COMP2300/6300 Final Exam 2017

Student ID:

u							

Time: 15 minutes reading, 180 minutes writing

Total marks: 100

Weighting: 50%

Permitted materials: 1 double-sided A4 cheat sheet

Make sure you read each question carefully. Questions are not equally weighted, and the size of the answer box is not necessarily related to the length of the expected answer or the number of marks given for the question.

All answers must be written in the boxes provided in this booklet. You will be provided with scrap paper for working, but only the answers written in this booklet will be marked. Do not remove this booklet from the examination room. There is additional space at the end of the booklet in case the boxes provided are insufficient. If you use these extra pages, make sure you clearly label which question the answer refers to.

Greater marks will be awarded for short, concrete answers than long, vague/rambling ones. Marks may be deducted for providing information that is irrelevant to a question.

For examiner use

Question 1 10 marks

You inherit a broken discoboard which can only execute the **immediate version** of the **`add`** instruction:

```
add{s}<c><q> <Rd>, <Rn>, #<const>
```

Assuming all of the registers still work, can you come up with equivalent ways of executing the following instructions using only the **`add`** instruction above?

Part 1 (2 marks) **`nop`**

Part 2 (2 marks) **`mov r2, r3`**

Part 3 6 marks

In an ARM assembly program there are many different ways to change the control flow of your program—i.e. to jump to a location *other than* the instruction directly following the one being currently executed. List as many of these ways as you can.

Question 2 15 marks

Part 1 10 marks

What are the main components of a CPU, and what are their roles?

Part 2 5 marks

Suppose there are two different programming languages: **LangA** (a functional-style language) and **LangB** (an imperative-style language). If you only see the ARM assembly code output generated by the compiler, can you tell whether the original program was written in LangA or LangB? If so, how? If not, why not?

Question 3 15 marks

Part 1 5 marks

Many programming languages have a **do-while** loop, which executes the loop body repeatedly until a certain condition is false (the conditional check occurs *after* the body is executed). Here's an example in C:

```
do{
  a = a + b;
  b = 2 * b;
}while(b < a)
```

Write a `dowhile` ARM assembler macro which performs a **do-while** loop (not for the specific C example above, but for *any* body & condition).

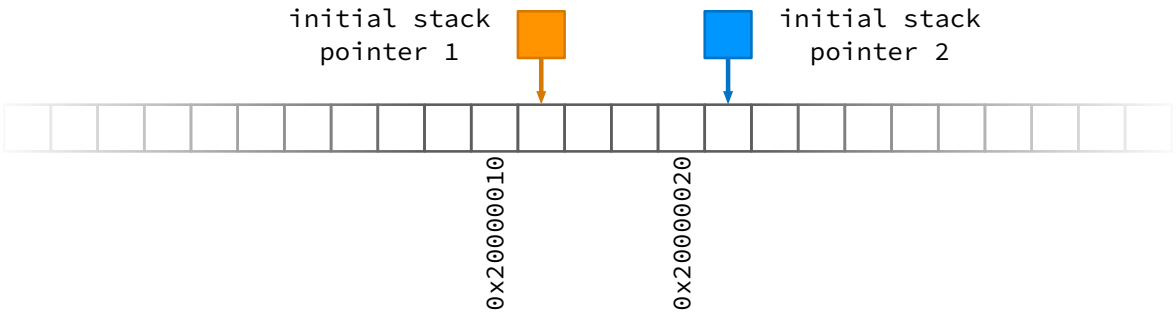
Part 2 10 marks

```
1  func:
2          push {lr}
3          push {r0}
4          push {r1}
5          cmp     r0, r1
6          bge     L4
7          bl      inner
8          mov     r2, r0
9          ldr     r3, [sp, #4]
10         mul     r3, r2, r3
11         b       L5
12 L4:
13         ldr     r3, [sp]
14 L5:
15         mov     r0, r3
16         add     sp, #8
17         pop {lr}
18         bx      lr
```

Write a short program in a programming language of your choice which might have compiled down to the above ARM assembly code (you may assume that the ARM code uses the standard ARM calling convention). Don't worry if you can't remember the exact syntax for your programming language, it's what it *does* that's important.

Question 4 15 marks

A discoboard is running a lightweight multi-tasking OS which supports two concurrent processes, each with its own stack: the base stack address is `0x20000010` for stack 1 and `0x20000020` for stack 2. This memory layout is shown in the figure below (each box represents a 32-bit word).




```
square:
  mul r0, r0, r0
  bx lr

square_plus_c:
  push {lr}
  push {r0}
  push {r1}

  bl square

  ldr r1, [sp]
  add r0, r1

  add sp, #8
  pop {lr}
  bx lr
```

Part 1 10 marks

If process 2 runs just the `square_plus_c` function as listed, will this affect the code running in process 1? Be as specific as you can—provide short code snippets or diagrams where necessary.

Part 2 5 marks

In general, how could you modify the OS so that different process stacks don't interfere with each other? You may give multiple answers—be as specific as you can, including diagrams and/or assembly code snippets where necessary.

Question 5 20 marks

Part 1 10 marks

Your discoboard is going to be connected up (by some jumper cables) to a different device with a different architecture (i.e. not ARMv7).

One thing you know about the other device is that it uses the **opposite** endianness to your discoboard. Write an ARM assembly program for your discoboard which takes a 32-bit value in `r0` and rearranges the bits to represent *the same 32-bit value* using the opposite endianness—store the result in `r1`.

Part 2 10 marks

The other device runs at 100Hz—*much* slower than your discoboard. You've been given the job of writing a program to process data coming from this external device.

Describe *at least* two different approaches you could use to do this in an ARM assembly program, and discuss the pros & cons of each approach. You may use diagrams or short assembly code snippets if necessary.

Question 6 15 marks

A large bank is re-designing their banking software to run on a discoboard using a multi-tasking OS. They've employed you to advise them on security issues in building this discoboard-powered banking system. The `deposit` function looks like this:

```
1  deposit:
2    push {lr}
3
4    push {r0} @ store deposit amount on the stack
5
6    @ get the memory address where the current account balance is
7    @ stored - the address is returned in r0
8    bl get_balance_address
9
10   @ read current balance
11   ldr r1, [r0]
12
13   @ get deposit amount (original argument, now on stack)
14   ldr r2, [sp]
15
16   @ add deposit amount to balance
17   add r1, r2
18
19   @ store updated balance
20   str r1, [r0]
21
22   add sp, #4
23   pop {lr}
24   bx lr
```

Part 1 10 marks

Is this code safe to run in a multi-tasking OS (i.e. with multiple processes which can each call the `deposit` function)? If not, what can go wrong? Be as specific as you can—provide short code snippets or diagrams where necessary.

Part 2 5 marks

How could you modify the `deposit` function so that it was safe to use in a multi-tasking OS? Be as specific as you can—provide short code snippets or diagrams where necessary.

Question 7 10 marks

What are the differences between an architecture which uses hyper-threading vs an architecture which uses a vector processing unit? What use cases is each one best suited to? Explain your answer.