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

COLÁISTE NA hOLLSCOILE, CORCAIGH
UNIVERSITY COLLEGE, CORK

SUMMER EXAMINATION 2003

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. What is a UNIX shell? How can it be regarded as a programming language?

   ii. What are *positional parameters*? Give examples of how they can be used in shell scripts.

   iii. Explain how the UNIX operating system handles access privileges to files and directories.

   iv. What is a multi-user computer?

   (*8 marks*)

   b) In the laboratory sessions for this course students were asked to develop simple shell scripts as exercises. Write a shell script for the Bourne Shell which fulfills the following specification.

   Name: `chex` - change a file to be executable.

   Syntax: `chex <filename>`

   Description: This is the outline for `chex`:

   - Select `sh`
   - Apply `chmod u+x` to the file named as argument ($1)
   - Inform the user that file is executable
   - Use `ls -l` to show the file's modes

   (*16 marks*)

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

   i. `mkdir www <return>`

   ii. `mv file1 file2 <return>`

   iii. `cp ../file1 ../../file2 <return>`

   iv. `mv file1 ../www <return>`

   v. `chmod uog-rwx file1 <return>`

   vi. `chmod a-r file1 <return>`

   (*8 marks*)

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

   i. How many bits are required to represent a hexadecimal digit? How many are required for an octal digit? Why?

   ii. Give examples of two approaches to introducing instruction level parallelism into a CPU.

   iii. Explain how to compute the excess notation representation of an n-bit binary number.

   iv. Constrast RISC and CISC architectures.

   (*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*:

   - 20
   - 28

   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*:

   - -32
   - 22

iv. Consider the following addition problems for 3-bit binary numbers in two's complement. For each state (i) if the sign bit of the result is 1 or 0, and (ii) if an overflow has occurred.

A. $000 + 001$

B. $000 + 111$

C. $111 + 110$

D. $100 + 111$

E. $100 + 100$

(*16 marks*)

c) On computer 1, all instructions take 10nsec to execute. On computer 2, they all take 5 nsec to execute. Can you say for certain that computer 2 is faster? Discuss.

(*8 marks*)

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

i. Use a truth table to show that *P = (P and Q) or (P and not Q)*.

ii. Illustrate, with the aid of a diagram, how a *delay* can be used in conjunction with a *clock* to implement an *asymmetric clock signal*. Explain what time references such circuits can give.

iii. Illustrate, with the aid of truth tables and/or diagrams, how the *half adder* differs from the *full adder* circuit.

iv. An 3-8 decoder circuit is illustrated in Figure 1. Explain how the circuit works.

(*8 marks*)

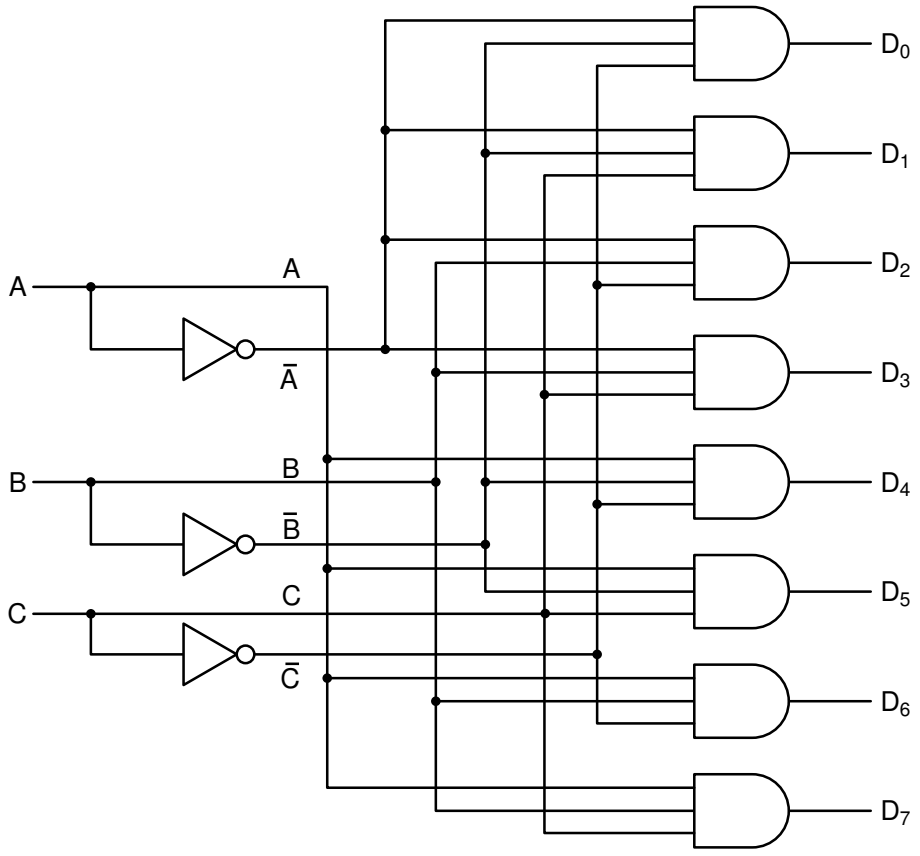

Figure 1: An 3-8 decoder circuit.

b) Consider the following truth-table – having 3-inputs (A,B,C) and 1-output (X):

A	B	C	X
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

i. Derive a Sum-of-Products expression for the output in the truth-table;
ii. Can you simplify this circuit?
iii. Draw a logic circuit of the simplified Sum-of-Products expression you have derived.

(*16 marks*)

c) Using De Morgan's Law demonstrate how NAND Gates can be used to implement a circuit which is equivalent to an OR Gate and how NOR Gates can be used to implement a circuit which is equivalent to an AND Gate. Give both the algebraic defintions and a circuit diagram for your solutions. (*8 marks*)

4. a) Explain any 3 of the following, making use of suitable examples:
i. ISA Level instructions can be approximately divided into a half dozen groups that are relatively similar from machine to machine, even though they may differ in the details. Give examples of some of the common categories.
ii. What is an *addressing mode*? Give examples of a number of modes and distinguish between their usages.
iii. The UltraSPARC Architecture is often described as a *load and store* architecture. What is meant by this phrase? What are the implications of such an architecture?
iv. Explain how a stack works. In particular describe how an *operand stack* works.

(*8 marks*)

b) i. Convert the following decimal numbers into IEEE 754 format single precision numbers. Give your answer in hexadecimal.
- 6.125
- -4.25

ii. Convert the following IEEE 754 format single precision numbers into decimal.
- 3F000000
- BFA00000

(*16 marks*)

c) The IEEE 754 Floating-Point Standard defines several numerical types: normalised and denormalised numbers, zero, infinity, and Not-a-Number. Explain how each of these numerical types is represented. In each case, justify its representation. (*8 marks*)

5. a) Explain what is meant by the term *Paging*. In particular, distinguish between *pages* and *page frames*. A diagram should be used to illustrate your explanation. (*8 marks*)

b) To understand how virtual I/O instructions are implemented, it is necessary to examine how files are organised and stored. A basic issue that must be dealth with by all file systems is allocation of storage. Explain what is meant by the term *allocation unit*. Give examples of strategies that can be used to allocate storage on a disk. Illustrate your examples with diagrams where appropriate.
(*8 marks*)

c) In the context of assembly languages:

i. Suggest a way to allow assembly language programmers to define synonyms for opcodes. How could this be implemented?

ii. Explain what is meant by the term *macro expansion*.

(*8 marks*)

d) In the context of the assembly process:

i. A symbol table can be regarded as an *associative memory*. What does this mean? There are many ways of implementing a symbol table. For example, amongst the alternatives are arrays and hash tables. What are the advantages of *hash coding* over methods such as *linear lookup* and *binary search*?

ii. Explain the processes *linking* and *loading*.

(*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	$\overline{AB} = \overline{A} + \overline{B}$	$\overline{A + B} = \overline{A}\,\overline{B}$