# Microprocessor and Computer Architecture Laboratory
# UE19CS256
## 4th Semester, Academic Year 2020-21

Date:02/01/2021

| Name: PRIYA MOHATA | SRN:PES2UG19CS301 | Section:E |
|---|---|---|

**Week#2.** **ProgramNumber:1**

**Title of the Program** : 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.

I. ARM Assembly Code for each program

TEST CASE 1

```
.text
MOV R0,#0
CMP R0,#0
BEQ label
BMI label2
MOV R1,#2
B exit
label:MOV R1,#1
B exit
label2:MOV R1,#3
exit:SWI 0x011
.end
```

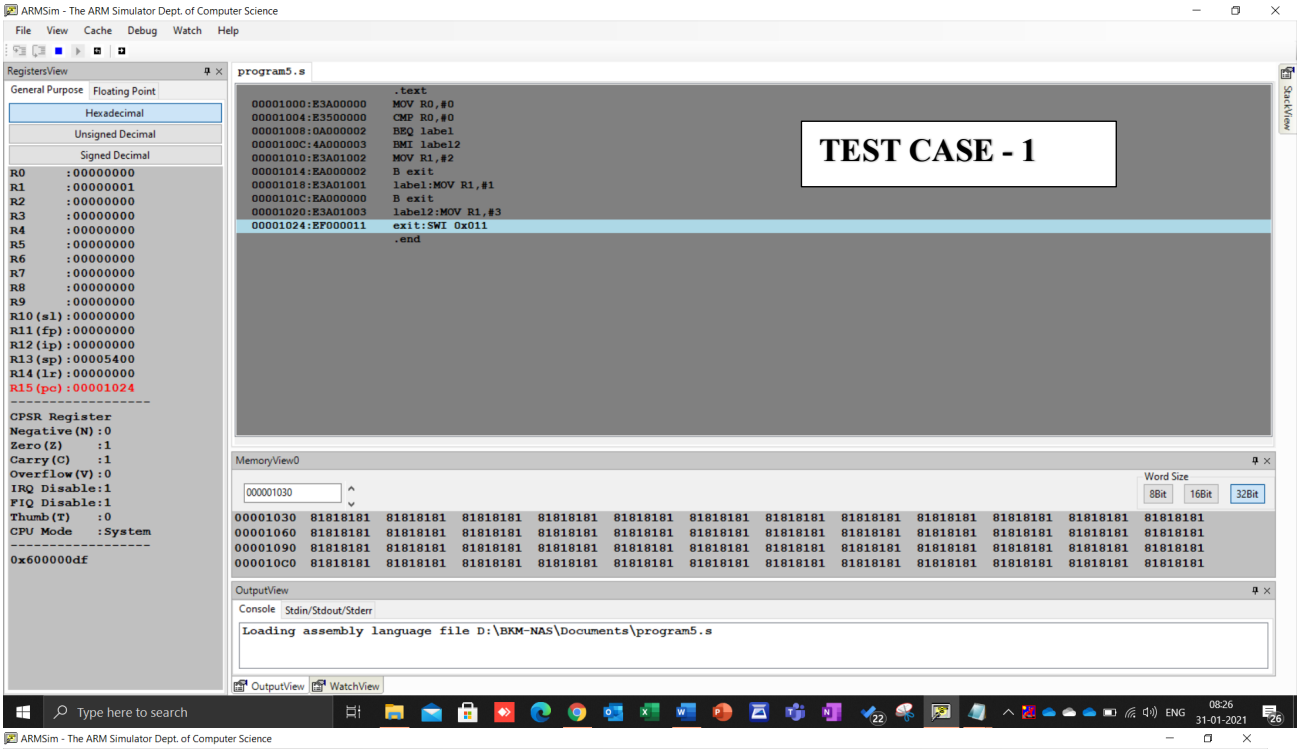
TEST CASE 2

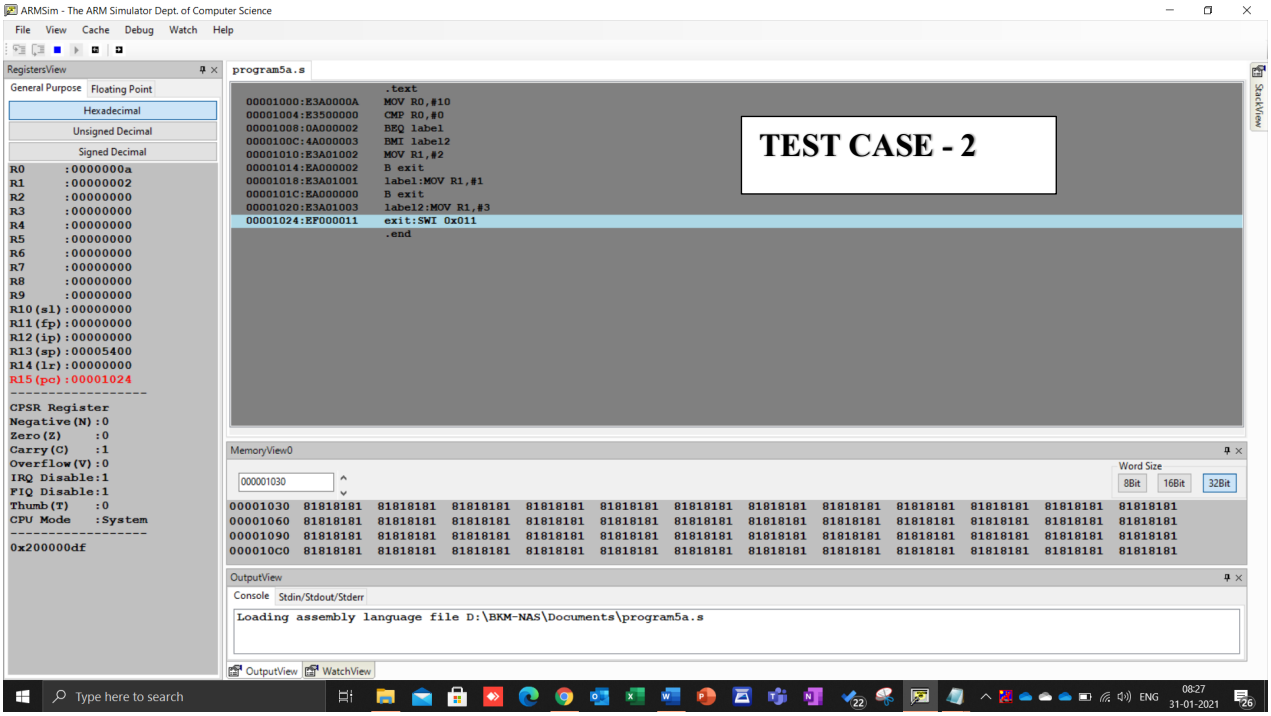
```
.text
MOV R0,#10
CMP R0,#0
BEQ label
BMI label2
MOV R1,#2
B exit
label:MOV R1,#1
B exit
label2:MOV R1,#3
exit:SWI 0x011
.end
```

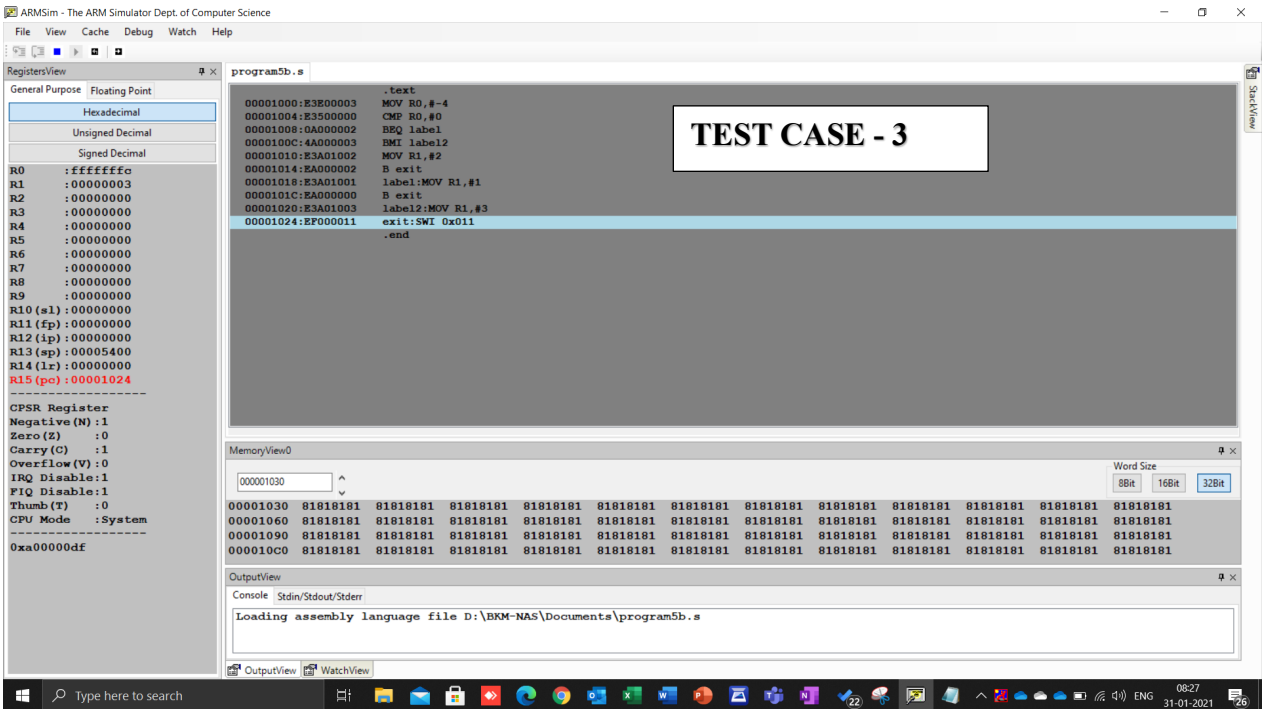
TEST CASE 3

```
.text
MOV R0,#-4
CMP R0,#0
BEQ label
BMI label2
MOV R1,#2
B exit
label:MOV R1,#1
B exit
label2:MOV R1,#3
exit:SWI 0x011
.end
```

# II.Output Screen Shot






## III.Output table for each program

| CASES | REGISTER | OPERATION | VALUE |
|---|---|---|---|
| CASE 1 | R0 | | 0x00000000 |
| | R1 | After compare | 1 |
| CASE 2 | R0 | | 0x0000000A |
| | R1 | After compare | 2 |
| CASE 3 | R0 | | 0xFFFFFFFC |
| | R1 | After compare | 3 |

**Week#2.** **Program Number:2**

**Title of the Program** : Write an ALP to compare the value of R0 and R1, add if R0 = R1, else subtract.

I. ARM Assembly Code for each program

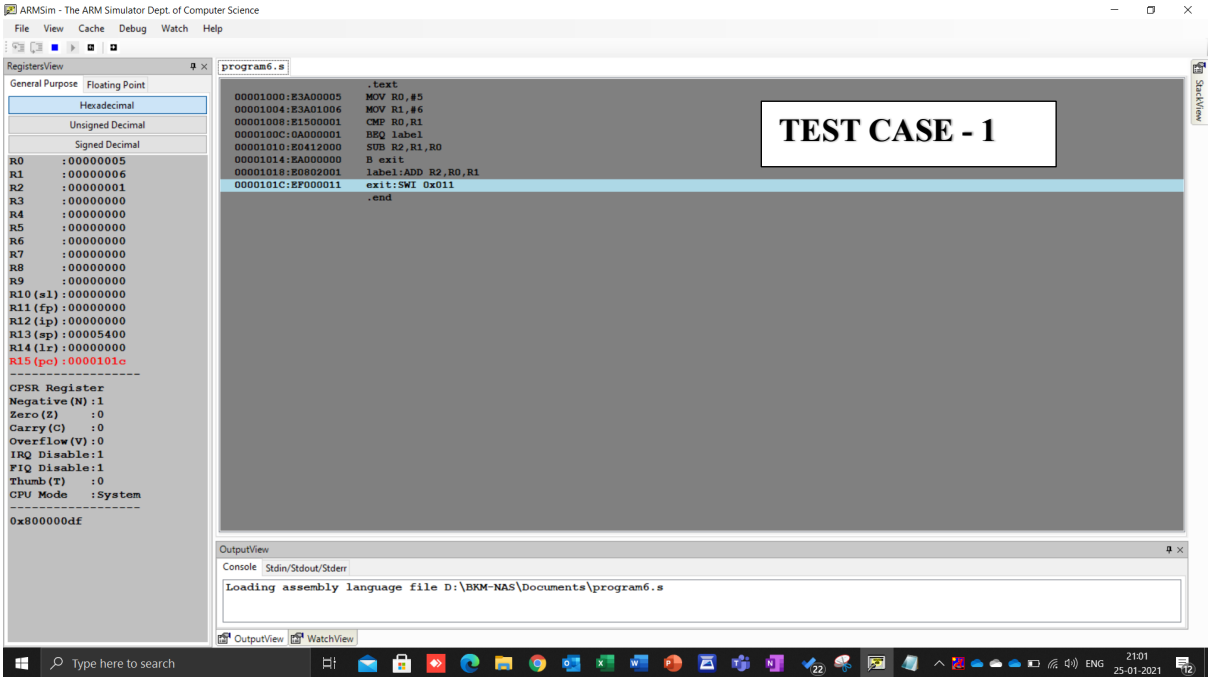
```
.text
MOV R0,#5
MOV R1,#6
CMP R0,R1
BEQ label
SUB R2,R1,R0
B exit
label:ADD R2,R0,R1
exit:SWI 0x011
.end
```

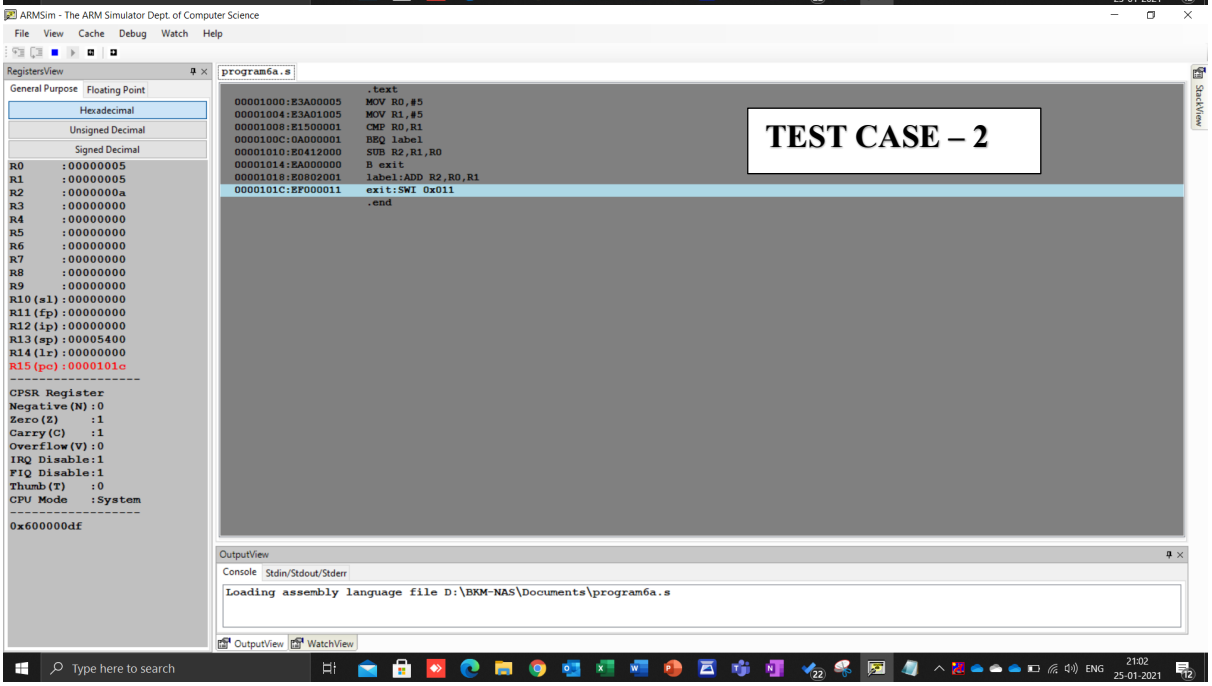
**TEST CASE - 1**

```
.text
MOV R0,#5
MOV R1,#5
CMP R0,R1
BEQ label
SUB R2,R1,R0
B exit
label:ADD R2,R0,R1
exit:SWI 0x011
.end
```

**TEST CASE - 2**

# II.Output Screen Shot



## III.OUTPUT TABLE

| CASES | |
|---|---|
| CASE 1 | R1=0x06, R0=0x05<br>R2=R1-R0=0x01 |
| | |
| CASE 2 | R1=0x05,R0=0x05<br>R2=R1+R0=0x0A |

**Week#2.** **Program Number:3**

**Title of the Program** : 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

I.ARM Assembly Code for each program

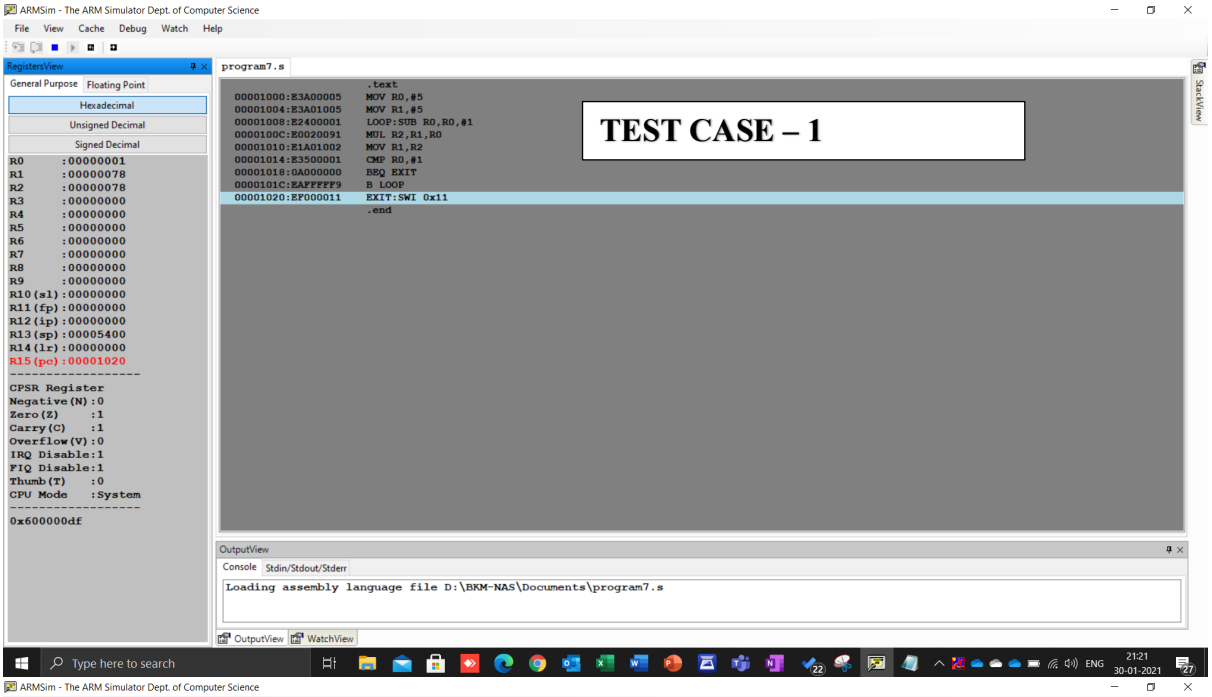
```
.text
MOV R0,#5
MOV R1,#5
LOOP:SUB R0,R0,#1
MUL R2,R1,R0
MOV R1,R2
CMP R0,#1
BEQ EXIT
B LOOP
EXIT:SWI 0x11
.end
```

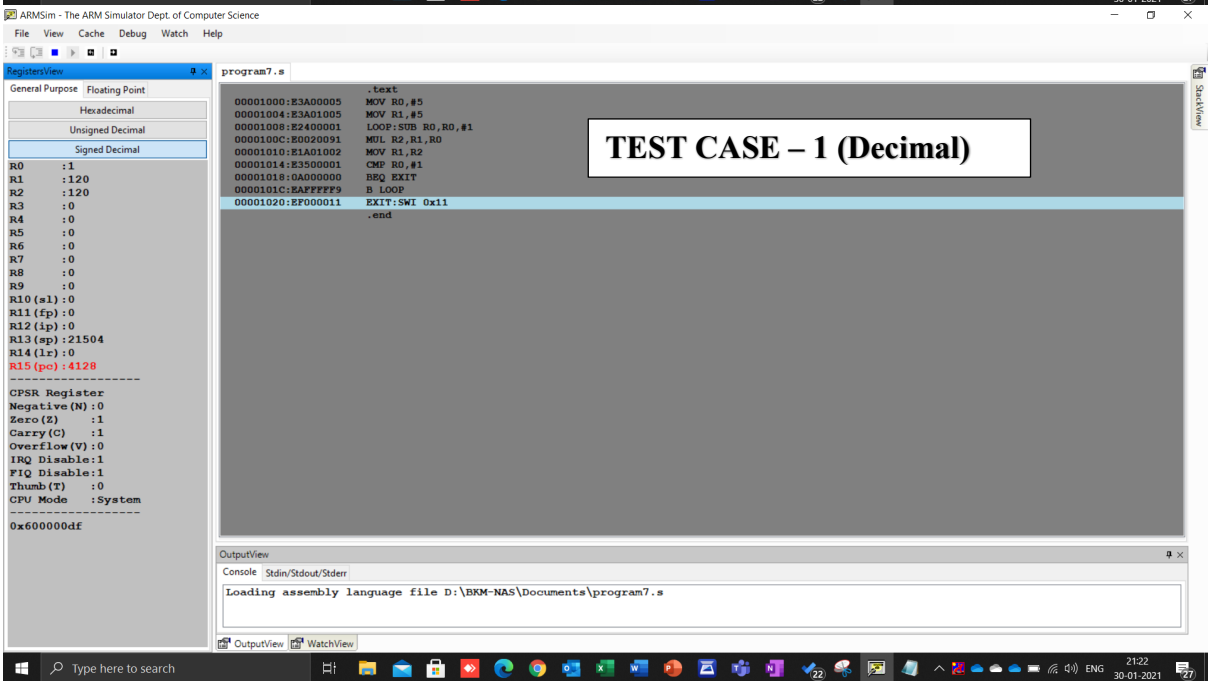
**TEST CASE – 1**

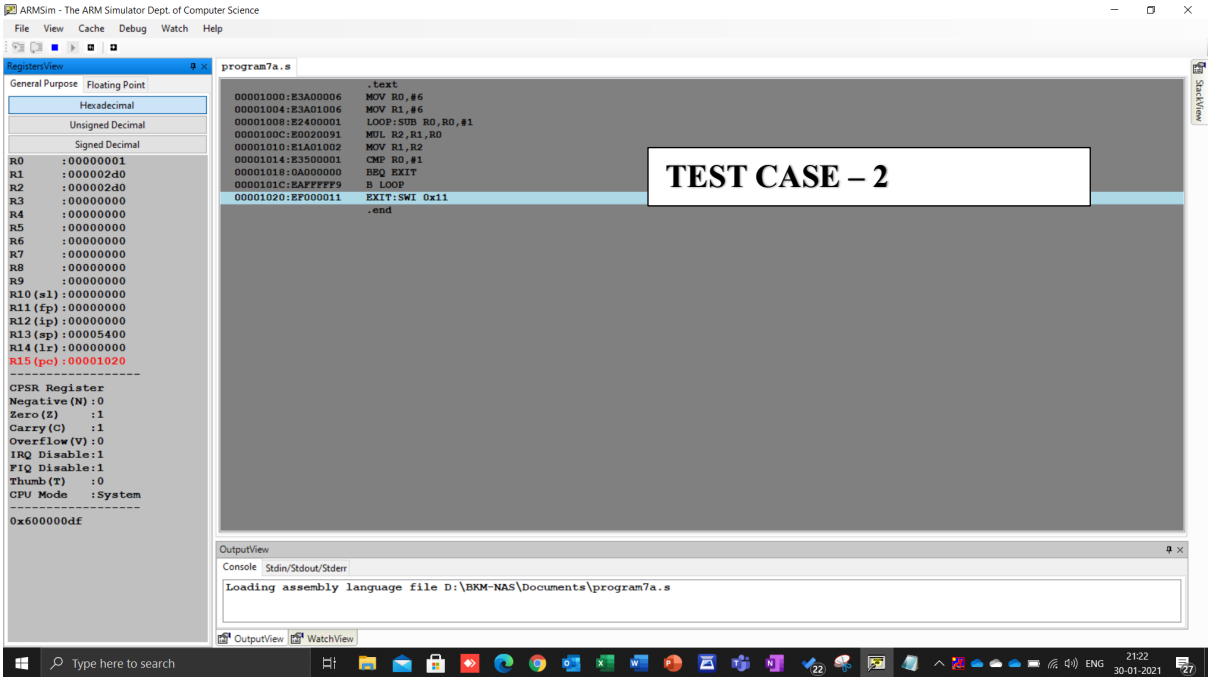
```
.text
MOV R0,#6
MOV R1,#6
LOOP:SUB R0,R0,#1
MUL R2,R1,R0
MOV R1,R2
CMP R0,#1
BEQ EXIT
B LOOP
EXIT:SWI 0x11
.end
```

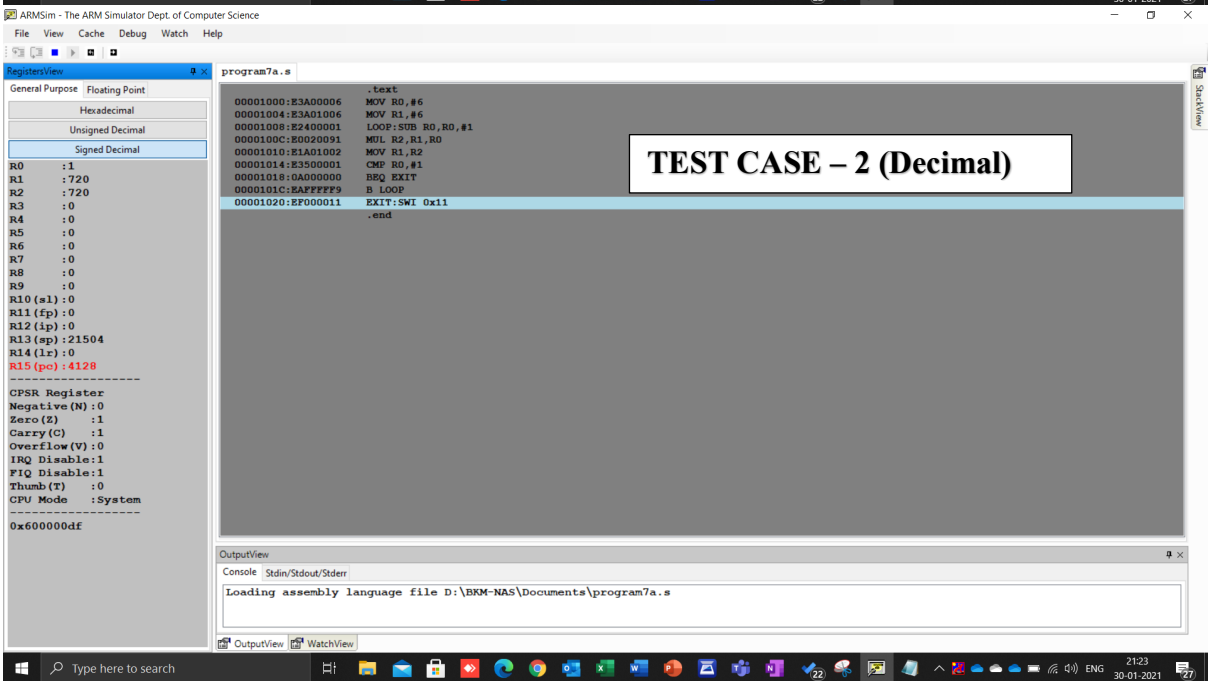
**TEST CASE – 2**

II.OUTPUT SCREENSHOTS


ARMSim - The ARM Simulator Dept. of Computer Science
File View Cache Debug Watch Help
RegistersView
General Purpose Floating Point
Hexadecimal
Unsigned Decimal
Signed Decimal
R0 :00000001
R1 :00000078
R2 :00000078
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):00001020
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 :System
0x600000df
program7.s
.text
00001000:E3A00005 MOV R0,#5
00001004:E3A01005 MOV R1,#5
00001008:E2400001 LOOP:SUB R0,R0,#1
0000100C:E0020091 MUL R2,R1,R0
00001010:E1A01002 MOV R1,R2
00001014:E3500001 CMP R0,#1
00001018:0A000000 BEQ EXIT
0000101C:EAFFFFF9 B LOOP
00001020:EF000011 EXIT:SWI 0x11
.end
TEST CASE – 1
OutputView
Console Stdin/Stdout/Stderr
Loading assembly language file D:\BKM-NAS\Documents\program7.s
OutputView WatchView
Type here to search
ENG
21:21
30-01-2021


ARMSim - The ARM Simulator Dept. of Computer Science
File View Cache Debug Watch Help
RegistersView
General Purpose Floating Point
Hexadecimal
Unsigned Decimal
Signed Decimal
R0 :1
R1 :120
R2 :120
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):21504
R14(lr):0
R15(pc):4128
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 :System
0x600000df
program7.s
.text
00001000:E3A00005 MOV R0,#5
00001004:E3A01005 MOV R1,#5
00001008:E2400001 LOOP:SUB R0,R0,#1
0000100C:E0020091 MUL R2,R1,R0
00001010:E1A01002 MOV R1,R2
00001014:E3500001 CMP R0,#1
00001018:0A000000 BEQ EXIT
0000101C:EAFFFFF9 B LOOP
00001020:EF000011 EXIT:SWI 0x11
.end
TEST CASE – 1 (Decimal)
OutputView
Console Stdin/Stdout/Stderr
Loading assembly language file D:\BKM-NAS\Documents\program7.s
OutputView WatchView
Type here to search
ENG
21:22
30-01-2021


ARMSim - The ARM Simulator Dept. of Computer Science
File View Cache Debug Watch Help
RegistersView
General Purpose Floating Point
Hexadecimal
Unsigned Decimal
Signed Decimal
R0 :00000001
R1 :000002d0
R2 :000002d0
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):00001020
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 :System
0x600000df
program7a.s
.text
00001000:E3A00006 MOV R0,#6
00001004:E3A01006 MOV R1,#6
00001008:E2400001 LOOP:SUB R0,R0,#1
0000100C:E0020091 MUL R2,R1,R0
00001010:E1A01002 MOV R1,R2
00001014:E3500001 CMP R0,#1
00001018:0A000000 BEQ EXIT
0000101C:EAFFFFF9 B LOOP
00001020:EF000011 EXIT:SWI 0x11
.end
TEST CASE – 2
OutputView
Console Stdin/Stdout/Stderr
Loading assembly language file D:\BKM-NAS\Documents\program7a.s
OutputView WatchView
Type here to search
ENG
21:22
30-01-2021


ARMSim - The ARM Simulator Dept. of Computer Science
File View Cache Debug Watch Help
RegistersView
General Purpose Floating Point
Hexadecimal
Unsigned Decimal
Signed Decimal
R0 :1
R1 :720
R2 :720
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):21504
R14(lr):0
R15(pc):4128
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 :System
0x600000df
program7a.s
.text
00001000:E3A00006 MOV R0,#6
00001004:E3A01006 MOV R1,#6
00001008:E2400001 LOOP:SUB R0,R0,#1
0000100C:E0020091 MUL R2,R1,R0
00001010:E1A01002 MOV R1,R2
00001014:E3500001 CMP R0,#1
00001018:0A000000 BEQ EXIT
0000101C:EAFFFFF9 B LOOP
00001020:EF000011 EXIT:SWI 0x11
.end
TEST CASE – 2 (Decimal)
OutputView
Console Stdin/Stdout/Stderr
Loading assembly language file D:\BKM-NAS\Documents\program7a.s
OutputView WatchView
Type here to search
ENG
21:23
30-01-2021

III.OUTPUT TABLE

| ITERATIONS | |
|---|---|
| 1st Iteration | R1=0x05 , R0=0x04<br>R2=0X14=Decimal 20 |
| 2nd Iteration | R1=0x14 , R0=0X03<br>R2=0x3C=Decimal 60 |
| 3rd Iteration | R1=0x3C ,R0=0X02<br>R2=0x78 =Decimal 120 |
| 4th Iteration | R1=0x78 , R0=0X01<br>R2=0x78 =Decimal 120 |

| ITERATIONS | |
|---|---|
| 1st Iteration | R1=0x06 , R0=0x05<br>R2=0X1e=Decimal 30 |
| 2nd Iteration | R1=0x1e , R0=0X04<br>R2=0x78=Decimal 120 |
| 3rd Iteration | R1=0x78 ,R0=0X03<br>R2=0x168 =Decimal 360 |
| 4th Iteration | R1=0x168 , R0=0X02<br>R2=0x2d0 =Decimal 720 |
| 5th Iteration | R1=0x2d0 , R0=0X01<br>R2=0x2d0 =Decimal 720 |

**Week#2.** **Program Number:4a**

**Title of the Program** : Write an ALP to add two 32 bit numbers loaded from memory and store the result in memory.

I. ARM Assembly Code for each program

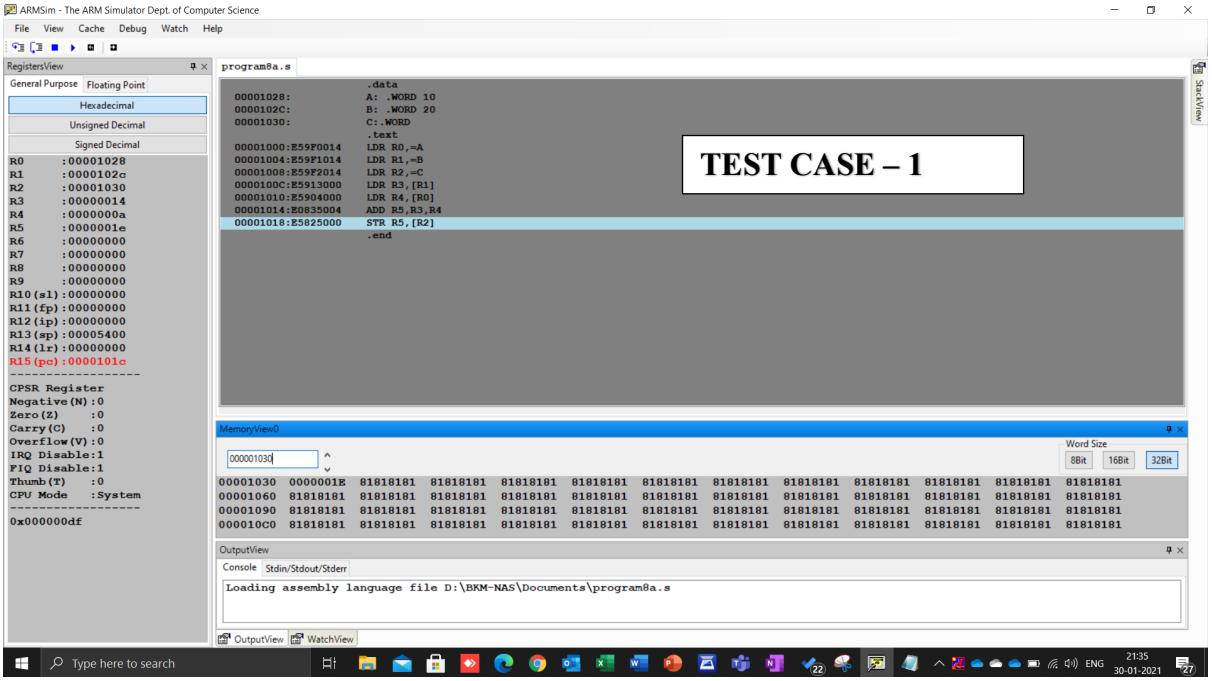
```
.data
A: .WORD 10
B: .WORD 20
C:.WORD
.text
LDR R0,=A
LDR R1,=B
LDR R2,=C
LDR R3,[R1]
LDR R4,[R0]
ADD R5,R3,R4
STR R5,[R2]
.end
```

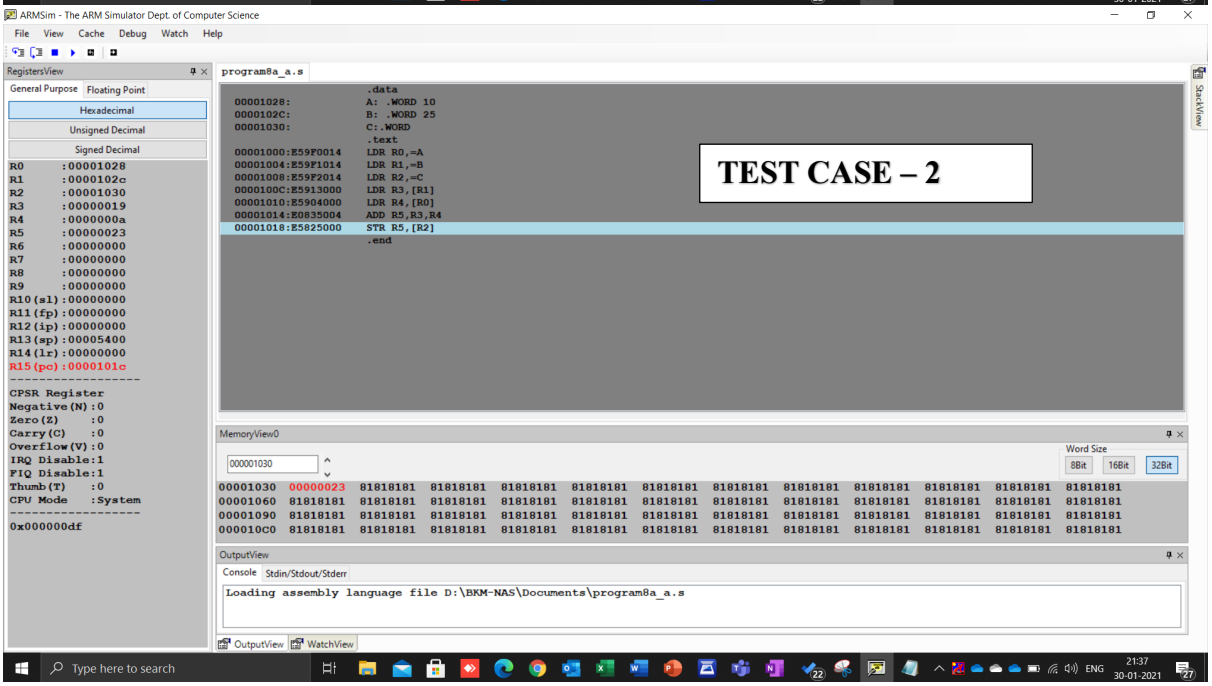
**TEST CASE – 1**

```
.data
A: .WORD 10
B: .WORD 25
C:.WORD
.text
LDR R0,=A
LDR R1,=B
LDR R2,=C
LDR R3,[R1]
LDR R4,[R0]
ADD R5,R3,R4
STR R5,[R2]
.end
```

**TEST CASE – 2**

# II.Output Screenshots



III.Output Table

| **TEST -1** | **A=0x0A, B=0x14** |
|---|---|
| R0 | Address of A |
| R1 | Address of B |
| R2 | Address of C |
| R3 | 0x14=Decimal 20 =Content of Location B |
| R4 | 0x0A=Decimal 10= Content of Location A |
| R5 | 0x1E=Decimal 30 |
| Location C | 0x1E=Decimal 30 |

**Week#2.** **Program Number:4b**

**Title of the Program** : Write an ALP to add two 16 bit numbers loaded from memory and store the result in memory.

I. ARM Assembly Code for each program

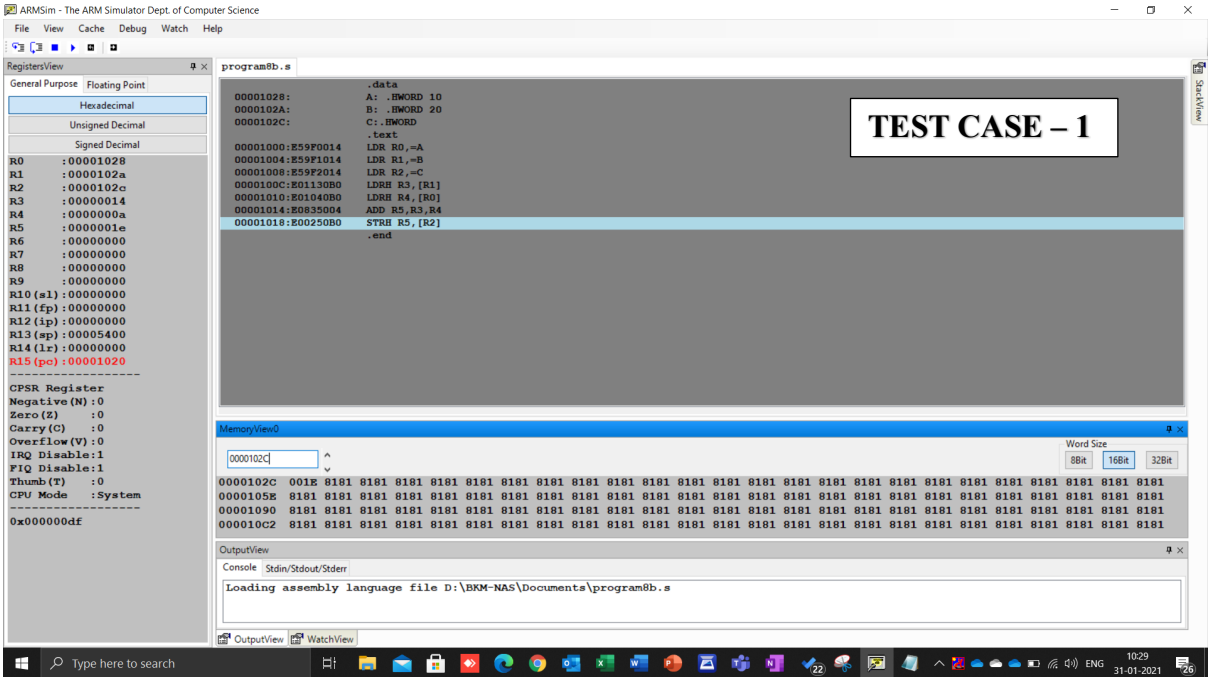
```
.data
A: .HWORD 10
B: .HWORD 20
C:.HWORD
.text
LDR R0,=A
LDR R1,=B
LDR R2,=C
LDRH R3,[R1]
LDRH R4,[R0]
ADD R5,R3,R4
STRH R5,[R2]
.end
```

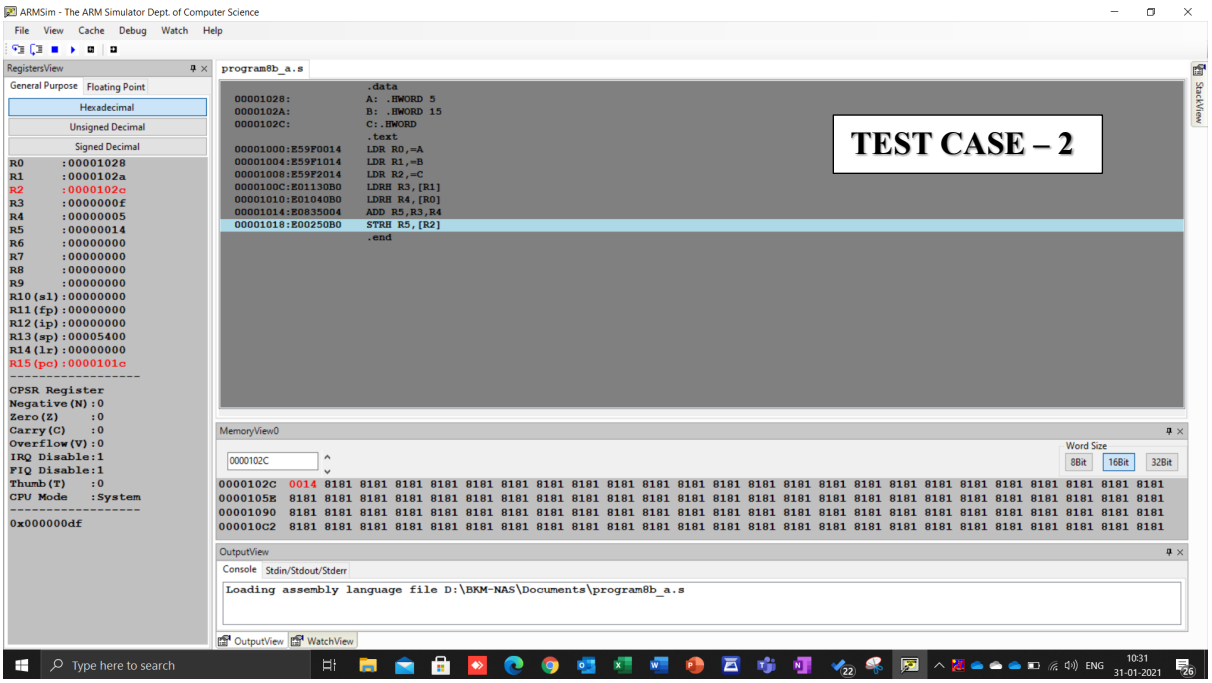
**TEST CASE – 1**

```
.data
A:  .HWORD 5
B:  .HWORD 15
C:.HWORD
.text
LDR R0,=A
LDR R1,=B
LDR R2,=C
LDRH R3,[R1]
LDRH R4,[R0]
ADD R5,R3,R4
STRH R5,[R2]
.end
```

**TEST CASE – 2**

# II.OUTPUT SCREENSHOTS






## III.Output Table

| TEST CASE 1 | A=0x0A, B=0x14 |
|---|---|
| R0 | Address of A (0x1028) |
| R1 | Address of B (0x102A) |
| R2 | Address of C (0x102C) |
| R3 | 0x14=Decimal 20=Content of Location B |
| R4 | 0x0A=Decimal 10= Content of Location A |
| R5 | 0x1E=Decimal 30 |
| Location C(0x102C) | 0x1E=Decimal 30 |
| TEST CASE 2 | A=0x05, B=0x0f |

| | |
|---|---|
| **R0** | **Address of A (0x1028)** |
| **R1** | **Address of B (0x102A)** |
| **R2** | **Address of C (0x102C)** |
| **R3** | **0x0f=Decimal 15=Content of Location B** |
| **R4** | **0x05=Decimal 05= Content of Location A** |
| **R5** | **0x14=Decimal 20** |
| **Location C(0x102C)** | **0x14=Decimal 20** |

**Week#2.** **Program Number:5a**

**Title of the Program** :Write an ALP to find GCD of two numbers (without using LDR and STR instructions).Both numbers are in registers. Use only registers

I. ARM Assembly Code for each program

```
.text
MOV R0,#10
MOV R1,#15
MOV R2,R0
MOV R3,R1
LOOP:CMP R2,R3
BEQ EXIT
BLT LABEL1
SUB R2,R2,R3
B LOOP
LABEL1:SUB R3,R3,R2
B LOOP
EXIT:SWI 0x11
```

**TEST CASE – 1**

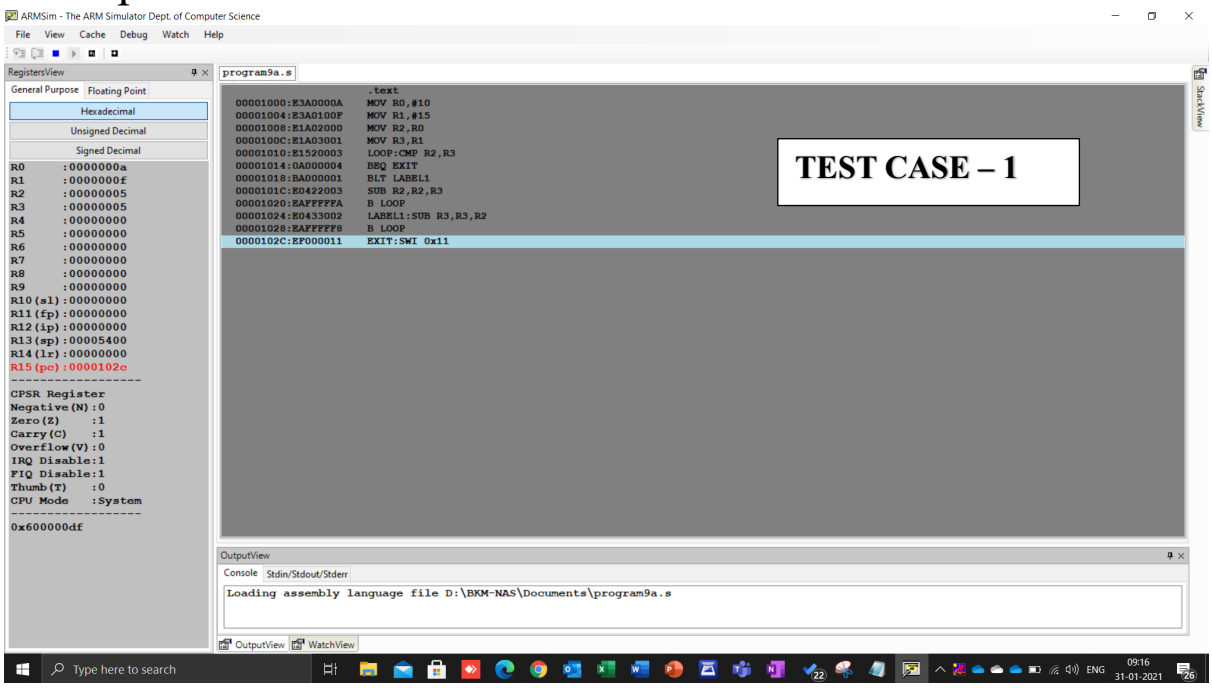
```
.text
MOV R0,#20
MOV R1,#10
MOV R2,R0
MOV R3,R1
LOOP:CMP R2,R3
BEQ EXIT
BLT LABEL1
SUB R2,R2,R3
B LOOP
LABEL1:SUB R3,R3,R2
B LOOP
EXIT:SWI 0x11
```

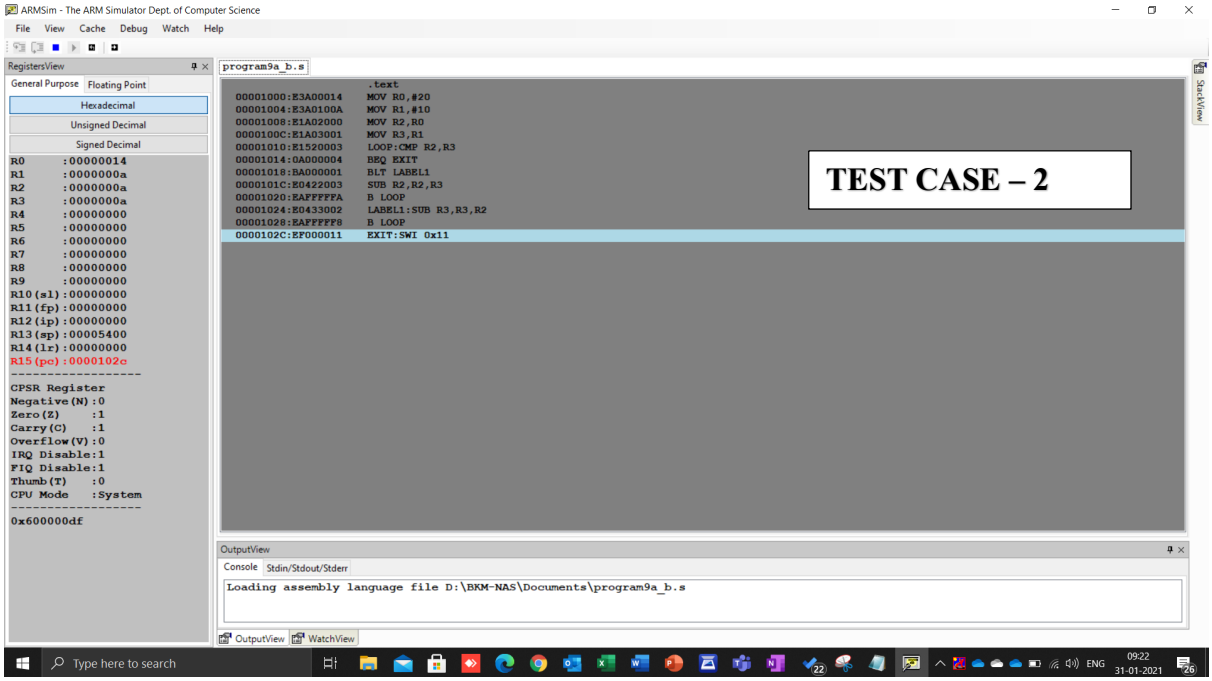
**TEST CASE – 2**

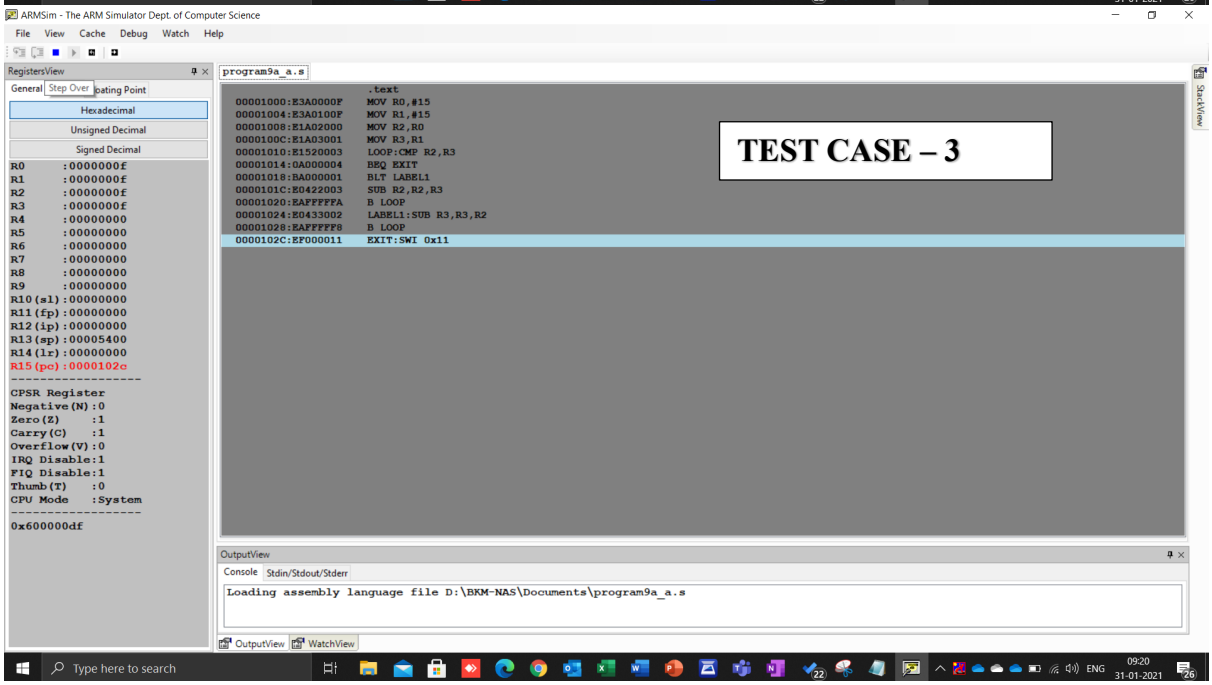
```
.text
MOV R0,#15
MOV R1,#15
MOV R2,R0
MOV R3,R1
LOOP:CMP R2,R3
BEQ EXIT
BLT LABEL1
SUB R2,R2,R3
B LOOP
LABEL1:SUB R3,R3,R2
B LOOP
EXIT:SWI 0x11
```

**TEST CASE – 3**

# II.Output Screenshots




ARMSim - The ARM Simulator Dept. of Computer Science
File View Cache Debug Watch Help
RegistersView
General Purpose Floating Point
Hexadecimal
Unsigned Decimal
Signed Decimal
R0 :00000014
R1 :0000000a
R2 :0000000a
R3 :0000000a
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):0000102c
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 :System
0x600000df
program9a_b.s
.text
00001000:E3A00014 MOV R0,#20
00001004:E3A0100A MOV R1,#10
00001008:E1A02000 MOV R2,R0
0000100C:E1A03001 MOV R3,R1
00001010:E1520003 LOOP:CMP R2,R3
00001014:0A000004 BEQ EXIT
00001018:BA000001 BLT LABEL1
0000101C:E0422003 SUB R2,R2,R3
00001020:EAFFFFFA B LOOP
00001024:E0433002 LABEL1:SUB R3,R3,R2
00001028:EAFFFFF8 B LOOP
0000102C:EF000011 EXIT:SWI 0x11
TEST CASE – 2
OutputView
Console Stdin/Stdout/Stderr
Loading assembly language file D:\BKM-NAS\Documents\program9a_b.s
OutputView WatchView
Type here to search
ENG 09:22 31-01-2021


ARMSim - The ARM Simulator Dept. of Computer Science
File View Cache Debug Watch Help
RegistersView
General Step Over oating Point
Hexadecimal
Unsigned Decimal
Signed Decimal
R0 :0000000f
R1 :0000000f
R2 :0000000f
R3 :0000000f
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):0000102c
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 :System
0x600000df
program9a_a.s
.text
00001000:E3A0000F MOV R0,#15
00001004:E3A0100F MOV R1,#15
00001008:E1A02000 MOV R2,R0
0000100C:E1A03001 MOV R3,R1
00001010:E1520003 LOOP:CMP R2,R3
00001014:0A000004 BEQ EXIT
00001018:BA000001 BLT LABEL1
0000101C:E0422003 SUB R2,R2,R3
00001020:EAFFFFFA B LOOP
00001024:E0433002 LABEL1:SUB R3,R3,R2
00001028:EAFFFFF8 B LOOP
0000102C:EF000011 EXIT:SWI 0x11
TEST CASE – 3
OutputView
Console Stdin/Stdout/Stderr
Loading assembly language file D:\BKM-NAS\Documents\program9a_a.s
OutputView WatchView
Type here to search
ENG 09:20 31-01-2021

## III.Output Table

| | | |
|---|---|---|
| CASE 1 | R0 | 0x0A |
| | R1 | 0x0F |
| 1st iteration | R2=0x0A , R3=0x0F<br>R3=0x0F-0x0A=0x05 | |
| 2nd iteration | R2=0x0A , R3=0x05<br>R2=0x0A-0x05=0x05 | |
| 3rd iteration | R2=0x05 ,R3=0x05<br>GCD=5 | |
| CASE 2 | R0 | 0x14 |
| | R1 | 0x0A |
| 1st iteration | R2=0x14 , R3=0x0A<br>R2=0x14-0x0A=0x0A | |
| 2nd iteration | R2=0x0A , R3=0x0A<br>GCD=0x0A(ie 10 in decimal) | |
| CASE 3 | R0 | 0x0F |
| | R1 | 0x0F |
| 1st iteration | R2=0x0F , R3=0x0F<br>GCD = 15 ( 0x0F) | |

**Week#2.** **Program Number:5b**

**Title of the Program** : Write an ALP to find the GCD of given numbers (both numbers in memory).Store result in memory

I.ARM Assembly Code for each program

```
.data
A:.WORD 20
B:.WORD 4
C:.WORD
.text
LDR R0,=A
LDR R1,=B
LDR R5,=C
LDR R2,[R0]
LDR R3,[R1]
LOOP:CMP R2,R3
BEQ EXIT
BLT LABEL1
SUB R2,R2,R3
B LOOP
LABEL1:SUB R3,R3,R2
B LOOP
EXIT:STR R3,[R5]
SWI 0x11
.end
```

**TEST CASE – 1**

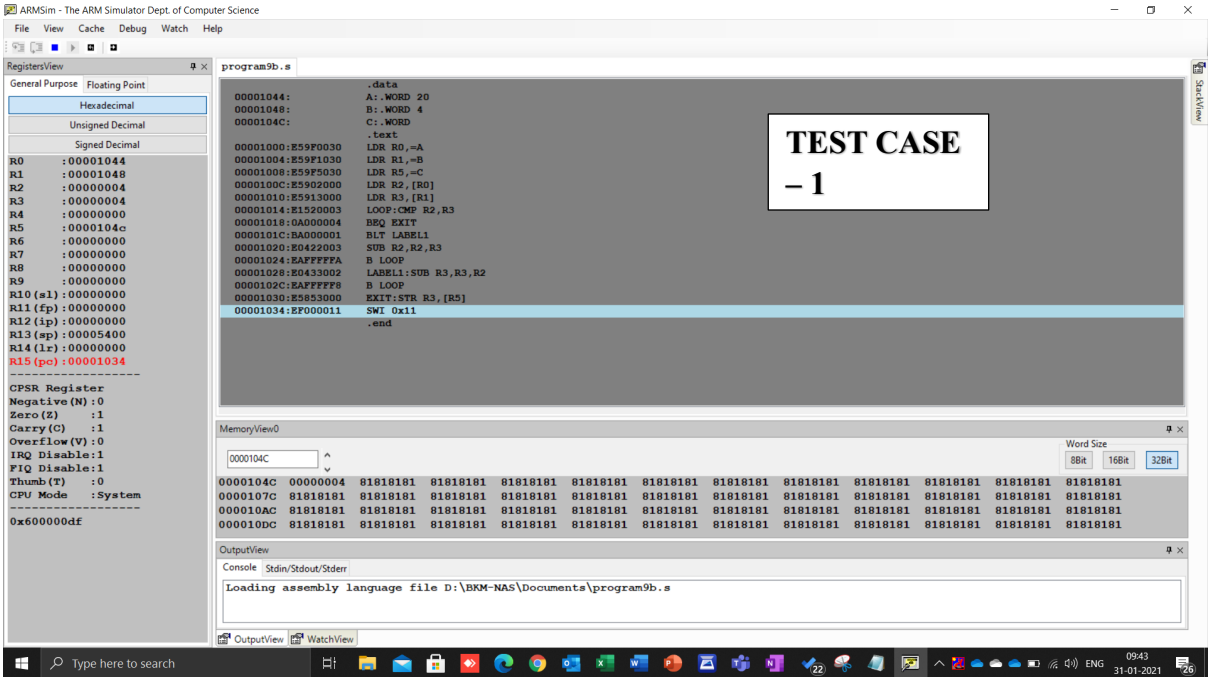
```
.data
A:.WORD 10
B:.WORD 15
C:.WORD
.text
LDR R0,=A
LDR R1,=B
LDR R5,=C
LDR R2,[R0]
LDR R3,[R1]
LOOP:CMP R2,R3
BEQ EXIT
BLT LABEL1
SUB R2,R2,R3
B LOOP
LABEL1:SUB R3,R3,R2
B LOOP
EXIT:STR R3,[R5]
SWI 0x11
.end
```

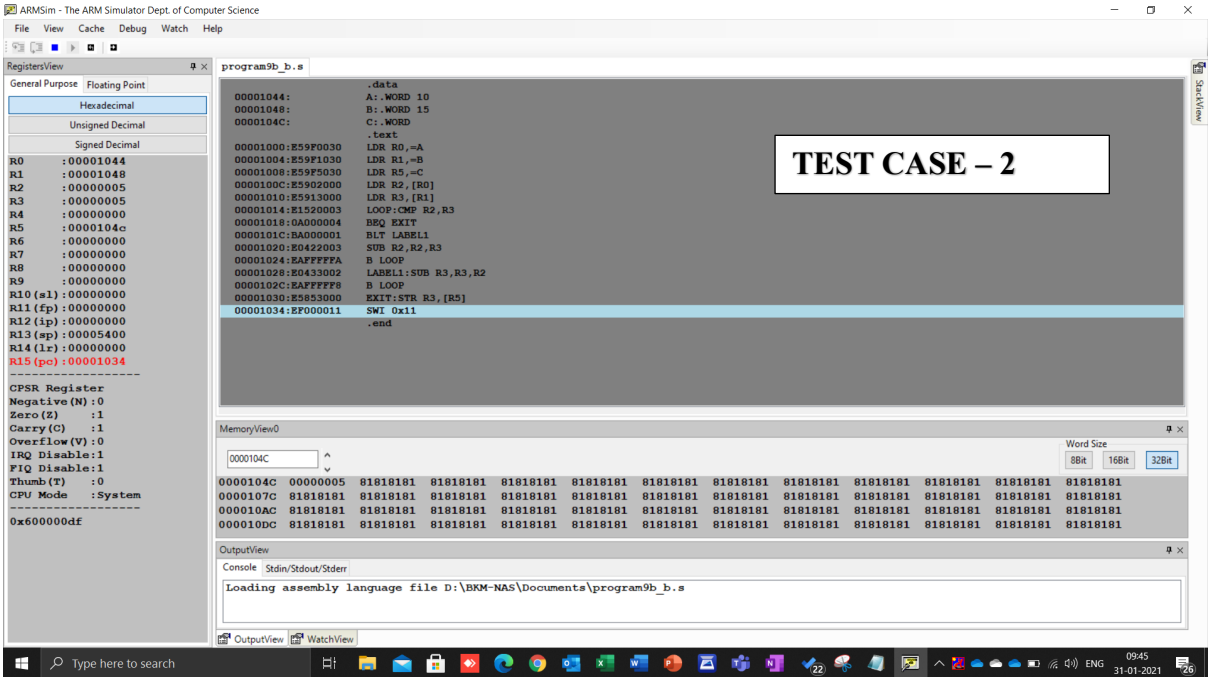
**TEST CASE – 2**

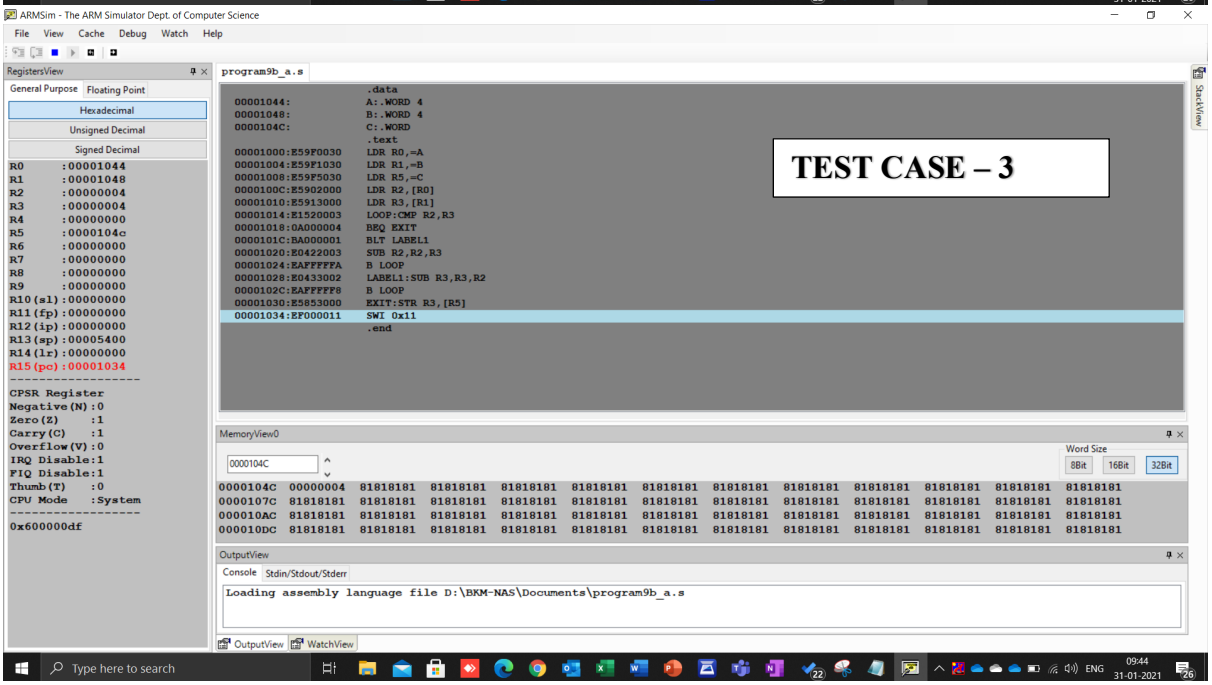
```
.data
A:.WORD 4
B:.WORD 4
C:.WORD
.text
LDR R0,=A
LDR R1,=B
LDR R5,=C
LDR R2,[R0]
LDR R3,[R1]
LOOP:CMP R2,R3
BEQ EXIT
BLT LABEL1
SUB R2,R2,R3
B LOOP
LABEL1:SUB R3,R3,R2
B LOOP
EXIT:STR R3,[R5]
SWI 0x11
.end
```

**TEST CASE – 3**

# II.Output Screenshots






# III.Output Table

| CASE 1 | A=0x14,B=0x04 |
|---|---|
| 1st iteration | R2=0x14=Decimal 20<br>R3=0x04<br>R2=0x14-0x04=0x10 |

| | |
|---|---|
| 2nd iteration | R2=0x10=Decimal 16<br>R3=0x04<br>R2=0x10-0x04=0x0C |
| 3rd iteration | R2=0x0C=Decimal 12<br>R3=0x04<br>R2=0x0C-0x04=0x08 |
| 4th iteration | R2=0x08<br>R3=0x04<br>R2=0x08-0x04=0x04 |
| 5th iteration | R2=0x04<br>R3=0x04<br>GCD=C=0x04 |
| **CASE 2** | **A=0x0A,B=0x0F** |
| 1st iteration | R2=0x0A=Decimal 10<br>R3=0x0F=Decimal 15<br>R3=0x0F-0x0A=0x05 |
| 2nd iteration | R2=0x0A=Decimal 10<br>R3=0x05<br>R2=0x0A-0x05=0x05 |
| 3rd iteration | R2=0x05=Decimal 5<br>R3=0x05<br>GCD=0x05 ( C=0x05) |
| **CASE 3** | **A=0x04,B=0x04** |
| 1st iteration | R2=0x04=Decimal 04<br>R3=0x04=Decimal 04<br>**GCD C=0x04** |

**Week#2** **Program Number:6a**

**Title of the Program** : Write an ALP to add an array of ten 32 bit numbers from memory

I.ARM Assembly Code for each program

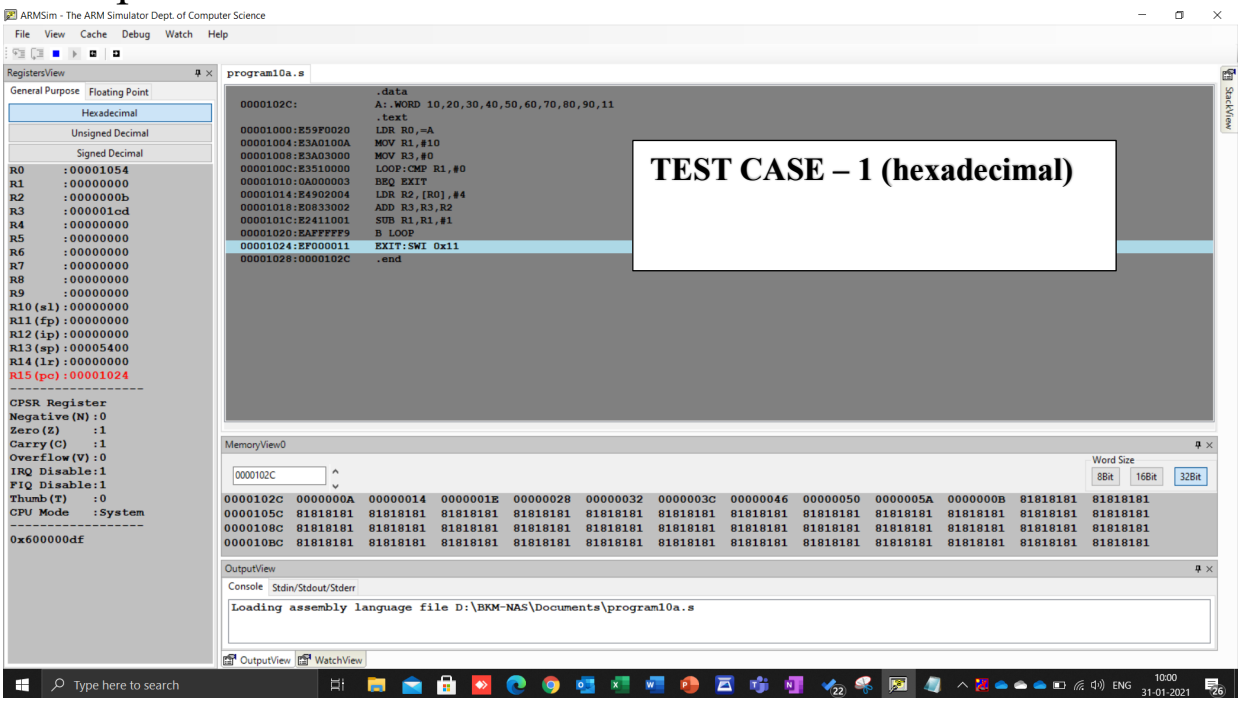
```
.data
A:.WORD 10,20,30,40,50,60,70,80,90,11
.text
LDR R0,=A
MOV R1,#10
MOV R3,#0
LOOP:CMP R1,#0
BEQ EXIT
LDR R2,[R0],#4
ADD R3,R3,R2
SUB R1,R1,#1
B LOOP
EXIT:SWI 0x11
.end
```

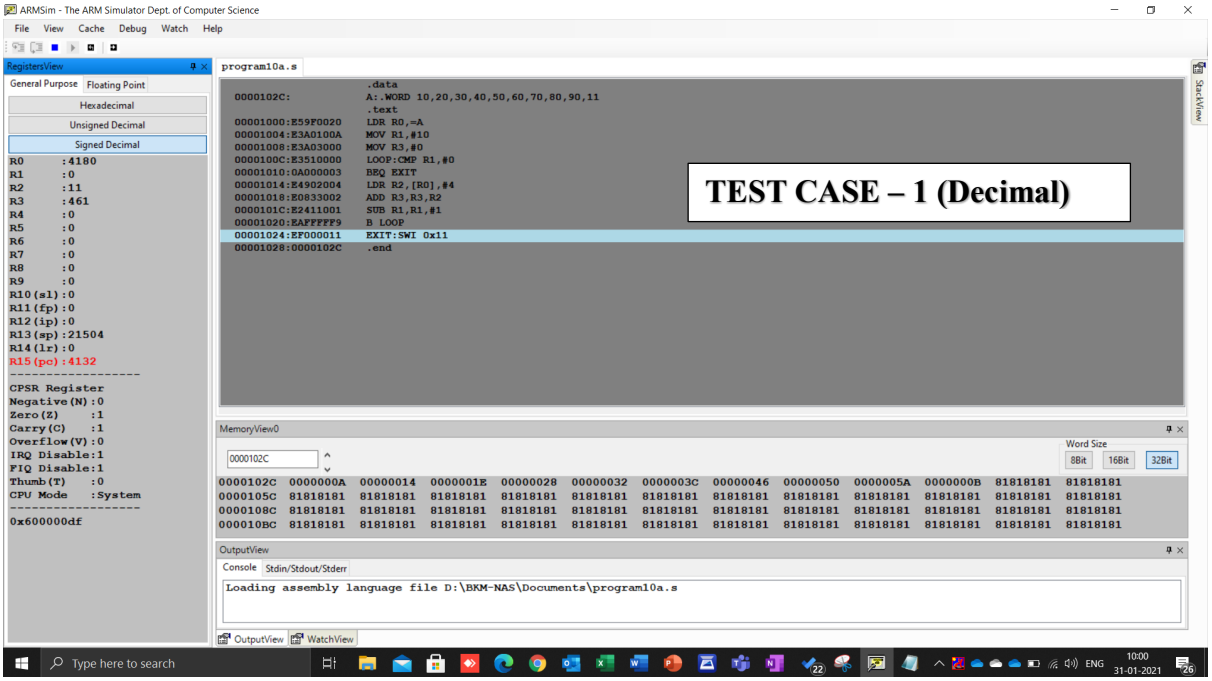
**TEST CASE – 1**

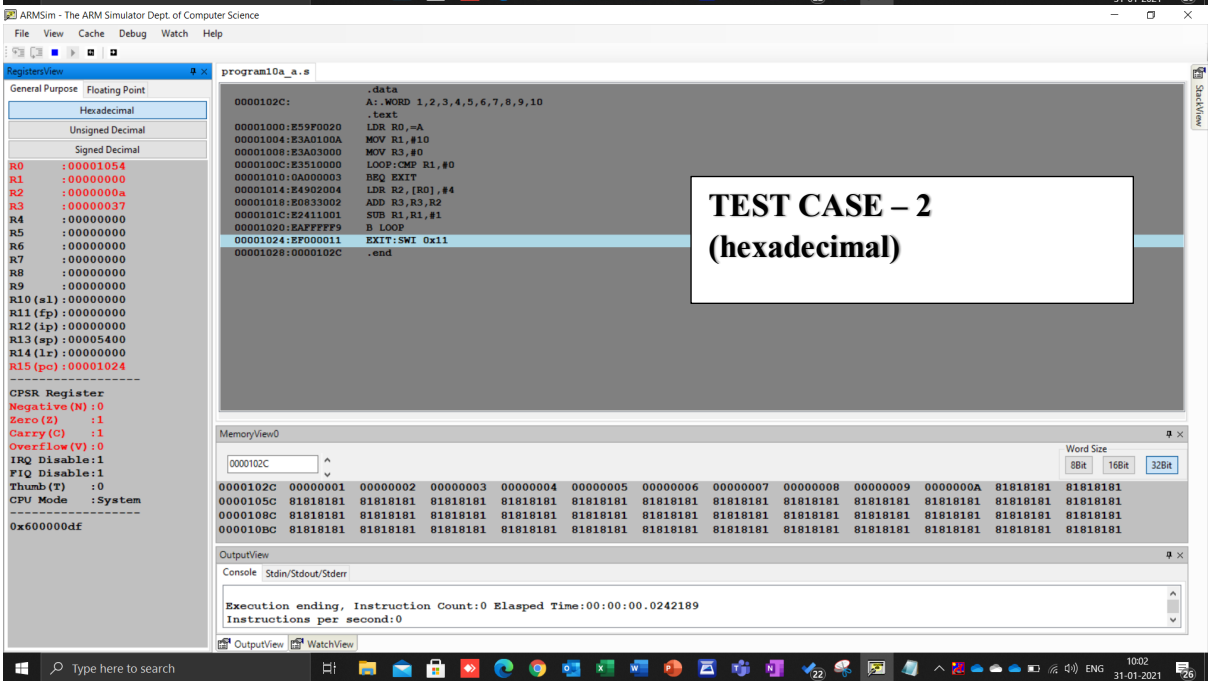
```
.data
A:.WORD 1,2,3,4,5,6,7,8,9,10
.text
LDR R0,=A
MOV R1,#10
MOV R3,#0
LOOP:CMP R1,#0
BEQ EXIT
LDR R2,[R0],#4
ADD R3,R3,R2
SUB R1,R1,#1
B LOOP
EXIT:SWI 0x11
.end
```

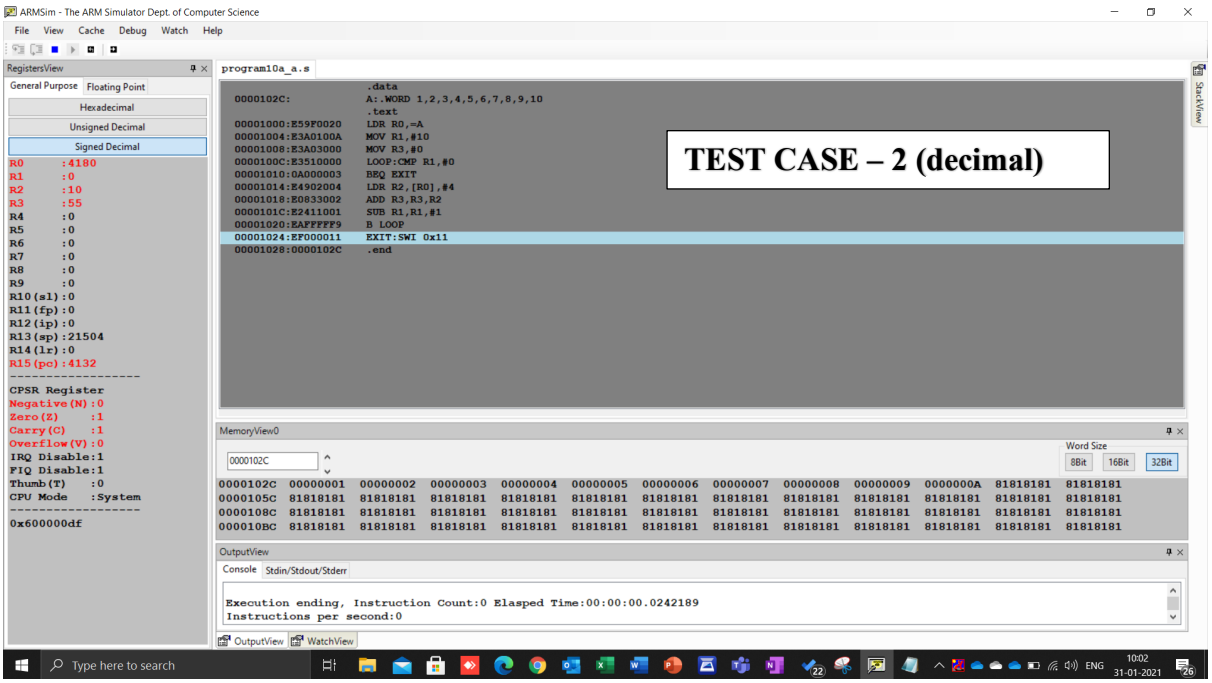
**TEST CASE – 2**

# II.Output Screenshots




ARMSim - The ARM Simulator Dept. of Computer Science
File View Cache Debug Watch Help
RegistersView
General Purpose Floating Point
Hexadecimal
Unsigned Decimal
Signed Decimal
R0 :4180
R1 :0
R2 :11
R3 :461
R4 :0
R5 :0
R6 :0
R7 :0
R8 :0
R9 :0
R10(sl):0
R11(fp):0
R12(ip):0
R13(sp):21504
R14(lr):0
R15(pc):4132
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 :System
0x600000df
program10a.s
0000102C: .data
A:.WORD 10,20,30,40,50,60,70,80,90,11
.text
00001000:E59F0020 LDR R0,=A
00001004:E3A0100A MOV R1,#10
00001008:E3A03000 MOV R3,#0
0000100C:E3510000 LOOP:CMP R1,#0
00001010:0A000003 BEQ EXIT
00001014:E4902004 LDR R2,[R0],#4
00001018:E0833002 ADD R3,R3,R2
0000101C:E2411001 SUB R1,R1,#1
00001020:EAFFFFF9 B LOOP
00001024:EF000011 EXIT:SWI 0x11
00001028:0000102C .end
TEST CASE – 1 (Decimal)
MemoryView0
0000102C
Word Size
8Bit 16Bit 32Bit
0000102C 0000000A 00000014 0000001E 00000028 00000032 0000003C 00000046 00000050 0000005A 0000000B 81818181 81818181
0000105C 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
0000108C 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
000010BC 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
OutputView
Console Stdin/Stdout/Stderr
Loading assembly language file D:\BKM-NAS\Documents\program10a.s
OutputView WatchView
Type here to search
ENG 10:00 31-01-2021


ARMSim - The ARM Simulator Dept. of Computer Science
File View Cache Debug Watch Help
RegistersView
General Purpose Floating Point
Hexadecimal
Unsigned Decimal
Signed Decimal
R0 :00001054
R1 :00000000
R2 :0000000a
R3 :00000037
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):00001024
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 :System
0x600000df
program10a_a.s
0000102C: .data
A:.WORD 1,2,3,4,5,6,7,8,9,10
.text
00001000:E59F0020 LDR R0,=A
00001004:E3A0100A MOV R1,#10
00001008:E3A03000 MOV R3,#0
0000100C:E3510000 LOOP:CMP R1,#0
00001010:0A000003 BEQ EXIT
00001014:E4902004 LDR R2,[R0],#4
00001018:E0833002 ADD R3,R3,R2
0000101C:E2411001 SUB R1,R1,#1
00001020:EAFFFFF9 B LOOP
00001024:EF000011 EXIT:SWI 0x11
00001028:0000102C .end
TEST CASE – 2 (hexadecimal)
MemoryView0
0000102C
Word Size
8Bit 16Bit 32Bit
0000102C 00000001 00000002 00000003 00000004 00000005 00000006 00000007 00000008 00000009 0000000A 81818181 81818181
0000105C 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
0000108C 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
000010BC 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
OutputView
Console Stdin/Stdout/Stderr
Execution ending, Instruction Count:0 Elasped Time:00:00:00.0242189
Instructions per second:0
OutputView WatchView
Type here to search
ENG 10:02 31-01-2021




# III.Output Table

| A:.word 10,20,30,40,50,60,70,80,90,11 [TEST CASE 1] | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| R1 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |
| R0 | A | A+4 | A+8 | A+12 | A+16 | A+20 | A+24 | A+28 | A+32 | A+36 |
| R2 | 10 | 20 | 30 | 40 | 50 | 60 | 70 | 80 | 90 | 11 |
| R3 | 0 | 10 | 30 | 60 | 100 | 150 | 210 | 280 | 360 | 450 |
| R3(after exec | 10 | 30 | 60 | 100 | 150 | 210 | 280 | 360 | 450 | 461 |

| ution ) | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Values (In hexadecimal) | 0x 0A | 0x 1E | 0x4C | 0x64 | 0x96 | 0xD2 | 0x118 | 0x168 | 0x1C2 | 0x1CD |
| A:.word 1,2,3,4,5,6,7,8,9,10 [TEST CASE 2] | | | | | | | | | | |
| R1 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |
| R0 | A | A+4 | A+8 | A+12 | A+16 | A+20 | A+24 | A+28 | A+32 | A+36 |
| R2 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| R3 | 0 | 1 | 3 | 6 | 10 | 15 | 21 | 28 | 36 | 45 |
| R3(after execution ) | 1 | 3 | 6 | 10 | 15 | 21 | 28 | 36 | 45 | 55 |
| Values (In hexadecimal) | 0x 01 | 0x 03 | 0x06 | 0x0A | 0x 0F | 0x 15 | 0x1c | 0x24 | 0x2d | 0x37 |

**Week#2.** **Program Number:6b**

**Title of the Program** : Write an ALP to add an array of five 8 bit numbers from memory(use .byte to store the data instead of .word)

I.ARM Assembly Code for each program

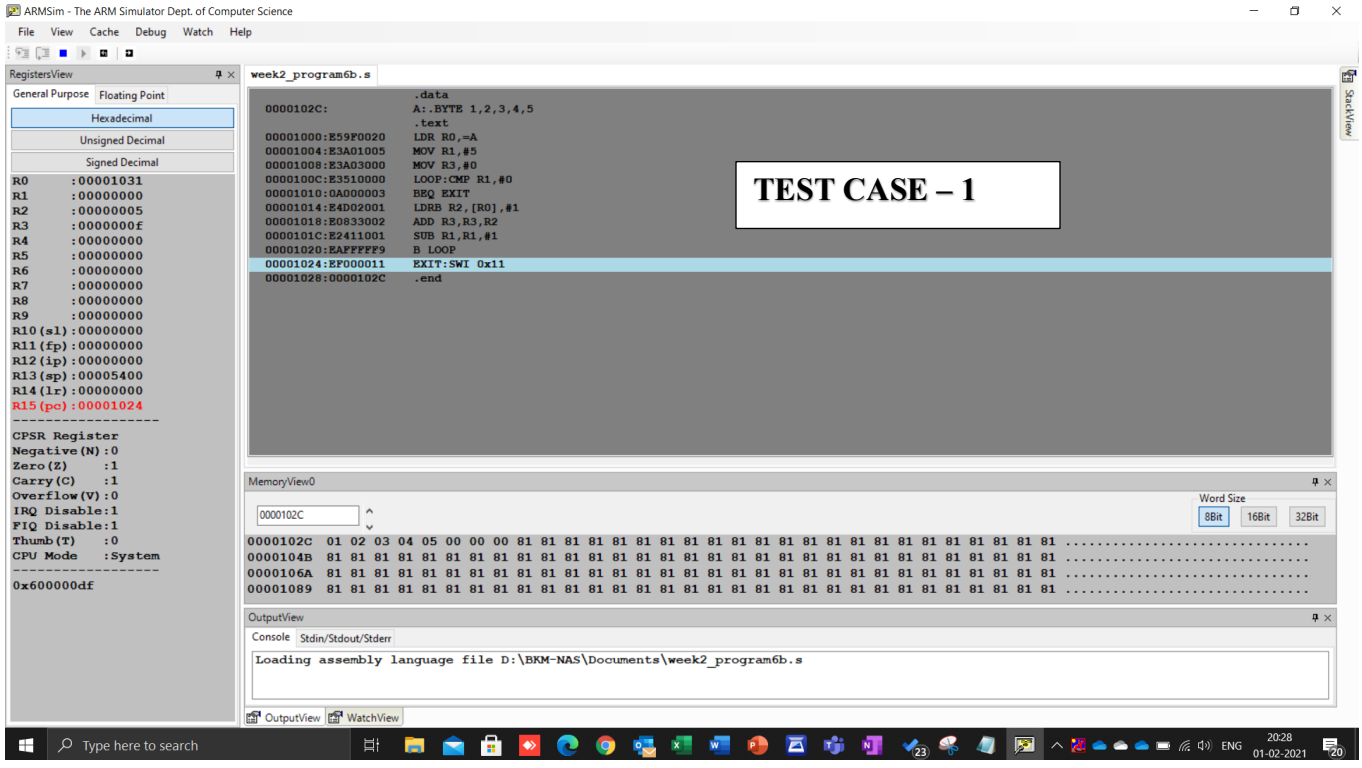
```
.data
A:.BYTE 1,2,3,4,5
.text
LDR R0,=A
MOV R1,#5
MOV R3,#0
LOOP:CMP R1,#0
BEQ EXIT
LDRB R2,[R0],#1
ADD R3,R3,R2
SUB R1,R1,#1
B LOOP
EXIT:SWI 0x11
.end
```

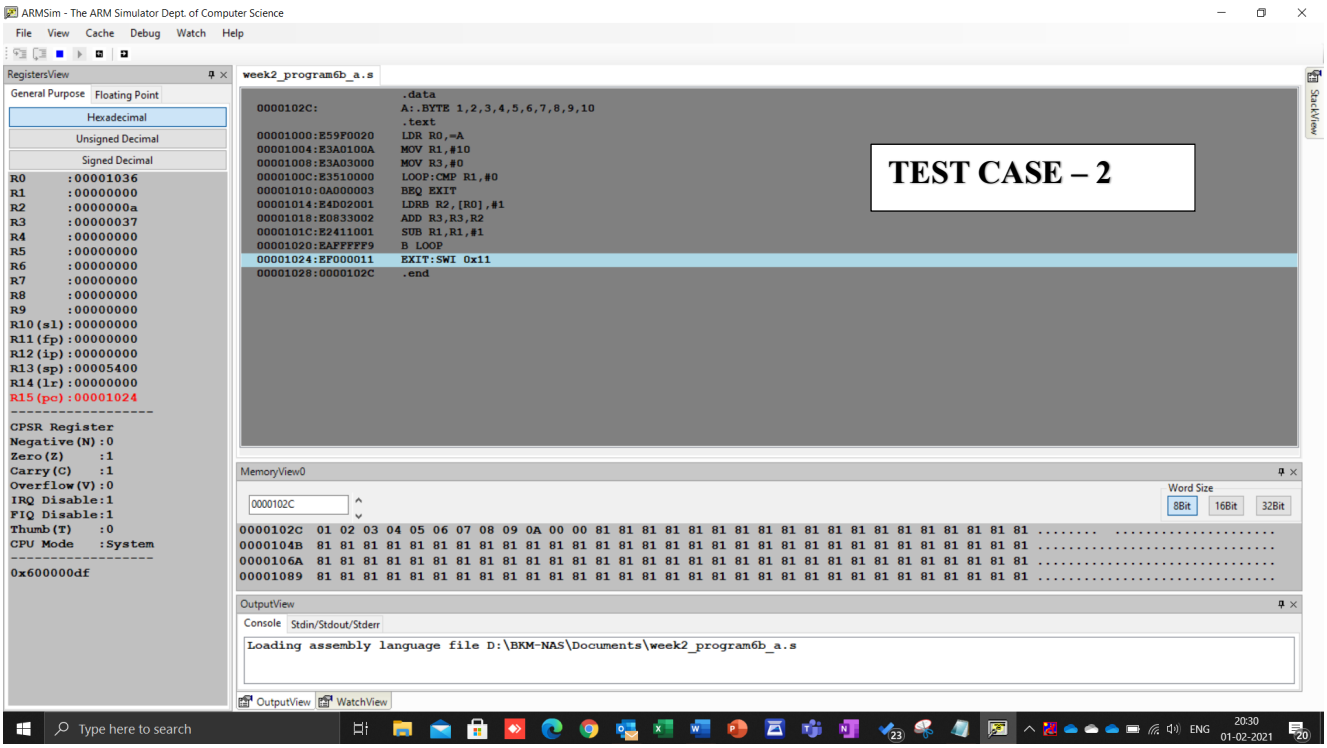
TEST CASE 1

```
.data
A:.BYTE 1,2,3,4,5,6,7,8,9,10
.text
LDR R0,=A
MOV R1,#10
MOV R3,#0
LOOP:CMP R1,#0
BEQ EXIT
LDRB R2,[R0],#1
ADD R3,R3,R2
SUB R1,R1,#1
B LOOP
EXIT:SWI 0x11
.end
```

**TEST CASE – 2**

# II.Output Screenshots






# III.Output Table

| A:.byte 1,2,3,4,5 | | | | | |
|---|---|---|---|---|---|
| R1 | 5 | 4 | 3 | 2 | 1 |
| R0 | A | A+1 | A+2 | A+3 | A+4 |
| R3 | 0 | 1 | 3 | 6 | 10 |
| R4 | 1 | 2 | 3 | 4 | 5 |
| R3 (After Execution) | 1 | 3 | 6 | 10 | 15 |
| Values in hex | 0x 01 | 0x 03 | 0x 06 | 0x 0A | 0x 0F |

| A:.byte 1,2,3,4,5,6,7,8,9,10 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| R1 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R0 | A | A+1 | A+2 | A+3 | A+4 | A+5 | A+6 | A+7 | A+8 | A+9 | A+10 |
| R3 | 0 | 1 | 3 | 6 | 10 | 15 | 21 | 28 | 36 | 45 | 55 |
| R4 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 10 |
| R3 (After Execution) | 1 | 3 | 6 | 10 | 15 | 21 | 28 | 36 | 45 | 55 | 55 |
| Values in hex | 0x 01 | 0x 03 | 0x 06 | 0x 0A | 0x 0F | 0x 15 | 0x 1c | 0x 24 | 0x 2d | 0x 37 | 0x 37 |

**Week#2** **ProgramNumber:7**

**Title of the Program : Write an ALP to multiply 35*R0.Use LSL instruction for multiplication**

## I.ARM ASSEMBLY CODE

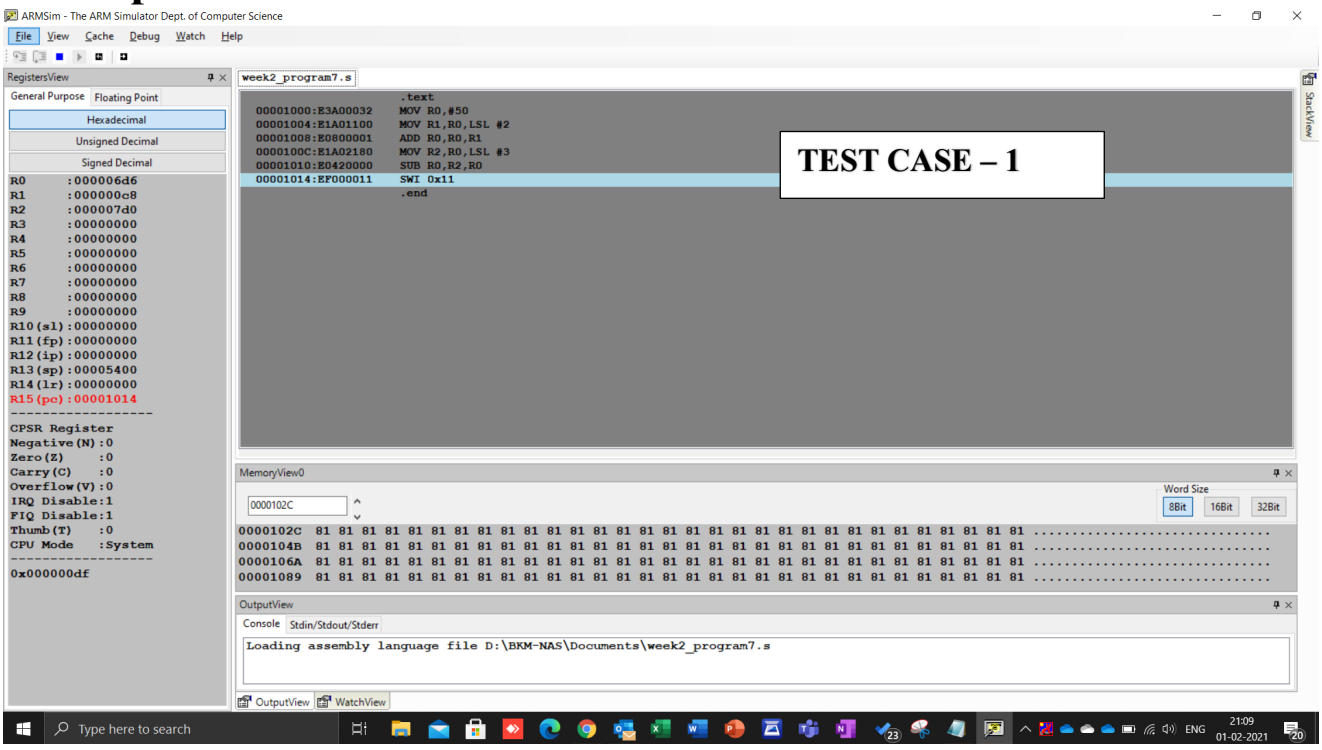
```
.text
MOV R0,#50
MOV R1,R0,LSL #2
ADD R0,R0,R1
MOV R2,R0,LSL #3
SUB R0,R2,R0
SWI 0x11
.end
```

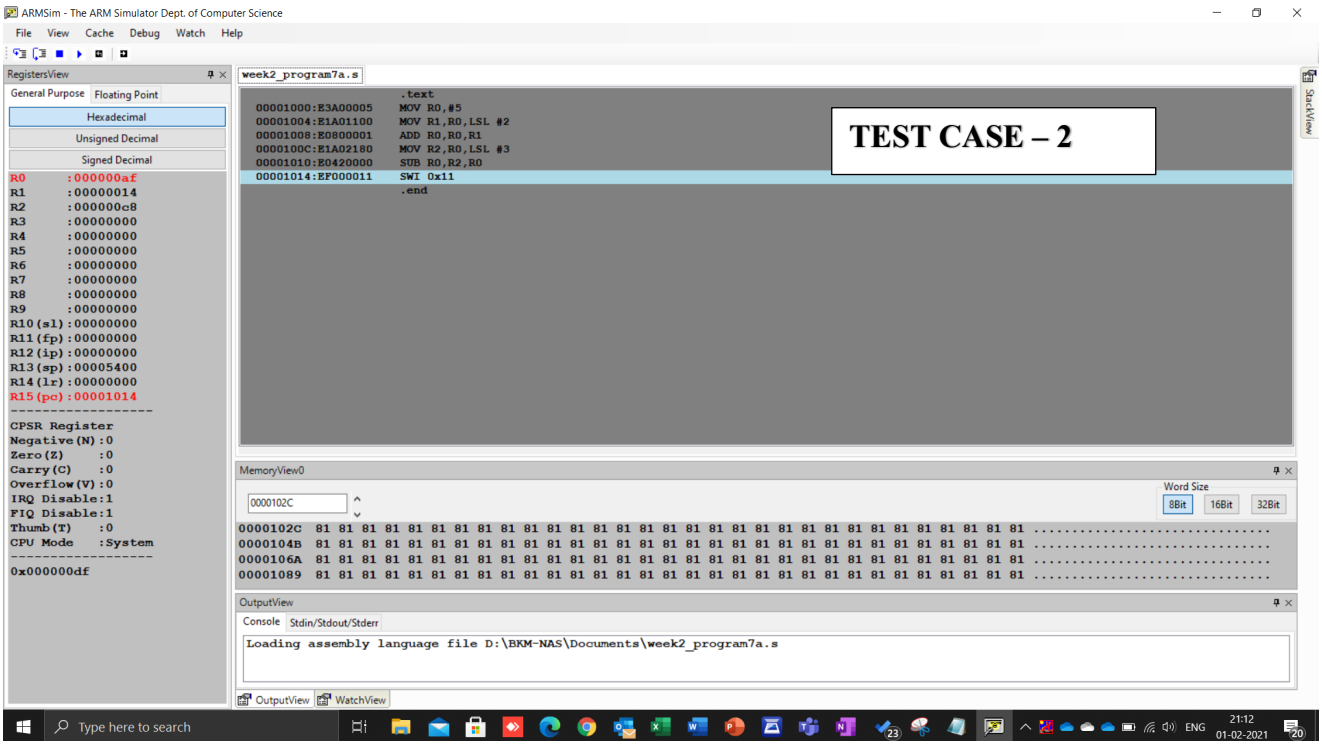
**TEST CASE – 1**

```
.text
MOV R0,#5
MOV R1,R0,LSL #2
ADD R0,R0,R1
MOV R2,R0,LSL #3
SUB R0,R2,R0
SWI 0x11
.end
```

**TEST CASE – 2**

# II.Output Screenshots






# III.Output Table

| TEST - 1 | Decimal | Hexadecimal |
|---|---|---|
| R0(1st line of the Program) | 50 | 32 |
| R0(2nd line of the Program | 4R0=200 | C8 |
| | 4R0+R0= 5R0=250 | FA |
| R0(3rd line of | 8R0=250*8=2000 | 0x7D0 |

| | | |
|---|---|---|
| **the Program** | | |

| TEST - 2 | Decimal | Hexadecimal |
|---|---|---|
| R0(1st line of the Program) | 5 | 5 |
| R0(2nd line of the Program | 4R0=20 | 14 |
| | 4R0+R0= 5R0=20+5=25 | 19 |
| R0(3rd line of the Program | 8R0=25*8=200 | 0xc8 |
| | 8R0-R0=200-25=175 | af |

## Week#2 Program Number:8

**Title of the Program : Write an ALP to evaluate the expression (A+B) + (5*B), where A and B are available in memory location. 9.Store the final result in memory Location C.Use LSL instruction for multiplication**

**I.ARM ASSEMBLY CODE**

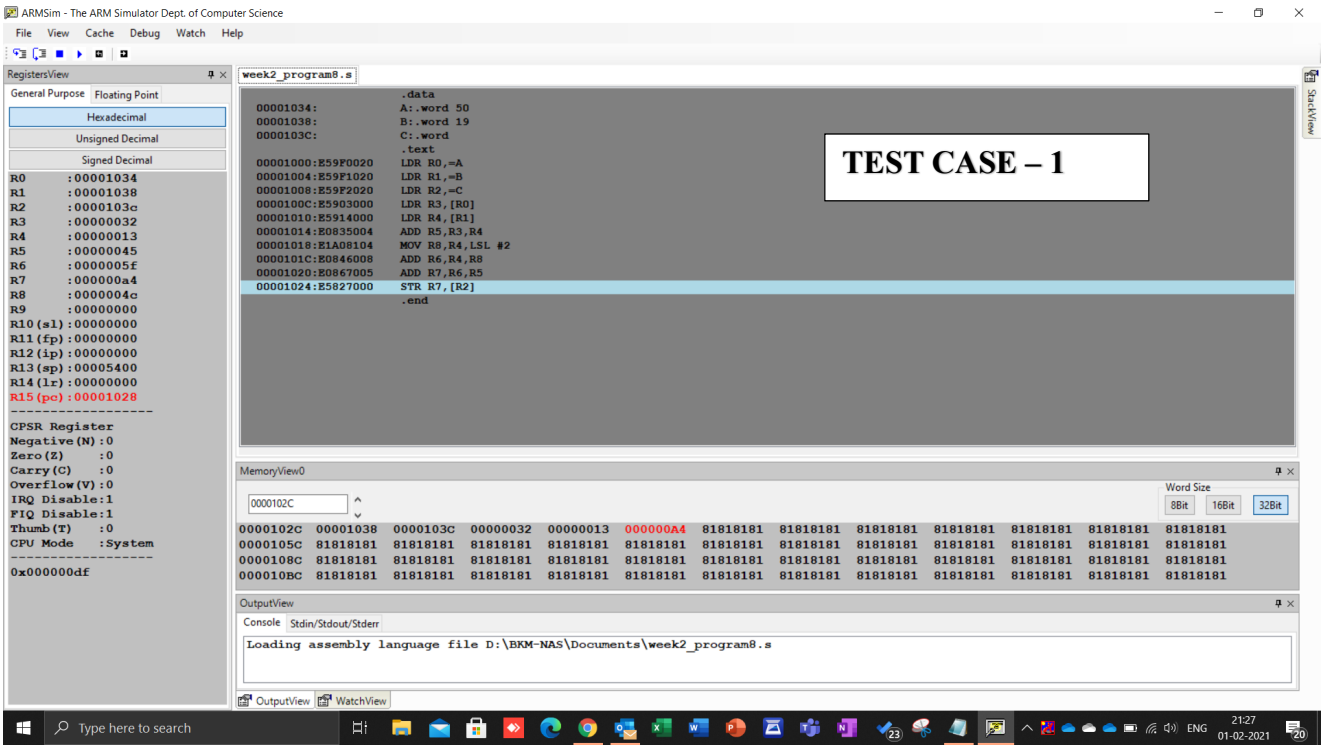
TEST CASE – 1

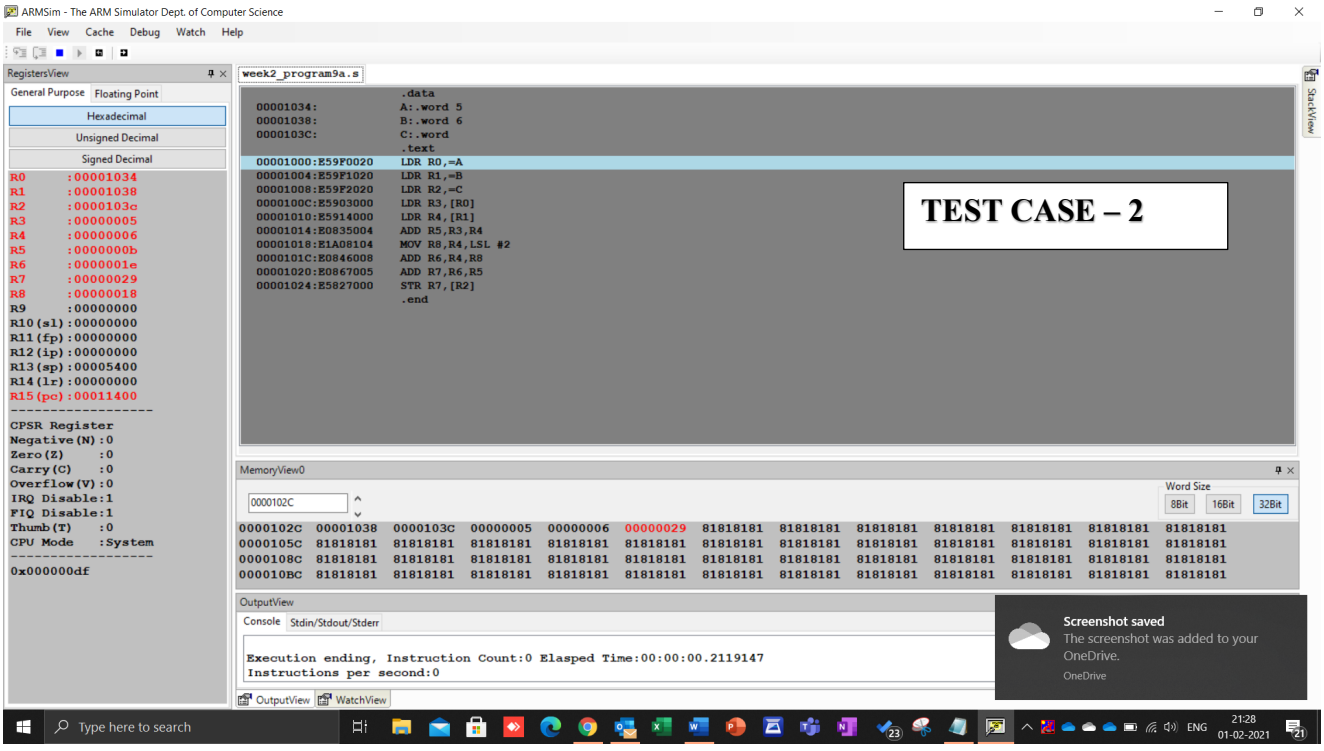
```
.data
A:.word 50
B:.word 19
C:.word
.text
LDR R0,=A
LDR R1,=B
LDR R2,=C
LDR R3,[R0]
LDR R4,[R1]
ADD R5,R3,R4
MOV R8,R4,LSL #2
ADD R6,R4,R8
ADD R7,R6,R5
STR R7,[R2]
.end
```

TEST CASE – 2

```
.data
A:.word 5
B:.word 6
C:.word
.text
LDR R0,=A
LDR R1,=B
LDR R2,=C
LDR R3,[R0]
LDR R4,[R1]
ADD R5,R3,R4
MOV R8,R4,LSL #2
ADD R6,R4,R8
ADD R7,R6,R5
STR R7,[R2]
.end
```

## II.Output Screenshots




# III.Output Table

| A=Decimal 50,B=Decimal 19 TEST CASE 1 | | |
|---|---|---|
| | Decimal | Hexadecimal |
| R3<br>R4 | 50<br>19 | 32<br>13 |
| R5<br>=R3+R4<br>=A+B | 69 | 45 |
| Calculate 4*B | 19*4=76 | 4C |
| Calculate 5*B | 76+19=95 | 5F |
| Calculate<br>(A+B)+5*B | 69+95=164 | A4 |

| A=Decimal 5,B=Decimal 6 | | |
|---|---|---|
| | Decimal | Hexadecimal |
| R3<br>R4 | 5<br>6 | 5<br>6 |

| | | |
|---|---|---|
| **R5 =R3+R4 =A+B** | **11** | **B** |
| **Calculate 4*B** | **4*6=24** | **18** |
| **Calculate 5*B** | **24+6=30** | **1e** |
| **Calculate (A+B)+5*B** | **11+30=41** | **29** |

## 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: Priya Mohata
Name: Priya Mohata
SRN: PES2UG19CS301
Section: E
Date: 02/01/2021