

Name: Nguyen Quoc Thang

ID : 104193360

LAB 8 : SUBMISSION

Excercise 8.1.1

Program

```
1 MOV R0, #15
2 STR R0, .WriteSignedNum
3 MOV R1, #str
4 STR R1, .WriteString
5 HALT
str: .ASCIZ "remaining\n"
```

Processor

PC: 0x00000018
LR: 0x00000000
SP: 0x00100000
R12: 0x00000000
R11: 0x00000000
R10: 0x00000000
R9: 0x00000000
R8: 0x00000000
R7: 0x00000000
R6: 0x00000000
R5: 0x00000000
R4: 0x00000000
R3: 0x00000000
R2: 0x00000000
R1: 0x00000014
R0: 0x0000000f

Count: 6
Current Instruction: 616d6572
Status bits: NZCV 0000

Input/Output

15 remaining
Bad instruction at line 6

Memory

000	0x0	0x4	0x8	0xc
0x0000	0xe3a0000f	0xe50f00fc	0xe3a01014	0xe50f10f4
0x0001	0xe1000070	0xe6e696e9	0x00000000	0x00000000
0x0002	0x00000000	0x00000000	0x00000000	0x00000000
0x0003	0x00000000	0x00000000	0x00000000	0x00000000
0x0004	0x00000000	0x00000000	0x00000000	0x00000000
0x0005	0x00000000	0x00000000	0x00000000	0x00000000
0x0006	0x00000000	0x00000000	0x00000000	0x00000000
0x0007	0x00000000	0x00000000	0x00000000	0x00000000
0x0008	0x00000000	0x00000000	0x00000000	0x00000000
0x0009	0x00000000	0x00000000	0x00000000	0x00000000
0x000a	0x00000000	0x00000000	0x00000000	0x00000000
0x000b	0x00000000	0x00000000	0x00000000	0x00000000
0x000c	0x00000000	0x00000000	0x00000000	0x00000000
0x000d	0x00000000	0x00000000	0x00000000	0x00000000
0x000e	0x00000000	0x00000000	0x00000000	0x00000000
0x000f	0x00000000	0x00000000	0x00000000	0x00000000
0x0010	0x00000000	0x00000000	0x00000000	0x00000000
0x0011	0x00000000	0x00000000	0x00000000	0x00000000
0x0012	0x00000000	0x00000000	0x00000000	0x00000000
0x0013	0x00000000	0x00000000	0x00000000	0x00000000
0x0014	0x00000000	0x00000000	0x00000000	0x00000000
0x0015	0x00000000	0x00000000	0x00000000	0x00000000
0x0016	0x00000000	0x00000000	0x00000000	0x00000000
0x0017	0x00000000	0x00000000	0x00000000	0x00000000
0x0018	0x00000000	0x00000000	0x00000000	0x00000000
0x0019	0x00000000	0x00000000	0x00000000	0x00000000
0x001a	0x00000000	0x00000000	0x00000000	0x00000000
0x001b	0x00000000	0x00000000	0x00000000	0x00000000
0x001c	0x00000000	0x00000000	0x00000000	0x00000000
0x001d	0x00000000	0x00000000	0x00000000	0x00000000
0x001e	0x00000000	0x00000000	0x00000000	0x00000000
0x001f	0x00000000	0x00000000	0x00000000	0x00000000

Hex

Clear

ARMLite Simulator V1.2.4 © Peter Higginson 2020-23
[Documentation](#)

Excercise 8.1.2

Program

```
1 MOV R1, #15
2 STR R0, .WriteSignedNum
3 MOV R1, #str
4 STR R1, .WriteString
5 MOV R2, #int
6 STR R2, .WriteString
7 LDR R2, .InputNum
8 STR R2, .WriteSignedNum
9 HALT
10 str: .ASCIZ "remaining\n"
11 int: .ASCIZ "How many do you want to remove(1-3) ?\n"
```

Processor

PC: 0x0000002c
LR: 0x00000000
SP: 0x00100000
R12: 0x00000000
R11: 0x00000000
R10: 0x00000000
R9: 0x00000000
R8: 0x00000000
R7: 0x00000000
R6: 0xfe696e69
R5: 0x00000000
R4: 0x00000000
R3: 0x00000000
R2: 0x00000004
R1: 0x00000024
R0: 0x00000000

Count: 11
Current Instruction: e696e69
Status bits: NZCV 0000

Input/Output

How many do you want to remove(1-3) ?
4
Done instruction MOV Rd, #im at line
undefined

Memory

000	0x0	0x4	0x8	0xc
0x0000	0xe3a0100f	0xe50f00fc	0xe3a01024	0xe50f10f4
0x0001	0xe3a0202f	0xe50f20fc	0xe51f2118	0xe50f2114
0x0002	0xe1000070	0xe61d6572	0xe6e696e9	0x48000a67
0x0003	0xe6d20776f	0x20796e61	0x79206f64	0x7720756f
0x0004	0x20746e61	0x72206f74	0x766f6d65	0x2d312865
0x0005	0xf2029333	0x0000000a	0x00000000	0x00000000
0x0006	0x00000000	0x00000000	0x00000000	0x00000000
0x0007	0x00000000	0x00000000	0x00000000	0x00000000
0x0008	0x00000000	0x00000000	0x00000000	0x00000000
0x0009	0x00000000	0x00000000	0x00000000	0x00000000
0x000a	0x00000000	0x00000000	0x00000000	0x00000000
0x000b	0x00000000	0x00000000	0x00000000	0x00000000
0x000c	0x00000000	0x00000000	0x00000000	0x00000000
0x000d	0x00000000	0x00000000	0x00000000	0x00000000
0x000e	0x00000000	0x00000000	0x00000000	0x00000000
0x000f	0x00000000	0x00000000	0x00000000	0x00000000
0x0010	0x00000000	0x00000000	0x00000000	0x00000000
0x0011	0x00000000	0x00000000	0x00000000	0x00000000
0x0012	0x00000000	0x00000000	0x00000000	0x00000000
0x0013	0x00000000	0x00000000	0x00000000	0x00000000
0x0014	0x00000000	0x00000000	0x00000000	0x00000000
0x0015	0x00000000	0x00000000	0x00000000	0x00000000
0x0016	0x00000000	0x00000000	0x00000000	0x00000000
0x0017	0x00000000	0x00000000	0x00000000	0x00000000
0x0018	0x00000000	0x00000000	0x00000000	0x00000000
0x0019	0x00000000	0x00000000	0x00000000	0x00000000
0x001a	0x00000000	0x00000000	0x00000000	0x00000000
0x001b	0x00000000	0x00000000	0x00000000	0x00000000
0x001c	0x00000000	0x00000000	0x00000000	0x00000000
0x001d	0x00000000	0x00000000	0x00000000	0x00000000
0x001e	0x00000000	0x00000000	0x00000000	0x00000000
0x001f	0x00000000	0x00000000	0x00000000	0x00000000

Decimal (unsigned)

Clear

ARMLite Simulator V1.2.4 © Peter Higginson 2020-23
[Documentation](#)

Exercise 8.1.3

Program

```

1 MOV R0, #15
2 STR R0, .WriteSignedNum
3
4 MOV R1, #str
5 STR R1, .WriteString
6 MOV R2, #int
7 LDR R2, .InputNum
8 STR R2, .WriteSignedNum
9 MOV R3, #rem
10 STR R3, .WriteString
11 SUB R0, R0, R2
12 STR R0, .WriteSignedNum
13 HALT
14 str: .ASCII "remaining\n"
15 int: .ASCII "How many do you want to remove(1-3) ?\n"
16 rem: .ASCII "The remaining match sticks are\n"

```

Processor

PC: 0x0000000c
 LR: 0x00000000
 SP: 0x00100000
 R12: 0x00000000
 R11: 0x00000000
 R10: 0x00000000
 R9: 0x00000000
 R8: 0x00000000
 R7: 0x00000000
 R6: 0xf696e69
 R5: 0x00000000
 R4: 0x00000000
 R3: 0x00000000
 R2: 0x00000000
 R1: 0x00000034
 R0: 0x00000005

Count: 15
 Current Instruction: MOV Rd, #im
 Status bits: NZCV 0000

Input/Output

10 The remaining match sticks are 5
 Done instruction MOV Rd, #im at line undefined
 10

Memory

000 0x0000 0xe3a0000f 0xe50f00fc 0xe3a01034 0xe50f10f4
 0x0001 0xe3a0203f 0xe50f20fc 0xe51f2118 0xe50f2114
 0x0002 0xe3a03066 0xe50f310c 0xe0400002 0xe50f0124
 0x0003 0xe1000070 0xe516d572 0xe696e6e9 0xe4800a67
 0x0004 0xe6d2076f 0xe20796e1 0xe7206f74 0xe772075f
 0x0005 0xe20746e1 0xe7206f74 0xe766f6d5 0xe2d31285
 0x0006 0xe3f20293 0xe685400a 0xe6572205 0xe6e6961d
 0x0007 0xe2067e69 0xe637461d 0xe7473208 0xe736b639
 0x0008 0xe5726120 0xe000000a 0xe0000000 0xe0000000
 0x0009 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x000a 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x000b 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x000c 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x000d 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x000e 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x000f 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x0010 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x0011 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x0012 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x0013 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x0014 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x0015 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x0016 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x0017 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x0018 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x0019 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x001a 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x001b 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x001c 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x001d 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x001e 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x001f 0xe0000000 0xe0000000 0xe0000000 0xe0000000

Hex Clear

ARMLite Simulator V1.2.4 © Peter Higginson 2020-23
[Documentation](#)

Exercise 8.2.1

Program

```

1 MOV R0, #15
2 STR R0, .WriteSignedNum
3 Loop:
4 MOV R1, #str
5 STR R1, .WriteString
6 MOV R2, #int
7 STR R2, .WriteString
8 LDR R2, .InputNum
9 STR R2, .WriteSignedNum
10 MOV R3, #rem
11 STR R3, .WriteString
12 SUB R0, R0, R2
13 STR R0, .WriteSignedNum
14 B Loop
15 HALT
16 str: .ASCII "remaining\n"
17 int: .ASCII "How many do you want to remove(1-3) ?\n"
18 rem: .ASCII "The remaining match sticks are\n"

```

Processor

PC: 0x0000001c
 LR: 0x00000000
 SP: 0x00100000
 R12: 0x00000000
 R11: 0x00000000
 R10: 0x00000000
 R9: 0x00000000
 R8: 0x00000000
 R7: 0x00000000
 R6: 0xf696e69
 R5: 0x00000000
 R4: 0x00000000
 R3: 0x00000000
 R2: 0x00000034
 R1: 0x00000018
 R0: 0x00000000

Count: 51
 Current Instruction: LDR Rd, addr
 Status bits: NZCV 0000

Input/Output

0 remaining
 How many do you want to remove(1-3) ?
 Done instruction STR Rd, addr at line 7
 Input expected

Memory

000 0x0000 0xe3a0000f 0xe50f00fc 0xe3a01038 0xe50f10f4
 0x0001 0xe3a02043 0xe50f20fc 0xe51f2118 0xe50f2114
 0x0002 0xe3a0306a 0xe50f310c 0xe0400002 0xe50f0124
 0x0003 0xeaffff44 0xe1000070 0xe516d572 0xe696e6e9
 0x0004 0xe4800a67 0xe6d2076f 0xe20796e1 0xe7206f74
 0x0005 0xe772075f 0xe20746e1 0xe7206f74 0xe766f6d5
 0x0006 0xe2d31285 0xe3f20293 0xe685400a 0xe6572205
 0x0007 0xe6e961d 0xe2067e69 0xe637461d 0xe7473208
 0x0008 0xe736b639 0xe5726120 0xe000000a 0xe0000000
 0x0009 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x000a 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x000b 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x000c 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x000d 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x000e 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x000f 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x0010 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x0011 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x0012 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x0013 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x0014 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x0015 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x0016 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x0017 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x0018 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x0019 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x001a 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x001b 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x001c 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x001d 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x001e 0xe0000000 0xe0000000 0xe0000000 0xe0000000
 0x001f 0xe0000000 0xe0000000 0xe0000000 0xe0000000

Hex Clear

ARMLite Simulator V1.2.4 © Peter Higginson 2020-23
[Documentation](#)

What happens if you enter a number that takes the number of matchsticks remaining beyond 0 (i.e., into negative values)? What do you think is going on here? Hint - take a look at the value in the register!

- The value of the R0 will be negative.

Exercise 8.2.2

Question 8.2.2(a) - What is the condition that needs to be satisfied in order for this loop to occur ? Write this as a comparison using an inequality (ie., less than, greater than, less than or equal, greater than or equal)

- Satisfaction condition for the loop to occur is that the input value must not be between 1-3, it mean value entered must be < 1 and > 3

Question 8.2.2(b) - What two ARM assembly instructions could be used to create a branch that only occurs under this condition ?

- BLT, BGT constructions could be used to create a branch that only occurs under this condition

Question 8.2.2(c) - Based on the instructions you outlined in 8.2.2(b), what status bit would be set to 1 if the loop was to repeat ?

- N, Z, C status bit would be set to 1 if the loop was to repeat. N set to 1 when input value is 1 or 2 means values less than 3. Z set to 1 when input value is 3 and C set to one when we compare register with value 1.

Question 8.2.2(d) - What are all the modifications needed to the current program to implement this feature ? Make the required modifications to your program to perform the task.

- I would add an iterator and use BLT to compare input value and value 1 and use BGT to compare input value and value 3 if the input value is less than 1 or greater than 3 then it will reverse the loop and fill in the value again from the beginning otherwise it will continue the program.

The screenshot displays the ARMLite Simulator V1.2.4 interface, which is divided into several panels:

- Program Panel:** Shows the assembly code. The current instruction being executed is line 19: `cmp R0, #0`. The program includes instructions for moving registers, writing strings, and conditional branching.
- Processor Panel:** Displays the current state of the processor. The Program Counter (PC) is 64, and the Status Register (CPSR) is 0000. The instruction being executed is `CHP Rn, #im`.
- Memory Panel:** Shows a memory dump starting at address 0x0000. The memory contains various hexadecimal values, including 3818913807, 3842965756, and 3818917968.
- Input/Output Panel:** Shows the current input and output. The input is 1, and the output is 1. The panel also displays the current instruction being executed: `CHP Rn, #im`.

The simulator is running on a Windows operating system, and the interface is in English.

Question 8.3.1(a) What bit-wise operation can we perform on the register holding the 32 bit pattern to set all bits in the register to zero except the least significant 2 bits ? Write this as a single line of code.

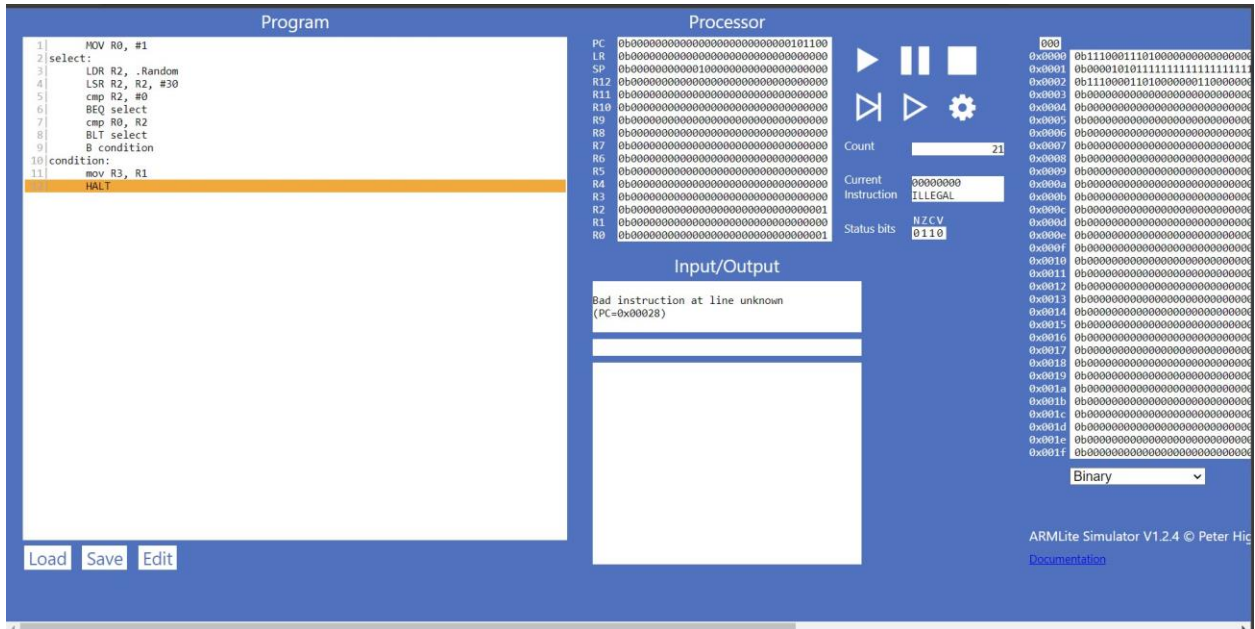
- LSR R0, #30 - (It allows us to shift 30 bits of R0 to the right which means 30 bits of the value in R0 will become 0 only the last 2 bits will also be 2 bits random , it can be 0 or 1).

The screenshot shows the ARMLite Simulator V1.2.4 interface. The Program window contains two instructions: `LDR R2, .Random` and `LSR R1, R2, #30`. The Processor window shows the PC at `0x00000008`, the current instruction as `LSR R1, R2, #30`, and the status bits as `NZCV 0000`. The Memory window shows a hex value of `0xe1a01f22` at address `0xe51f20e8`. The Count register is set to 2.

Question 8.3.1(b) Using a label named "`select:`" Write the code to repeatedly sample a random number (from `.Random`) until the value is in the range 1-3. For now, just write this as a separate program and test it

The screenshot shows the ARMLite Simulator V1.2.4 interface. The Program window contains a loop labeled `select:` with the following instructions: `LDR R2, .Random`, `LSR R1, R2, #30`, `cmp R1, #0`, `BEQ select`, `B condition`, and `mov R3, R1`. The Processor window shows the PC at `0x00000008`, the current instruction as `ILLEGAL`, and the status bits as `NZCV 0000`. The Memory window shows a binary value of `0b11100101000111100100000111` at address `0x00000000`. The Count register is set to 12.

Question 8.3.2(a) - Write the ARM assembly code that implements the algorithm expressed in the psuedo code above. Implement this as a seperate stand alone program and initialise R0 with a number at the start of your program to allow you to test the functionality. You will want to test it using different values in R0.



Exercise 8.4

My code:

```

1|  MOV R0, #15
2| Loop1:
3|  STR R0, .WriteSignedNum
4|  MOV R1, #str
5|  STR R1, .WriteString
6| Loop2:
7|  MOV R2, #int
8|  STR R2, .WriteString
9|  LDR R2, .InputNum
10|  STR R2, .WriteSignedNum
11|  cmp R2, R0

```

```
12|    BGT Loop2
13|    cmp R2, #1
14|    BLT Loop2
15|    cmp R2, #3
16|    BGT Loop2
17|    B cont
18|cont:
19|    SUB R0, R0, R2
20|    STR R0, .WriteSignedNum
21|    cmp R0, #0
22|    BEQ result1
23|    BGT select
24|select:
25|    MOV R1, #str
26|    STR R1, .WriteString
27|    B comp
28|select2:
29|    LDR R6, .Random
30|    LSR R6, R6, #30
31|    cmp R6, #0
32|    BEQ select2
33|    cmp R0, R6
34|    BLT select2
35|    SUB R0, R0, R6
36|    cmp R0, #0
37|    BEQ result
```

```
38|    BGT Loop1
39|comp:
40|    MOV R5, #compt
41|    STR R5, .WriteString
42|    B select2
43|result1:
44|    MOV R4, #none
45|    str R4, .WriteString
46|    MOV R8, #rst
47|    STR R8, .WriteString
48|    B end
49|result:
50|    MOV R4, #none
51|    str R4, .WriteString
52|    MOV R9, #rst1
53|    STR R9, .WriteString
54|end:
55|    HALT
56|str: .ASCIZ "remaining\n"
57|int: .ASCIZ "How many do you want to remove(1-3) ?\n"
58|none: .ASCIZ "There are no match stick\n"
59|compt: .ASCIZ "-----Computer turn-----"
60|rst: .ASCIZ "You lose\n"
61|rst1: .ASCIZ "You win\n"
```