

Microprocessor and Computer Architecture Laboratory

UE19CS256

4th Semester, Academic Year 2020-21

Date:

Name: A Narendiran	SRN:PES1UG19CS001	Section
		A

Week# 2 Program Number: 1

1. Based on the value of the number in R0, Write an ALP to store 1 in R1 if R0 is zero, Store 2 in R1 if R0 is positive, Store 3 in R1 if R0 is negative.

ARM Assembly Code for the program

```
.text
MOV r0 , #-10
CMP r0, #0
BEQ if_zero
BMI if_negative
MOV r1, #2
SWI 0x011
if_negative:
MOV r1, #3
SWI 0x011
if_zero:
MOV r1, #1
SWI 0x011
.end
```

Output Screen Shot

RegistersView	
General Purpose	Floating Point
Hexadecimal	
Unsigned Decimal	
Signed Decimal	
R0	: -10
R1	: 3
R2	: 0
R3	: 0
R4	: 0
R5	: 0
R6	: 0
R7	: 0
R8	: 0
R9	: 0
R10 (sl)	: 0
R11 (fp)	: 0
R12 (ip)	: 0
R13 (sp)	: 0
R14 (lr)	: 4128
R15 (pc)	: 8

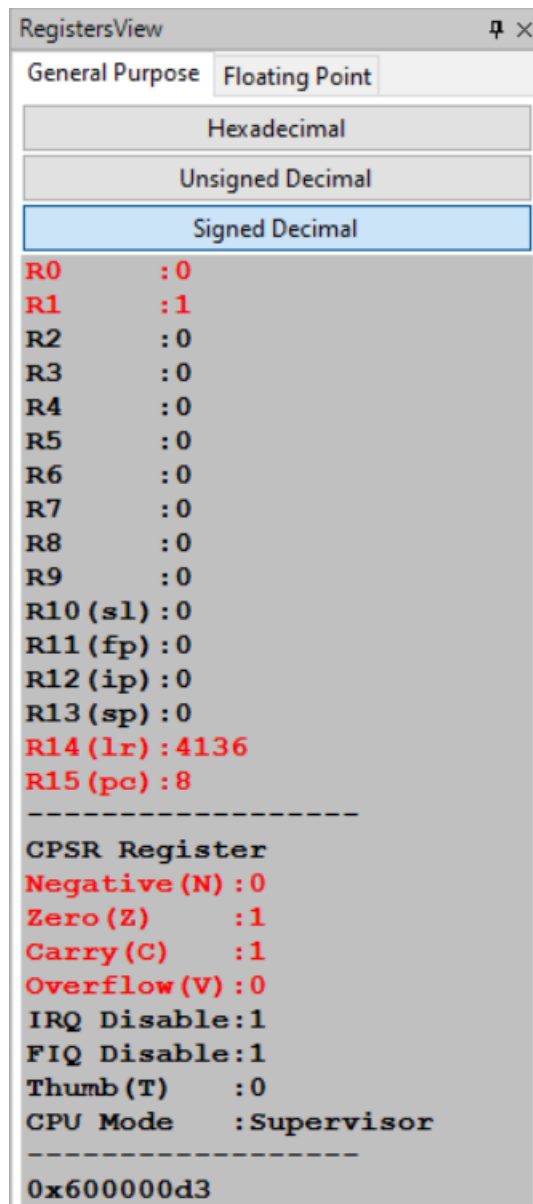
CPSR Register	
Negative (N)	: 1
Zero (Z)	: 0
Carry (C)	: 1
Overflow (V)	: 0
IRQ Disable	: 1
FIQ Disable	: 1
Thumb (T)	: 0
CPU Mode	: Supervisor

0xa00000d3	

RegistersView	
General Purpose	Floating Point
Hexadecimal	
Unsigned Decimal	
Signed Decimal	
R0	: 99
R1	: 2
R2	: 0
R3	: 0
R4	: 0
R5	: 0
R6	: 0
R7	: 0
R8	: 0
R9	: 0
R10 (sl)	: 0
R11 (fp)	: 0
R12 (ip)	: 0
R13 (sp)	: 0
R14 (lr)	: 4120
R15 (pc)	: 8

CPSR Register	
Negative (N)	: 0
Zero (Z)	: 0
Carry (C)	: 1
Overflow (V)	: 0
IRQ Disable	: 1
FIQ Disable	: 1
Thumb (T)	: 0
CPU Mode	: Supervisor

0x200000d3	



Week# 2 Program Number: 2

2. Write an ALP to compare the value of R0 and R1, add if R0 = R1, else subtract

ARM Assembly Code for the program

```
.text
MOV r0, #106
MOV r1, #106
CMP r0, r1
BEQ if_equal
```

```

SUB r2, r0, r1
SWI 0x011
if_equal:
ADD r2, r1, r0
SWI 0x011
.end

```

Output Screen Shot

RegistersView		RegistersView	
General Purpose		General Purpose	
Floating Point		Floating Point	
Hexadecimal		Hexadecimal	
Unsigned Decimal		Unsigned Decimal	
Signed Decimal		Signed Decimal	
R0	:106	R0	:106
R1	:106	R1	:204
R2	:212	R2	:-98
R3	:0	R3	:0
R4	:0	R4	:0
R5	:0	R5	:0
R6	:0	R6	:0
R7	:0	R7	:0
R8	:0	R8	:0
R9	:0	R9	:0
R10 (s1)	:0	R10 (s1)	:0
R11 (fp)	:0	R11 (fp)	:0
R12 (ip)	:0	R12 (ip)	:0
R13 (sp)	:0	R13 (sp)	:0
R14 (lr)	:4128	R14 (lr)	:4120
R15 (pc)	:8	R15 (pc)	:8
-----		-----	
CPSR Register		CPSR Register	
Negative (N)	:0	Negative (N)	:1
Zero (Z)	:1	Zero (Z)	:0
Carry (C)	:1	Carry (C)	:0
Overflow (V)	:0	Overflow (V)	:0
IRQ Disable	:1	IRQ Disable	:1
FIQ Disable	:1	FIQ Disable	:1
Thumb (T)	:0	Thumb (T)	:0
CPU Mode	:Supervisor	CPU Mode	:Supervisor
-----		-----	
0x600000d3		0x800000d3	

Week# 2 Program Number: 3

3. Write an ALP to find the factorial of a number stored in R0. Store the value in R1 (without using LDR and STR instructions). Use only registers.

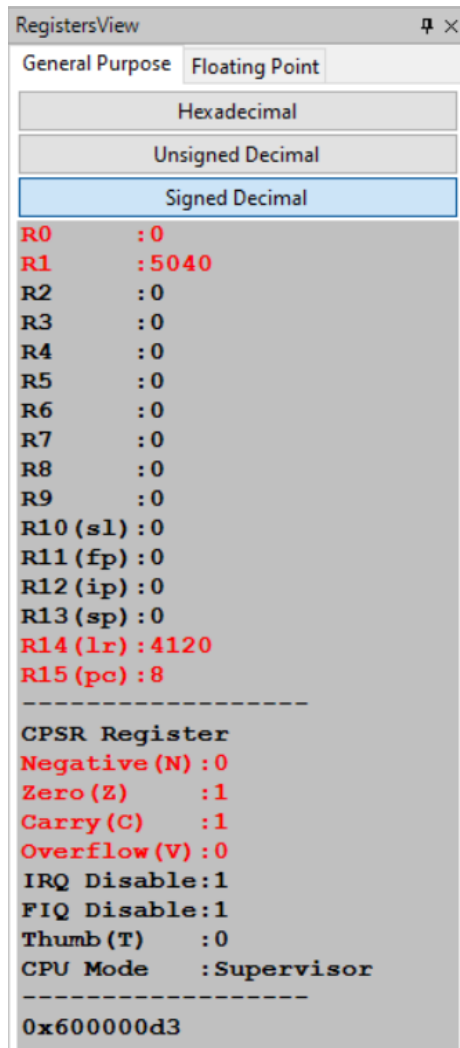
ARM Assembly Code for the program

```
.text

MOV r0, #7
MOV r1, #1
loop:
    MUL r1, r0, r1
    SUBS r0, r0, #1
    BNE loop
    SWI 0x011

.end
```

Output Screen Shot



Week#__2__

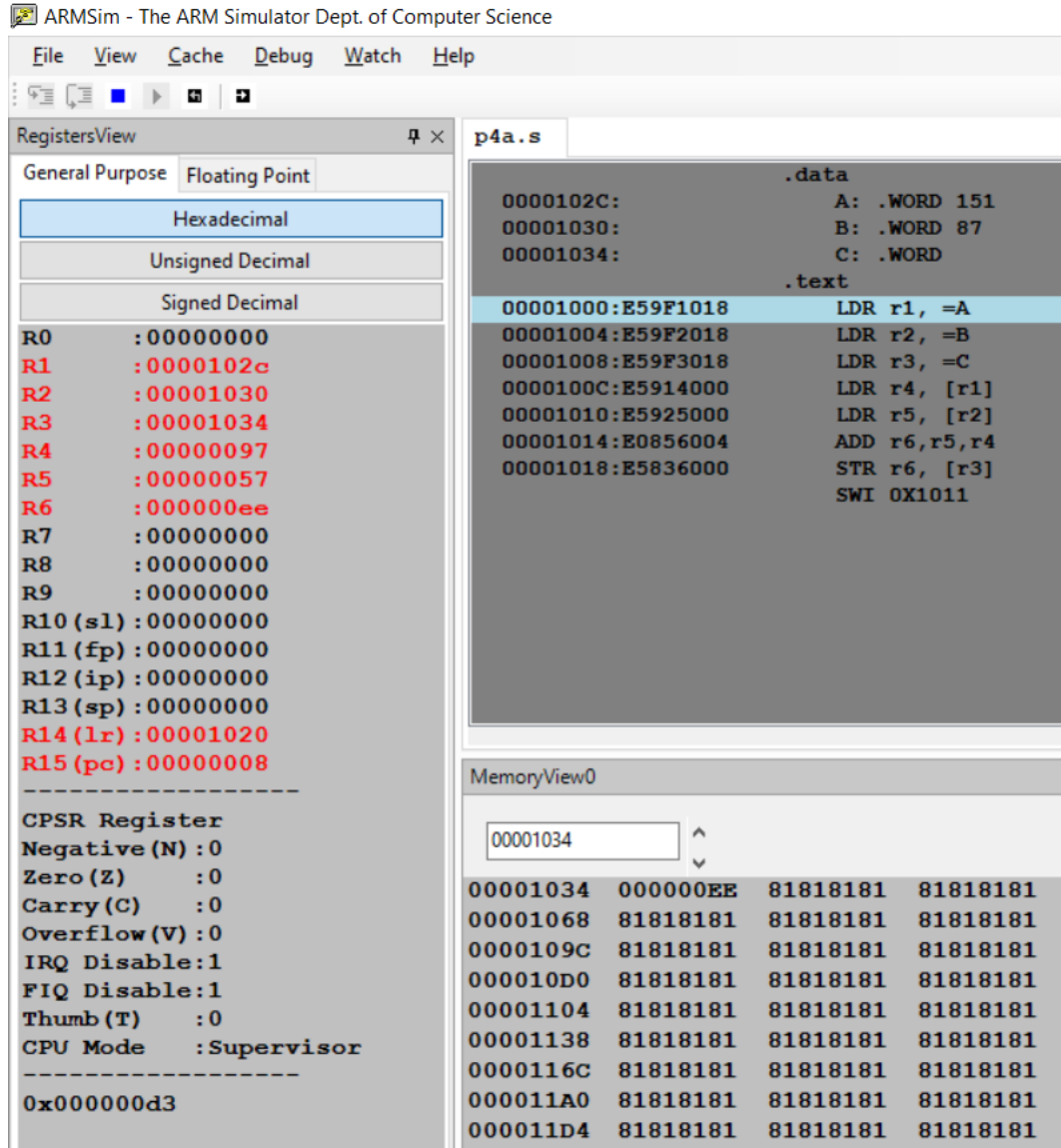
Program Number: __4a__

4. a) Write an ALP to add two 32 bit numbers loaded from memory and store the result in memory.

ARM Assembly Code for the program

```
.data
A: .WORD 151
B: .WORD 87
C: .WORD
.text
LDR r1, =A
LDR r2, =B
LDR r3, =C
LDR r4, [r1]
LDR r5, [r2]
ADD r6, r5, r4
STR r6, [r3]
SWI 0X1011
.end
```

Output Screen Shot



Week# 2 Program Number: 4b

4 b) Write an ALP to add two 16 bit numbers loaded from memory and store the result in memory.

ARM Assembly Code for the program

```
.data
A: .HWORD 18
B: .HWORD 11
C: .HWORD
.text
LDR r1, =A
```

```

LDR r2, =B
LDR r3, =C
LDRH r4, [r1]
LDRH r5, [r2]
ADD r6, r5, r4
STRH r6, [r3]
SWI 0X1011
.end

```

Output Screen Shot

The screenshot displays a debugger interface with two main windows: RegistersView and MemoryView0.

RegistersView: This window shows the state of the ARM registers. The 'General Purpose' tab is selected, and the 'Hexadecimal' view is chosen. The registers R0 through R15 are listed, with their values in hexadecimal. R14 (lr) and R15 (pc) are highlighted in red. Below the registers, the CPSR (Current Program Status Register) is shown with its various flags and modes.

MemoryView0: This window shows the contents of memory starting at address 00001030. The address 00001030 is entered in the search box. The memory is displayed in a table format with columns for address, hex value, and ASCII characters. The hex values are shown in red.

RegistersView Details:

Register	Value
R0	:00000000
R1	:0000102c
R2	:0000102e
R3	:00001030
R4	:00000012
R5	:0000000b
R6	:0000001d
R7	:00000000
R8	:00000000
R9	:00000000
R10 (s1)	:00000000
R11 (fp)	:00000000
R12 (ip)	:00000000
R13 (sp)	:00000000
R14 (lr)	:00001020
R15 (pc)	:00000008

CPSR Register:

Negative (N)	:0
Zero (Z)	:0
Carry (C)	:0
Overflow (V)	:0
IRQ Disable	:1
FIQ Disable	:1
Thumb (T)	:0
CPU Mode	:Supervisor

MemoryView0 Details:

Address	Hex Value	ASCII
00001030	001D	
00001064	8181 8181 8181 8181 8181 8181 8181 8181	
00001098	8181 8181 8181 8181 8181 8181 8181 8181	
000010CC	8181 8181 8181 8181 8181 8181 8181 8181	
00001100	8181 8181 8181 8181 8181 8181 8181 8181	
00001134	8181 8181 8181 8181 8181 8181 8181 8181	
00001168	8181 8181 8181 8181 8181 8181 8181 8181	
0000119C	8181 8181 8181 8181 8181 8181 8181 8181	

Week# 2

Program Number: 5a

5. a) Write an ALP to find GCD of two numbers (without using LDR and STR instructions). Both numbers are in registers. Use only registers.

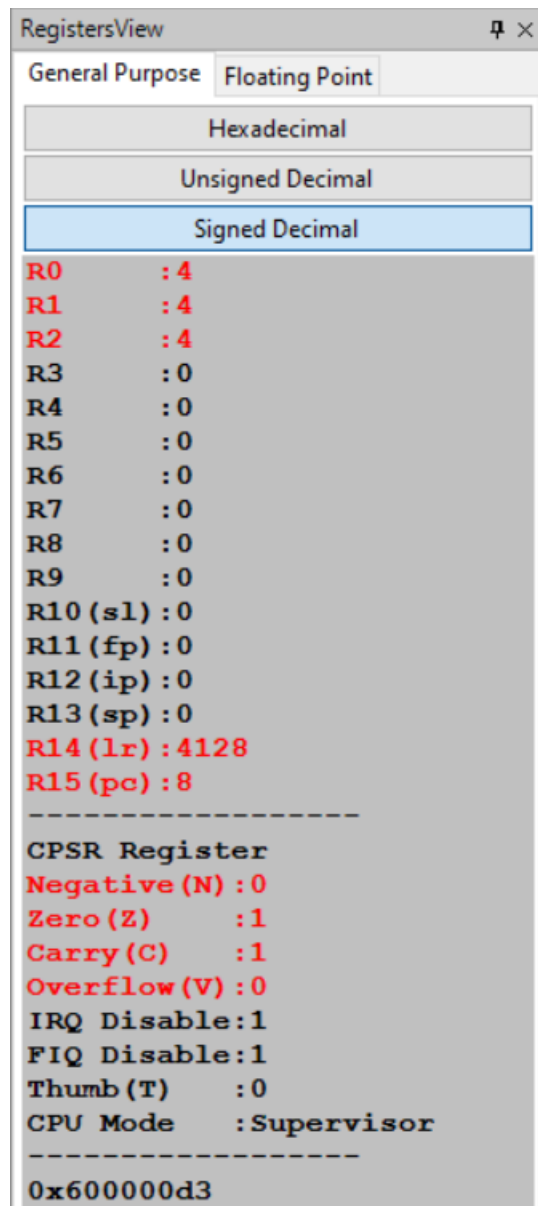
ARM Assembly Code for the program

```
.text

MOV r0, #36
MOV r1, #88
loop:
    CMP r1, r0
    BEQ L1
    BMI L2
    B L3
L1:
    MOV r2, r0
    SWI 0x011
L2:
    SUB r0, r0, r1
    B loop
L3:
    SUB r1, r1, r0
    B loop

.end
```

Output Screen Shot



Week# ____2____

Program Number: __5b__

5 b) Write an ALP to find the GCD of given numbers (both numbers in memory) Store result in memory.

ARM Assembly Code for the program

```
.data
    A:.WORD 36
    B:.WORD 88
    C: .WORD
.text
    LDR R0, =A
    LDR R1, =B
    LDR R2, =C
    LDR R0, [R0]
    LDR R1, [R1]

loop1: CMP R0,R1
        BEQ exit
        BLT loop2
        SUB R0,R0,R1
        B loop1
loop2: SUB R1,R1,r0
        B loop1
exit:
        STR R1,[R2]
        SWI 0x1011
.end
```

Output Screen Shot

```
RegistersView
General Purpose Floating Point
Hexadecimal
Unsigned Decimal
Signed Decimal
R0 : 4
R1 : 4
R2 : 4172
R3 : 0
R4 : 0
R5 : 0
R6 : 0
R7 : 0
R8 : 0
R9 : 0
R10 (s1) : 0
R11 (fp) : 0
R12 (ip) : 0
R13 (sp) : 0
R14 (lr) : 4152
R15 (pc) : 8
-----
CPSR Register
Negative (N) : 0
Zero (Z) : 1
Carry (C) : 1
Overflow (V) : 0
IRQ Disable: 1
FIQ Disable: 1
Thumb (T) : 0
CPU Mode : Supervisor
-----
0x600000d3
```

Week# 2

Program Number: 6a

6. a) Write an ALP to add an array of ten 32 bit numbers from memory.

ARM Assembly Code for the program

```
.data

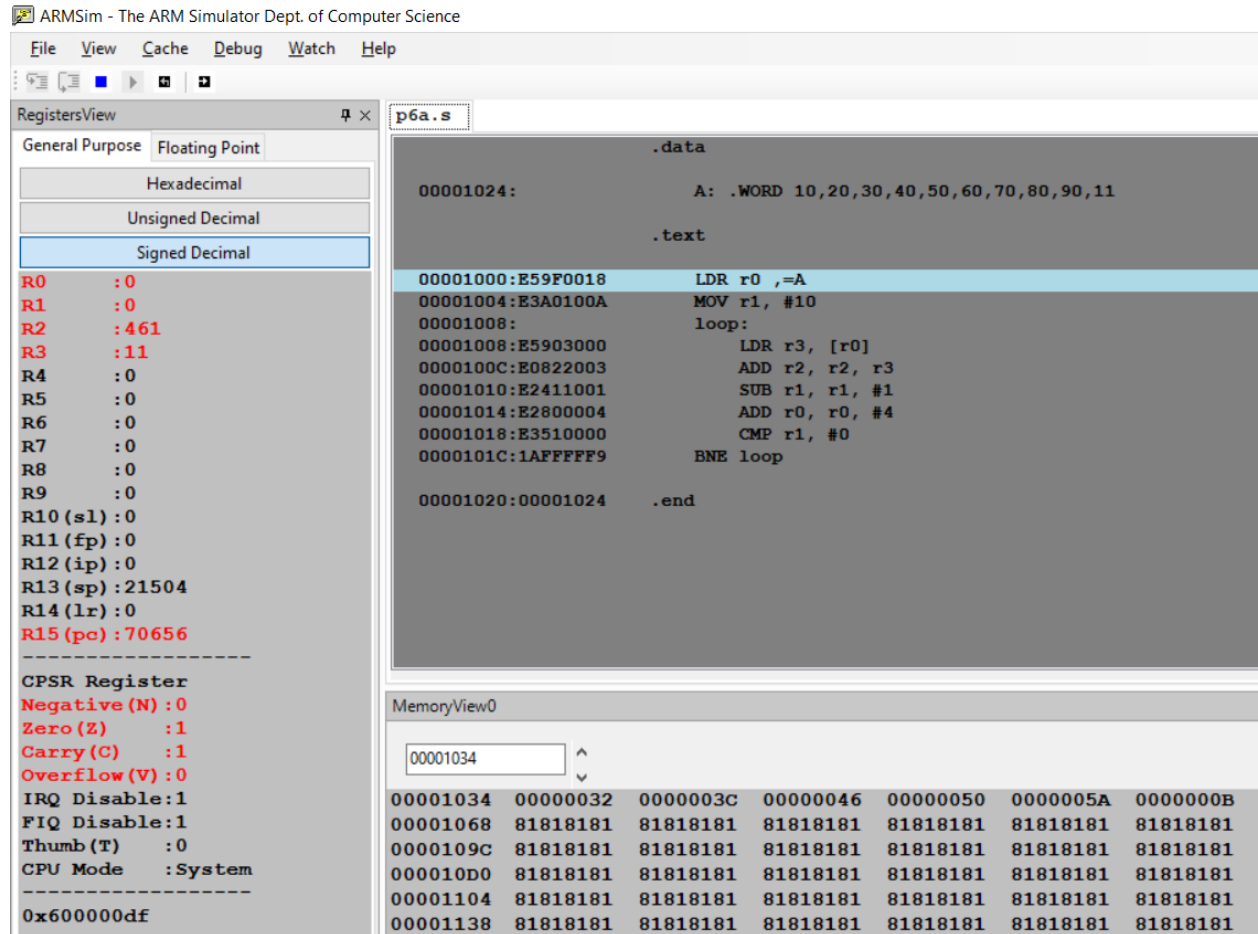
A: .WORD 10,20,30,40,50,60,70,80,90,11

.text

LDR r0 ,=A
MOV r1, #10
loop:
    LDR r3, [r0]
    ADD r2, r2, r3
    SUB r1, r1, #1
    ADD r0, r0, #4
    CMP r1, #0
    BNE loop

.end
```

Output Screen Shot



Week# 2 Program Number: 6b

6 b) Write an ALP to add array of ten 8 bit numbers taking data from memory location stored as byte data (use .byte to store the data instead of .word)

ARM Assembly Code for the program

```
.data

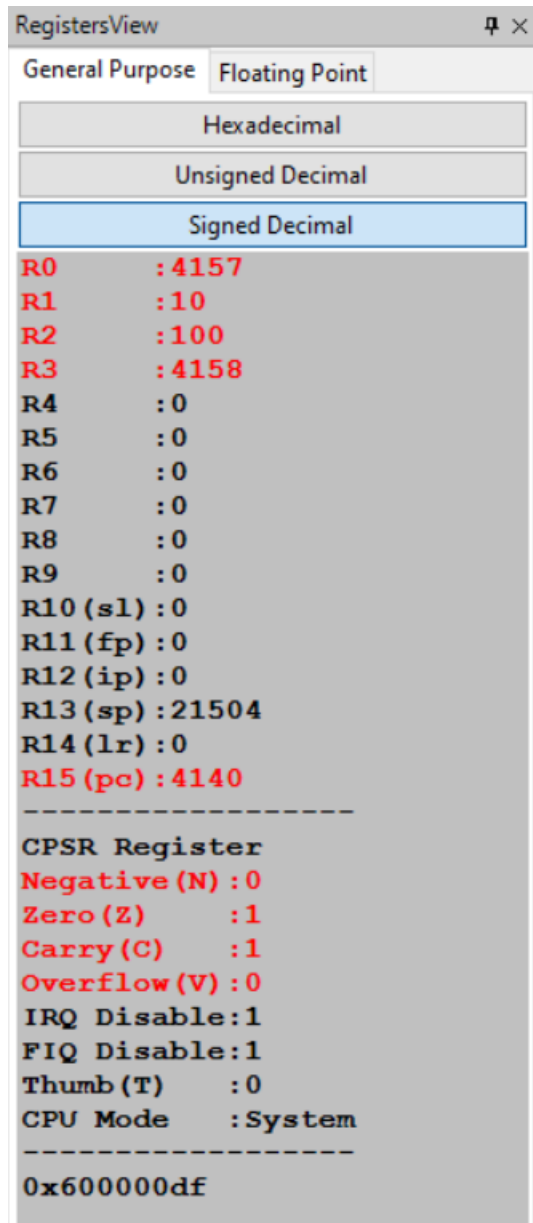
A: .BYTE 10,10,10,10,10,10,10,10,10,10
B: .BYTE

.text

LDR R0,=A
```

```
MOV R1,#1
LDRB R2,[R0]
loop:LDRB R3,[R0,#1]!
ADD R2,R2,R3
ADD R1,R1,#1
CMP R1,#10
BEQ exit
B loop
exit:
LDR R3,=B
STR R2,[R3]
.end
```

Output Screen Shot



Week# 2

Program Number: 7

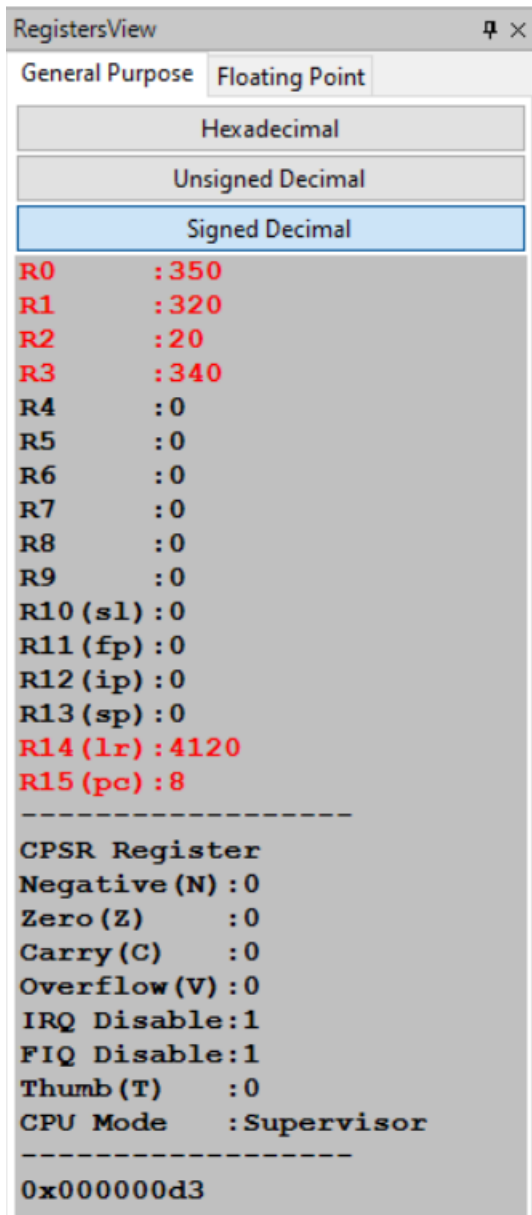
7. Write an ALP to multiply using barrel shifter
ARM Assembly Code for the program

```
.text
MOV R0,#10
MOV R1,R0, LSL #5;
MOV R2,R0, LSL #1;
```



```
ADD R3,R1,R2
ADD R0,R0,R3
SWI 0x1011
.end
```

Output Screen Shot



Week# 2

Program Number: 8

8. Write an ALP to evaluate the expression $(A+B) + (3*B)$, where A and B are memory location.

ARM Assembly Code for the program

```
.data
    A: .WORD 10
    B: .WORD 10
.text
    LDR R0,=A
    LDR R1,[R0]
    LDR R0,=B
    LDR R2, [R0]
    ADD R0,R1,R2
    MOV R4,R2 ,LSL #1
    ADD R4,R4,R2
    ADD R0,R0,R4
    SWI 0x1011
.end
```

Output Screen Shot

RegistersView		🚩 ×
General Purpose		Floating Point
Hexadecimal		
Unsigned Decimal		
Signed Decimal		
R0	: 50	
R1	: 10	
R2	: 10	
R3	: 0	
R4	: 30	
R5	: 0	
R6	: 0	
R7	: 0	
R8	: 0	
R9	: 0	
R10 (s1)	: 0	
R11 (fp)	: 0	
R12 (ip)	: 0	
R13 (sp)	: 0	
R14 (lr)	: 4132	
R15 (pc)	: 8	

CPSR Register		
Negative (N) : 0		
Zero (Z) : 0		
Carry (C) : 0		
Overflow (V) : 0		
IRQ Disable: 1		
FIQ Disable: 1		
Thumb (T) : 0		
CPU Mode : Supervisor		

0x000000d3		