UNIVERSITY OF DUBLIN
TRINITY COLLEGE

Faculty of Engineering, Mathematics and Science

School of Computer Science and Statistics

BA (Mod) Business and Computing
BA (Mod) Computer Science, Linguistics and a Language
Junior Freshman Examination

Trinity Term 2011

CS1021 – Introduction to Computing I

Wednesday 18th May 2011 | **Drawing Office, Museum Building** | **14:00 – 16:00**

Dr Jonathan Dukes

You **MUST** Answer
Question 1 from **Section A** and
TWO out of **THREE Questions** from **Section B**

Non-programmable calculators are permitted for this examination

Please indicate the make and model of your calculator
on each answer book used

To be accompanied by an **ARM Instruction Set and Addressing Mode Summary** booklet

Where you are asked to write an assembly language program, you must provide an adequate explanation of your program, for example, in the form of pseudo-code comments.

SECTION A (20 Marks)

You MUST answer Question 1 from this section

Suggested time allocation: 25-30 minutes

1. (a) Write an ARM Assembly Language program to evaluate the following function:

$$f(x) = ax^2 + bx + c$$

where a, b and c are values stored in **R1**, **R2** and **R3** respectively and x is a word-size value stored in memory at address **0xA1001000**. Store the result in **R0**.

(4 marks)

(b) Provide an ARM Assembly Language instruction, or a sequence of instructions, to multiply a value in **R1** by the following constants, **without using the MUL instruction**. Store the result in each case in **R0**.

(i) 16 (1 mark)

(ii) 12 (1 mark)

(c) Consider the following ARM Assembly Language listing, which also shows the address at which each assembled instruction will be stored in memory. Calculate the branch target offset that would be encoded in each of the two highlighted branch instructions and, in each case, show how you computed your result.

```
    address  label   instruction
1 00000000 start   MOV     R0, #0
2 00000004         MOV     R1, #1
3 00000008 wh      CMP     R1, #15
4 0000000C         BHI     endwh
5 00000010         ADD     R0, R0, R1
6 00000014         ADD     R1, R1, #1
7 00000018         B       wh
8          endwh
9 0000001C stop    B       stop
```

(3 marks)

(d) Convert the following pseudo-code program into ARM Assembly Language. You must provide suitable comments to explain your answers. Assume that **i**, **j**, **k** and **N** are unsigned values stored in **R1**, **R2**, **R3** and **R4** respectively.

(**Note:** The syntax **Memory.Word[address]=0** is intended to represent storing the word-size value 0x0 in memory at the specified **address**.)

```
if (i * i <= N)
{
    j = 2;
    while (i * j <= N)
    {
        address = k + (i * j * 4);
        Memory.Word[address] = 0;
        j = j + 1;
    }
}
```

(7 marks)

(e) Provide pairs of values which, if stored in registers **R1** and **R2**, would cause the following condition code flag states after the execution of the instruction **ADDS R0, R1, R2**. Explain your answer in each case.

(i) N=0; Z=1; C=1; V=0 (2 marks)

(ii) N=0; Z=0; C=1; V=1 (2 marks)

SECTION B (80 marks)

You MUST answer TWO out of THREE questions from this Section
All questions are worth 40 marks

Suggested time allocation: 40-45 minutes per question

2. Design and write an ARM Assembly Language program to count the number of pairs of identical adjacent words in a text string. Your program should store the result in **R0**. Assume that **R1** contains the address of the first character in the string.

You may assume that strings are composed of byte-size ASCII characters and are NULL (zero) terminated. You may also assume that the words in the string contain only lower-case alphabetic characters (i.e. 'a' ... 'z') and that adjacent words are separated by a single space character. For example, given the following string:

"one two two three three three"

your program should store the value 3 in **R0**, since "two two", the first "three three" and the second "three three" are pairs of adjacent identical words.

Your answer must include:

(i) An explanation of your approach to solving the problem

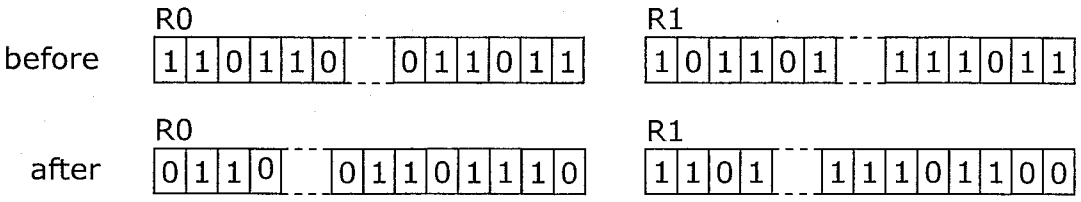
(10 marks)

(ii) An ARM Assembly Language listing for your program. **You must provide adequate comments to explain your program.**

(30 marks)

3. (a) Design and write an ARM Assembly Language program that will perform a logical shift operation on a 64-bit value stored in registers **R0** and **R1**. Assume that the most-significant 32-bits are stored in **R0** and the least-significant 32-bits are stored in **R1**. The count of the number of bits to shift is stored in **R2**. If the count in **R2** is negative, your program should shift left. Conversely, if the count is positive, your program should shift right.

The following example shows how your program should logically shift left the 64-bit value stored in **R0** and **R1** when **R2** contains **-2** (0xFFFFFFFE using the 2's Complement representation of negative values.)



Your answer must include:

(i) An explanation of your approach to solving the problem

(5 marks)

(ii) An ARM Assembly Language listing for your program. **You must provide adequate comments to explain your program.**

(15 marks)

(b) Design and write an ARM Assembly Language program that will round a non-negative integer *i* up to the nearest multiple of another integer *j*. Assume that *i* is stored in **R1** and *j* is stored in **R2**. Your program should store the result in **R0**.

For example, if *i* is 11 and *j* is 5, your program should store 15 in **R0**.

Your answer must include:

(i) An explanation of your approach to solving the problem

(5 marks)

(ii) An ARM Assembly Language listing for your program. **You must provide adequate comments to explain your program.**

(15 marks)

4. (a) Design and write an ARM Assembly Language program that will read a sequence of ASCII characters typed by a user in a "console" window and convert the sequence of characters to the value they represent. The only valid characters that may be entered are those representing the decimal digits '0' ... '9'. The result should be stored in **R0**. For example, if a user enters the characters '2', '4', '7', '5' and '3', your program should store the value 24753 in **R0**. The sequence of characters ends when the user presses the **RETURN** key, which you may assume corresponds to ASCII character code **0x0D**.

A single ASCII character can be read from a "console" window by executing the **BL getkey** instruction, which will store the ASCII character code in **R0**. Since the key pressed is not automatically also displayed in the window, your program should also execute the **BL sendchar** instruction to display each character as it is read. The character to be displayed should be stored in **R0**.

Your answer must include:

(i) An explanation of your approach to solving the problem

(5 marks)

(ii) An ARM Assembly Language listing for your program. **You must provide adequate comments to explain your program.**

(15 marks)

(b) Design and write an ARM Assembly Language program that will output a sequence of ASCII characters to a "console" window representing, in hexadecimal form, the value stored in register **R0**.

Your program should output the value one ASCII character at a time using the **BL sendchar** instruction as described in part (a) above.

Your answer must include:

(i) An explanation of your approach to solving the problem

(5 marks)

(ii) An ARM Assembly Language listing for your program. **You must provide adequate comments to explain your program.**

(15 marks)