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 **Trinity Term 2016**
Year 1 Annual Examinations

CS1022 – Introduction to Computing II

14 MAY 2016 **SPORTS CENTRE** **14:00 – 16:00**

Ms Rebekah Clarke

Instructions to Candidates:
Answer any **TWO out of THREE questions**.
All questions are marked out of 25.
Where you are asked to write an assembly language program you must provide suitable comments to explain your program, for example, in the form of pseudo- code comments.

You may not start this examination until you are instructed to do so by the Invigilator.

Materials permitted for this examination:
An **ARM Instruction Set and Addressing Mode Summary** booklet is available on request.
Non-programmable calculators are permitted for this examination — please indicate the make and model of your calculator on each answer book used.

1. (a) Describe in detail the operation performed by each of the instructions below. For each instruction, state the memory address that is accessed, the new value contained in any register that is modified, and the addressing mode used (if any). Provide a diagram to illustrate the effect of any instruction that modifies the system stack.

   (i) `LDR R0, [R1], #4`

   (ii) `STR R0, [R1, R2]!`

   (iii) `LDR R0, [R1, R2, LSL #4]`

   (iv) `STMFD SP!, {R4-R6, R1, R9, LR}`

   Assume the following initial values in R1, R2 and SP (R13):

   `R1=0xA1000800, R2=0x00000020, SP=0xA1000700`

   [6 marks]

   (b) Provide ARM Assembly Language instructions to perform the following stack operations without the use of LDM or STM instructions. Use SP (R13) as the stack pointer.

   (i) Push a word-sized value in R2 onto a Empty Ascending Stack.

   (ii) Pop off and **discard** the top word on an Empty Descending stack. No register should be modified other than the stack pointer.

   [2 marks]

   (c) Explain how the Branch and Link (BL) instruction allows you to invoke and subsequently return from a subroutine. Why do subroutines usually save the contents of the Link Register on the system stack at the start of the subroutine and restore it at the end?

   [3 marks]

(d) Design and write an ARM Assembly Language **subroutine** that will reverse the order of the elements in an array. The reversed array must overwrite the original array in memory. Document a suitable interface for your subroutine. (Higher marks will be awarded if you can avoid using a stack or other memory to store a temporary copy of the array.)

[8 marks]

(e) Design and write an ARM Assembly Language subroutine that will implement the Pseduo-code below for a recursive factorial function.

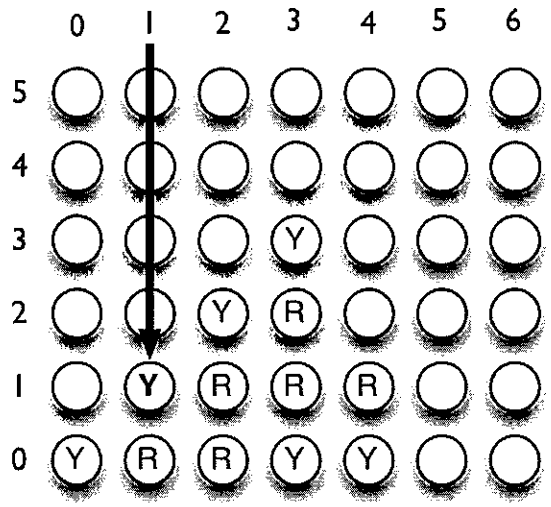
```
int fact(int n)
{
  if(n==0)
    return 1;
  else
    result = fact(n-1) * n;

  return result;
}
```

Your answer must include:

(i) a description of an appropriate interface for your subroutine [1 mark]

(ii) an ARM assembly Language listing for your subroutine [5 marks]

2. Connect 4 is a two-player puzzle game in which players take turns to drop discs into a vertically mounted board with seven columns and six rows. One of the players inserts yellow discs and the other inserts red discs, attempting to create a horizontal, vertical or diagonal run of four of their colour discs. In the diagram below, yellow discs are represented by 'Y' and red discs are represented by 'R'. Here the yellow player wins after dropping a disc into column 1 to connect a diagonal run of 4 yellow discs.



Assume that the board is represented in memory by a two-dimensional array of **bytes** with 6 rows and 7 columns. The value 1 in an element of the array indicates the presence of a yellow disc, the value 2 indicates a red disc and 0 indicates an empty slot.

(a) Design and write an ARM Assembly Language subroutine that, if given a column number and a disc colour, will update the state of the game to reflect that colour disc being dropped into the specified column.

Your answer must include:

(i) a detailed explanation of your approach to solving the problem, using pseudo-code to support your explanation

[2 marks]

(ii) a description of the interface to your subroutine

[1 mark]

(iii) an ARM Assembly Language listing for your subroutine with adequate comments

[7 marks]

(b) Design and write an ARM Assembly Language subroutine that will determine if the current move is a winning move i.e. connects 4 discs of the same colour. Assume that the column number and disc colour are passed as parameters along with the row number the disc landed in. The subroutine should return 1 if the move is a winning move and zero otherwise.

Note: You need only provide a program that checks one direction for a winning move along with an explanation or pseudo-code explaining how you would check the remaining directions.

Your answer must include:

(i) a detailed explanation of your approach to solving the problem, using pseudo-code to support your explanation

[4 marks]

(ii) a description of the interface to your subroutine

[1 mark]

(iii) an ARM Assembly Language listing for your subroutine with adequate comments

[10 marks]

3. (a) Show how you would represent each of the following two floating point values using the IEEE754 single-precision format

   (i) 8.25_{10}

   (ii) 21.625_{10}

   [4 marks]

   (b) The ARM7TDMI-S microcontroller raises an Undefined Instruction exception if an attempt is made to decode an undefined instruction. Show how this mechanism can be used to extend the instruction set with a DIV instruction that divides one integer (the dividend) by another (the divisor).

   Use the following undefined instruction template, where Rd is the destination register, Rm is the dividend and Rn is the divisor.

DIV instruction template

cond	0111	1111	0100	Rn	Rd	1111	Rm

Your answer must include:

(i) a new exception handler for Undefined Instruction exceptions that tests for your new DIV instruction and, when it is detected, decodes the instruction and implements the division operation

[13 marks]

(ii) a program to install the new exception handler in the exception vector table and test the new DIV instruction

[4 marks]

(iii) a discussion of the steps that take place when the undefined DIV instruction is fetched and the Undefined Instruction exception is raised

[4 marks]