McGill University
Department of Electrical and Computer Engineering

ECSE 324 Mid-Term Exam Fall 2022

Name: _____

Mcgill ID: _____

Closed Book Exam. No calculator allowed. Answer all questions directly on the exam paper. You can do rough work on the blank pages. Rough work will not be used for grading.

Part 1 : 10pt

Part 2 : 7pt

Part 3 : 10pt

(Bonus : 1pt)

Total : 27 pt

Part A: General knowledge **[10 points]**

A.1: multiple choice questions [1 point] each.

Only select a single answer. If more than one answer is selected, you will get zero for that question.

1. In a Von Neumann architecture:

a. Instructions are 32 bits wide
b. Instructions and data are stored in the same memory
c. The transistor count doubles every two years
d. Only Load/Store instructions can access memory
e. The register files contains 16 registers

2. What is -2 in 8-bits two's complement in hexadecimal?

a. 01
b. 09
c. FF
d. FE
e. None of these

3. Which ARM instruction modifies the Link Register?

a. BX LR
b. BL foo
c. ADD V1, LR, V2
d. None of them
e. All of them

4. Which ARM instruction leaves the CPSR unmodified?

a. BLE END
b. CMP SP, R2
c. SUBS R1, R2, R3
d. TST R6, R6
e. All of them

5. What applies *specifically* to the callee-save convention?

a. The caller must save all registers it has used before calling a subroutine
b. The callee must return to the caller at the end of the subroutine
c. The caller can read the stack to read the return value from the callee
d. The callee uses the stack to read the input arguments
e. None of these

6. Which registers are modified by this ARM instruction: “LDR R1, [R2], R3” ?

a. R1, R2, R3
b. R2
c. R1, R3
d. R1, R2
e. R1

7. When are shadow registers used on ARM?

a. When calling a subroutine
b. When executing an ISR
c. When polling an I/O device
d. When the status register of an I/O device is read
e. None of these

8. Fill up the answer card on the next page based on your answers for questions 1-7 for an additional bonus point. We suggest that you do it towards the end of the exam once you are sure about your answers.

A.2: Short answer questions [1 point] each.

9. The target of a branch instruction are encoded as an offset from the PC in multiple of 4 bytes on ARM as seen in the course. Why is it a multiple of 4?

Instructions are always aligned at a 4 bytes boundary because they are 4 bytes wide. Hence the only address we can branch to are multiple of 4 bytes.

10. Why does the assembler usually requires two passes to produce an object file?

There may be forward references: e.g. branches that target labels not yet encountered.

11. What does the interrupt vector table contain?

The address of the interrupt service routine corresponding to each interrupt/exception.

Part B: Instruction Set **[7 points]**

Assume a 12-bits RISC CPU with a CPSR. Both registers and instructions are 12 bits wide. The memory is byte-addressable and the address size is 12 bits.

The only registers available for programmers are R0, R1, R2, R3, R4, R5, R6 and PC.

The only instructions available are:

- `MOV  Rd, #imm        // Rd <- #imm  (immediate signed value)`
- `ADD  Rd, Rs1, Rs2    // Rd <- Rs1 + Rs2`
- `LD   Rd, Rs1, Rs2    // Rd <- MEM[Rs1+Rs2]`
- `ST   Rd, Rs1, Rs2    // MEM[Rs1+Rs2] <- Rd`

An instruction can be conditionally executed (when Z==1) based on a 1-bit condition field in each instruction, indicated by the EQ suffix to an instructions (we assume we only have one condition and that every instruction modifies the CPSR). E.g. `ADDEQ R0,R1,R2` will only execute if the preceding instruction produces a zero into the destination register.

1. What is the addressable memory capacity for this CPU in KB? **[1 point]**

2^12B = 4096B = 4KB
1pt for 4KB, 0.5pt for 4096B or 2^12B, 0pt otherwise

2. Show how each of these instructions could be encoded with 12bits using the minimal number of bits for the opcode and register encoding. Make sure to make it clear how many bits, and which bits are used for the opcode, condition, register operands and immediate value for each instruction. **[3 points]**

For ADD, LD, ST:
1 bit cond + 2 bits opcode + 3 bits destReg + (3 bits srcReg1+ 3 bits srcReg2)
For MOV:
1 bit cond + 2 bits opcode + 3 bits destReg + 6 bits immediate value

0.5 pt for cond bit
0.5 pt for opcode bits
1 pt for dst+src regs
1 pt for imm

3. Using only instructions from the set above, write a sequence of instructions that emulate a branch two instructions back (two instructions above the first instruction in the sequence). We assume that the PC points 6 bytes ahead of the currently executing instruction in this machine. **[2 points]**

Question was probably too hard hard given that the instruction size is 1.5 byte.

MOV R0, #0 // any instruction that only modifies R0 will do here
MOV R0, #-12
ADD PC, PC, R0

1 pt for general idea (MOV + ADD to PC)
1 pt for getting the offset right or identifying issue of targeting instruction that are in the middle of a byte

4. Assuming this system is BIG ENDIAN and the word size is 3-byte. Given the content of the memory below, what is the hexadecimal value stored in R0 after these two instructions execute? **[1 point]**

```
MOV    R0, #0
LD     R0, R0, R0
```

Address	Value
0x000	0xAB
0x001	0xFF
0x002	0x1E
0x003	0xBE
0x004	0x14
0x005	0xAD
0x006	0X42
0x007	0x7F
0x008	0x2B

0xABFF1E or 0XABF   1pt   given conflicting information (registers hold 12 bits yet question mentions 3-byte word size)

Part C: I/O Device & Assembly **[10 points]**

Suppose that you have a temperature sensor which continuously measure temperature to control a heating device.

Sensor device: The sensor device has two memory-mapped I/O register of 32 bits each: a data register, mapped at address 0x00007000, which contains the temperature in degrees; and a status register mapped at address 0x00007004, where bit **N** indicates that a new temperature reading is available as shown below. A read of the status register will clear the **N** bit.

Data register (0x00070000):

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
Temperature (signed integer)																															

Status register (0x00070004):

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
																								N							

Heating device: The heating device has a single memory mapped data register at address 0x00008000. If a non zero value is written into this register, the heating will start and remain on. If a zero value is written into the register, the heating will turn off and remain off.

Complete the following assembly program on the next page by filling in the blanks (underlined). The effect of the program should be that the heating is turned on when the temperature is below 20 degrees and is turned off when the temperature is equal or higher to 20 degrees.

```
sensor:         .word       0x00007000      1pt (or 0x00070000 1pt)
heating:        .word       0x00008000      1pt

_start:

off:            BL          readSensor
                CMP         A1, #20         1pt
                BGE         off
                BL          startHeating
on:             BL          readSensor
                CMP         A1, #20
                BLT         on              1pt
                BL          stopHeating
                B           off



readSensor:
                PUSH        {V1,V2}         1pt
                LDR         V1, sensor
readLoop:
                LDR         V2, [V1,#4]
                TST         V2, #128        1pt
                BEQ         readLoop        1pt
                LDR         A1, [V1]   or   A1, [V1,#0]   1pt
                POP         {V1,V2}
                BX          LR



startHeating:   MOV         A1, #1
                B           setHeating

stopHeating:    MOV         A1, #0
                B           setHeating

setHeating:
                PUSH        {V1}
                LDR         V1, heating
                STR         A1, [V1]   or  STR A1,[V1,#0]    1pt
                POP         {V1}            1pt
                BX          LR
```