CG2028 Sample Quiz – Part 1

Consists of multiple parts – MCQs &/or True-False &/or short-answer questions, and a long question.

Question #: 1 (Just for your reference. Not emphasized this semester, you may ignore this question)
Which one of the following statements about the Load & Store architecture is TRUE?
(2 marks)

A. Highly pipelined and multiple instructions take one cycle to execute.
√B. Small instruction set with each instruction typically takes one cycle to execute.
C. The instructions have fixed length and all of them have direct access to the memory operands.
D. The operands of the logic instructions can either be in the registers or memory.

Question #: 2
Which one of the following ARMv7E-M instructions will NOT cause the assembler to issue a syntax error message?
(2 marks)

A. LDR R0, [R0, #4!]
B. LDR R0, [NUM1]
C. STR R0, R0, #4!
√D. STR R0, [R0, #4]

Question #: 3
Which one of the following ARMv7E-M instructions will cause the assembler to issue a syntax error message?
(2 marks)

A. ANDS R1, R2, R3
B. POP {R3, R2, R1}
C. RORS R1, R2, #3
√D. CMNS R1, R2

Question #: 4
In as few lines as possible, write an ARMv7E-M IF-THEN (IT) instruction block to best represent the following pseudocode for unsigned numbers conditional execution:

```
                CMP   R0, R1
                BCC   IF_LOWER
                SUB   R2, R1, R0          @ Some instructions for R0 >= R1
                B     SKIP_LOWER
IF_LOWER:       ADD   R2, R1, R0          @ Some instructions for R0 <R1
SKIP_LOWER:     MOV   R3, R2              @ Continue onto the rest of the program
```

(4 marks)

Solution:

```
CMP R0, R1
ITE  CC
ADDCC R2, R1, R0          @ Some instructions for R0 < R1
SUBCS  R2, R1, R0         @ Some instructions for R0 >= R1
MOV R3, R2                @ Continue onto the rest of the program
```

Question #: 5 One of the key embedded subsystems in electric vehicles (EVs) is the one that controls the power electronics devices and manages the batteries' operation, safety and efficiency. It also estimates the internal states of the batteries, i.e., state of charge (SOC) and state of health (SOH). The SOC, similar to a fuel gauge, is the main indicator on which a driver can rely to determine whether the energy in the EV battery is sufficient to travel the desired distance without recharging. The SOH is the ratio of the maximum battery charge of a used battery to its rated capacity, indicating the aging of the battery cells.

As one of the engineers in charge of developing this subsystem in an advanced EVs production plant, you have been tasked to program the function of the embedded system that dynamically assesses the EV's ability to complete a journey. The program must:

- Read in the *estimated distance to travel* (calculated based on the driver's desired destination), which is stored in the memory location labelled **DestDist**.
- Compare it to the *estimated maximum distance* (derived from the most recent reading of the continually-updated SOC), stored in the memory location labelled **SOCDist**.
- Retrieve the *four most recent SOH readings* which are continually updated and stored sequentially in the memory, starting from the memory location labelled **SOH**.
- Determine the mean of the four SOH readings.
- Compare the mean to a known *healthy threshold*, whose value is stored in the memory location labelled **HT**.
- Raise an alarm in the *driver's console* by writing 0xBAD to a peripheral register whose address is labelled **CONSOLE**, whenever: SOCDist is less than DestDist, *and* the mean of the four most recent SOH readings is less than the healthy threshold, HT. However, if *only one* of the above two conditions is true, a predefined error message will be sent to the driver's console instead, by writing 0xC0DE1 to CONSOLE.

The alarm or error message will be reset automatically when the driver stops the car to change to a nearer destination, to recharge the EV, or to replace the unhealthy batteries if it is an SOH issue. Your program should run continuously throughout the entire journey to ensure proper monitoring of the batteries' internal states and the safety of the EV.

Assume all values used are unsigned integers. In addition, assume that all necessary peripheral hardware configurations/settings/data declarations have been successfully completed for you.
Write an ARMv7E-M assembly language program to implement the functionalities of the embedded system described above. Comment all your instructions clearly and adequately.

Hint: multiple IT blocks may be needed for conditional execution.
Marks breakdown:

- 3 marks: Initialization and forever loop
- 2 marks: Find mean
- 6 marks: Events confirmation & alarm/error
- 2 marks: Syntax
- 2 marks: Adequate comments

(15 marks)

Solution:
Not provided, but you may email your solution to eletanh@nus.edu.sg for consultation.