Microprocessor and Computer Architecture Laboratory

UE19CS256

4th Semester, Academic Year 2020-21

Date:2/2/2021

Name: Ramya Narasimha Prabhu	SRN:PES1UG19CS380	Section: F

Week#____2_____ Program Number: ___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.

I. ARM Assembly Code for each program

```
;case positive
.text
   MOV r0, #10
   CMP r0, #0
   BEQ zero
   BMI negative
   MOV r1, #2
   SWI 0x011
   zero:
      MOV r1, #1
```

```
    SWI 0x011
  negative:
    MOV r1, #3
    SWI 0x011
.end
```

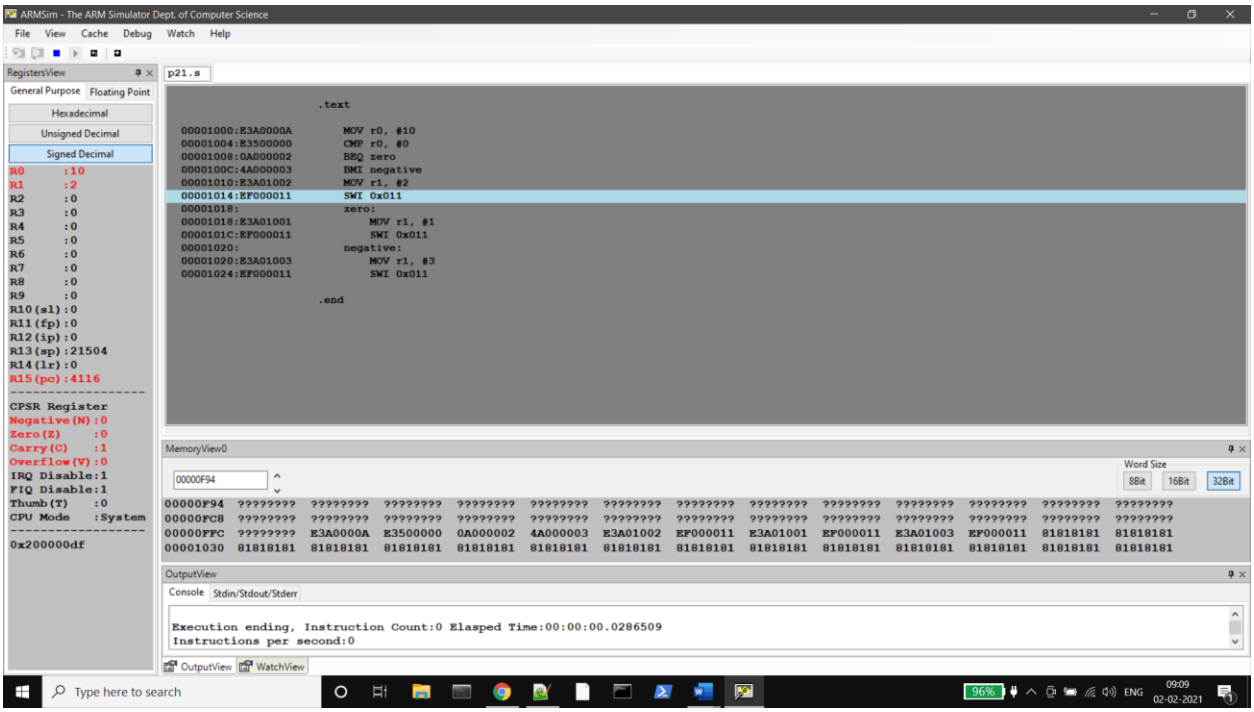
```
;case negative
.text

  MOV r0, #-20
  CMP r0, #0
  BEQ zero
  BMI negative
  MOV r1, #2
  SWI 0x011
  zero:
    MOV r1, #1
    SWI 0x011
  negative:
    MOV r1, #3
    SWI 0x011

.end
```

```
.text
;case zero
  MOV r0, #0
```

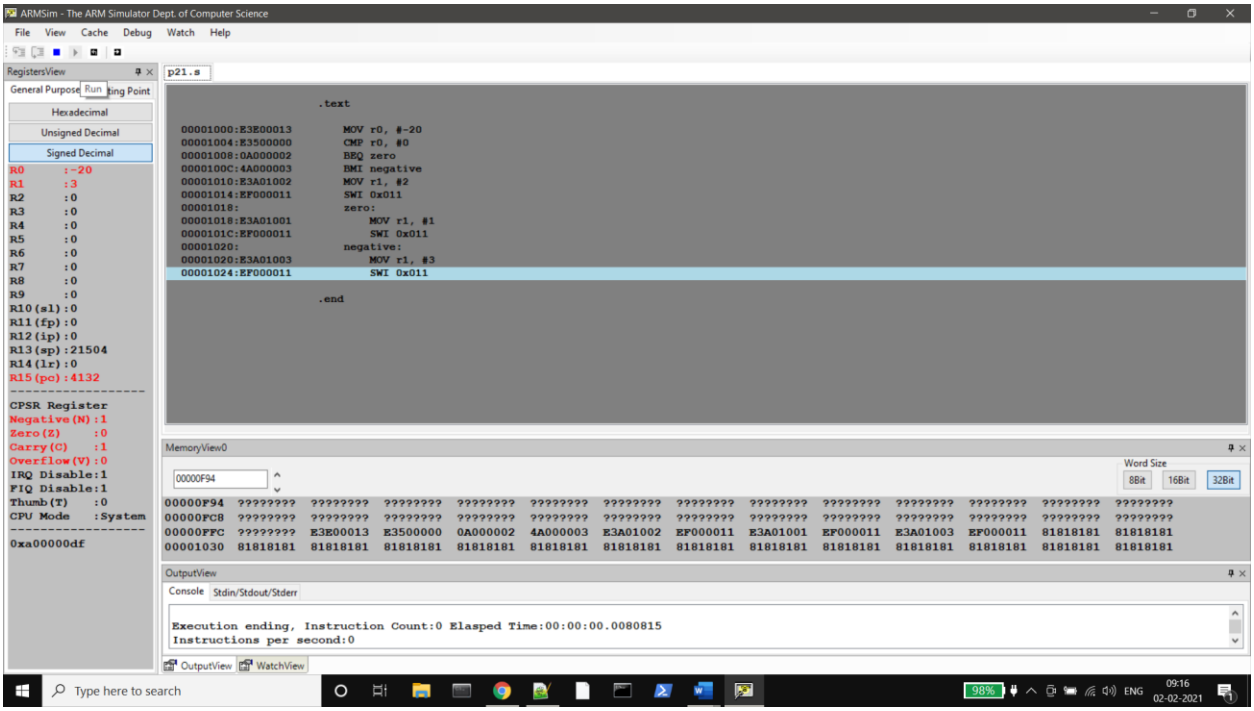
```
    CMP r0, #0
    BEQ zero
    BMI negative
    MOV r1, #2
    SWI 0x011
    zero:
        MOV r1, #1
        SWI 0x011
    negative:
        MOV r1, #3
        SWI 0x011
.end
```

II. Output Screen Shot

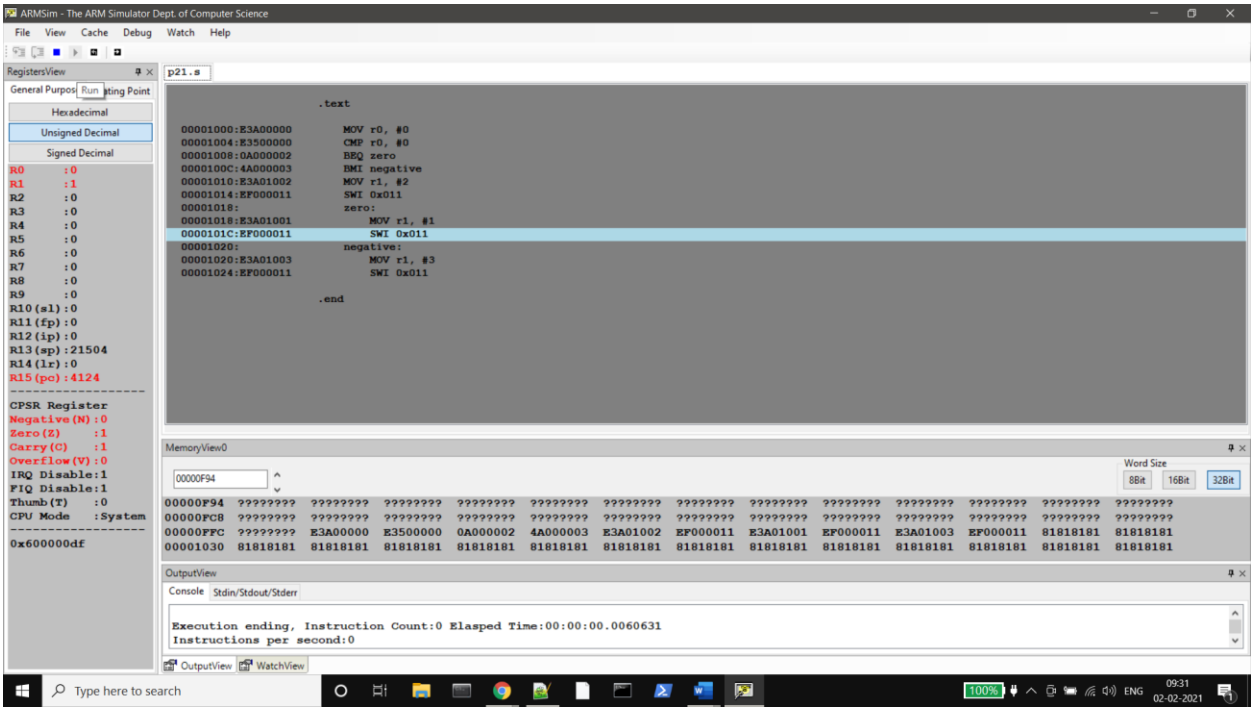
Case Positive:



Case Negative:



Case Zero:



Week#____2_____ Program Number: ___2___

Write an ALP to compare the value of R0 and R1, add if R0 = R1, else subtract

I. ARM Assembly Code for each program

;case equal

```
.text
        MOV r0, #10
        MOV r1, #10
        CMP r0, r1
        BNE subtract
        ADD r2, r1, r0
        SWI 0x011
        subtract:
                SUB r2, r1, r0
                SWI 0x011
.end
```

;case unequal

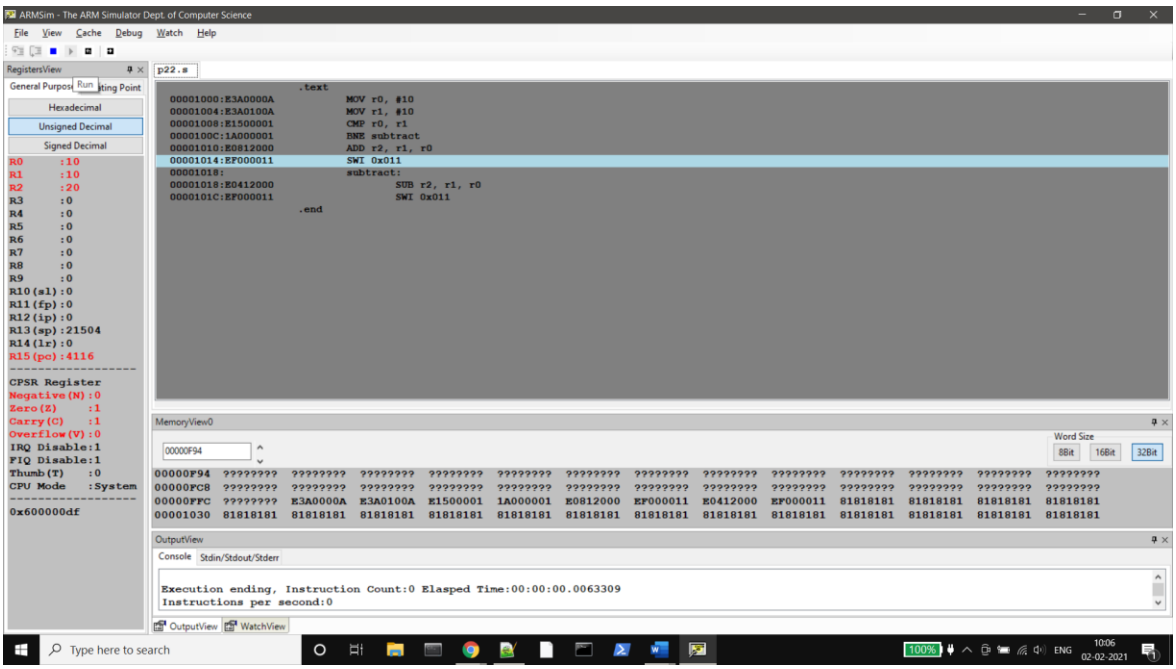
```
.text
        MOV r0, #56
        MOV r1, #10
        CMP r0, r1
        BNE subtract
        ADD r2, r1, r0
        SWI 0x011
        subtract:
                SUB r2, r1, r0
```

SWI 0x011

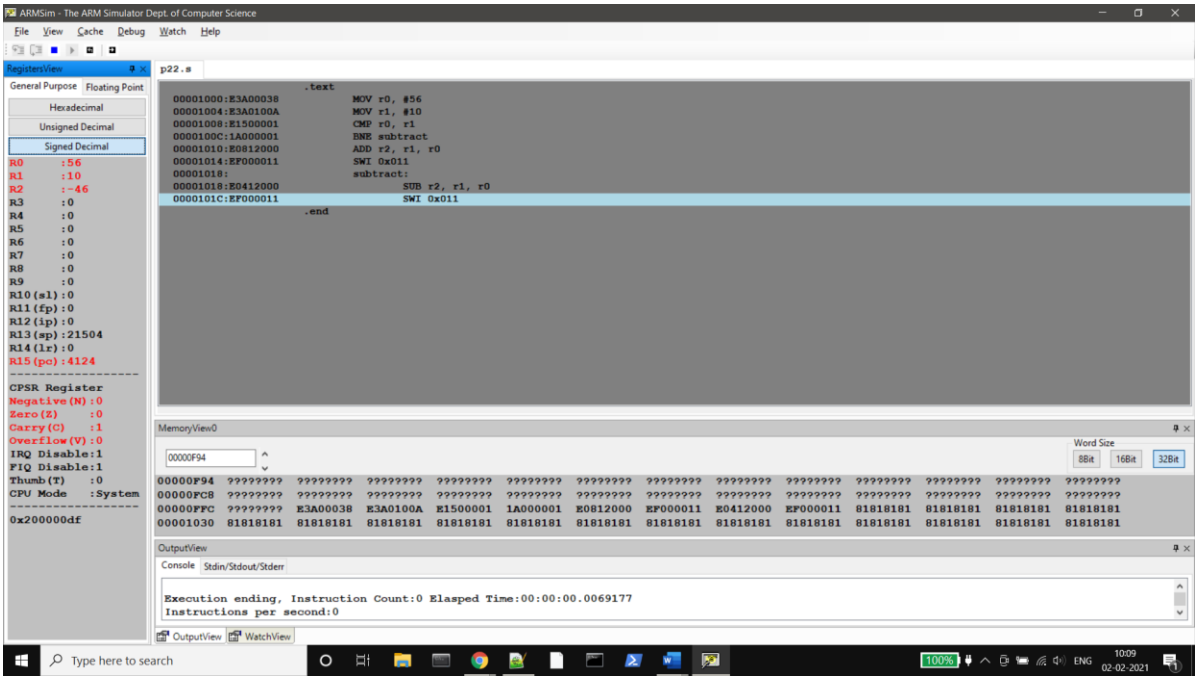
.end

II. Output Screen Shot

Case Equal:



Case Unequal:



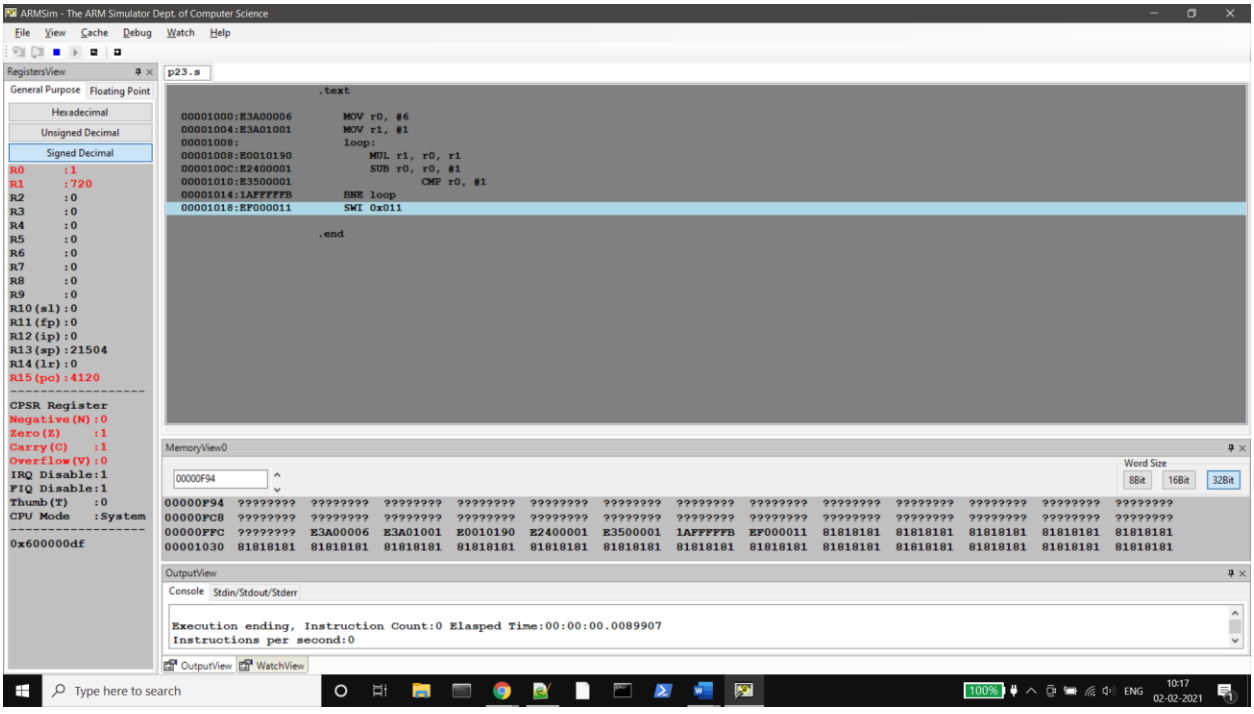
Week#____2_____ Program Number: ___3___

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.

I. ARM Assembly Code for each program

```
.text
   MOV r0, #6
   MOV r1, #1
   loop:
      MUL r1, r0, r1
      SUB r0, r0, #1
      CMP r0, #1
   BNE loop
   SWI 0x011
.end
```

Output Screen Shots



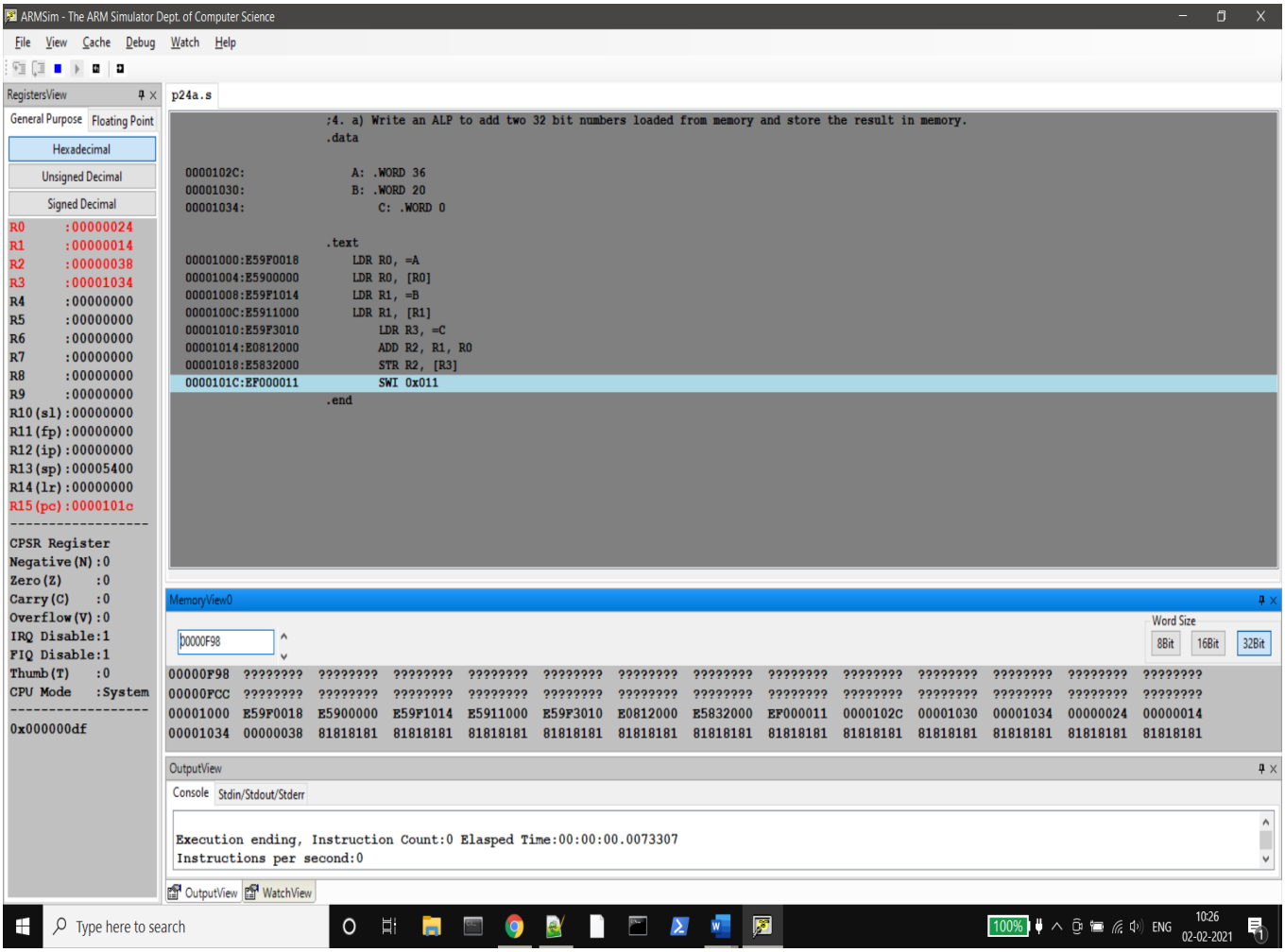
Week#____2_____ Program Number: ___4, a___

4. a) 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 36
   B: .WORD 20
        C: .WORD 0
.text
   LDR R0, =A
   LDR R0, [R0]
   LDR R1, =B
   LDR R1, [R1]
        LDR R3, =C
        ADD R2, R1, R0
        STR R2, [R3]
        SWI 0x011
.end
```

Output Screen Shot


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 :00000024
R1 :00000014
R2 :00000038
R3 :00001034
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):0000101c
CPSR Register
Negative(N):0
Zero(Z) :0
Carry(C) :0
Overflow(V):0
IRQ Disable:1
FIQ Disable:1
Thumb(T) :0
CPU Mode :System
0x000000df
p24a.s
;4. a) Write an ALP to add two 32 bit numbers loaded from memory and store the result in memory.
.data
0000102C: A: .WORD 36
00001030: B: .WORD 20
00001034: C: .WORD 0
.text
00001000:E59F0018 LDR R0, =A
00001004:E5900000 LDR R0, [R0]
00001008:E59F1014 LDR R1, =B
0000100C:E5911000 LDR R1, [R1]
00001010:E59F3010 LDR R3, =C
00001014:E0812000 ADD R2, R1, R0
00001018:E5832000 STR R2, [R3]
0000101C:EF000011 SWI 0x011
.end
MemoryView0
00000F98
Word Size
8Bit 16Bit 32Bit
00000F98 ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ????????
00000FCC ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ????????
00001000 E59F0018 E5900000 E59F1014 E5911000 E59F3010 E0812000 E5832000 EF000011 0000102C 00001030 00001034 00000024 00000014
00001034 00000038 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
OutputView
Console Stdin/Stdout/Stderr
Execution ending, Instruction Count:0 Elasped Time:00:00:00.0073307
Instructions per second:0
OutputView WatchView
Type here to search
100%
ENG
10:26
02-02-2021

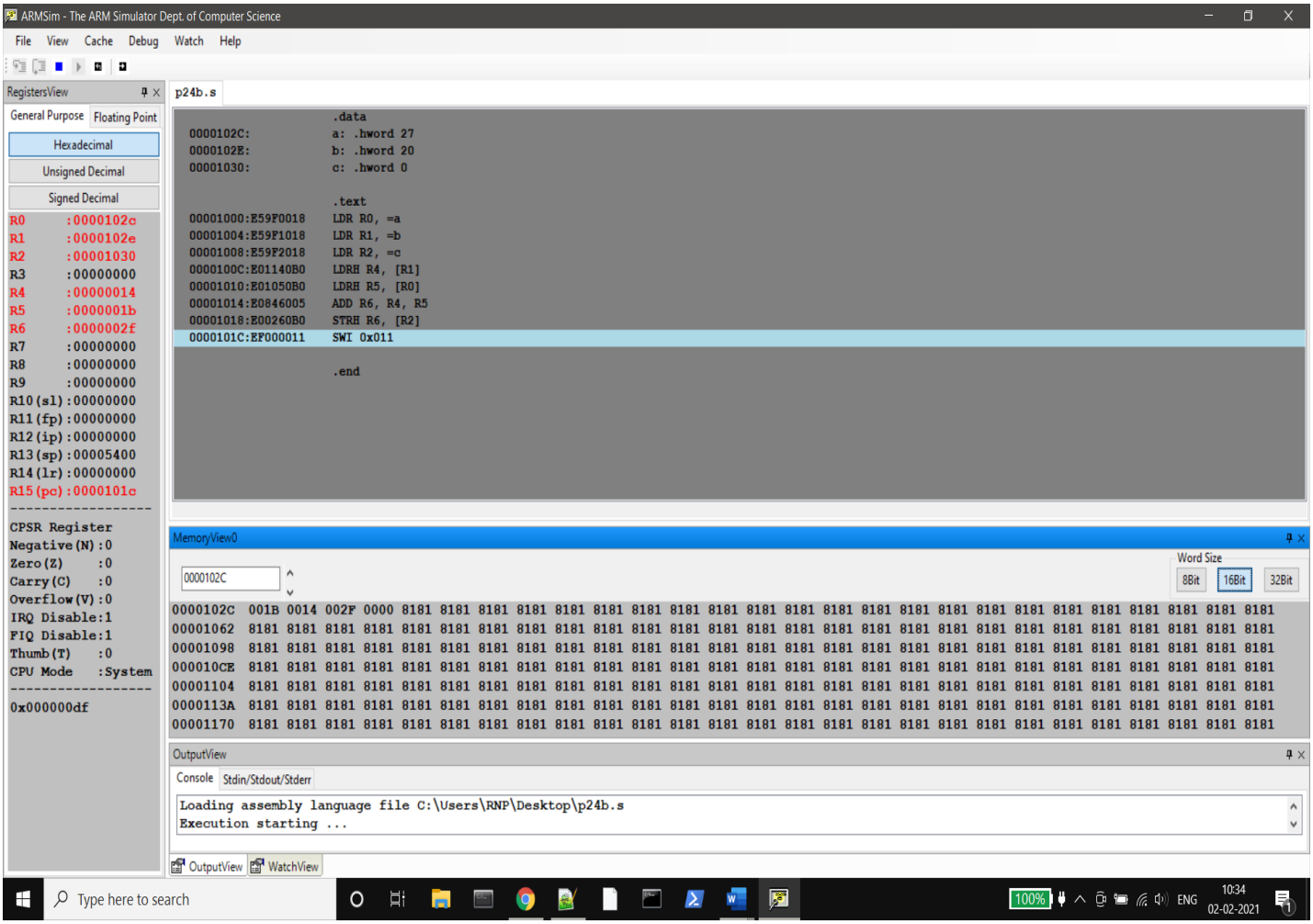
Week#____2_____ Program Number: ___4, b___

4. b.) 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 27
b: .hword 20
c: .hword 0
.text
LDR R0, =a
LDR R1, =b
LDR R2, =c
LDRH R4, [R1]
LDRH R5, [R0]
ADD R6, R4, R5
STRH R6, [R2]
SWI 0x011
.end
```

Output screenshot


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 :0000102c
R1 :0000102e
R2 :00001030
R3 :00000000
R4 :00000014
R5 :0000001b
R6 :0000002f
R7 :00000000
R8 :00000000
R9 :00000000
R10(sl):00000000
R11(fp):00000000
R12(ip):00000000
R13(sp):00005400
R14(lr):00000000
R15(pc):0000101c
CPSR Register
Negative(N):0
Zero(Z) :0
Carry(C) :0
Overflow(V):0
IRQ Disable:1
FIQ Disable:1
Thumb(T) :0
CPU Mode :System
0x000000df
p24b.s
.data
0000102C: a: .hword 27
0000102E: b: .hword 20
00001030: c: .hword 0
.text
00001000:E59F0018 LDR R0, =a
00001004:E59F1018 LDR R1, =b
00001008:E59F2018 LDR R2, =c
0000100C:E01140B0 LDRH R4, [R1]
00001010:E01050B0 LDRH R5, [R0]
00001014:E0846005 ADD R6, R4, R5
00001018:E00260B0 STRH R6, [R2]
0000101C:EF000011 SWI 0x011
.end
MemoryView0
0000102C
Word Size
8Bit 16Bit 32Bit
0000102C 001B 0014 002F 0000 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181
00001062 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181
00001098 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181
000010CE 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181
00001104 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181
0000113A 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181
00001170 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181
OutputView
Console Stdin/Stdout/Stderr
Loading assembly language file C:\Users\RNP\Desktop\p24b.s
Execution starting ...
OutputView WatchView
Type here to search
100%
ENG
10:34
02-02-2021

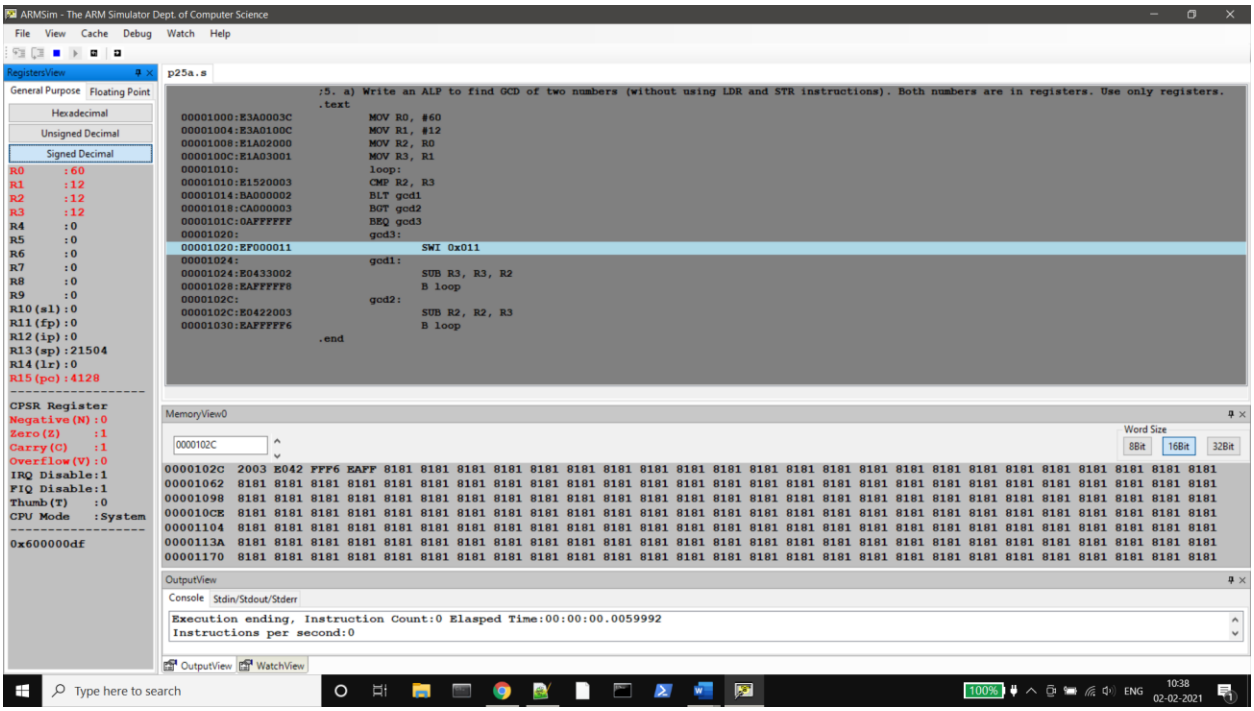
Week#____2_____ Program Number: ___5,a___

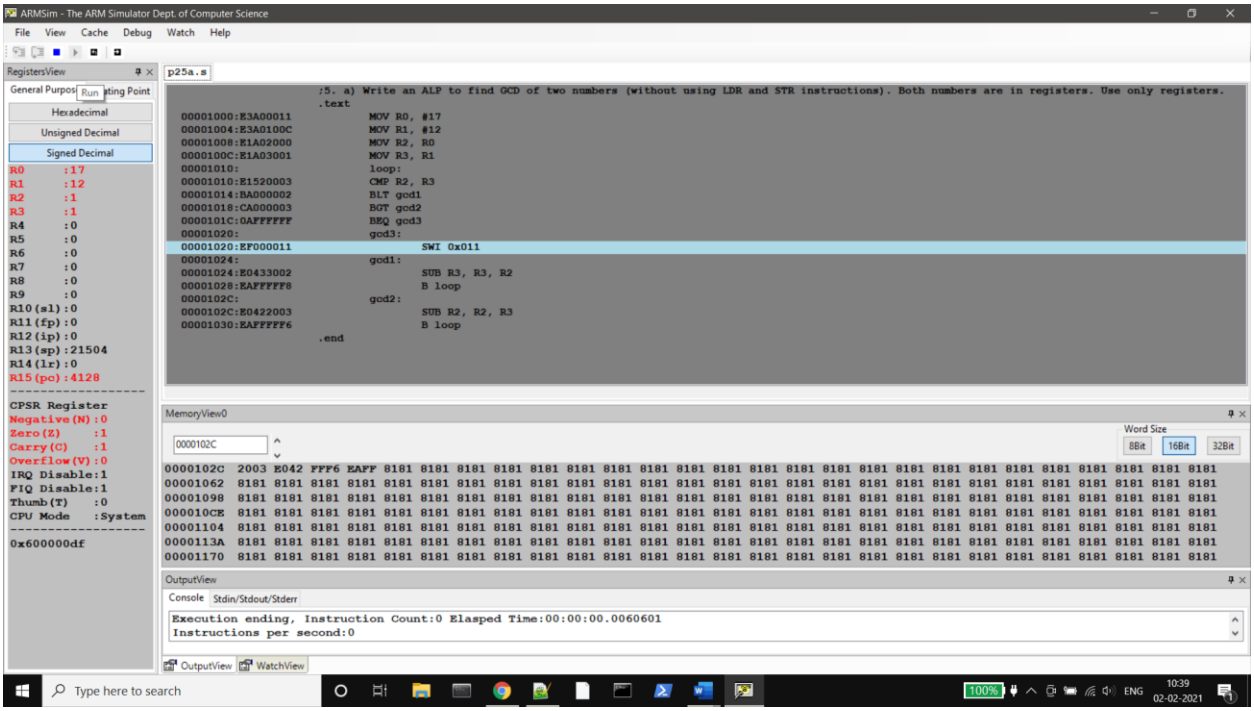
5. a) Write an ALP to find GCD of two numbers (without using LDR and STR instructions). Both numbers are in registers. Use only registers.

I. ARM Assembly Code for each program

```
.text
        MOV R0, #60
        MOV R1, #12
        MOV R2, R0
        MOV R3, R1
        loop:
        CMP R2, R3
        BLT gcd1
        BGT gcd2
        BEQ gcd3
        gcd3:
                SWI 0x011
        gcd1:
                SUB R3, R3, R2
                B loop
        gcd2:
                SUB R2, R2, R3
                B loop
.end
```

Output Screenshot


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 :60
R1 :12
R2 :12
R3 :12
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
p25a.s
;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.
.text
00001000:E3A0003C MOV R0, #60
00001004:E3A0100C MOV R1, #12
00001008:E1A02000 MOV R2, R0
0000100C:E1A03001 MOV R3, R1
00001010: loop:
00001010:E1520003 CMP R2, R3
00001014:BA000002 BLT god1
00001018:CA000003 BGT god2
0000101C:0AFFFFFF BEQ god3
00001020: god3:
00001020:EF000011 SWI 0x011
00001024: god1:
00001024:E0433002 SUB R3, R3, R2
00001028:EAFFFFF8 B loop
0000102C: god2:
0000102C:E0422003 SUB R2, R2, R3
00001030:EAFFFFF6 B loop
.end
MemoryView0
0000102C
Word Size
8Bit 16Bit 32Bit
0000102C 2003 E042 FFF6 EAFF 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181
00001062 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181
00001098 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181
000010CE 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181
00001104 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181
0000113A 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181
00001170 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181
OutputView
Console Stdin/Stdout/Stderr
Execution ending, Instruction Count:0 Elasped Time:00:00:00.0059992
Instructions per second:0
OutputView WatchView
Type here to search
100%
ENG
10:38
02-02-2021


ARMSim - The ARM Simulator Dept. of Computer Science
File View Cache Debug Watch Help
RegistersView
Run
Hexadecimal
Unsigned Decimal
Signed Decimal
R0 :17
R1 :12
R2 :1
R3 :1
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
p25a.s
;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.
.text
00001000:E3A00011 MOV R0, #17
00001004:E3A0100C MOV R1, #12
00001008:E1A02000 MOV R2, R0
0000100C:E1A03001 MOV R3, R1
00001010: loop:
00001010:E1520003 CMP R2, R3
00001014:BA000002 BLT god1
00001018:CA000003 BGT god2
0000101C:0AFFFFFF BEQ god3
00001020: god3:
00001020:EF000011 SWI 0x011
00001024: god1:
00001024:E0433002 SUB R3, R3, R2
00001028:EAFFFFF8 B loop
0000102C: god2:
0000102C:E0422003 SUB R2, R2, R3
00001030:EAFFFFF6 B loop
.end
MemoryView0
0000102C
Word Size
8Bit 16Bit 32Bit
0000102C 2003 E042 FFF6 EAFF 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181
00001062 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181
00001098 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181
000010CE 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181
00001104 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181
0000113A 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181
00001170 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181
OutputView
Console Stdin/Stdout/Stderr
Execution ending, Instruction Count:0 Elasped Time:00:00:00.0060601
Instructions per second:0
OutputView WatchView
Type here to search
100%
ENG
10:39
02-02-2021

Week#____2_____ Program Number: ___5,b___

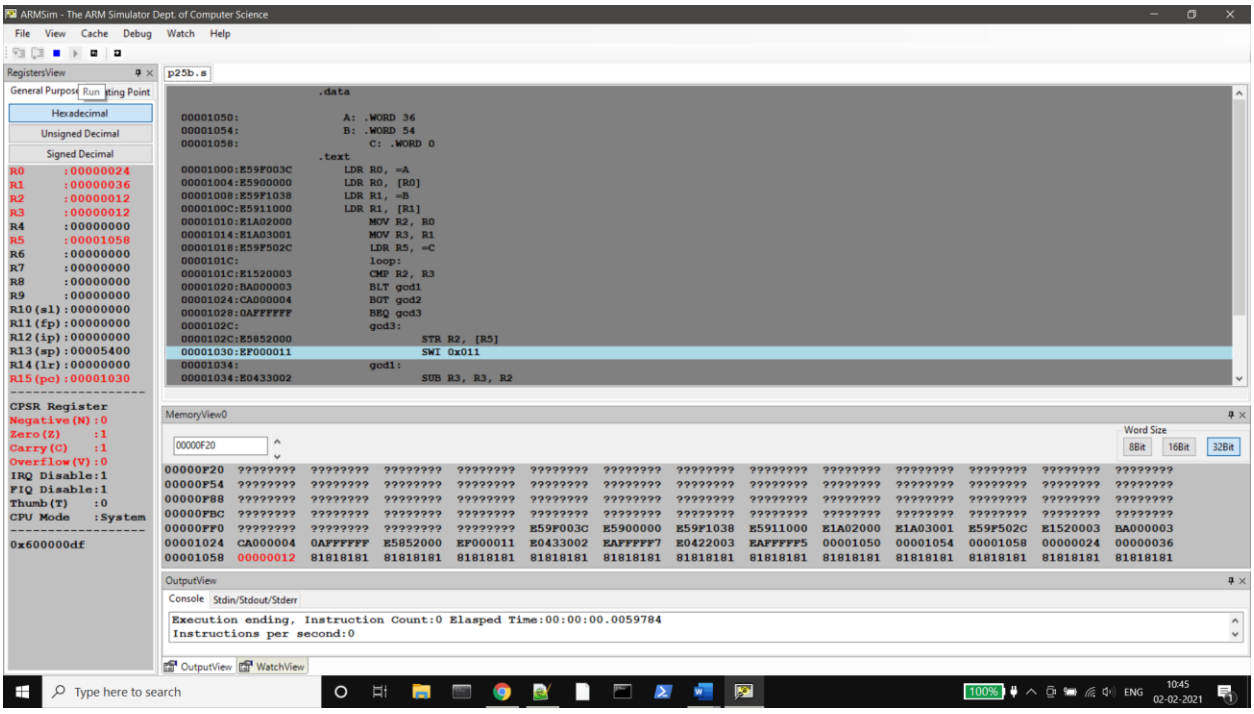
5 b) Write an ALP to find the GCD of given numbers (both numbers in memory). Store result in memory.

I. ARM Assembly Code for each program

```
.data


   A: .WORD 36
   B: .WORD 54
                        C: .WORD 0
.text
   LDR R0, =A
   LDR R0, [R0]
   LDR R1, =B
   LDR R1, [R1]
                        MOV R2, R0
                        MOV R3, R1
                        LDR R5, =C
                        loop:
                        CMP R2, R3
                        BLT gcd1
                        BGT gcd2
                        BEQ gcd3
                        gcd3:
                         STR R2, [R5]
                         SWI 0x011
                        gcd1:
```

```
    SUB R3, R3, R2
    B loop
gcd2:
    SUB R2, R2, R3
    B loop

.end
```

Output Screenshot



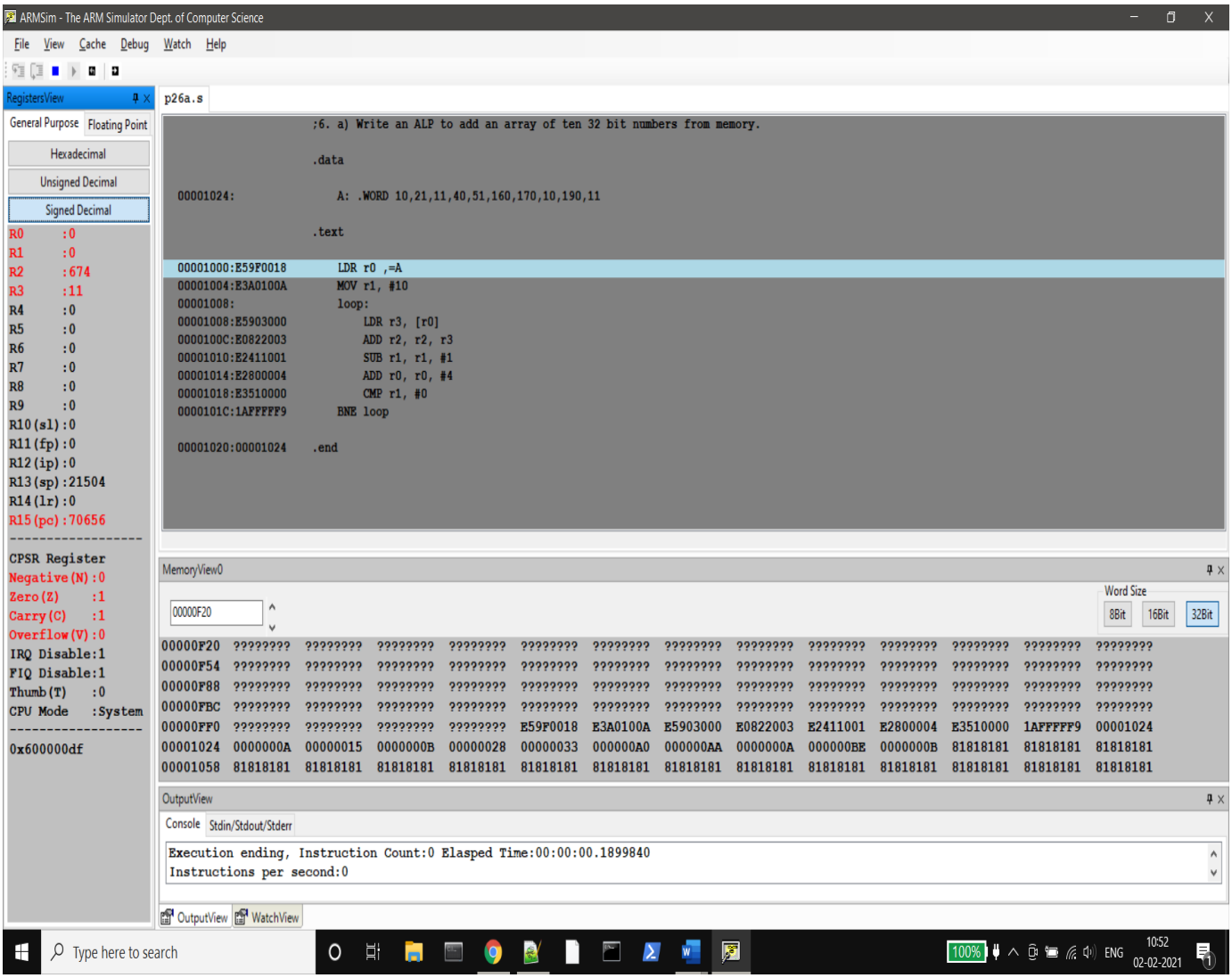
Week#____2_____ Program Number: ___6,a___

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

ARM Assembly Code for each program

```
.data
   A: .WORD 10,21,11,40,51,160,170,10,190,11
.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 Screenshot


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 :0
R1 :0
R2 :674
R3 :11
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):70656
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
p26a.s
;6. a) Write an ALP to add an array of ten 32 bit numbers from memory.
.data
00001024: A: .WORD 10,21,11,40,51,160,170,10,190,11
.text
00001000:E59F0018 LDR r0 ,=A
00001004:E3A0100A MOV r1, #10
00001008: loop:
00001008:E5903000 LDR r3, [r0]
0000100C:E0822003 ADD r2, r2, r3
00001010:E2411001 SUB r1, r1, #1
00001014:E2800004 ADD r0, r0, #4
00001018:E3510000 CMP r1, #0
0000101C:1AFFFFF9 BNE loop
00001020:00001024 .end
MemoryView0
00000F20
Word Size
8Bit 16Bit 32Bit
00000F20 ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ????????
00000F54 ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ????????
00000F88 ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ????????
00000FBC ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ???????? ????????
00000FF0 ???????? ???????? ???????? ???????? E59F0018 E3A0100A E5903000 E0822003 E2411001 E2800004 E3510000 1AFFFFF9 00001024
00001024 0000000A 00000015 0000000B 00000028 00000033 000000A0 000000AA 0000000A 000000BE 0000000B 81818181 81818181 81818181
00001058 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
OutputView
Console Stdin/Stdout/Stderr
Execution ending, Instruction Count:0 Elasped Time:00:00:00.1899840
Instructions per second:0
OutputView WatchView
Type here to search
100%
ENG
10:52
02-02-2021

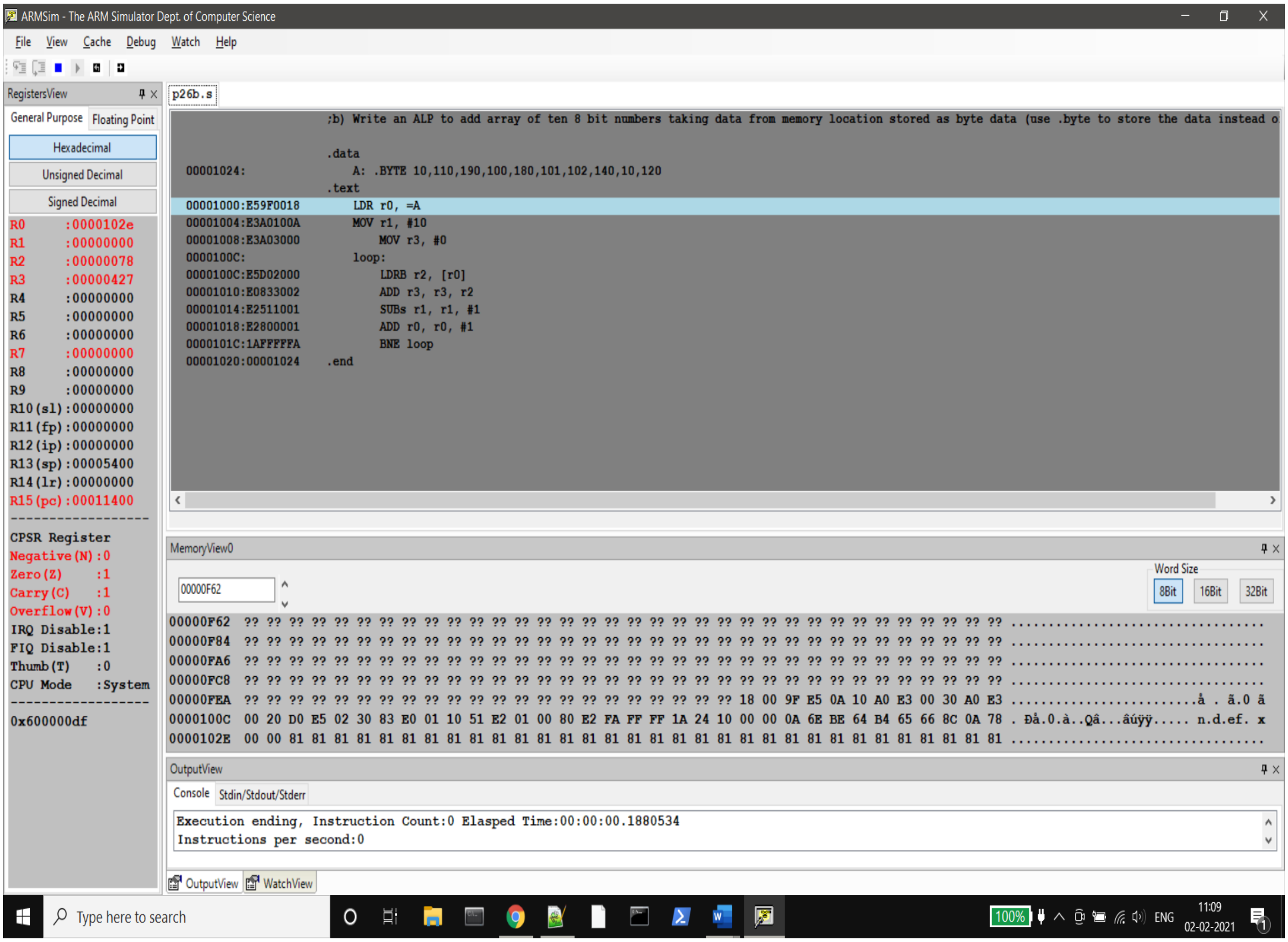
Week#____2_____ Program Number: ___6,b___

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 each program

```
.data
   A: .BYTE 10,110,190,100,180,101,102,140,10,120
.text
   LDR r0, =A
   MOV r1, #10
   MOV r3, #0
   loop:
      LDRB r2, [r0]
      ADD r3, r3, r2
      SUBs r1, r1, #1
      ADD r0, r0, #1
      BNE loop
.end
```

Output Screenshot


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 :0000102e
R1 :00000000
R2 :00000078
R3 :00000427
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):00011400
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
p26b.s
;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 o
.data
00001024: A: .BYTE 10,110,190,100,180,101,102,140,10,120
.text
00001000:E59F0018 LDR r0, =A
00001004:E3A0100A MOV r1, #10
00001008:E3A03000 MOV r3, #0
0000100C: loop:
0000100C:E5D02000 LDRB r2, [r0]
00001010:E0833002 ADD r3, r3, r2
00001014:E2511001 SUBs r1, r1, #1
00001018:E2800001 ADD r0, r0, #1
0000101C:1AFFFFFA BNE loop
00001020:00001024 .end
MemoryView0
00000F62
Word Size
8Bit
16Bit
32Bit
00000F62 ??
00000F84 ??
00000FA6 ??
00000FC8 ??
00000FEA ?? 18 00 9F E5 0A 10 A0 E3 00 30 A0 E3
0000100C 00 20 D0 E5 02 30 83 E0 01 10 51 E2 01 00 80 E2 FA FF FF 1A 24 10 00 00 0A 6E BE 64 B4 65 66 8C 0A 78
0000102E 00 00 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81
OutputView
Console
Stdin/Stdout/Stderr
Execution ending, Instruction Count:0 Elasped Time:00:00:00.1880534
Instructions per second:0
OutputView
WatchView
Type here to search
100%
ENG
11:09
02-02-2021

Week#____2_____ Program Number: ___7____

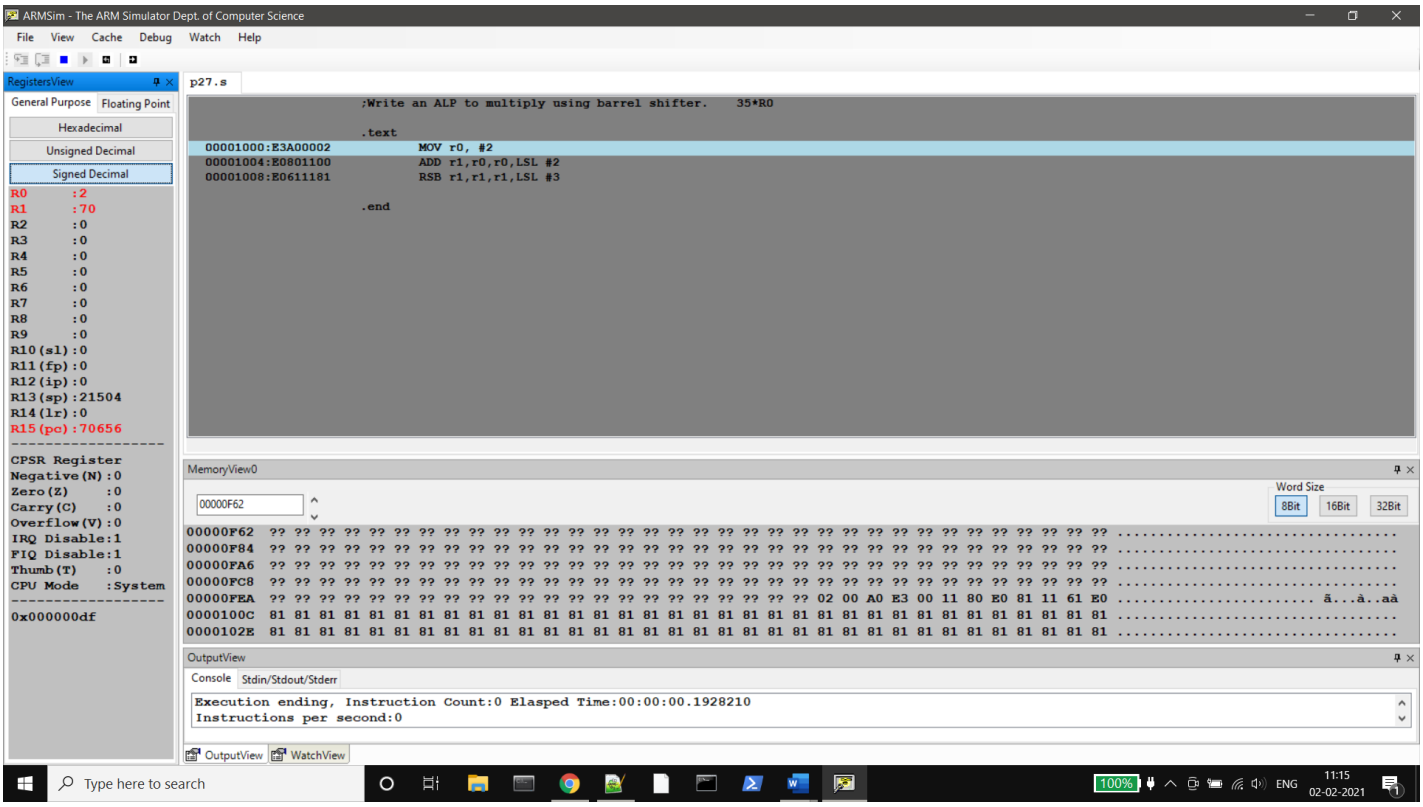
7. Write an ALP to multiply using barrel shifter.

35*R0

ARM Assembly Code for each program

```
.text
        MOV r0, #2
        ADD r1,r0,r0,LSL #2
        RSB r1,r1,r1,LSL #3
.end
```

Output ScreenShot



Week#____2_____ Program Number: ___8___

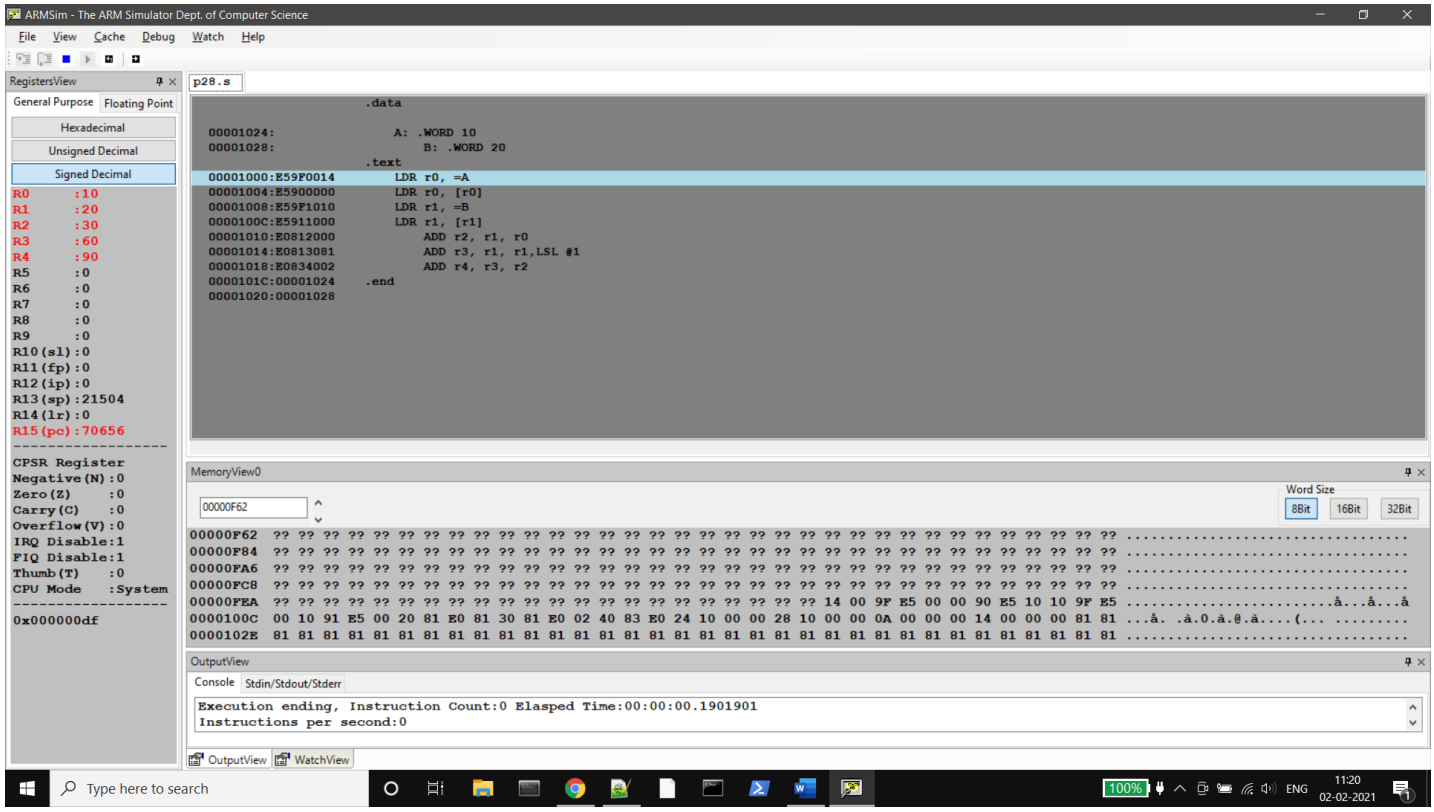
8. Write an ALP to evaluate the expression (A+B) + (3*B), where A and B are memory location.

* Use LSL instruction for multiplication

I. ARM Assembly Code for each program

```
.data
   A: .WORD 10
        B: .WORD 20
.text
   LDR r0, =A
   LDR r0, [r0]
   LDR r1, =B
   LDR r1, [r1]
        ADD r2, r1, r0
        ADD r3, r1, r1, LSL #1
        ADD r4, r3, r2
.end
```

Output Screenshot


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 :10
R1 :20
R2 :30
R3 :60
R4 :90
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):70656
CPSR Register
Negative(N):0
Zero(Z) :0
Carry(C) :0
Overflow(V):0
IRQ Disable:1
FIQ Disable:1
Thumb(T) :0
CPU Mode :System
0x000000df
p28.s
.data
00001024: A: .WORD 10
00001028: B: .WORD 20
.text
00001000:E59F0014 LDR r0, =A
00001004:E5900000 LDR r0, [r0]
00001008:E59F1010 LDR r1, =B
0000100C:E5911000 LDR r1, [r1]
00001010:E0812000 ADD r2, r1, r0
00001014:E0813081 ADD r3, r1, r1,LSL #1
00001018:E0834002 ADD r4, r3, r2
0000101C:00001024 .end
00001020:00001028
MemoryView0
00000F62
Word Size
8Bit 16Bit 32Bit
00000F62 ??
00000F84 ??
00000FA6 ??
00000FC8 ??
00000FEA ?? 14 00 9F E5 00 00 90 E5 10 10 9F E5
0000100C 00 10 91 E5 00 20 81 E0 81 30 81 E0 02 40 83 E0 24 10 00 00 28 10 00 00 0A 00 00 00 14 00 00 00 81 81
0000102E 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81 81
OutputView
Console Stdin/Stdout/Stderr
Execution ending, Instruction Count:0 Elasped Time:00:00:00.1901901
Instructions per second:0
OutputView WatchView
Type here to search
100%
ENG
11:20
02-02-2021