**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

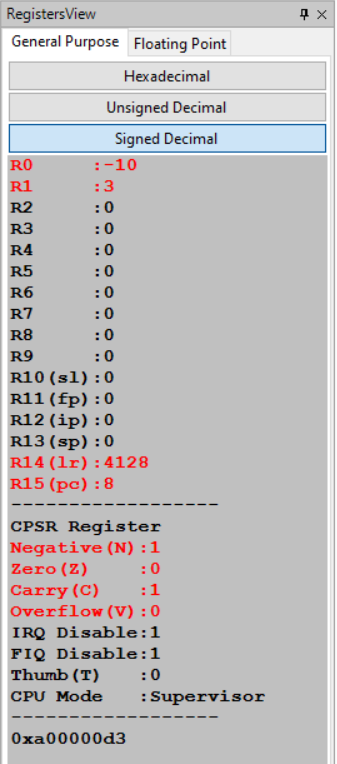
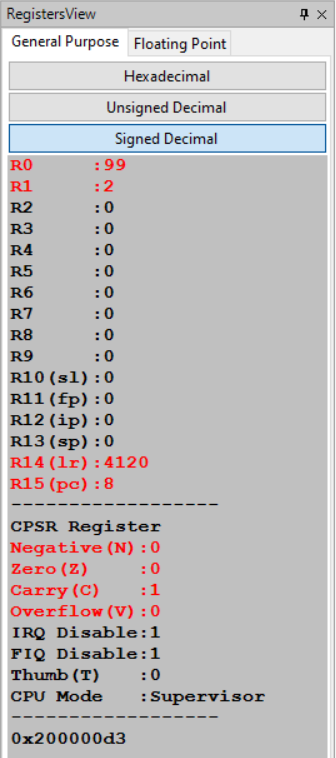
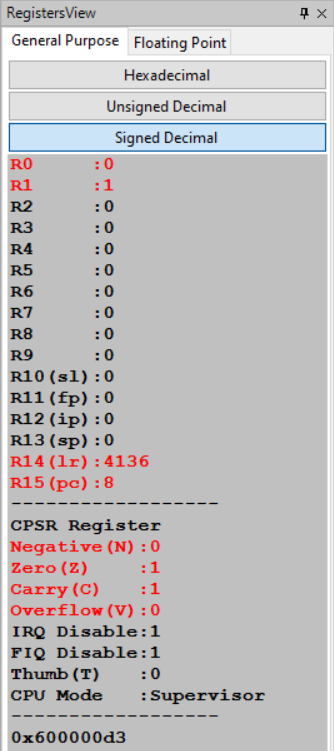
if\_zero:

MOV r1, #1

SWI 0x011

.end

Output Screen Shot

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

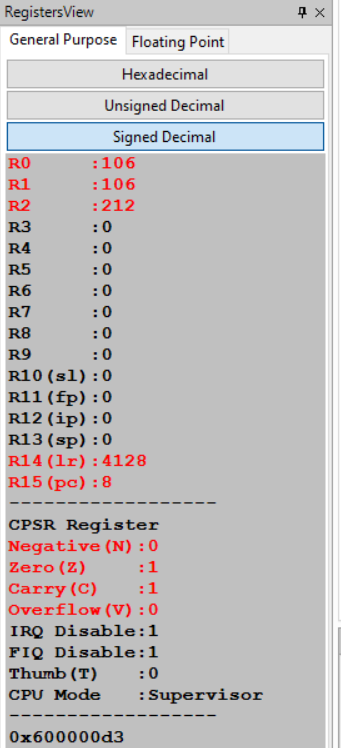
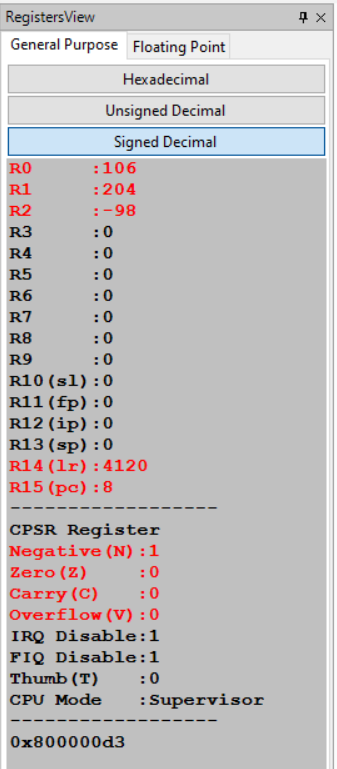
if\_equal:

ADD r2, r1, r0

SWI 0x011

.end

Output Screen Shot

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

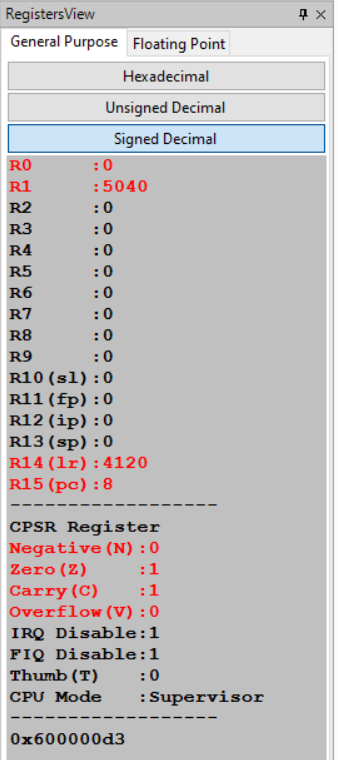
        SUBS r0, r0, #1

    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]

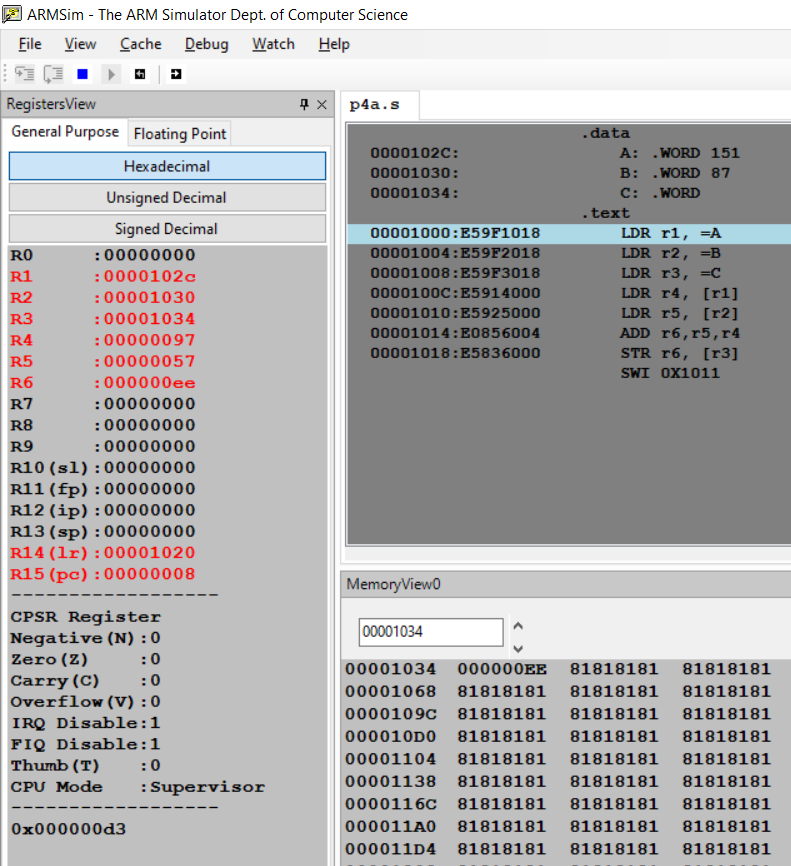
    ADD r6,r5,r4

    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]

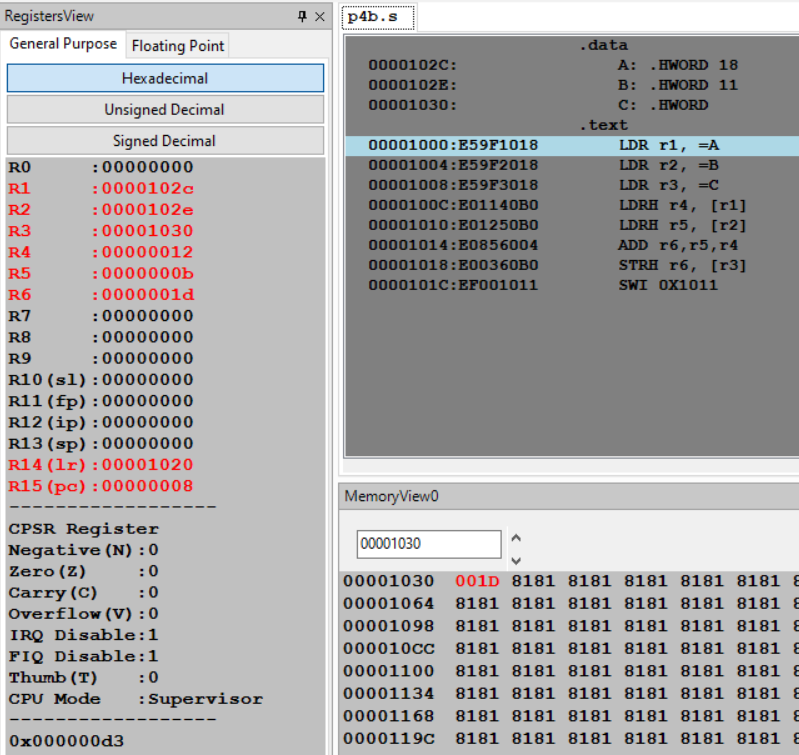
    ADD r6,r5,r4

    STRH r6, [r3]

    SWI 0X1011

.end

Output Screen Shot



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

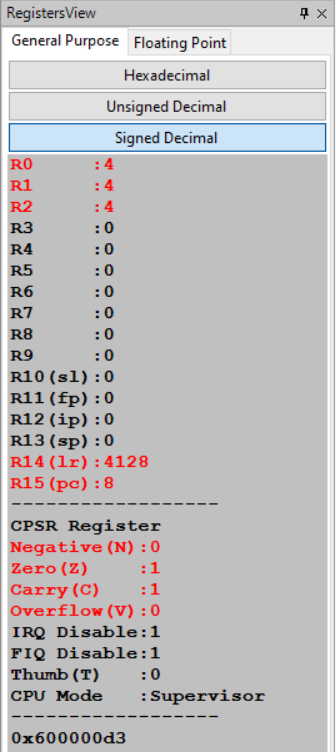
    L3:

        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

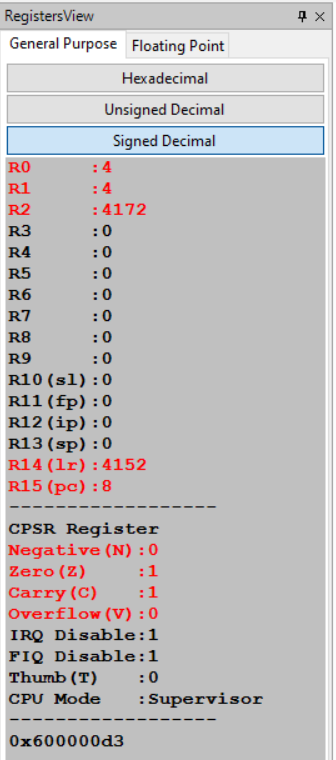
exit:

    STR R1,[R2]

    SWI 0x1011

.end

Output Screen Shot

****

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

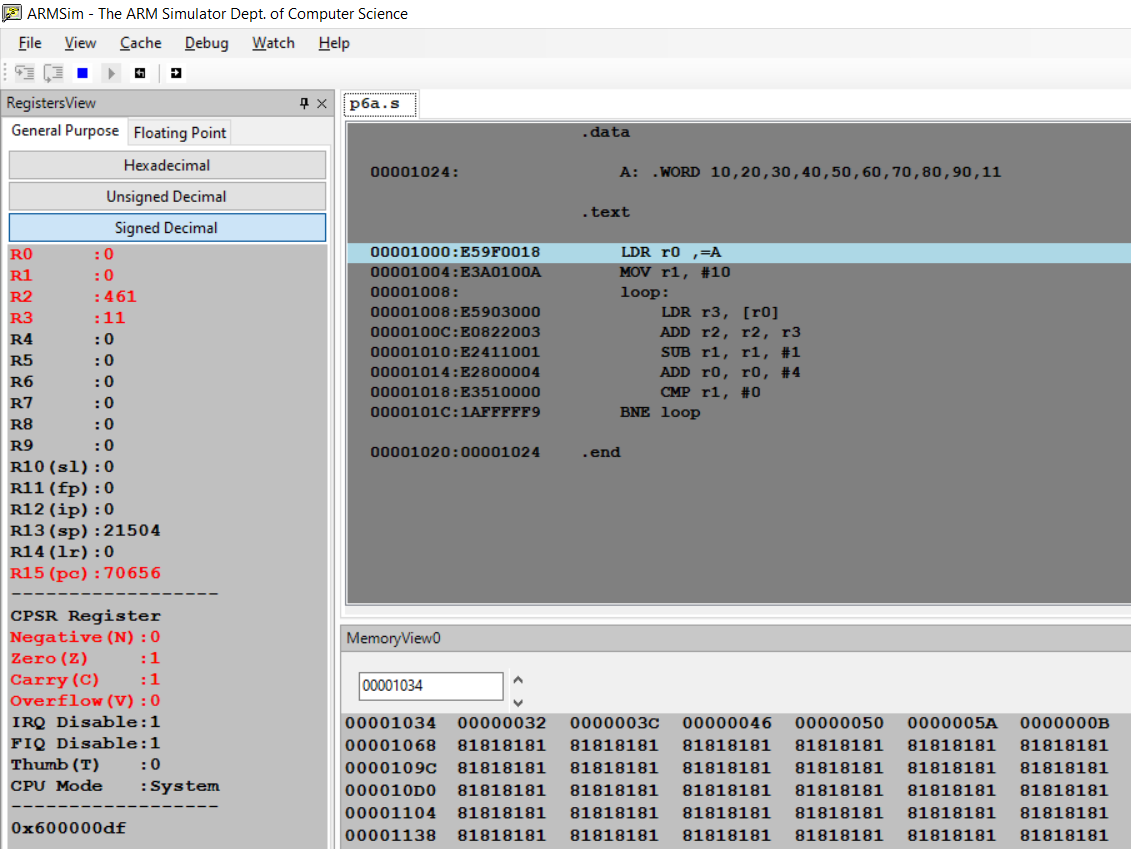
        ADD r0, r0, #4

        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

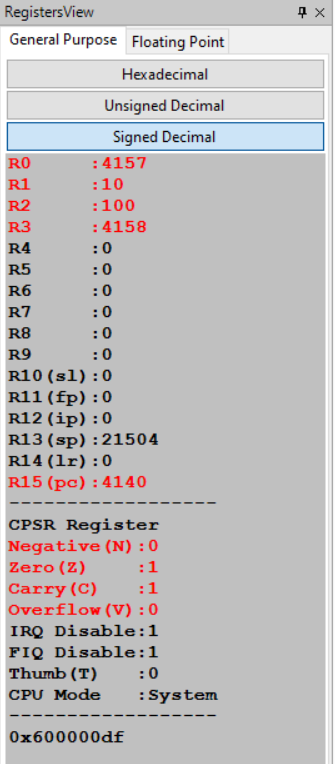
    exit:

    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*;*

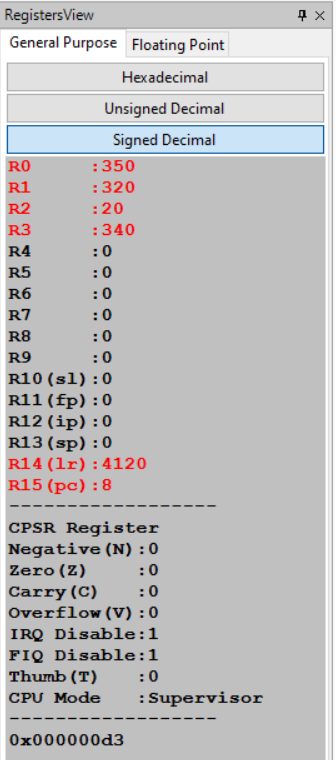
    ADD R3,R1,R2

    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

