Coláiste na Tríonóide, Baile Átha Cliath
Trinity College Dublin
Ollscoil Átha Cliath | The University of Dublin

Faculty of Engineering, Mathematics and Science

School of Computer Science & Statistics

Integrated Computer Science
Computer Science and Business
Computer Science and Language
Junior Fresh

SAMPLE PAPER

CSU11021 – Introduction to Computing I

Dr Jonathan Dukes

Instructions to Candidates

Attempt all questions.

Materials permitted for this examination

Non-programmable calculators are permitted for this examination. You must indicate the make and model of your calculator on the front of your first answer book.

1. Consider the following sequence of ARM Assembly Language instructions. For each highlighted instruction, **give the final value in the destination register** and **state whether each of the N (negative), Z (zero), C (carry) and V (overflow) flags is set or clear** after the execution of the instruction. **NB: Answers without a brief supporting explanation or calculation will receive zero marks.**

```
1        LDR     R0, =0xFFFFFFFE
2        LDR     R1, =0x00000004
3        SUBS    R5, R3, R4              ; flags? result?
4        LDR     R6, =0x80000000
5        LDR     R7, =0x80000000
6        ADDS    R8, R6, R7              ; flags? result?
7        LDR     R9, =0xE0000000
8        LDR     R10, =0xC0000000
9        ADDS    R11, R9, R10            ; flags? result?
```

[10 marks]

2. Provide sequences of ARM Assembly Language instructions to perform each of the following bit-manipulation operations.

   (a) Clear the most significant byte of the 32-bit word in R0.

   (b) Set bits 0, 8, 16 and 24 of the 32-bit word in R1.

   (c) Invert the most significant bit of the 32-bit word in R2.

   (d) Swap the first and last bytes of the 32-bit word in R3.

[10 marks]

3. Translate the pseudo-code below into ARM Assembly Language. The syntax `x<<y` represents the contents of `x` logically shifted left by the number of bits specified by `y`. Similarly, `>>` represents logical shift right. Assume a, b, c and r are stored in R0, R1, R2 and R3 respectively.

```
r=0;
for ( c=0; b >> c != 0; c++ ) {
  if ( (b >> c) & 1 !=0 ) {
    r = r + (a << c);
  }
}
```

[10 marks]

4. Write an ARM Assembly Language program that will determine the length of the longest contiguous sequence of 1s in the 32-bit value stored in R1. For example, given the binary value shown below in R1, your program should store the result 6 in R0.

10000111 11100100 10110100 01000011

[10 marks]

5. Write an ARM Assembly Language program that will count the number of **odd** values in a sequence of word-size integer values stored in memory. Assume that the sequence of values begins in memory at the address contained in R1. The length of the sequence is stored in R2. Store your result in R0. [10 marks]

6. Assume a sequence of N word-size values is stored in memory. Design and write an ARM Assembly Language program to determine the *mode* of the values. (The *mode* is the value that appears most often.) For example, given the values $[5, 3, 7, 5, 3, 5, 1, 9]$, the *mode* would be 5.

   Assume the value of N is stored in R1 and the start address of the sequence of values is stored in R2. Store the result (the *mode*) in R0. You may assume there is only one mode.

   Your answer must include:

   (a) an explanation of your approach (e.g. using pseudo-code, diagrams and/or examples) and [5 marks]

   (b) your ARM Assembly Language program with comments. [15 marks]

7. Write an ARM Assembly Language program to convert a NULL-terminated ASCII string to lower case. Assume that the original string may contain either upper- or lowercase alphabetic characters, as well as other characters. The string is stored in memory at the address contained in R0. [10 marks]

8. Design and write an ARM Assembly Language program that will convert a NULL-terminated string to proper case, overwriting the original string in memory. (i.e. Capitalise The First Letter Of Every Word.) Assume the string contains only alphanumeric characters (a-z, A-Z and 0-9) and spaces. Don't forget that the second and subsequent letters of each word may need to be converted to lower case!

   Your answer must include:

   (i) an explanation of your approach (e.g. using pseudo-code, diagrams and/or examples) and [5 marks]

   (ii) your ARM Assembly Language program with comments. [15 marks]