

CSE 112 Lab 1

2.1

1.

1) $2^{15} + 2^{13} + 2^{11} + 2^9 + 2^8 + 2^7 + 2^6 + 2^3 + 2^2 + 1 = 43,981$

2) $2^{15} + 2^{14} + 2^{13} + 2^{12} + 2^{11} + 2^{10} + 2^9 + 2^7 + 2^6 + 2^4 + 12 = 65,244$

3) $2^{14} + 2^{13} + 2^{12} + 2^{11} + 2^{10} + 2^8 + 2^7 + 2^6 + 2^5 + 2^4 + 8 = 32,248$

4) $2^{13} + 2^{12} + 2^5 + 2^4 + 2^3 + 1 = 12,345$

2.

1) $15 \times (16^3) = 40,960$

2) $8 \times 16^3 + 10 \times 16^2 + 8 \times 16 + 9 = 35,465$

3) $16^2 + 9 \times 16 = 400$

4) $10 \times 16^3 + 15 \times 16^2 + 12 \times 16 + 13 = 45,005$

3. First converting all the numbers in binary and then forming triplets from right hand side to convert in octal:

1) $(a000)_{16} = (1010\ 0000\ 0000\ 0000)_2 = (120000)_8$

2) $(8a89)_{16} = (1000\ 1010\ 1000\ 1001)_2 = (105211)_8$

3) $(0190)_{16} = (0000\ 0001\ 1001\ 0000)_2 = (0620)_8$

4) $(afcd)_{16} = (1010\ 1111\ 1100\ 1101)_2 = (127715)_8$

4.

1) $(1010\ 1011\ 1100\ 1101)_2 = (001\ 010\ 101\ 111\ 001\ 101)_2 = (125715)_8$

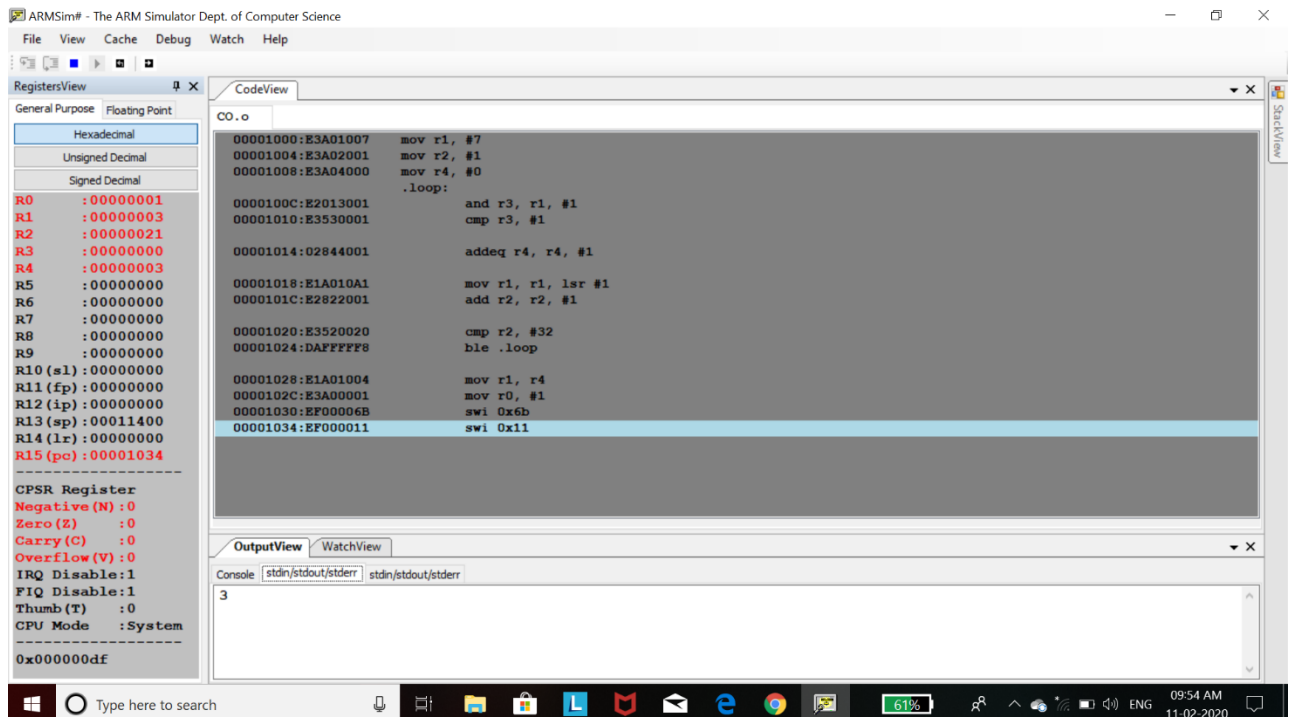
2) $(1111\ 1110\ 1101\ 1100)_2 = (001\ 111\ 111\ 011\ 011\ 100)_2 = (177334)_8$

3) $(0111\ 1101\ 1111\ 1000)_2 = (000\ 111\ 110\ 111\ 111\ 000)_2 = (076770)_8$

4) $(0011\ 0000\ 0011\ 1001)_2 = (000\ 011\ 000\ 000\ 111\ 001)_2 = (030071)_8$

2.2

2.

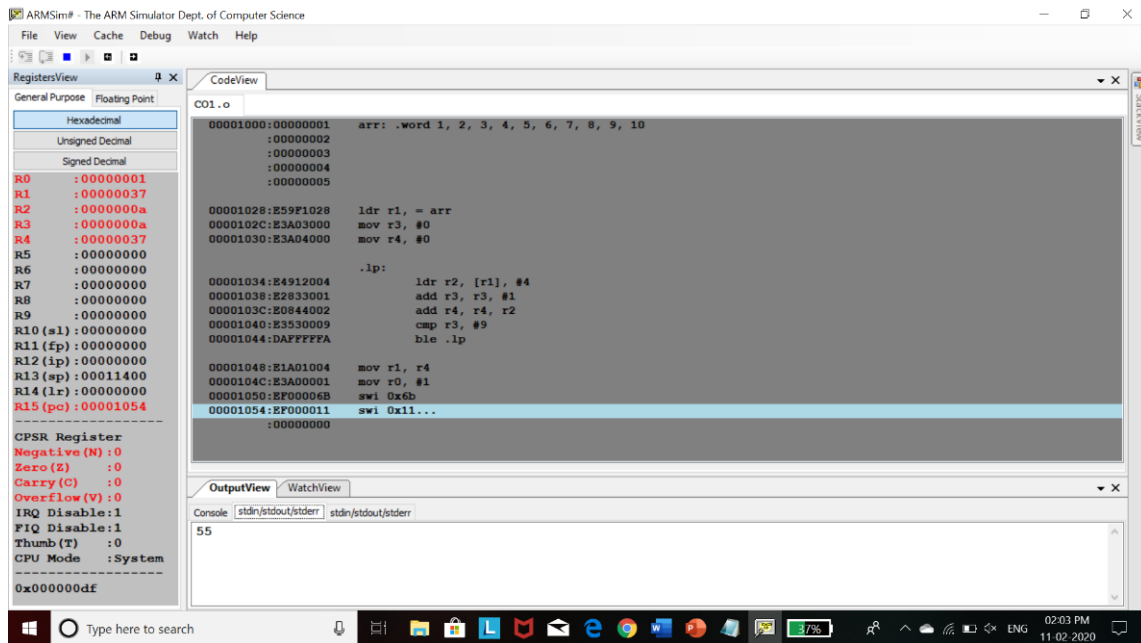


This works as follows:

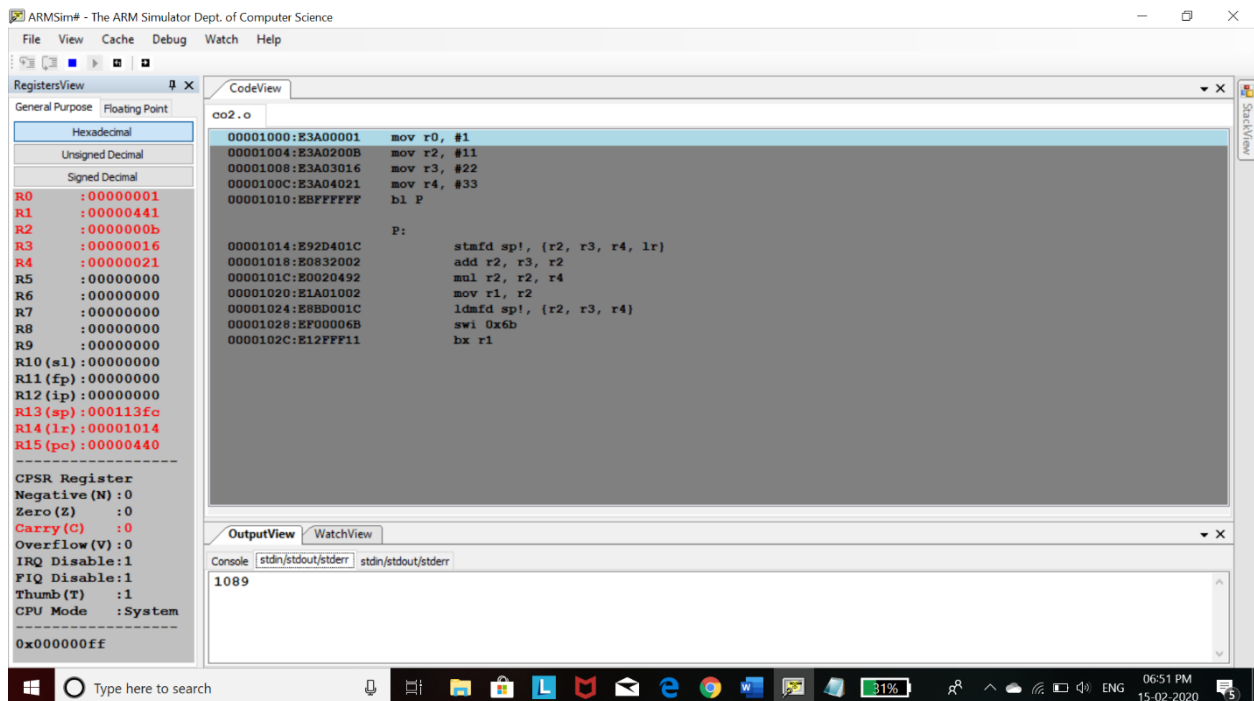
The value 7 is stored register r1, value 1 in r2 and value 0 in r4. Now, the loop executes: and operation is applied on 1 and the LSB of the binary number stored in r1, the answer gets stored in r3. The value in r3 is compared with 1 and if it is 1, 1 is added to the initial value stored in r4. Next, the bit in r2 gets shifted towards right so as to check if the next bit is 1. Since one operation is complete 1 is added to r2 which depicts the count of the bit being checked. r2 is compared with 32 to check if all the bits are compared or not. If it comes out to be less than or equal to 32 it branches to loop otherwise the programme proceeds to end by storing the value of r4 in r1 and 1 in r0 for stdout handle and print the value stored in r1 using swi0x6b. The last line ends the programme.

2.3

1. Sum the numbers of array a[]. Find a way to initialize this array with values you choose. Print sum.



2. The procedure should add the first two and multiply it with the third. Invoke the procedure P with arguments 11, 22 and 33. Print the result returned by P.



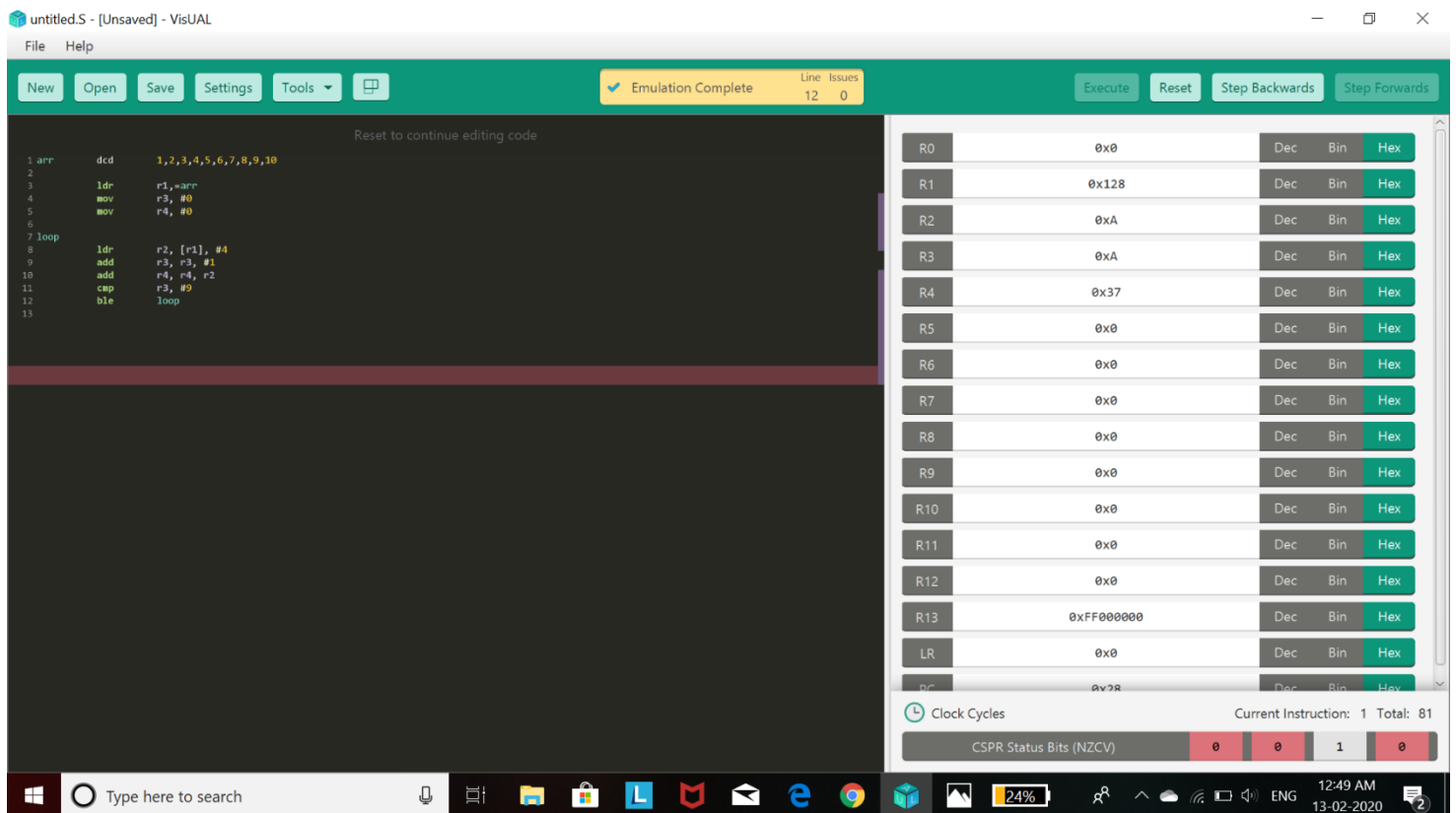
In visual

2.2)

1. The explanation is same as ARMSim.

2.3)

1. Sum the numbers of array a[]. Find a way to initialize this array with values you choose. Print the sum.



The result is stored in r4. Please zoom in if the font size is small.

2. The procedure should add the first two and multiply it with the third. Invoke the procedure P with arguments 11, 22 and 33. Print the result returned by P.

The screenshot shows the VisUAL emulator interface. The main window displays assembly code for a procedure named 'proc'. The code is as follows:

```
1  mov    r0, #1
2  mov    r2, #11
3  mov    r3, #22
4  mov    r4, #33
5  mov    r5, #0
6  bl     proc
7  end
8
9
10 proc
11 stm    sp!, {r2, r3, r4, lr}
12 add    r2, r3, r2
13 mov    r6, #0
14
15 loopmul
16 add    r5, r5, r4
17 add    r6, r6, #1
18 cmp    r6, r2
19 lne    loopmul
20
21 ldm    sp!, {r2, r3, r4, pc}
22 mov    pc, lr
23
24
25 end
26
27
```

On the right side, a register window shows the values of registers R0 through R15. The values are:

Register	Value	Dec	Bin	Hex
R0	0x1			
R1	0x0			
R2	0x21			
R3	0x16			
R4	0x21			
R5	0x441			
R6	0x21			
R7	0x0			
R8	0x0			
R9	0x0			
R10	0x0			
R11	0x0			
R12	0x0			
R13	0xFF000000			
LR	0x18			
PC	0x30			

At the bottom, the status bar shows 'Clock Cycles: 0', 'Current Instruction: 0', and 'Total: 209'. The system tray at the bottom right shows the date and time: '08:38 PM 15-02-2020'.

The result is stored in r5. Please zoom if the font size is small.