Australian National University

First Semester Examination 2012

Introduction to Computer Systems

(COMP2300/COMP6300)

Writing Period: 3 hour duration
Study Period: 15 minutes duration
Permitted Materials: One A4 page with notes on both sides. The rPeANUt specification. Note also the standard lab tools are available including: Java, eclipse, ooffice, kate, dia, gcc, man, rPeANUt ...
NO calculator permitted.

Please Read The Following Instructions Carefully.

This exam will be marked out of 100 and consists of 4 questions. Questions are of unequal value. The value of each question is shown in square brackets. Questions that are partitioned into parts show the number of marks given to each part within square brackets.

Students should attempt all questions. Answers must be saved into the question's directory (Q1, Q2, Q3, Q4) using the file(s) described in the question statement. Marks may be lost for giving information that is irrelevant.

Network traffic may be monitored for inappropriate communications between students, or attempts to gain access to the Internet.

The marking scheme will put a high value on clarity so, as a general guide, it is better to give fewer answers in a clear manner than to outline a greater number in a sketchy, half-answered fashion.

Question 1 [40 marks]

Highest marks are gained by providing: clear, concise, and short answers. Save your answers in the in the file 'Q1Answers.odt' in the directory Q1. This file can be edited using 'ooffice'. Please make certain that this file is saved both as you progress through the exam and before the exam ends.

i. [4 marks] How many different values can be stored in a 6 bit word? What is the range of numbers that is stored in a 6 bit word when two's complement representation is used? Convert the octal number 072 to hexadecimal. What decimal number would the octal number 072 represent if it was interpreted as a 6 bit two's complement number.
ii. [4 marks] The IEEE 32-bit single-precision floating-point standard is: 1 bit sign, 8 bits exponent with a bias of 127 (normalized numbers), and the remaining 23 bits are the significand (mantissa). What decimal number does the float 0xC1F20000 represent? In this IEEE 32-bit floating-point standard what is the smallest number representable that is greater than zero (give your answer either as an expression or just the decimal number using scientific notation)?
iii. [4 marks] How does a 1-bit register maintain state information about the value of the bit? Draw a circuit diagram of a simple flip-flop. (Hint: draw the diagram using 'dia' or 'gimp', save as an image, and insert into this document as a picture).
iv. [4 marks] How many bytes are there in a Megabyte (MB)? How does this value depend on the context in which Megabyte is used? How many bytes are there in a Mebibyte (MiB)?
v. [4 marks] What is a system call? Why are they important for modern operating systems? How are system calls implemented in Linux based x86 machines?
vi. [4 marks] In rPeANUt why is it impossible to use load immediate to set a register's value to greater than 32767 or less than -32768? If you wish to set a register to a value such as 40000 what are two different approaches you could use?

vii. [4 marks] Suppose you had a CPU with: 8 bit addresses, byte addressable memory, a 3-way set associative data cache with a total of 96 bytes, each cache line containing 8 bytes, a LRU replacement approach is used within each of the sets; also there is no pre-fetching. How many cache lines are there in the cache? How is the 8 bit address partitioned into tag, set, and word sections (give the number of bits in each section)? Assuming the cache is initially completely empty and the below sequence of addresses is read by the CPU. Place brackets around the reads that generate a cache miss for the sequence: 0x03, 0x25, 0x0F, 0xE3, 0x23, 0x0A, 0x28, 0x86, 0x03.

viii. [4 marks] What is an operating system? What is a multiprogramming operating system? How does a multiprogramming OS generally improve performance of the computing system?

ix. [4 marks] What is a TLB? What information is stored in each entry of a TLB?

x. [4 marks] UDP provides an unreliable way of sending datagrams from a process on one host to a process on another host without requiring a connection to be set up . In what way is UDP 'unreliable'? When is an 'unreliable' way of sending datagrams useful? What implications does this unreliability have for the applications using UDP?

Question 2 [20 marks]
(a) [5 marks] Write a program in rPeANUt that prints “Hello World!” to the terminal. Place your answer in a file called 'hello.s' in the Q2 directory. (Hint you do not need to use a loop.)

(b) [5 marks] Disassemble the program given in the image below. Place your answer in a file called 'disassemble.s' in the Q2 directory. Note your disassembled program should be able to be assembled by rPeANUt assembler without any errors. Also exactly what will the program output to the terminal when it is run (answer this question within a comment of the disassembled program)?

address	data
0x0100	0xc0020061
0x0101	0xc003007a
0x0102	0xc00400df
0x0103	0xc107fff1
0x0104	0xa4170103
0x0105	0xc101fff0
0x0106	0x21200000
0x0107	0xa420010b
0x0108	0x23100000
0x0109	0xa420010b
0x010a	0x64110000
0x010b	0xd110fff0
0x010c	0xa4000103

(c) [10 marks] The *C* code below calculates the greatest common divisor (gcd) of a series of pairs of numbers. Convert this *C* code into rPeANUt assembler code. Your solution must implement a procedure for 'gcd' using the recursive approach given below. Also, use the conventional rPeANUt stack frame approach for this procedure. Note within 'gcd.s' you are given some code that will print out the decimal numbers to the terminal. Place your answer in a file called 'gcd.s' in the Q2 directory.

```
#include <stdio.h>
int gcd(int m, int n) {
   if(m == n)
      return m;
   else if (m > n)
      return gcd(m-n, n);
   else
      return gcd(m, n-m);
}
void main() {
    printf("%d\n", gcd(6,15)); // prints 3
    printf("%d\n", gcd(6,4)); // prints 2
    printf("%d\n", gcd(30,84));  // prints 6
}
```

Question 3 [20 marks]

a) [10 marks] Write a program in *C* that converts binary numbers into hexadecimal numbers (only required to work on positive numbers). The binary numbers are provided to the program via standard input and are at most 16 bits long. Each binary line is separated via a return character. Place your answer in the file called 'bin2hex.c' in the Q3 directory. An example of the program running is given below (input typed by the user is given in bold):

$ ./bin2hex
1111111111111111
0xFFFF
0000000000000000
0x0
0
0x0
10
0x2
1010
0xA
11110000
0xF0
101001011111
0xA5F
01010
0xA
111
0x7

(b) [10 marks] Write a program in *C* that determines if two files are 'similar'. Two files are defined to be 'similar' if and only if:

- the files are exactly the same length, and
- all but at most 10 bytes in the files have the same value in the same position.

The program is provided the names of the two files as arguments. The program gives the result to standard out saying either "similar" or "different". Place your answer in the file called 'similar.c' in the Q3 directory.

Question 4 [20 marks]

You have been given the task of implementing a program that takes data from a pen device (such as the one used in lectures) and displays what is drawn. This involves drawing lines between the points at which the pen has contact with the device panel. You may assume code has already been written that collects these pen events and this program can provide the data via a pipe to your program. Your program needs to be able to read this collected pen event information via standard input and display it to the screen. The stream of data contains: x position information, y position information, and information about when the pen is in contact with the panel.

The good news: example code for creating a screen and drawing lines to the screen has been given in lines.c.

The catch: you do not know the format of the stream of data that contains the pen information! However, you have been given four files that contain dumps of this data ("singlehorizontal.dump", "singlevertical.dump", "shapes.dump", "resolution.dump").

The program you implement should be called 'display' and placed in the 'display.c' file in the Q4 directory. You should be able to test your solution by redirecting the dump files into your program. e.g.
$ display < shapes.dump
or they should also work when concatenated together
$ cat singlehorizontal.dump singlevertical.dump | display

(a) [5 marks] What approaches/tools could you use to work out the format of the stream of data? What is the format of this data? (place your answers to this question in a comment at the top of the 'display.c' file in the Q4 directory)
(b) [15 marks] Implement your solution in the 'display.c' file in the Q4 directory.