
   - 3F000000

(*16 marks*)

c) Give examples and discuss 3 different addressing modes used at the Instruction-Set Architecture Level of a computer. Make use of examples. (*8 marks*)

5. a) Explain what is meant by the term *virtual memory*. Discuss how it could be implemented. A diagram should be used to illustrate your explanation. (*8 marks*)

b) What is the difference between the Instruction-Set Architecture Level and Operating System Level of a computer? What additional features does an operating system bring? (*8 marks*)

c) In the context of the assembly process, explain the processes *linking* and *loading*. (*8 marks*)

d) In the context of assembly languages, briefly explain the following terms:

   i. pseudo-instruction

   ii. macro

   iii. macro-expansion

   iv. machine code

(*8 marks*)

Name	AND form	OR form
Identity law	1A = A	0 + A = A
Null law	0A = 0	1 + A = 1
Idempotent law	AA = A	A + A = A
Inverse law	$A\overline{A} = 0$	$A + \overline{A} = 1$
Commutative law	AB = BA	A + B = B + A
Associative law	(AB)C = A(BC)	(A + B) + C = A + (B + C)
Distributive law	A + BC = (A + B)(A + C)	A(B + C) = AB + AC
Absorption law	A(A + B) = A	A + AB = A
De Morgan's law	$\overline{AB} = \overline{A} + \overline{B}$	$\overline{A + B} = \overline{A}\,\overline{B}$

Figure 1: Table of Boolean identities.