Microprocessor and Computer Architecture

UE21CS251B

4th Semester, Academic Year 2022-23

Date:

Name: Nihal T M	SRN:PES2UG21CS333	Section: F

Week#____4_____ Program Number: ____1__

Title of the Program

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

I. ARM Assembly Code

Code:

```
@ ALP program to add two 64-bit numbers

.data
    a:.word 12312734,77823438
    b:.word 24629982,56436738
    c:.word 0,0
```

```
.text
        ldr r1,=a
        ldr r2,=b
        ldr r3,=c
        ldr r4,[r1],#4
        ldr r5,[r2],#4
        add r6,r4,r5
        str r6,[r3],#4
        ldr r4,[r1],#4
        ldr r5,[r2],#4
        add r6,r4,r5
        str r6,[r3],#4
        swi 0x11
.end
```

II. Output Screen Shot (One)

General Purpose | Floatin

Hexadecimal
Unsigned Decimal
Signed Decimal

```
R0      :00000000
R1      :000011a4
R2      :000011ac
R3      :000011b4
R4      :04a37dce
R5      :035d2802
R6      :0800a5d0
R7      :00000000
R8      :00000000
R9      :00000000
R10(sl):00000000
R11(fp):00000000
R12(ip):00000000
R13(sp):00005400
R14(lr):00000000
R15(pc):0000102c
----------------
CPSR Register
Negative(N):0
```

1.s | 2.s | 3.s | 4.s | 5.s | 6.s

```
@ ALP program to add two 64-bit numbers

.data
0000119C:            a:.word 12312734,77823438
000011A4:            b:.word 24629982,56436738
000011AC:            c:.word 0,0

.text
00001000:E59F1028    ldr r1,=a
00001004:E59F2028    ldr r2,=b
00001008:E59F3028    ldr r3,=c
0000100C:E4914004    ldr r4,[r1],#4
00001010:E4925004    ldr r5,[r2],#4
00001014:E0846005    add r6,r4,r5
00001018:E4836004    str r6,[r3],#4
0000101C:E4914004    ldr r4,[r1],#4
00001020:E4925004    ldr r5,[r2],#4
00001024:E0846005    add r6,r4,r5
00001028:E4836004    str r6,[r3],#4
0000102C:EF000011    swi 0x11
.end
```

OutputView | WatchView | MemoryView0

0000119c

```
0000119C  00BBE09E  04A37DCE  0177D2DE  035D2802  0233B37C  0800A5D0
000011D4  61762065  2065756C  6E207369  74616765  00657669  20656854
```

Week#____4_____ Program Number: ____2___

Title of the Program

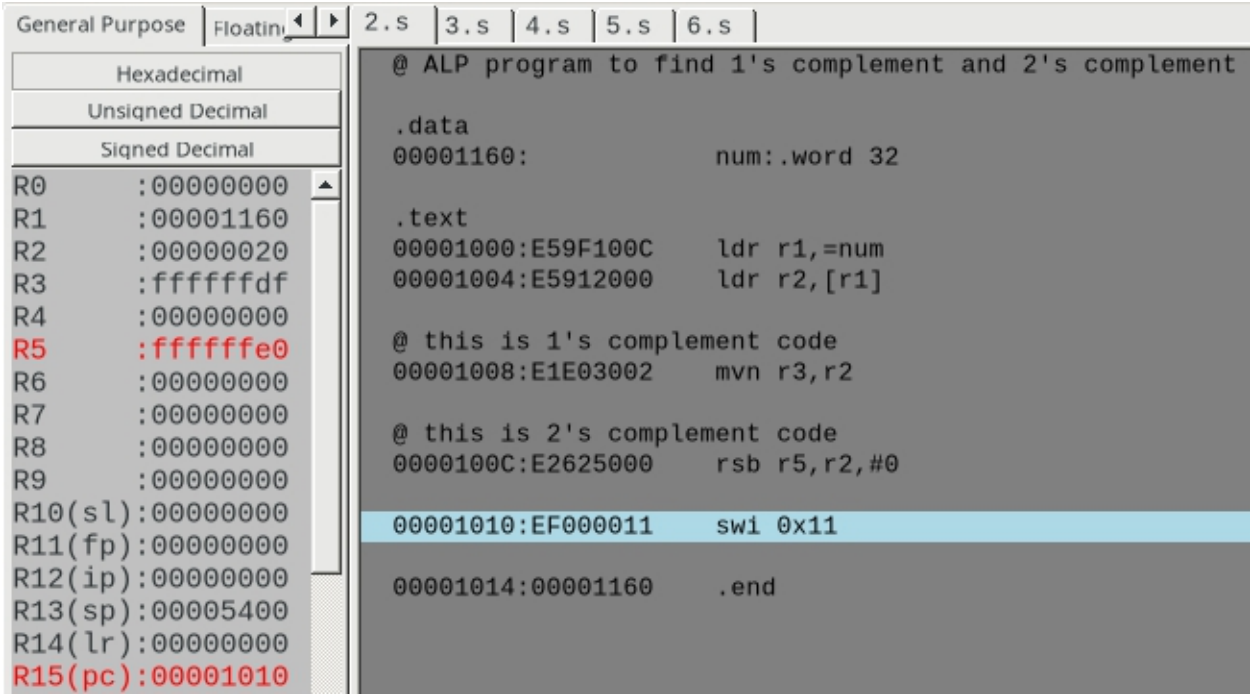
Write an ALP to find 1’s and 2’s complement of a 32 bit number

I. ARM Assembly Code

Code:

```
@ ALP program to find 1's complement and 2's complement of a 32-bit number
.data
	num:.word 32

.text
	ldr r1,=num
	ldr r2,[r1]

	@ this is 1's complement code
	mvn r3,r2

	@ this is 2's complement code
	rsb r5,r2,#0
	swi 0x11
.end
```

II. Output Screen Shot (One)



Week#____4_____ Program Number: ____3 __

Title of the Program

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

I. ARM Assembly Code

Code:

```
@ ALP program to scan a number and find if it is positive or negative


.data
        num:.word 0
        pos:.asciz "The value is positive"
        neg:.asciz "The value is negative"
        zer:.asciz "The value is zero"


.text
        ldr r1,=num
        ldr r2,[r1]
        cmp r2,#0
        beq zero
        bpl positive
```

```
        bmi negative

        positive:
                ldr r0,=pos
                swi 0x02
                b end

        negative:
                ldr r0,=neg
                swi 0x02
                b end

        zero:
                ldr r0,=zer
                swi 0x02
                b end

        end: swi 0x11

.end
```

II. Output Screen Shot (One)

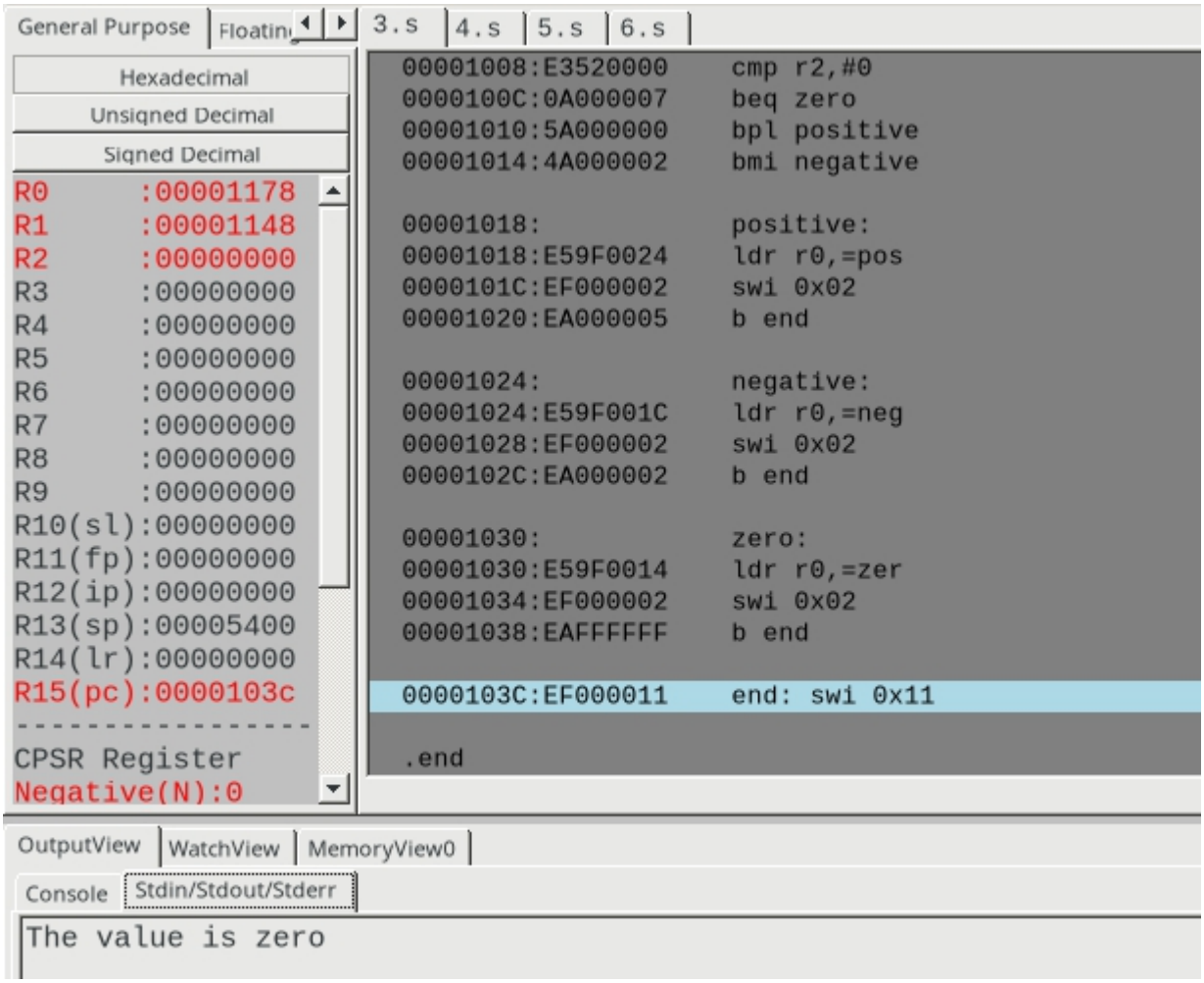
(i) when the value is positive:

```
General Purpose | Floatin | 3.s | 4.s | 5.s | 6.s |
Hexadecimal
Unsigned Decimal
Signed Decimal
R0      :0000114c
R1      :00001148
R2      :0000000a
R3      :00000000
R4      :00000000
R5      :00000000
R6      :00000000
R7      :00000000
R8      :00000000
R9      :00000000
R10(sl):00000000
R11(fp):00000000
R12(ip):00000000
R13(sp):00005400
R14(lr):00000000
R15(pc):0000103c
-----------------
CPSR Register
Negative(N):0

00001008:E3520000    cmp r2,#0
0000100C:0A000007    beq zero
00001010:5A000000    bpl positive
00001014:4A000002    bmi negative

00001018:            positive:
00001018:E59F0024    ldr r0,=pos
0000101C:EF000002    swi 0x02
00001020:EA000005    b end

00001024:            negative:
00001024:E59F001C    ldr r0,=neg
00001028:EF000002    swi 0x02
0000102C:EA000002    b end

00001030:            zero:
00001030:E59F0014    ldr r0,=zer
00001034:EF000002    swi 0x02
00001038:EAFFFFFF    b end

0000103C:EF000011    end: swi 0x11

.end

OutputView | WatchView | MemoryView0 |
Console | Stdin/Stdout/Stderr
The value is positive
```

(ii) when value is negative:

```
General Purpose | Floatin | 3.s | 4.s | 5.s | 6.s |
Hexadecimal
Unsigned Decimal
Signed Decimal
R0      :00001162
R1      :00001148
R2      :fffffff6
R3      :00000000
R4      :00000000
R5      :00000000
R6      :00000000
R7      :00000000
R8      :00000000
R9      :00000000
R10(sl):00000000
R11(fp):00000000
R12(ip):00000000
R13(sp):00005400
R14(lr):00000000
R15(pc):0000103c
-----------------
CPSR Register
Negative(N):1

00001008:E3520000    cmp r2,#0
0000100C:0A000007    beq zero
00001010:5A000000    bpl positive
00001014:4A000002    bmi negative

00001018:            positive:
00001018:E59F0024    ldr r0,=pos
0000101C:EF000002    swi 0x02
00001020:EA000005    b end

00001024:            negative:
00001024:E59F001C    ldr r0,=neg
00001028:EF000002    swi 0x02
0000102C:EA000002    b end

00001030:            zero:
00001030:E59F0014    ldr r0,=zer
00001034:EF000002    swi 0x02
00001038:EAFFFFFF    b end

0000103C:EF000011    end: swi 0x11

.end

OutputView | WatchView | MemoryView0 |
Console | Stdin/Stdout/Stderr
The value is negative
```

(iii) when the value is zero:



Week#____4_____ Program Number: ____4__

Title of the Program

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

I. ARM Assembly Code

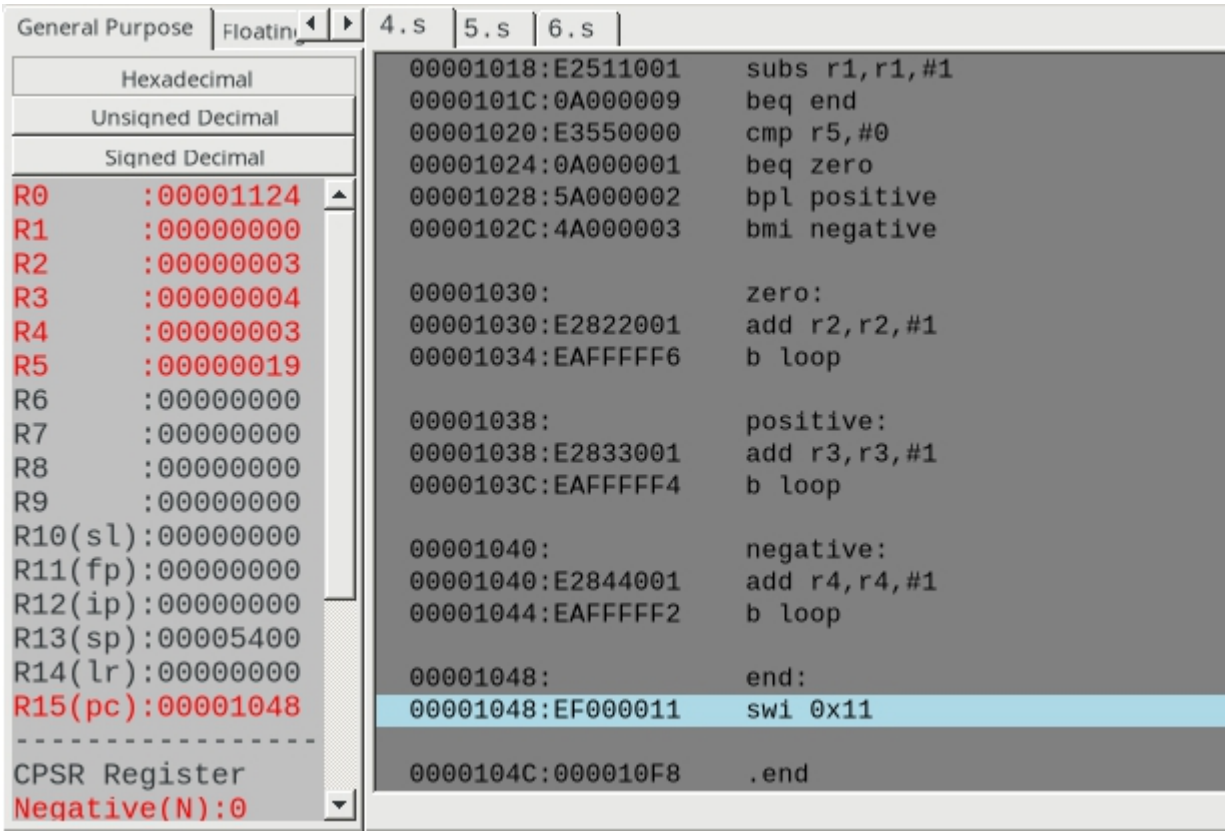
Code:

```
@ ALP to find the number of zeroes, positive numbers
and negative numbers in the array
.data
	arr:.word 0,5,-4,0,23,-4,32,66,-3,0


.text
	ldr r0,=arr
	mov r1,#11
	mov r2,#0	@ holds number of zeroes
	mov r3,#0	@ holds number of positive numbers
	mov r4,#0	@ holds number of negative numbers

	loop:
		ldr r5,[r0],#4
```

```
        subs r1,r1,#1
        beq end
        cmp r5,#0
        beq zero
        bpl positive
        bmi negative

    zero:
        add r2,r2,#1
        b loop

    positive:
        add r3,r3,#1
        b loop

    negative:
        add r4,r4,#1
        b loop

    end:
        swi 0x11

.end
```

II. Output Screen Shot (One)

here r2, r3, r4 holds the number of zeroes, positive numbers and negative numbers present in the arrys respectively:



Week#____4_____ Program Number: ____5__

Title of the Program

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

I. ARM Assembly Code

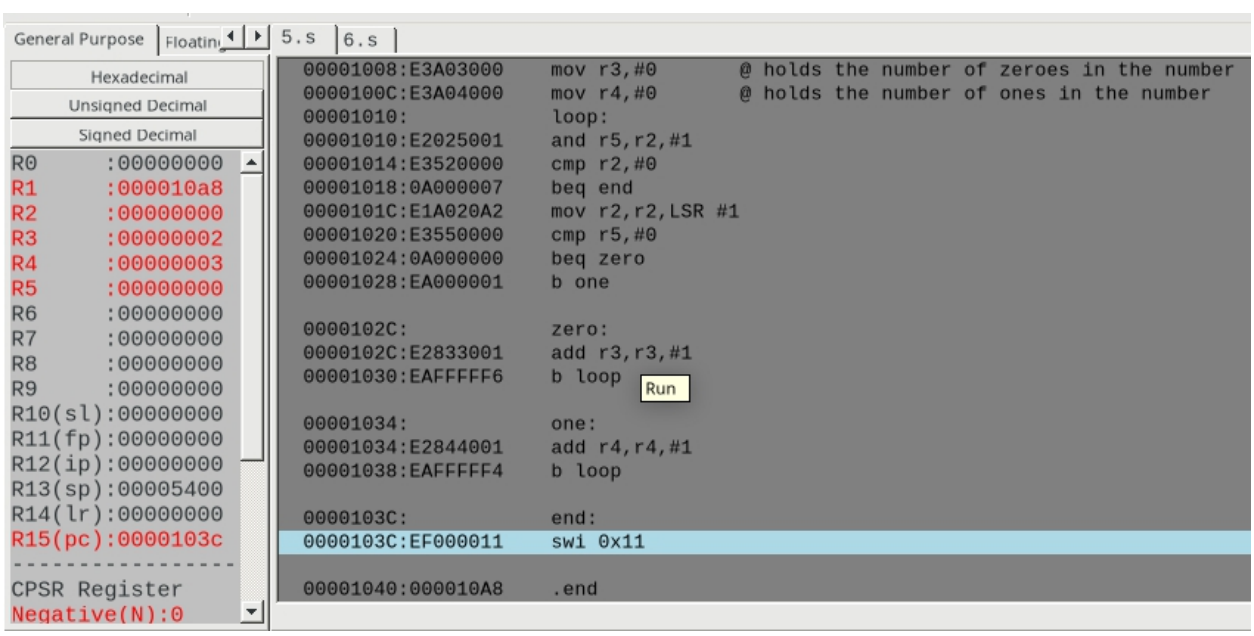
Code:

```
@ ALP program to count the number of zeroes and ones
in the binary number

.data

	num:.word 25

.text

	ldr r1,=num

	ldr r2,[r1]

	mov r3,#0	@ holds the number of zeroes in the no.

	mov r4,#0	@ holds the number of ones in the no.

	loop:and r5,r2,#1

		cmp r2,#0

		beq end

		mov r2,r2,LSR #1

		cmp r5,#0

		beq zero

		b one
```

```
zero:
        add r3,r3,#1
        b loop
one:
        add r4,r4,#1
        b loop
end:
        swi 0x11
.end
```

II. Output Screen Shot (One)

here r3 and r4 hold the number of zeroes and ones present in the binary number respectively.(excluding trailing left zeroes as they are not important)



Week#____4_____ Program Number: ____6__

Title of the Program

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

Code:

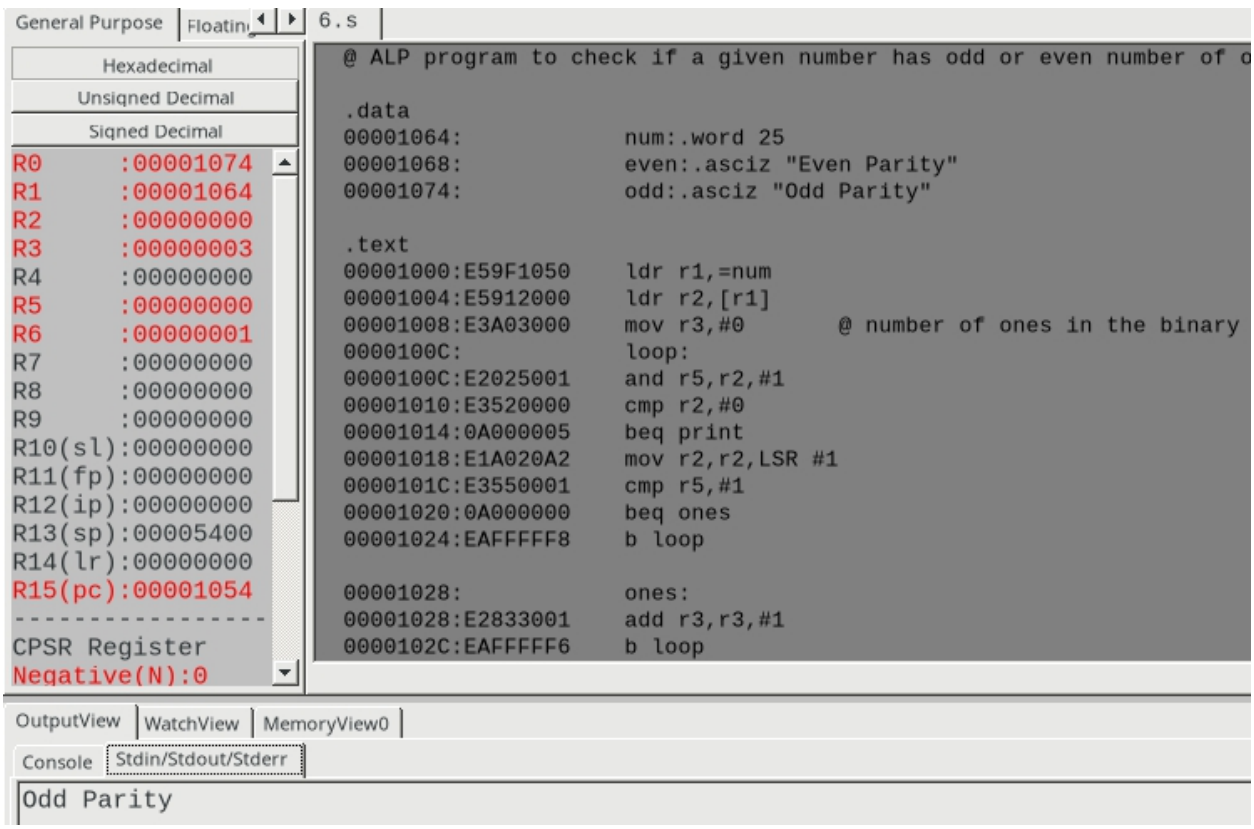
```
@ ALP program to check if a given number has odd or
even number of ones in binary


.data
	num:.word 25
	even:.asciz "Even Parity"
	odd:.asciz "Odd Parity"


.text
	ldr r1,=num
	ldr r2,[r1]
	mov r3,#0	@ number of ones in the binary format
of the given number
	loop:
```

```
        and r5,r2,#1
        cmp r2,#0
        beq print
        mov r2,r2,LSR #1
        cmp r5,#1
        beq ones
        b loop

ones:
        add r3,r3,#1
        b loop

print:
        ands r6,r3,#1
        beq is_even
        b is_odd

is_odd:
        ldr r0,=odd
        swi 0x02
        b end

is_even:
```

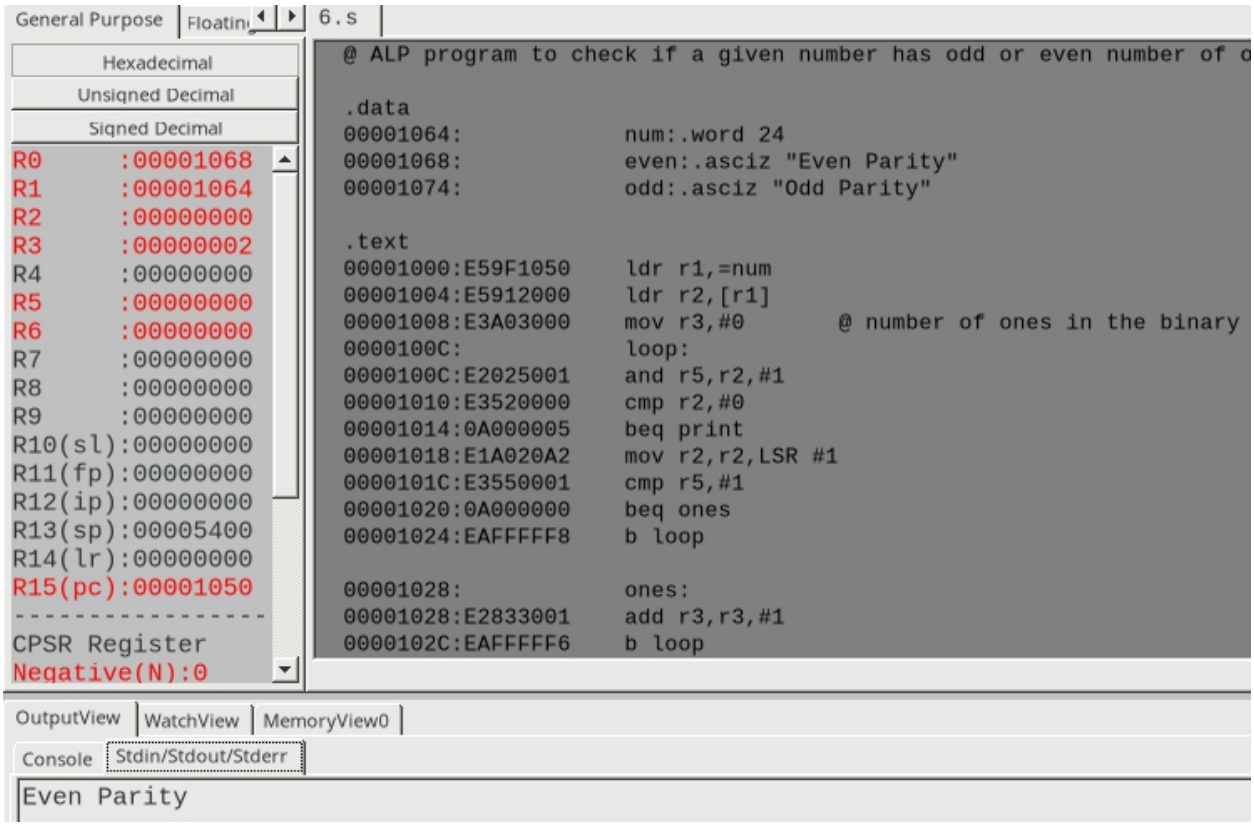
```
        ldr r0,=even
        swi 0x02
        swi 0x11

    end:
        swi 0x11
.end
```

II. Output Screen Shot (One)

(i) when it is odd parity:



(ii) when it is even parity:



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:
Name: Nihal T M
SRN: PES2UG21CS333
Section: F
Date: 11/02/2023