**Microprocessor and Computer Architecture**

**UE21CS251B**

**4th Semester, Academic Year 2022-23**

Date:1/03/2023



Name: YAMAN GUPTA SRN:PES2UG21CS619 j



Week#\_\_\_\_4\_\_\_\_\_\_\_ Program Number: \_\_\_\_1\_\_

Title of the Program

**1.Write an ALP to add two 64 bit numbers loaded from memory and store the result in memory.**

I.ARM Assembly Code

**.data**

1. **.word 0x0,0x80000000**
2. **.word 0x0,0x80000000 sum: .word 0**

**.text**

**LDR r0,=a**

**LDR r1,=b**

**LDR r2,=sum**

**LDR r3,[r0,#4]**

**LDR r4,[r1,#4]**

**ADDS r3,r3,r4**

**STR r3,[r2,#4]**

**LDR r3,[r0]**

**LDR r4,[r1]**

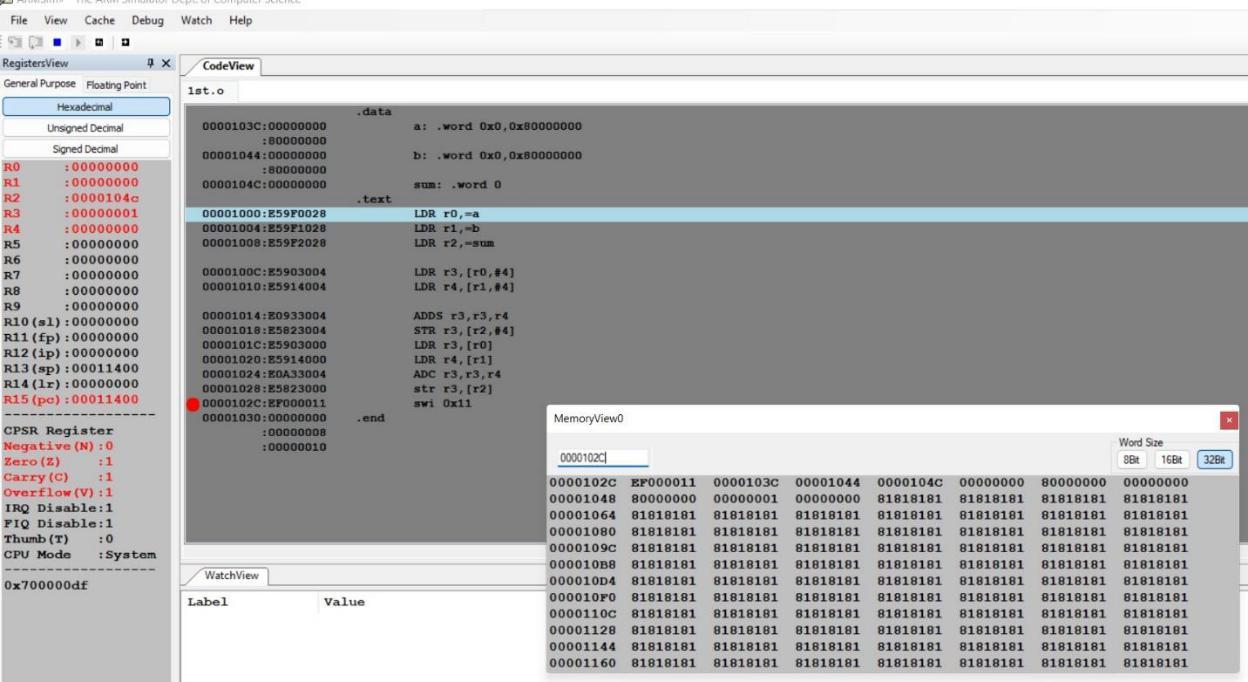
**ADC r3,r3,r4**

**str r3,[r2]**

**swi 0x11**

**.end**

II. Output Screen Shot (One)



**Microprocessor and Computer Architecture**

**UE21CS251B**

**4th Semester, Academic Year 2022-23**

Date:1/03/2023



Name:yaman gupta SRN:PES2UG21CS619 J



Week#\_\_\_\_4\_\_\_\_\_\_\_ Program Number: \_\_\_\_2\_\_\_

Title of the Program

**2.Write an ALP to find 1’s and 2’s complement of a 32 bit number**

I.ARM Assembly Code

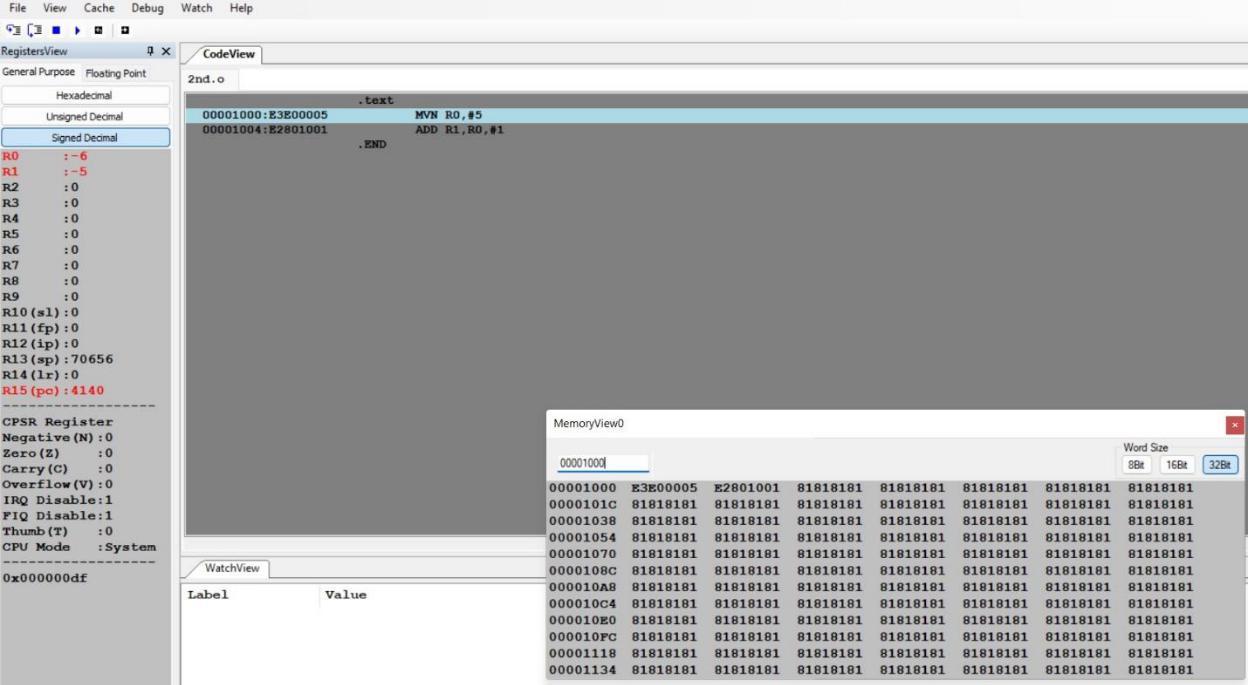
**.text**

**MVN R0,#5**

**ADD R1,R0,#1**

**.END**

II. Output Screen Shot (One)



**Microprocessor and Computer Architecture**

**UE21CS251B**

**4th Semester, Academic Year 2022-23**

Date:1/3/2023



Name:Yaman Gupta SRN:PES2UG21CS619 Section j



Week#\_\_\_\_4\_\_\_\_\_\_\_ Program Number: \_\_\_\_3 \_\_

Title of the Program

**3.Write an ALP to scan a 32 bit number if it is negative or positive**

I.ARM Assembly Code

**.data**

**a: .word -17**

**neg: .asciz "negative\n"**

**pos: .asciz "positive\n"**

**.text**

**LDR r0,=a**

**LDR r1,[r0]**

**ADDS r1,r1,#0**

**BMI negative**

**LDR r0,=pos**

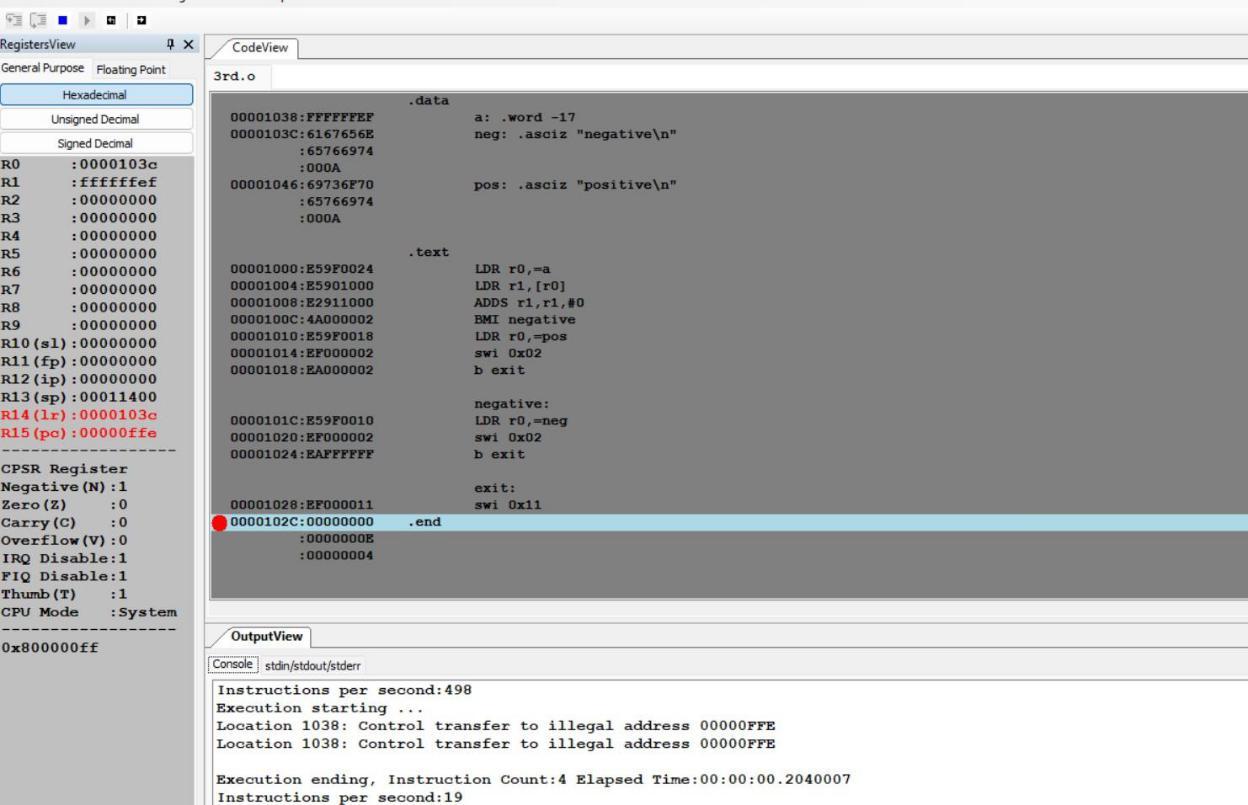
**swi 0x02**

1. **exit negative: LDR r0,=neg swi 0x02**

**b exit exit: swi 0x11**

**.end**

II. Output Screen Shot (One)



**Microprocessor and Computer Architecture**

**UE21CS251B**

**4th Semester, Academic Year 2022-23**

Date:1/03/2023



Name: YAMAN GUPTA SRN:PES2UG21CS619 Section:J



Week#\_\_\_\_4\_\_\_\_\_\_\_ Program Number: \_\_\_\_4\_\_

Title of the Program

**4.Write an ALP to find the number of zeroes, positive and negative numbers in a given array**

I.ARM Assembly Code

**.data**

**arr: .word 0,5,0,-3,-5,-7**

**.text**

**ldr r0,=arr**

**mov r1,#0 @zero**

**mov r2,#0 @positive**

**mov r3,#0 @negative**

**mov r4,#6**

**loop:**

**ldr r5,[r0],#4**

**cmp r5,#0**

**beq zeroes**

**adds r6,r5,#0**

**bmi negative**

1. **positive here:**

**sub r4,r4,#1 cmp r4,#0 beq exit**

**b loop**

**zeroes:**

**add r1,r1,#1**

1. **here**

**positive:**

**add r2,r2,#1 b here**

**negative:**

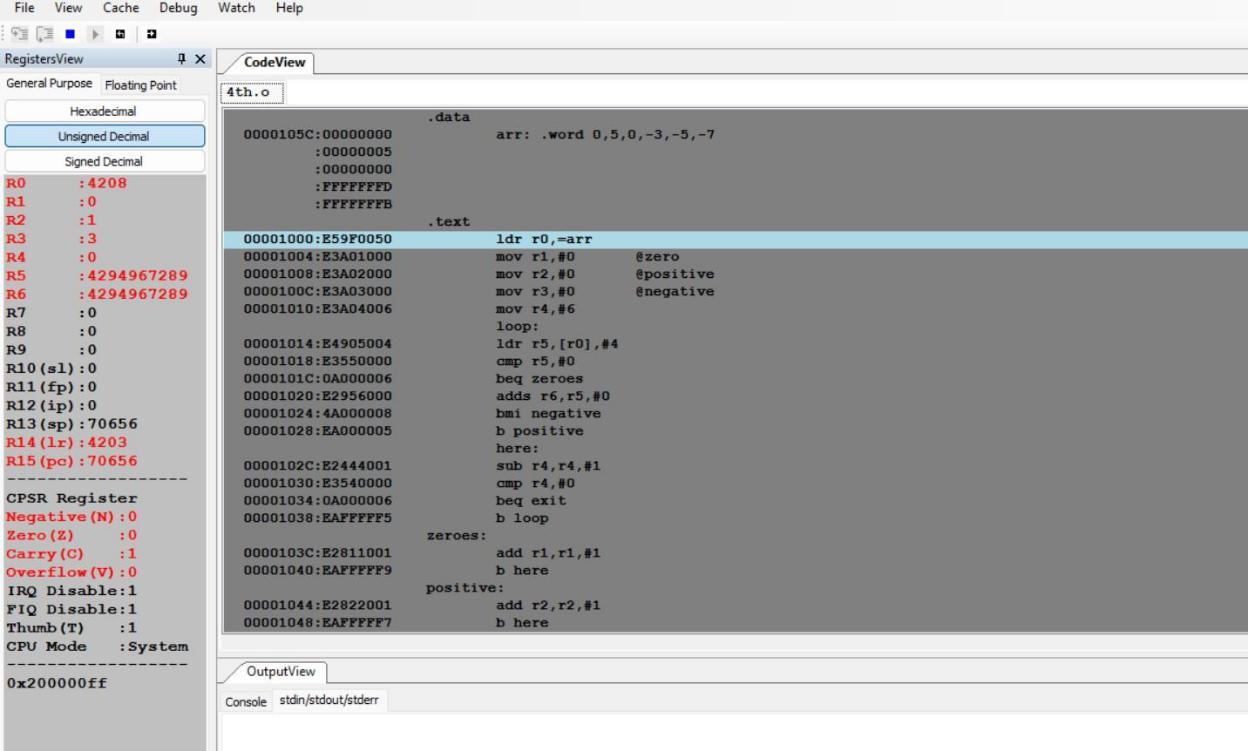
**add r3,r3,#1 b here**

**exit:**

**swi 0x11**

**.end**

II. Output Screen Shot (One)



**Microprocessor and Computer Architecture**

**UE21CS251B**

**4th Semester, Academic Year 2022-23**

Date:1/03/2023



Name: yaman gupta SRN:PES2UG21CS619 Section:J



Week#\_\_\_\_4\_\_\_\_\_\_\_ Program Number: \_\_\_\_5\_\_

Title of the Program

**5.Write an ALP to count the number of 1’s and 0’s in a given 32 bit number.**

I.ARM Assembly Code

**.data**

**a: .word 0xffff**

**.text**

**LDR r0,=a**

**ldr r4,[r0]**

**MOV r5,#0**

**mov r3,#32**

**loop:**

**and r2,r4,#1**

**cmp r2,#1**

**addeq r5,r5,#1**

**lsr r4,#1**

**sub r3,r3,#1**

**cmp r3,#0**

**beq exit**

**b loop**

**exit:**

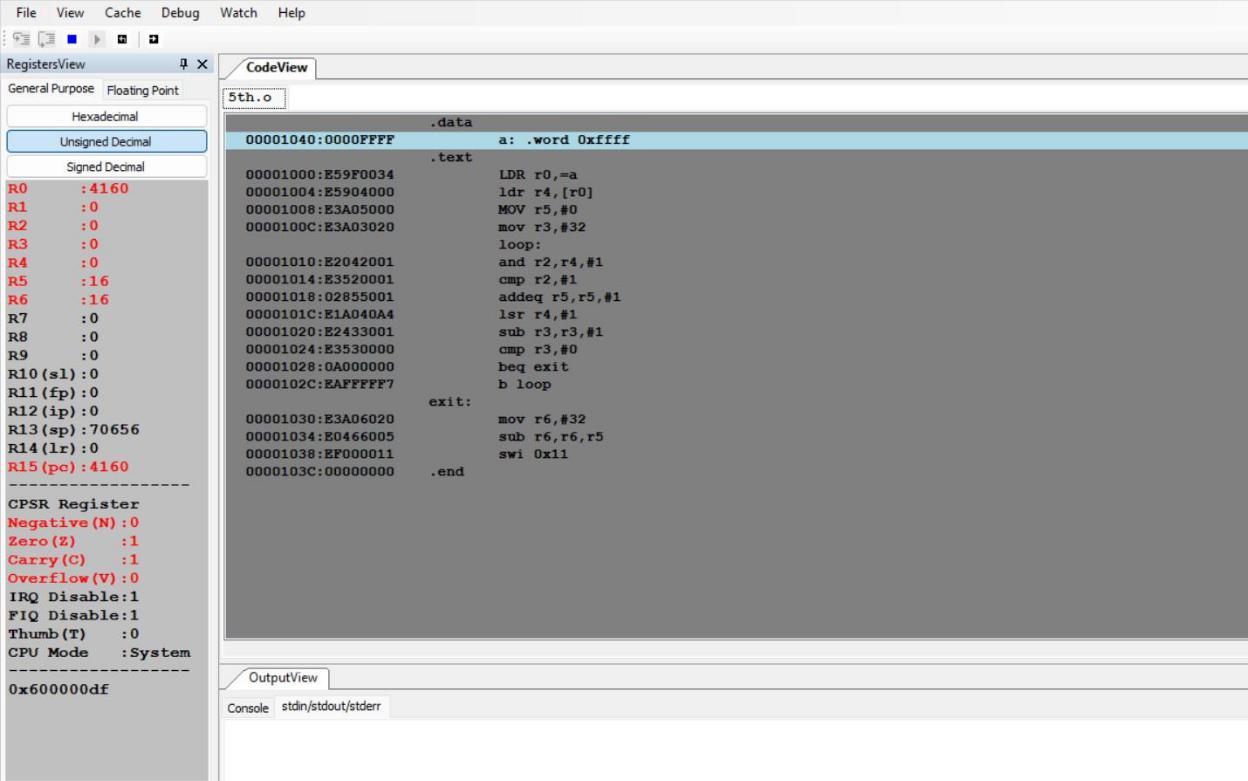
**mov r6,#32**

**sub r6,r6,r5**

**swi 0x11**

**.end**

II. Output Screen Shot (One)



**Microprocessor and Computer Architecture**

**UE21CS251B**

**4th Semester, Academic Year 2022-23**

Date:1/03/2023



Name: Yaman gupta SRN:PES2UG21CS619 J



Week#\_\_\_\_4\_\_\_\_\_\_\_ Program Number: \_\_\_\_6\_\_

Title of the Program

**6.Write an ALP to check the given number has odd or even number of 1’s and display the result. (Even Parity and Odd Parity)**

I.ARM Assembly Code

**.data**

**a: .word 0x4**

1. **.asciz "even\n"**
2. **.asciz "odd\n"**

**.text**

**LDR r0,=a**

**ldr r4,[r0]**

**MOV r5,#0**

**mov r3,#32**

**loop:**

**and r2,r4,#1**

**cmp r2,#1**

**addeq r5,r5,#1**

**lsr r4,#1**

**sub r3,r3,#1**

**cmp r3,#0**

**beq print**

**b loop**

**print:**

**and r6,r5,#1**

**cmp r6,#1**

**beq odd**

**ldr r0,=e**

**swi 0x02**

**b exit**

**odd:**

**ldr r0,=o**

**swi 0x02**

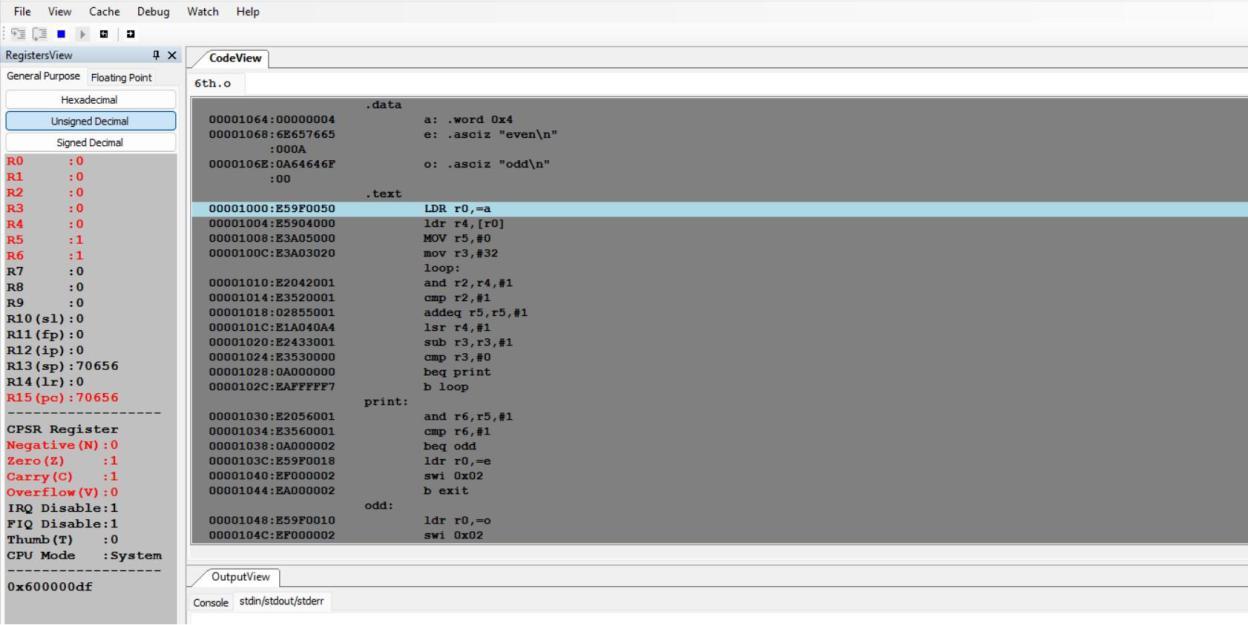
**b exit**

**exit:**

**swi 0x11**

**.end**

II. Output Screen Shot (One)



**Disclaimer:**

* The programs and output submitted is duly written, verified and executed by me.
* I have not copied from any of my peers nor from the external resource such as internet.
* If found plagiarized, I will abide with the disciplinary action of the University.

Signature: Yaman

Name:YamanGupta

SRN:PES2UG21CS619

Section:J

Date:1/03/2023