**Microprocessor and Computer Architecture Laboratory**

**UE19CS256**

**4th Semester, Academic Year 2020-21**

Date: 02/02/2021

|  |  |  |
| --- | --- | --- |
| Name: Pranav R. Hegde | SRN: PES1UG19CS343 | Section: F |

Week#\_\_\_\_2\_\_\_\_\_\_\_ Program Number: \_\_\_1\_\_\_

Title of the Program

1. ARM Assembly Code for each program
2. Output Screen Shot

Problem statement:

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. (Program shown in class)

ARM assembly code:

mov r0, #7

mov r1, #7

cmp r0, r1

beq equal

blt lessthan

bgt greaterthan

equal:

    mov r2, #1

    swi 0x11

greaterthan:

    mov r2, #2

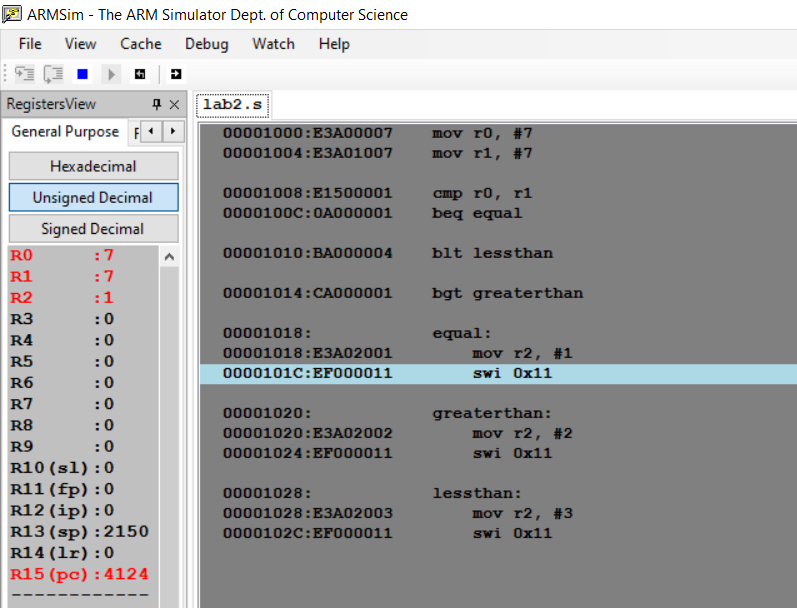
    swi 0x11

lessthan:

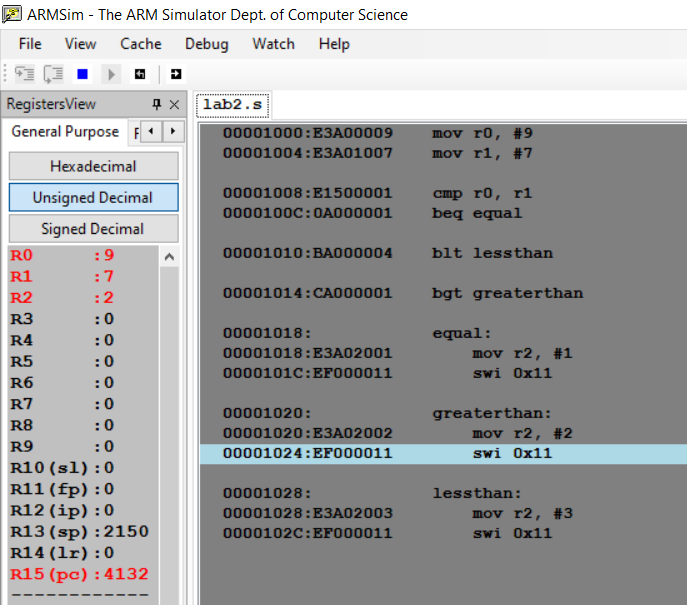
    mov r2, #3

    swi 0x11

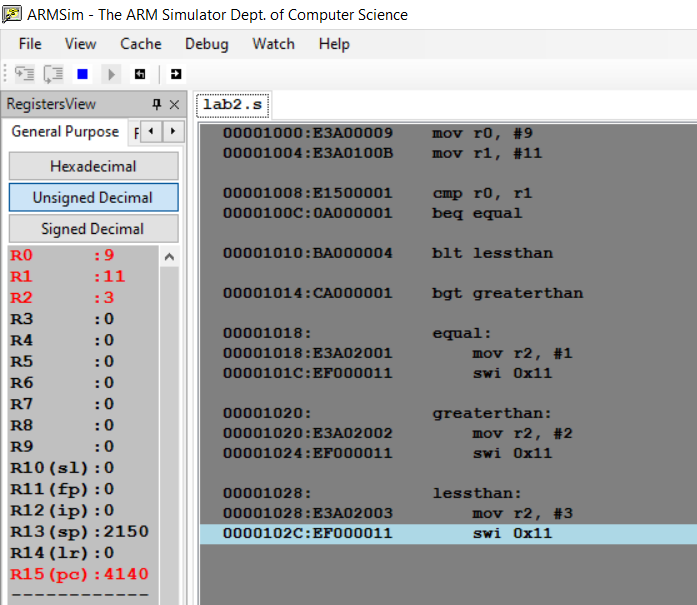
Case 1: r0 == r1



Case2: r0 > r1



Case3: r0<r1



**Microprocessor and Computer Architecture Laboratory**

**UE19CS256**

**4th Semester, Academic Year 2020-21**

Date: 02/02/2021

|  |  |  |
| --- | --- | --- |
| Name: Pranav R. Hegde | SRN: PES1UG19CS343 | Section: F |

Week#\_\_\_\_2\_\_\_\_\_\_\_ Program Number: \_\_\_2\_\_\_

Title of the Program

1. ARM Assembly Code for each program
2. Output Screen Shot

Problem statement:

2. Write an ALP to compare the value of R0 and R1, add if R0 = R1, else subtract (Program shown in class)

Code:

mov r0, #6

mov r1, #6

cmp r0, r1

beq equal

sub r2, r0, r1

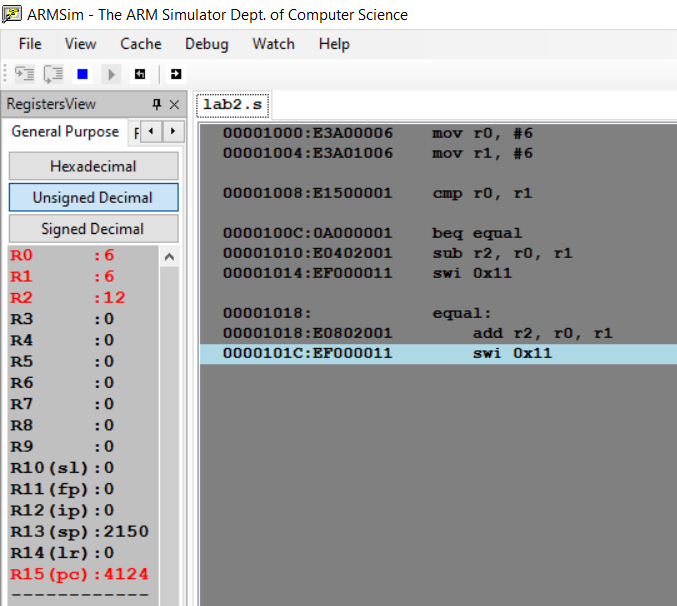
swi 0x11

equal:

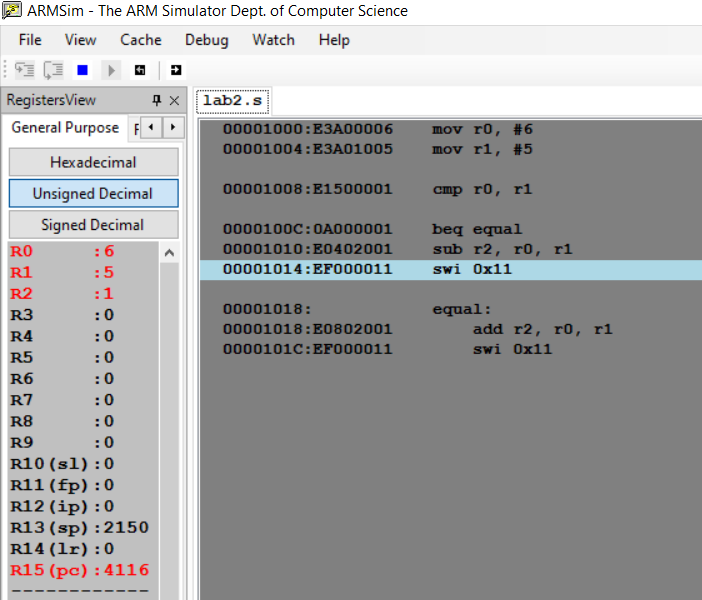
    add r2, r0, r1

    swi 0x11

Case1: both are equal



Case2: r0 and r1 are unequal



**Microprocessor and Computer Architecture Laboratory**

**UE19CS256**

**4th Semester, Academic Year 2020-21**

Date: 02/02/2021

|  |  |  |
| --- | --- | --- |
| Name: Pranav R. Hegde | SRN: PES1UG19CS343 | Section: F |

Week#\_\_\_\_2\_\_\_\_\_\_\_ Program Number: \_\_\_3\_\_\_

Title of the Program

1. ARM Assembly Code for each program
2. Output Screen Shot

Problem statement:

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. (Program shown in class)

Code:

mov r0, #4  ;this is the number of which factorial has to be found

mov r1, r0  ;just moving r0 to r1 for decrementing

mov r2, #1  ;the factorial of the above number ; results stored in r2

cmp r0, #1

beq factorial

factorial:

    mul r2, r1, r2

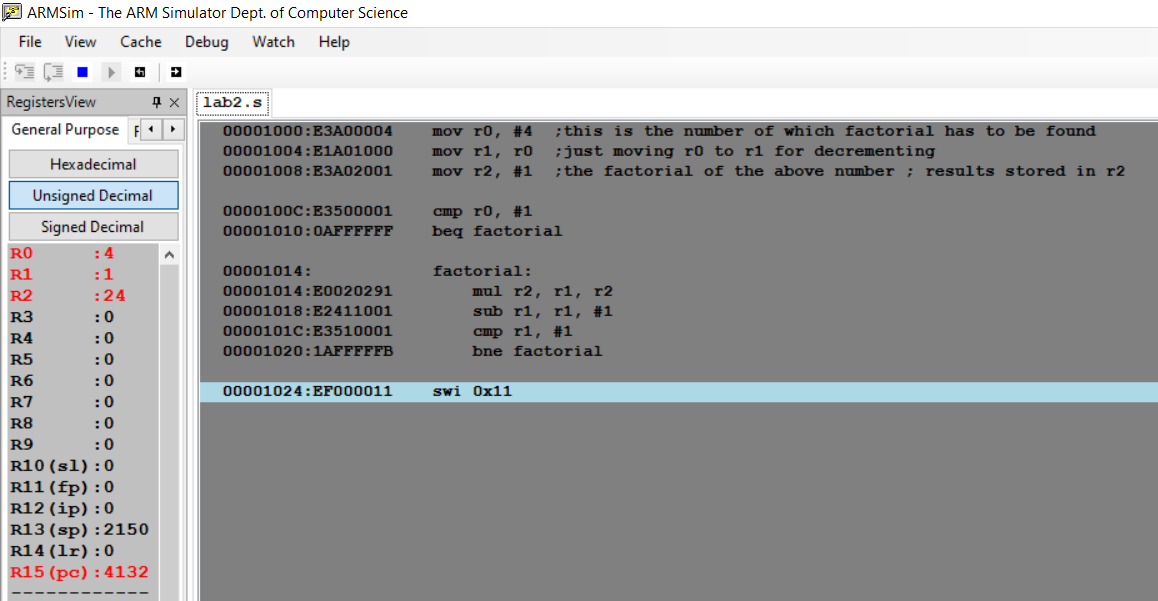
    sub r1, r1, #1

    cmp r1, #1

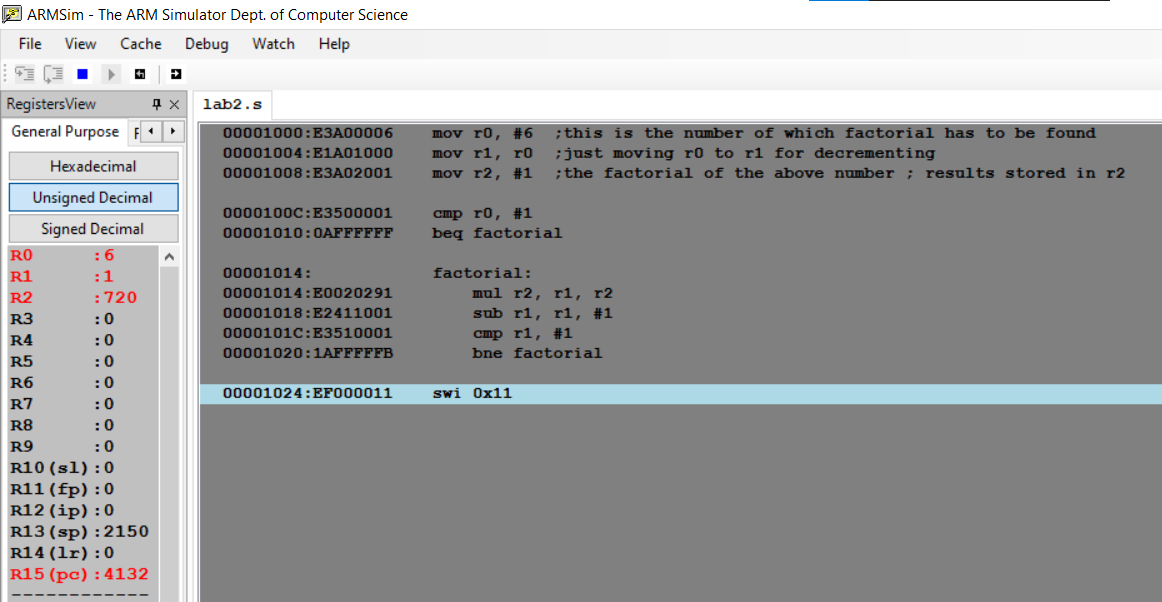
    bne factorial

swi 0x11

Case 1: r0 = 4



Case2: r0 = 6



**Microprocessor and Computer Architecture Laboratory**

**UE19CS256**

**4th Semester, Academic Year 2020-21**

Date: 02/02/2021

|  |  |  |
| --- | --- | --- |
| Name: Pranav R. Hegde | SRN: PES1UG19CS343 | Section: F |

Week#\_\_\_\_2\_\_\_\_\_\_\_ Program Number: \_\_\_4a\_\_\_

Title of the Program

1. ARM Assembly Code for each program
2. Output Screen Shot

Problem statement:

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

Code:

.TEXT

LDR R0, =A

LDR R1, =B

LDR R2, =C

LDR R0, [R0]

LDR R1, [R1]

ADD R3, R0, R1

STR R3, [R2]

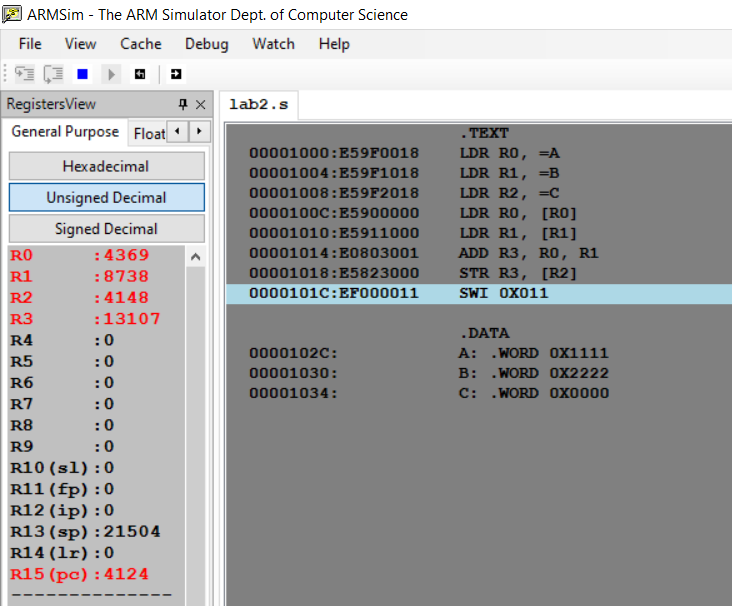
SWI 0X011

.DATA

A: .WORD 0X1111

B: .WORD 0X2222

C: .WORD 0X0000



**Microprocessor and Computer Architecture Laboratory**

**UE19CS256**

**4th Semester, Academic Year 2020-21**

Date: 02/02/2021

|  |  |  |
| --- | --- | --- |
| Name: Pranav R. Hegde | SRN: PES1UG19CS343 | Section: F |

Week#\_\_\_\_2\_\_\_\_\_\_\_ Program Number: \_\_\_4b\_\_\_

Title of the Program

1. ARM Assembly Code for each program
2. Output Screen Shot

Problem statement:

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

Code:

.TEXT

LDR R0, =A

LDR R1, =B

LDR R2, =C

LDR R0, [R0]

LDR R1, [R1]

ADD R3, R0, R1

STR R3, [R2]

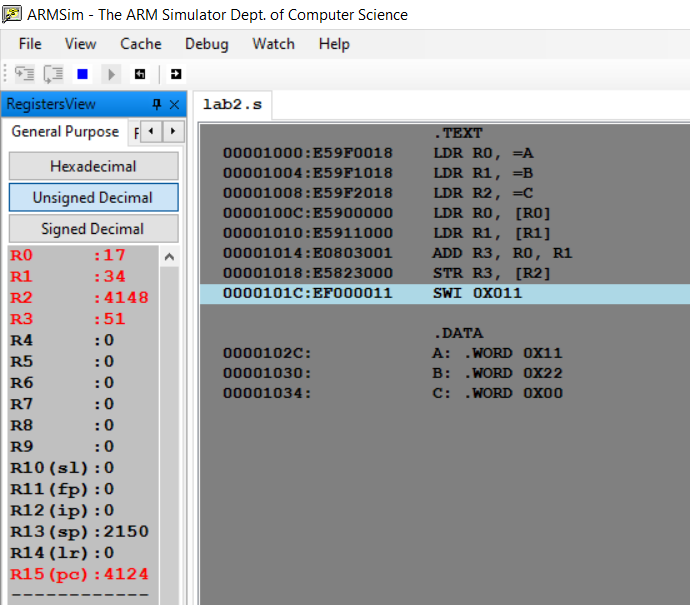
SWI 0X011

.DATA

A: .WORD 0X11

B: .WORD 0X22

C: .WORD 0X00



**Microprocessor and Computer Architecture Laboratory**

**UE19CS256**

**4th Semester, Academic Year 2020-21**

Date: 02/02/2021

|  |  |  |
| --- | --- | --- |
| Name: Pranav R. Hegde | SRN: PES1UG19CS343 | Section: F |

Week#\_\_\_\_2\_\_\_\_\_\_\_ Program Number: \_\_\_5a\_\_\_

Title of the Program

1. ARM Assembly Code for each program
2. Output Screen Shot

Problem statement:

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.

Code:

mov r0, #2

mov r1, #3

mov r2, r0

mov r3, r1

l1:

    cmp r3, r2

    beq l2

    cmp r3, r2

    bmi negative

    sub r3, r3, r2

    b l1

l2:

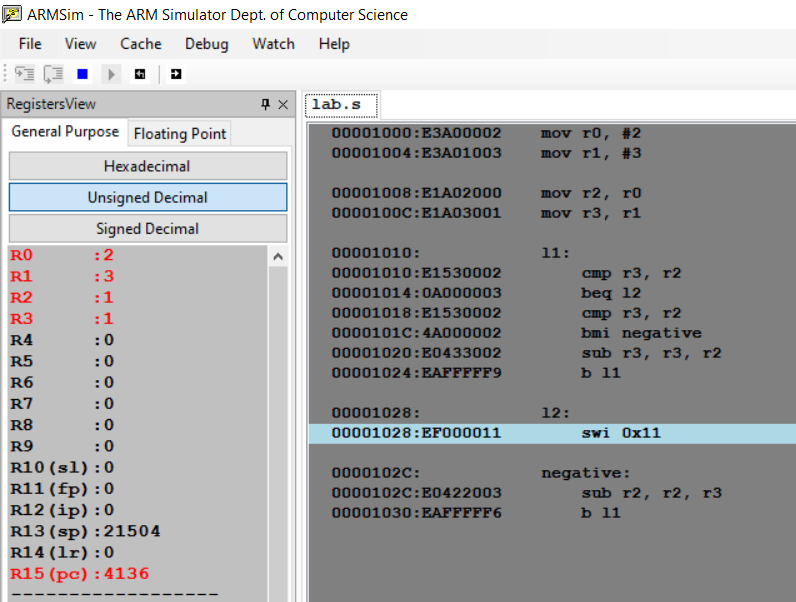
    swi 0x11

negative:

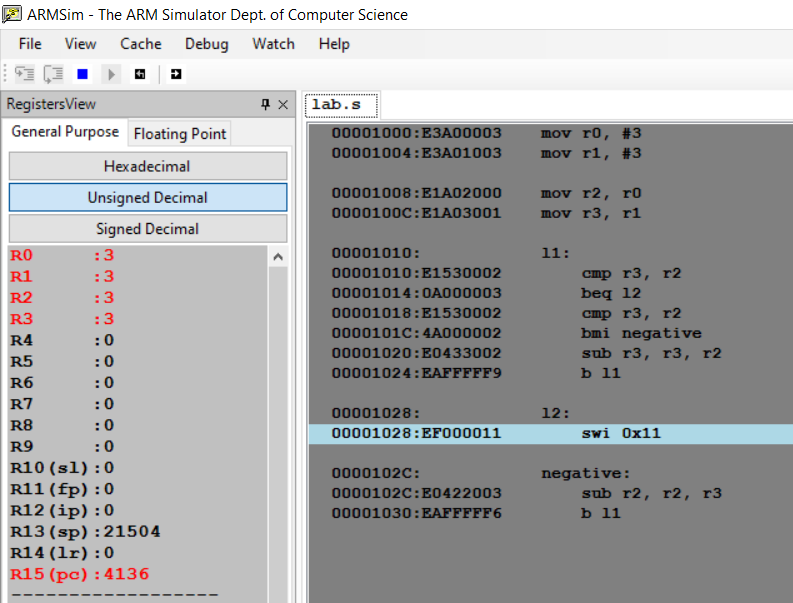
    sub r2, r2, r3

    b l1

Case: A < B



Case: A = B



**Microprocessor and Computer Architecture Laboratory**

**UE19CS256**

**4th Semester, Academic Year 2020-21**

Date: 02/02/2021

|  |  |  |
| --- | --- | --- |
| Name: Pranav R. Hegde | SRN: PES1UG19CS343 | Section: F |

Week#\_\_\_\_2\_\_\_\_\_\_\_ Program Number: \_\_\_5b\_\_\_

Title of the Program

1. ARM Assembly Code for each program
2. Output Screen Shot

Problem statement:

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

LDR r0, =A

LDR r1, =B

LDR r5, =C

LDR r2, [r0]

LDR r3, [r1]

l1:

    cmp r2, r3

    beq l2

    cmp r2, r3

    bmi negative

    sub r2, r2, r3

    B l1

l2:

    str r3, [r5]

    swi 0x11

    negative:

        sub r3, r3, r2

        B l1

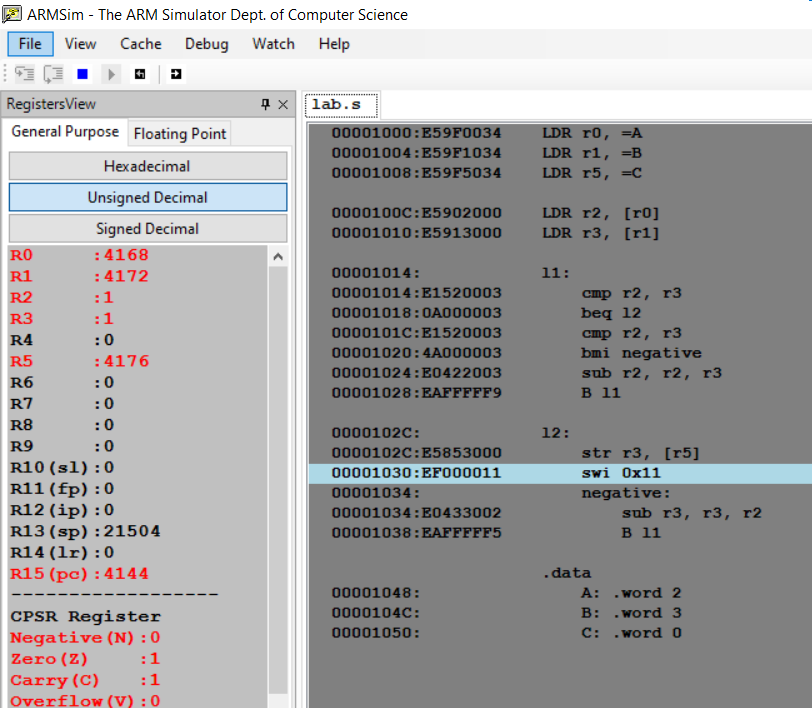
.data

    A: .word 2

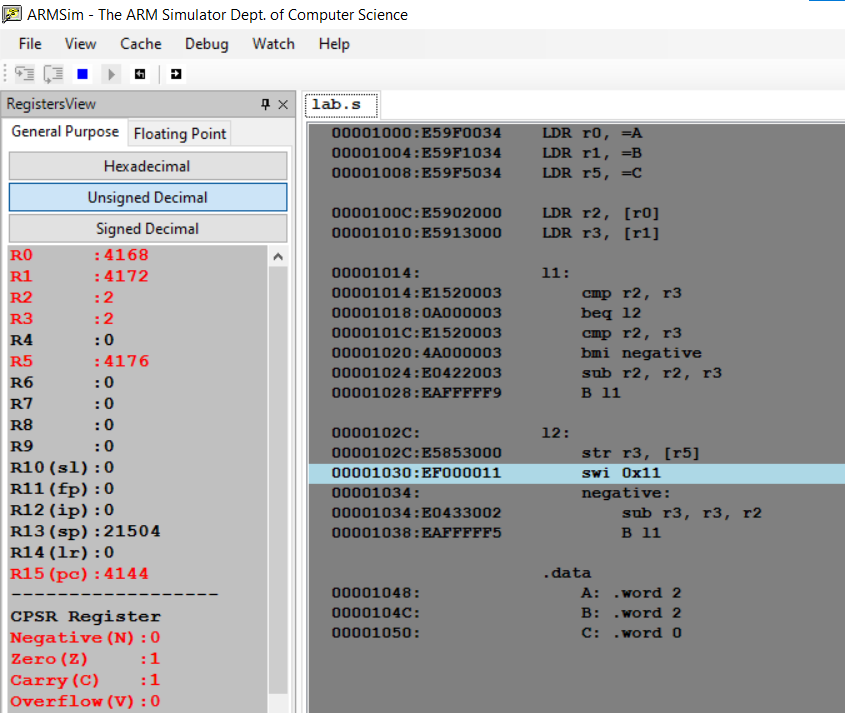
    B: .word 3

    C: .word 0

Case1: A > B



Case: A=B



**Microprocessor and Computer Architecture Laboratory**

**UE19CS256**

**4th Semester, Academic Year 2020-21**

Date: 02/02/2021

|  |  |  |
| --- | --- | --- |
| Name: Pranav R. Hegde | SRN: PES1UG19CS343 | Section: F |

Week#\_\_\_\_2\_\_\_\_\_\_\_ Program Number: \_\_\_6a\_\_\_

Title of the Program

1. ARM Assembly Code for each program
2. Output Screen Shot

Problem statement:

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

Code:

.text

MOV R0,#5

LDR R1,=array

loop:

      LDR R2,[R1],#4

      ADD R3,R3,R2

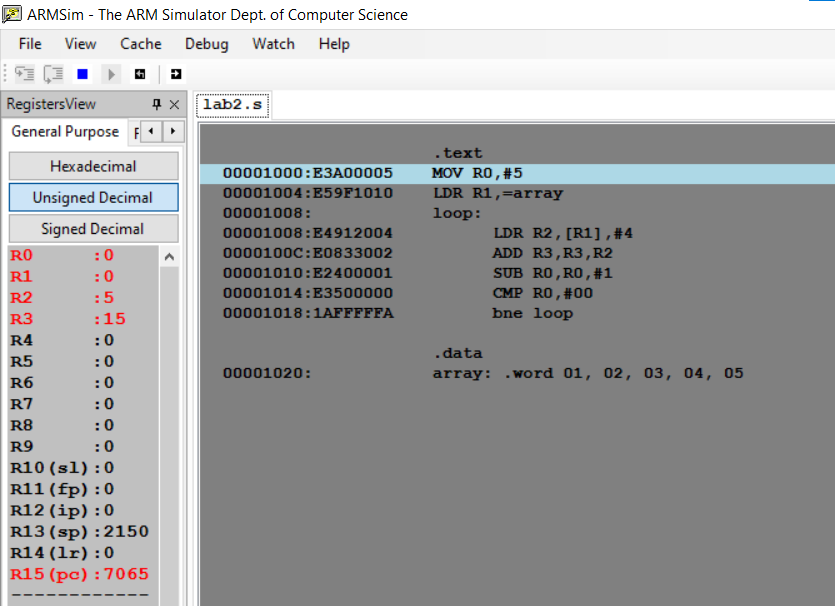
      SUB R0,R0,#1

      CMP R0,#00

      bne loop

.data

array: .word 01, 02, 03, 04, 05



**Microprocessor and Computer Architecture Laboratory**

**UE19CS256**

**4th Semester, Academic Year 2020-21**

Date: 02/02/2021

|  |  |  |
| --- | --- | --- |
| Name: Pranav R. Hegde | SRN: PES1UG19CS343 | Section: F |

Week#\_\_\_\_2\_\_\_\_\_\_\_ Program Number: \_\_\_6b\_\_\_

Title of the Program

1. ARM Assembly Code for each program
2. Output Screen Shot

Problem statement:

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)

Code:

.text

MOV R0,#10

LDR R1,=array

loop:

      LDR R2,[R1],#9

      ADD R3,R3,R2

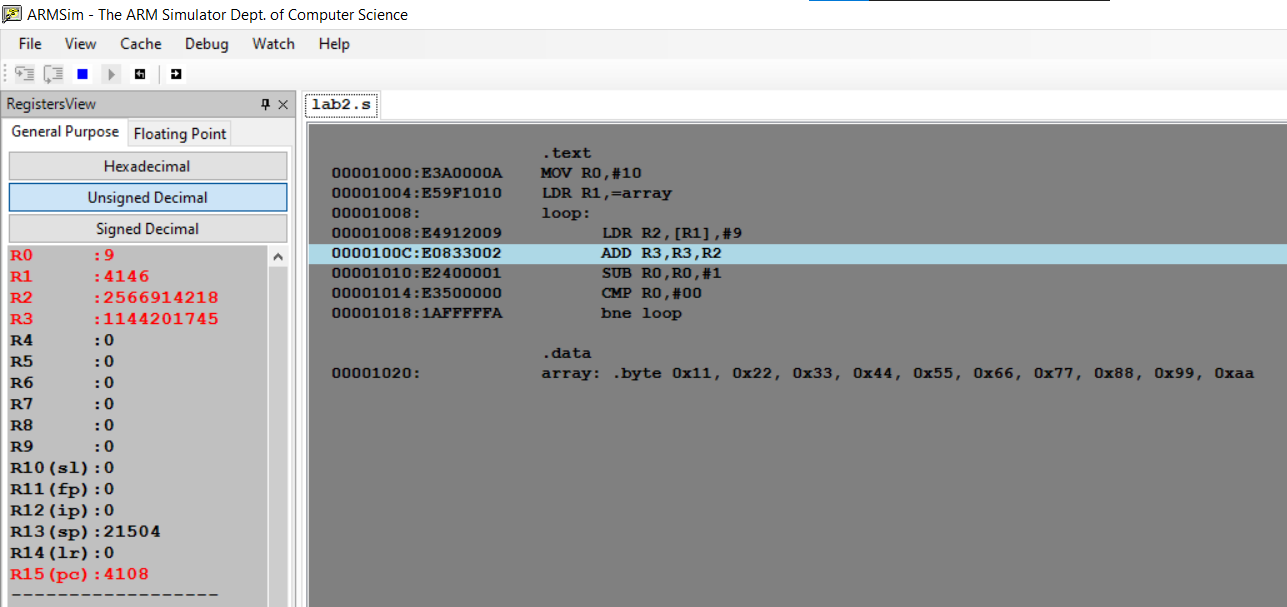
      SUB R0,R0,#1

      CMP R0,#00

      bne loop

.data

array: .byte 0x11, 0x22, 0x33, 0x44, 0x55, 0x66, 0x77, 0x88, 0x99, 0xaa



**Microprocessor and Computer Architecture Laboratory**

**UE19CS256**

**4th Semester, Academic Year 2020-21**

Date: 02/02/2021

|  |  |  |
| --- | --- | --- |
| Name: Pranav R. Hegde | SRN: PES1UG19CS343 | Section: F |

Week#\_\_\_\_2\_\_\_\_\_\_\_ Program Number: \_\_\_7\_\_\_

Title of the Program

1. ARM Assembly Code for each program
2. Output Screen Shot

Problem statement:

7. Write an ALP to multiply using barrel shifter.

35\*R0

Code:

.text

ldr r0, =A

ldr r1, =B

ldr r3, [r0]

mov r4, r3, lsl#3

sub r4, r4, r3

mov r2, r4, lsl#2

add r2, r2, r4

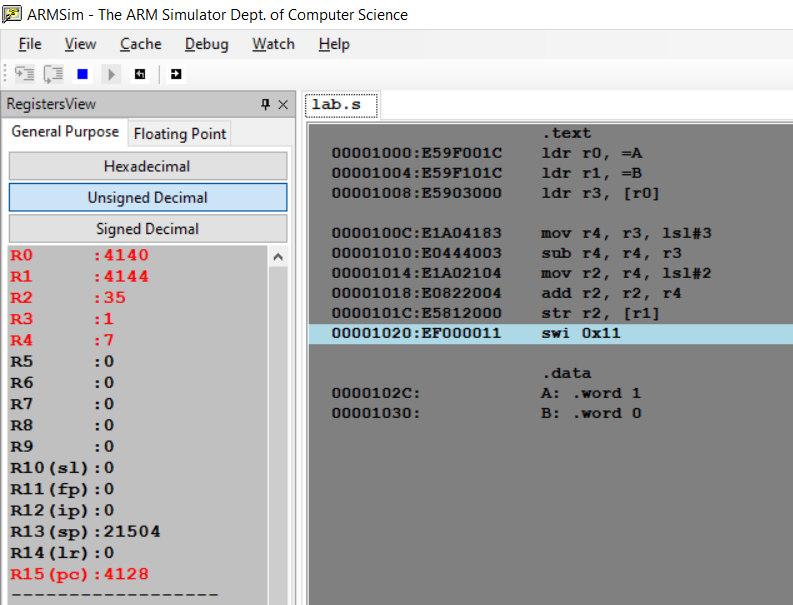
str r2, [r1]

swi 0x11

.data

A: .word 1

B: .word 0



**Microprocessor and Computer Architecture Laboratory**

**UE19CS256**

**4th Semester, Academic Year 2020-21**

Date: 02/02/2021

|  |  |  |
| --- | --- | --- |
| Name: Pranav R. Hegde | SRN: PES1UG19CS343 | Section: F |

Week#\_\_\_\_2\_\_\_\_\_\_\_ Program Number: \_\_\_8\_\_\_

Title of the Program

1. ARM Assembly Code for each program
2. Output Screen Shot

Problem statement:

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

\* Use LSL instruction for multiplication

Code:

.text

ldr r0, =A

ldr r1, =B

ldr r2, =C

ldr r3, [r0]

ldr r4, [r1]

add r5, r3, r4

mov r6, r4, lsl#1

add r6, r6, r4

add r7, r6, r5

str r7, [r2]

swi 0x11

.data

A: .word 2

B: .word 1

C: .word 0

