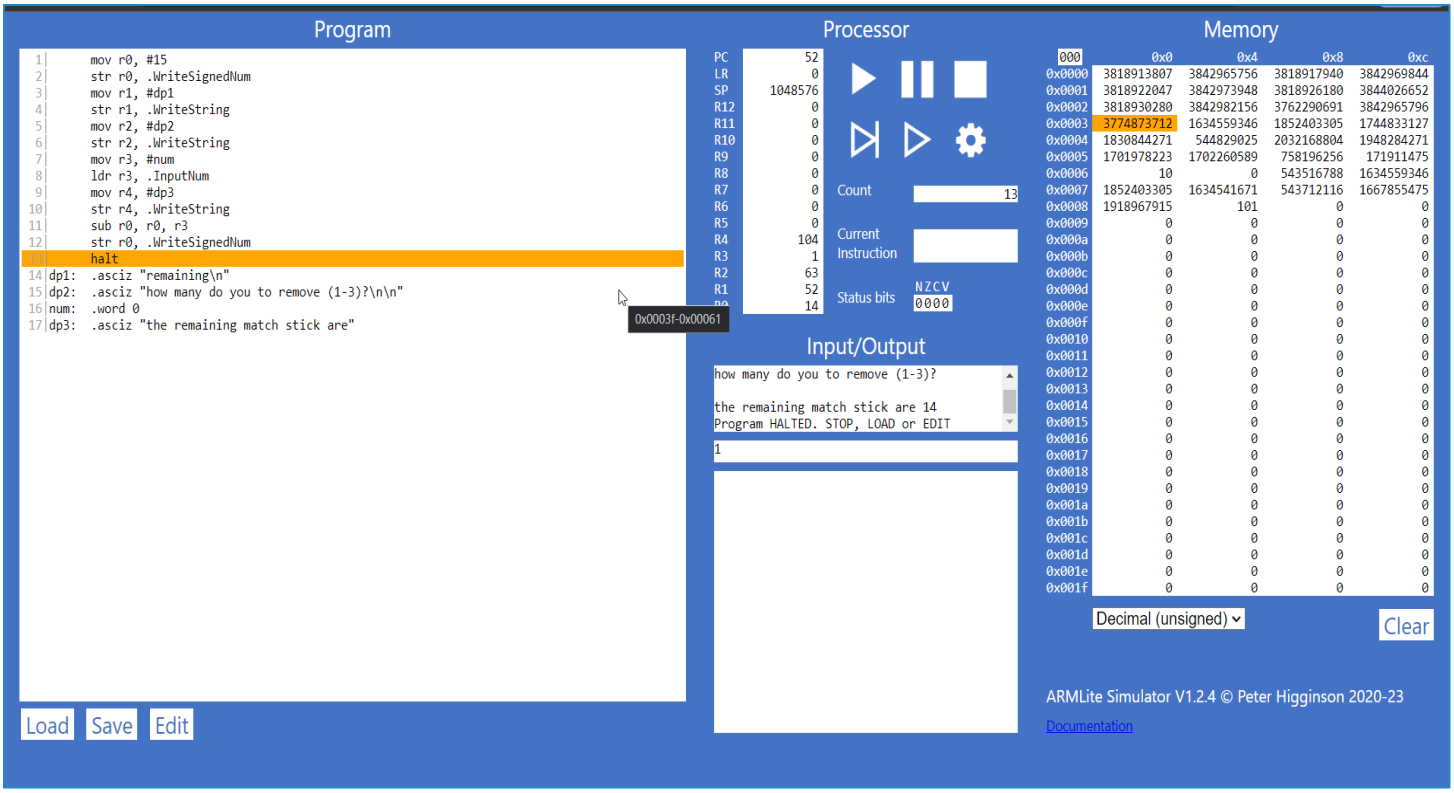
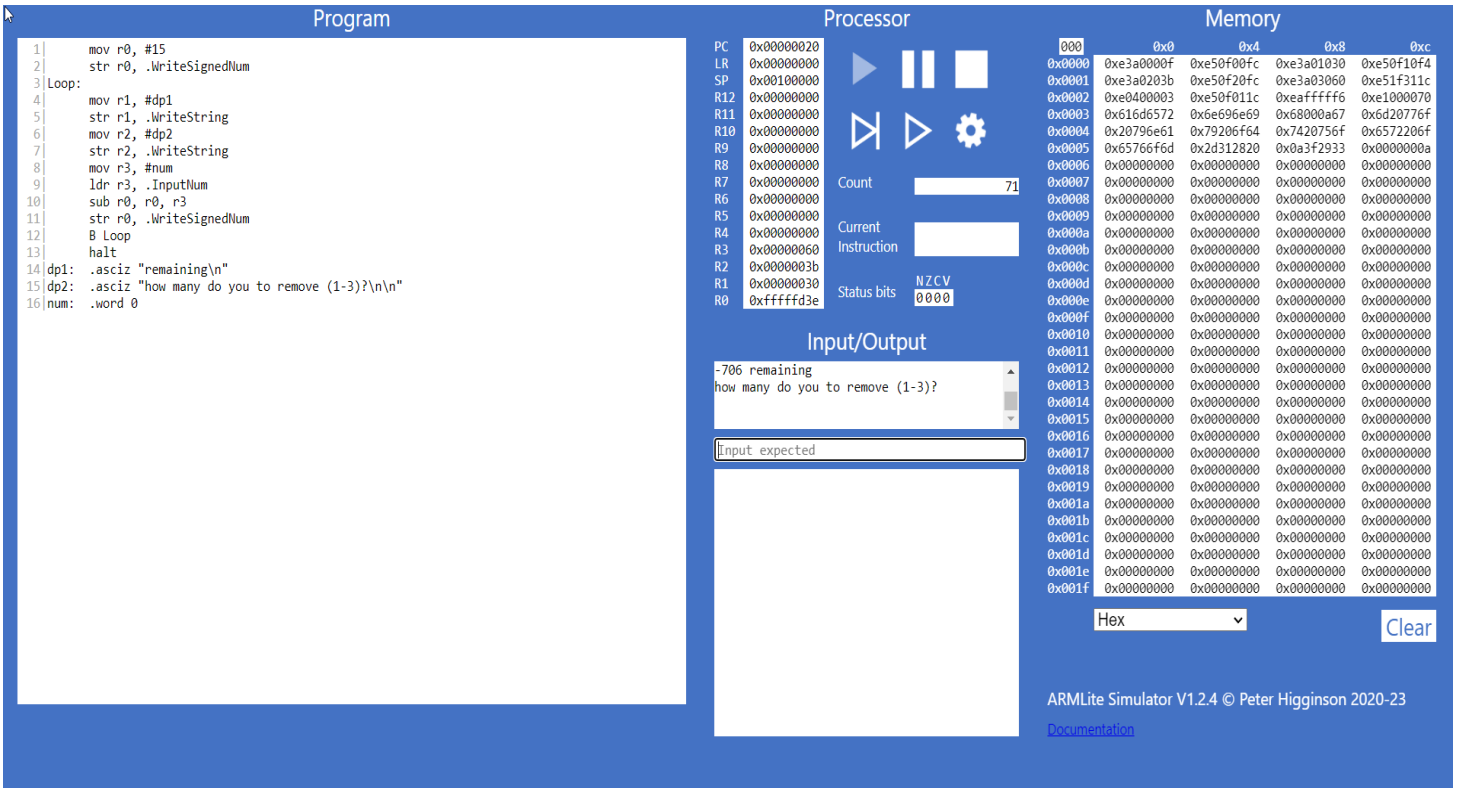
LAB 08

ARMlite

8.1



8.2



If the number of match sticks left cross 0 the value or answer goes into negative (-706). By using 2s compliment principal

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)?

Ans: - The input value must be between 1-3

8.2.2(b)

What two ARM assembly instructions could be used to create a branch that only occurs under this condition?

Ans: - cmp, blt and bgt

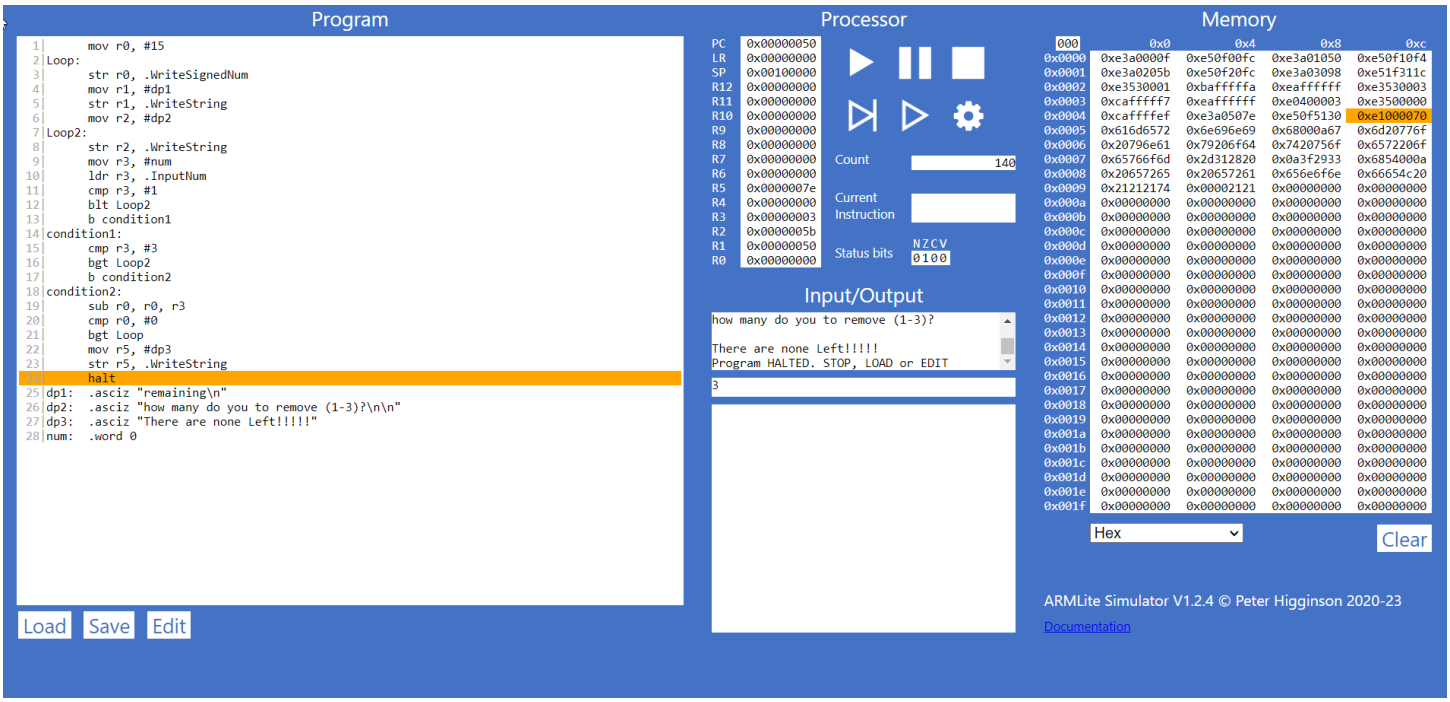
8.2.2(c)

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?

Ans: - The N and Z bits should be set to 1 when the instruction is executed. When the entered value is less than 3, only N bit is set, when we compared entered value with 3 the result was in this case negative.

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.

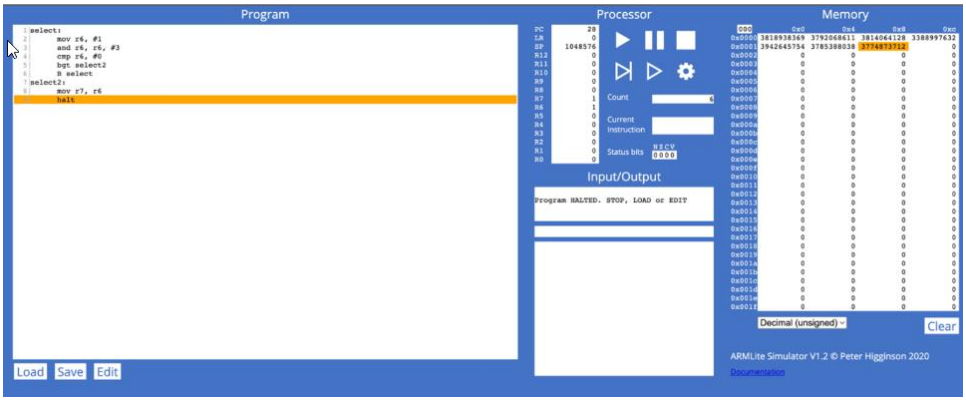


8.3.1(a)

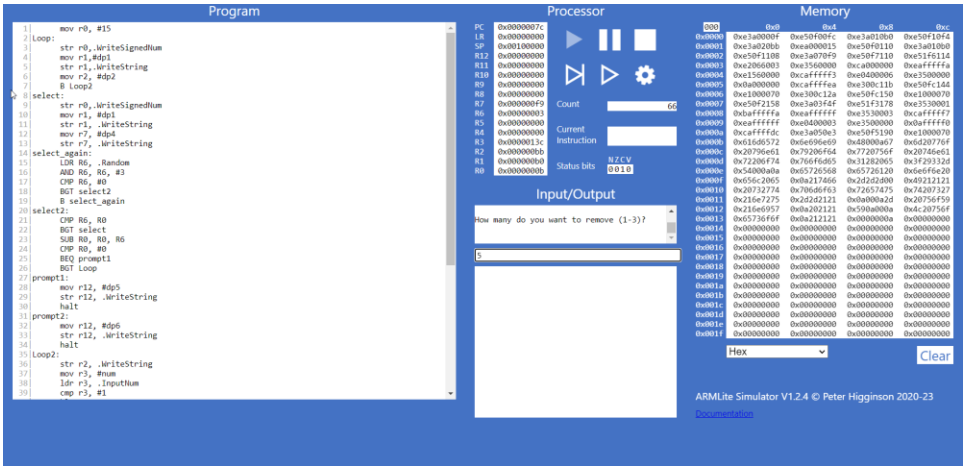
What bit-wise operation can we perform on the register holding the 32-bit pattern to set all bits in the register to zero except the least significant 2 bits? Write this as a single line of code.

Ans: - The bitwise operation is AND when we will use that operation with random number 0b00000000000000000000000000000011.

8.3.1(b)



8.4


Program
1 mov r0, #15
2 Loop:
3 str r0,.WriteSignedNum
4 mov r1,#dp1
5 str r1,.WriteString
6 mov r2, #dp2
7 B Loop2
8 select:
9 str r0,.WriteSignedNum
10 mov r1, #dp1
11 str r1, .WriteString
12 mov r7, #dp4
13 str r7, .WriteString
14 select_again:
15 LDR R6, .Random
16 AND R6, R6, #3
17 CMP R6, #0
18 BGT select2
19 B select_again
20 select2:
21 CMP R6, R0
22 BGT select
23 SUB R0, R0, R6
24 CMP R0, #0
25 BEQ prompt1
26 BGT Loop
27 prompt1:
28 mov r12, #dp5
29 str r12, .WriteString
30 halt
31 prompt2:
32 mov r12, #dp6
33 str r12, .WriteString
34 halt
35 Loop2:
36 str r2, .WriteString
37 mov r3, #num
38 ldr r3, .InputNum
39 cmp r3, #1
Processor
PC 0x0000007c
LR 0x00000000
SP 0x00100000
R12 0x00000000
R11 0x00000000
R10 0x00000000
R9 0x00000000
R8 0x00000000
R7 0x000000f9
R6 0x00000003
R5 0x00000000
R4 0x00000000
R3 0x0000013c
R2 0x000000bb
R1 0x000000b0
R0 0x000000bb
Count 66
Current Instruction
Status bits NZCV 0010
Input/Output
How many do you want to remove (1-3)?
5
Memory
000 0x0 0x4 0x8 0xc
0x0000 0xe3a0000f 0xe50f00fc 0xe3a010b0 0xe50f10f4
0x0001 0xe3a020bb 0xeaa000015 0xe50f0110 0xe3a010b0
0x0002 0xe50f1108 0xe3a070f9 0xe50f7110 0xe51f6114
0x0003 0xe2066003 0xe3560000 0xca000000 0xeafffffa
0x0004 0xe1560000 0xcafffff3 0xe0400006 0xe3500000
0x0005 0x0a000000 0xcafffffea 0xe300c11b 0xe50fc144
0x0006 0xe1000070 0xe300c12a 0xe50fc150 0xe1000070
0x0007 0xe50f2158 0xe3a03f4f 0xe51f3178 0xe3530001
0x0008 0xbafffffa 0xeafffffff 0xe3530003 0xcafffff7
0x0009 0xeafffffff 0xe0400003 0xe3500000 0x0afffff0
0x000a 0xcafffffdc 0xe3a050e3 0xe50f5190 0xe1000070
0x000b 0x616d6572 0x6e696e69 0x68000a67 0x6d20776f
0x000c 0x20796e61 0x79206f64 0x7720756f 0x20746e61
0x000d 0x72206f74 0x766f6d65 0x31282065 0x3f29332d
0x000e 0x54000a0a 0x65726568 0x65726120 0x6e6f6e20
0x000f 0x656c2065 0x0a217466 0x2d2d2d00 0x49212121
0x0010 0x20732774 0x706d6f63 0x72657475 0x74207327
0x0011 0x216e7275 0x2d2d2121 0x0a900a2d 0x20756f59
0x0012 0x216e6957 0x0a202121 0x590a000a 0x4c20756f
0x0013 0x65736f6f 0x0a212121 0x0000000a 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