Microprocessor and Computer Architecture Laboratory

UE19CS256

4th Semester, Academic Year 2020-21

Date:5/2/21

Name: Kartik Soni	SRN: PES1UG19CS212	Section D

Week#____3_______Program Number: ____1___

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

I.ARM Assembly Code

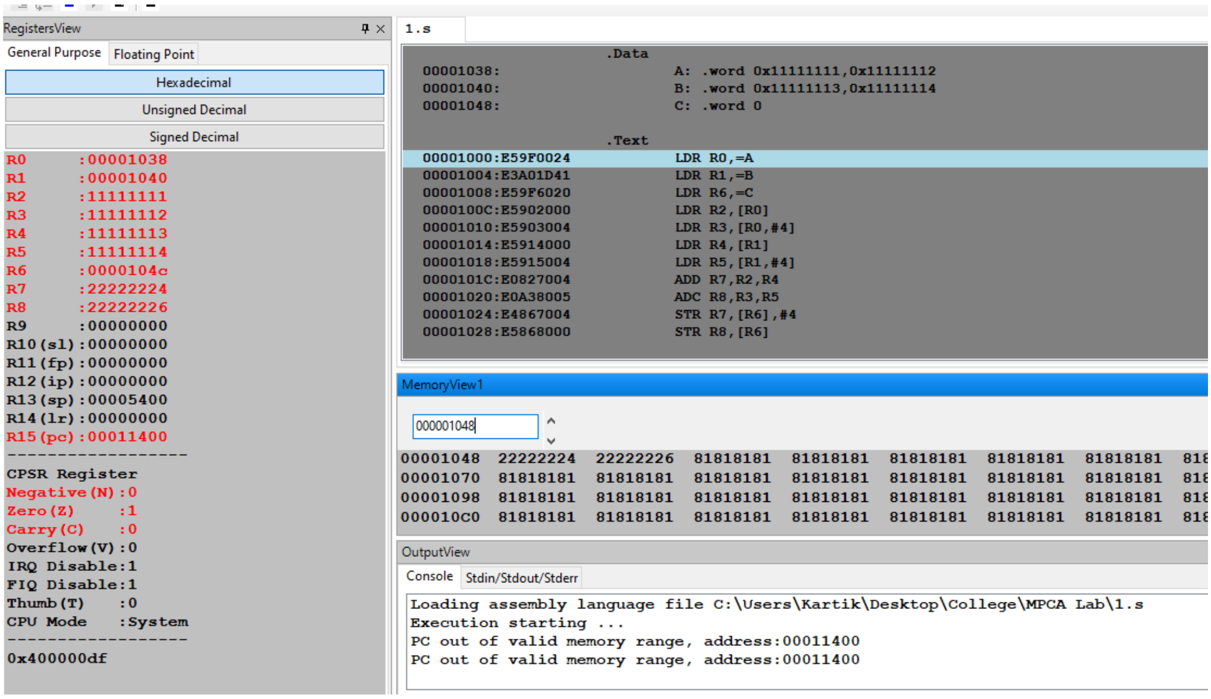
```
.Data
        A: .word 0x11111111,0x11111112
        B: .word 0x11111113,0x11111114
        C: .word 0
.Text
        LDR R0,=A
        LDR R1,=B
        LDR R6,=C
        LDR R2,[R0]
        LDR R3,[R0,#4]
        LDR R4,[R1]
        LDR R5,[R1,#4]
        ADD R7,R2,R4
```

ADC R8,R3,R5

STR R7,[R6],#4

STR R8,[R6]

II OUTPUT



```
R0       :00001038
R1       :00001040
R2       :11111111
R3       :11111112
R4       :11111113
R5       :11111114
R6       :0000104c
R7       :22222224
R8       :22222226
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)   :0
Overflow(V):0
IRQ Disable:1
FIQ Disable:1
Thumb(T)   :0
CPU Mode   :System
------------------
0x400000df
```

```
                          .Data
00001038:                         A: .word 0x11111111,0x11111112
00001040:                         B: .word 0x11111113,0x11111114
00001048:                         C: .word 0

                          .Text
00001000:E59F0024                 LDR R0,=A
00001004:E3A01D41                 LDR R1,=B
00001008:E59F6020                 LDR R6,=C
0000100C:E5902000                 LDR R2,[R0]
00001010:E5903004                 LDR R3,[R0,#4]
00001014:E5914000                 LDR R4,[R1]
00001018:E5915004                 LDR R5,[R1,#4]
0000101C:E0827004                 ADD R7,R2,R4
00001020:E0A38005                 ADC R8,R3,R5
00001024:E4867004                 STR R7,[R6],#4
00001028:E5868000                 STR R8,[R6]
```

```
000001048
00001048  22222224  22222226  81818181  81818181  81818181  81818181  81818181  818
00001070  81818181  81818181  81818181  81818181  81818181  81818181  81818181  818
00001098  81818181  81818181  81818181  81818181  81818181  81818181  81818181  818
000010C0  81818181  81818181  81818181  81818181  81818181  81818181  81818181  818
```

```
Loading assembly language file C:\Users\Kartik\Desktop\College\MPCA Lab\1.s
Execution starting ...
PC out of valid memory range, address:00011400
PC out of valid memory range, address:00011400
```

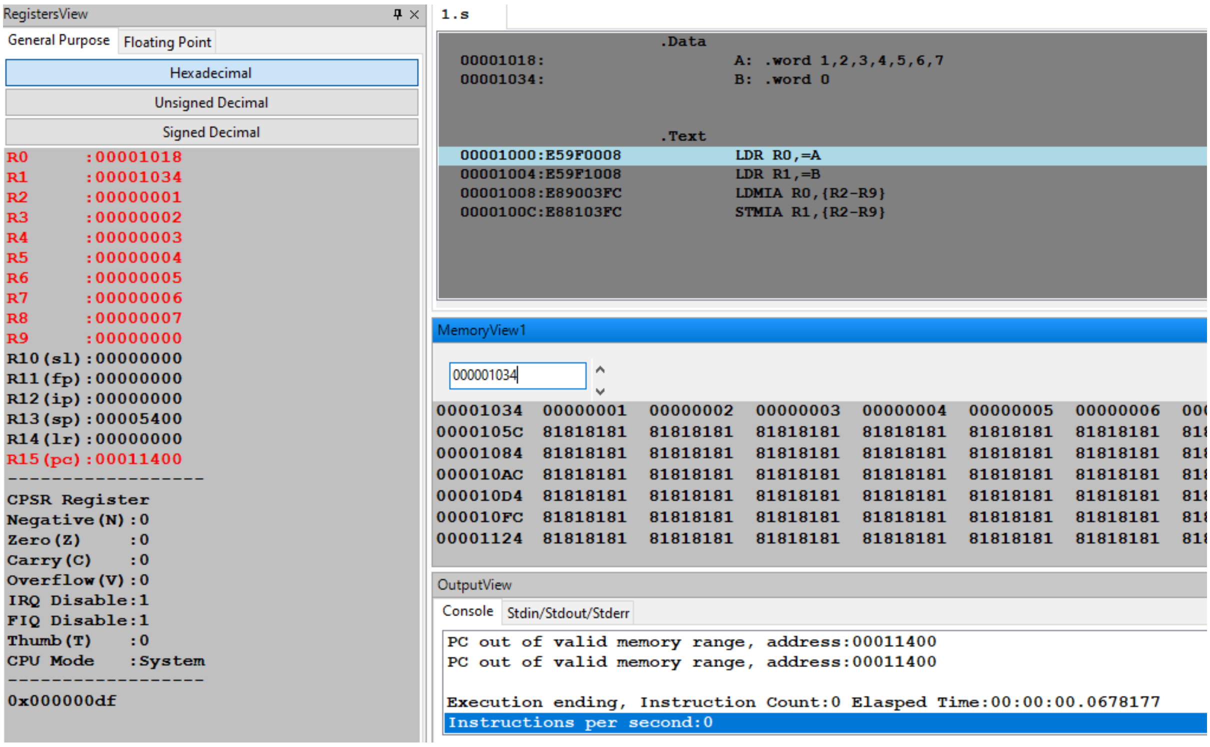
Week#____3_______Program Number: ____2___

Write an ALP to copy n numbers from Memory Location A to Memory Location B

I.ARM Assembly Code

```
.Data

        A: .word 1,2,3,4,5,6,7

        B: .word 0



.Text

        LDR R0,=A

        LDR R1,=B

        LDMIA R0,{R2-R9}

        STMIA R1,{R2-R9}
```

II. Output



```
R0       :00001018
R1       :00001034
R2       :00000001
R3       :00000002
R4       :00000003
R5       :00000004
R6       :00000005
R7       :00000006
R8       :00000007
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)    :0
Carry(C)   :0
Overflow(V):0
IRQ Disable:1
FIQ Disable:1
Thumb(T)   :0
CPU Mode   :System
------------------
0x000000df
```

```
                        .Data
00001018:                       A:  .word 1,2,3,4,5,6,7
00001034:                       B:  .word 0


                        .Text
00001000:E59F0008               LDR R0,=A
00001004:E59F1008               LDR R1,=B
00001008:E89003FC               LDMIA R0,{R2-R9}
0000100C:E88103FC               STMIA R1,{R2-R9}
```

```
000001034
00001034  00000001  00000002  00000003  00000004  00000005  00000006  00
0000105C  81818181  81818181  81818181  81818181  81818181  81818181  81
00001084  81818181  81818181  81818181  81818181  81818181  81818181  81
000010AC  81818181  81818181  81818181  81818181  81818181  81818181  81
000010D4  81818181  81818181  81818181  81818181  81818181  81818181  81
000010FC  81818181  81818181  81818181  81818181  81818181  81818181  81
00001124  81818181  81818181  81818181  81818181  81818181  81818181  81
```

```
PC out of valid memory range, address:00011400
PC out of valid memory range, address:00011400

Execution ending, Instruction Count:0 Elasped Time:00:00:00.0678177
Instructions per second:0
```

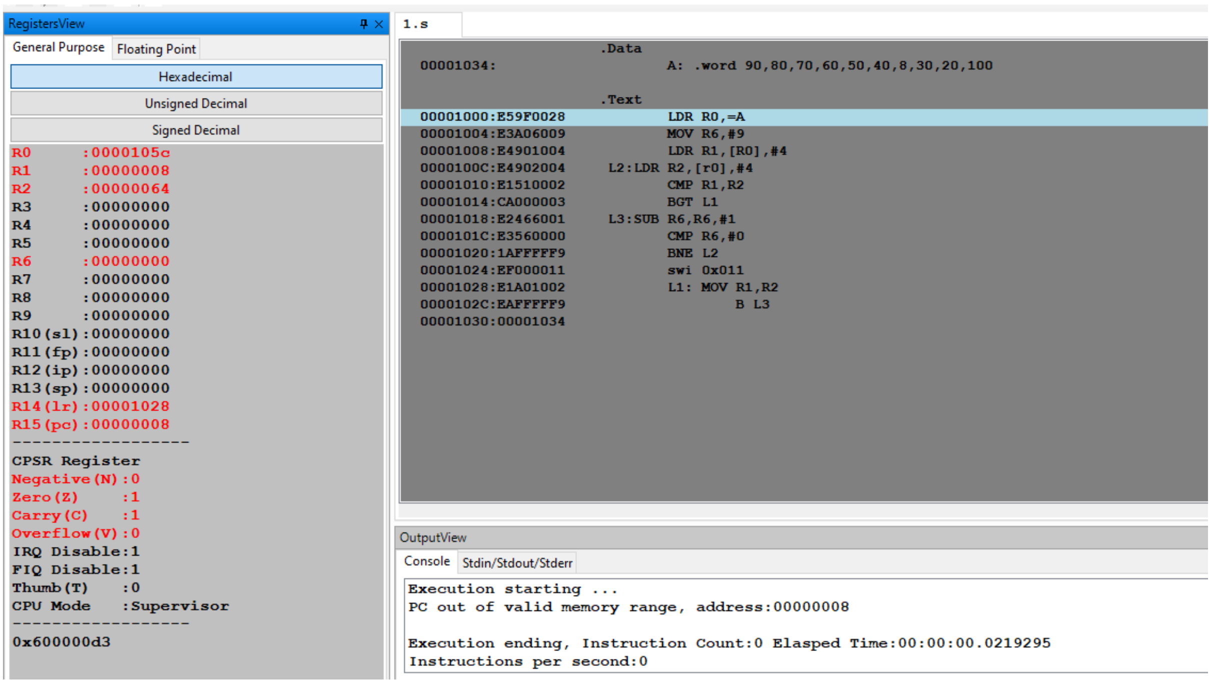
Week#____3_______Program Number: ____3___

Write an ALP to find smallest number in an array of n - 32 bit numbers

I.ARM Assembly Code

```
.Data
        A: .word 90,80,70,60,50,40,8,30,20,100


.Text
        LDR R0,=A
        MOV R6,#9
        LDR R1,[R0],#4
 L2:LDR R2,[r0],#4
        CMP R1,R2
        BGT L1
 L3:SUB R6,R6,#1
        CMP R6,#0
        BNE L2
        swi 0x011
        L1: MOV R1,R2
                B L3
```

II. Output



```
R0       :0000105c
R1       :00000008
R2       :00000064
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):00000000
R14(lr):00001028
R15(pc):00000008
------------------
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    :Supervisor
------------------
0x600000d3
```

```
                      .Data
00001034:                   A: .word 90,80,70,60,50,40,8,30,20,100

                      .Text
00001000:E59F0028           LDR R0,=A
00001004:E3A06009           MOV R6,#9
00001008:E4901004           LDR R1,[R0],#4
0000100C:E4902004       L2:LDR R2,[r0],#4
00001010:E1510002           CMP R1,R2
00001014:CA000003           BGT L1
00001018:E2466001       L3:SUB R6,R6,#1
0000101C:E3560000           CMP R6,#0
00001020:1AFFFFF9           BNE L2
00001024:EF000011           swi 0x011
00001028:E1A01002           L1: MOV R1,R2
0000102C:EAFFFFF9                   B L3
00001030:00001034
```

```
Execution starting ...
PC out of valid memory range, address:00000008

Execution ending, Instruction Count:0 Elasped Time:00:00:00.0219295
Instructions per second:0
```

Week#____3_______Program Number: ____4a___

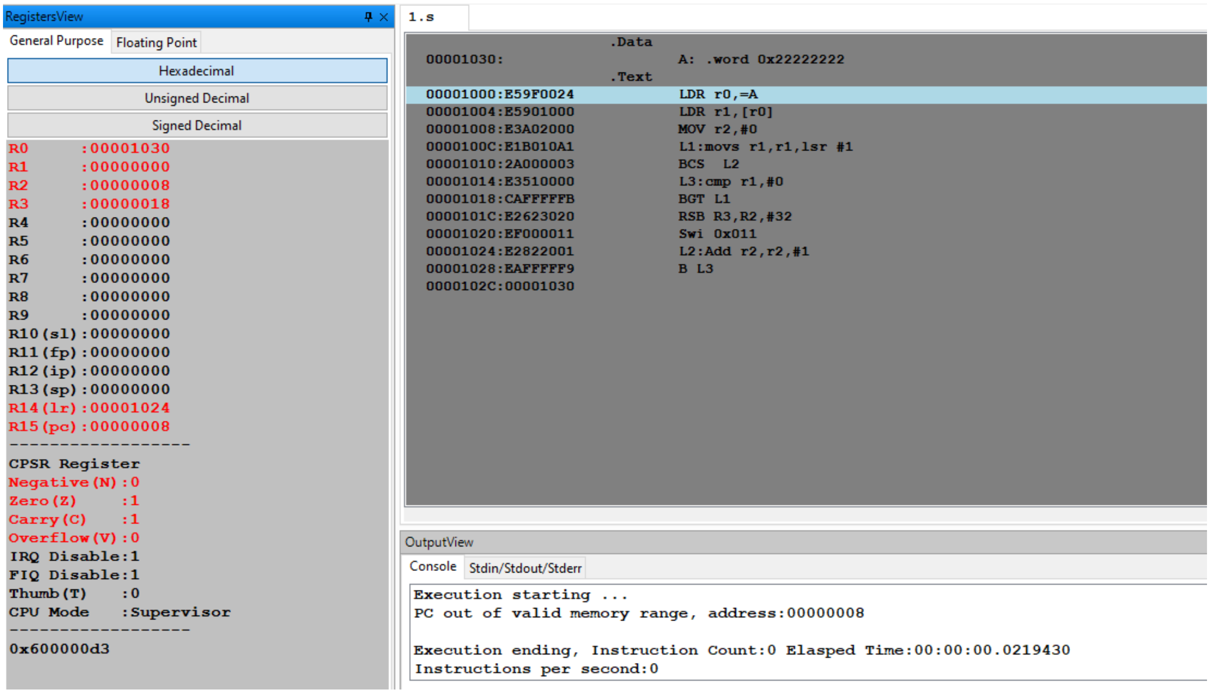
Write an ALP to count the number of 1’s and

0’s in a given 32 bit number.

I.ARM Assembly Code

```
. .Data
        A: .word 0x22222222
.Text
        LDR r0,=A
        LDR r1,[r0]
        MOV r2,#0
        L1:movs r1,r1,lsr #1
        BCS  L2
        L3:cmp r1,#0
        BGT L1
        RSB R3,R2,#32
        Swi 0x011
        L2:Add r2,r2,#1
        B L3
```

II. Output(R2 contains 8)



```
R0      :00001030
R1      :00000000
R2      :00000008
R3      :00000018
R4      :00000000
R5      :00000000
R6      :00000000
R7      :00000000
R8      :00000000
R9      :00000000
R10(sl):00000000
R11(fp):00000000
R12(ip):00000000
R13(sp):00000000
R14(lr):00001024
R15(pc):00000008
------------------
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   :Supervisor
------------------
0x600000d3
```

```
                        .Data
00001030:                       A: .word 0x22222222
                        .Text
00001000:E59F0024               LDR r0,=A
00001004:E5901000               LDR r1,[r0]
00001008:E3A02000               MOV r2,#0
0000100C:E1B010A1               L1:movs r1,r1,lsr #1
00001010:2A000003               BCS  L2
00001014:E3510000               L3:cmp r1,#0
00001018:CAFFFFFB               BGT L1
0000101C:E2623020               RSB R3,R2,#32
00001020:EF000011               Swi 0x011
00001024:E2822001               L2:Add r2,r2,#1
00001028:EAFFFFF9               B L3
0000102C:00001030
```

```
Execution starting ...
PC out of valid memory range, address:00000008

Execution ending, Instruction Count:0 Elasped Time:00:00:00.0219430
Instructions per second:0
```

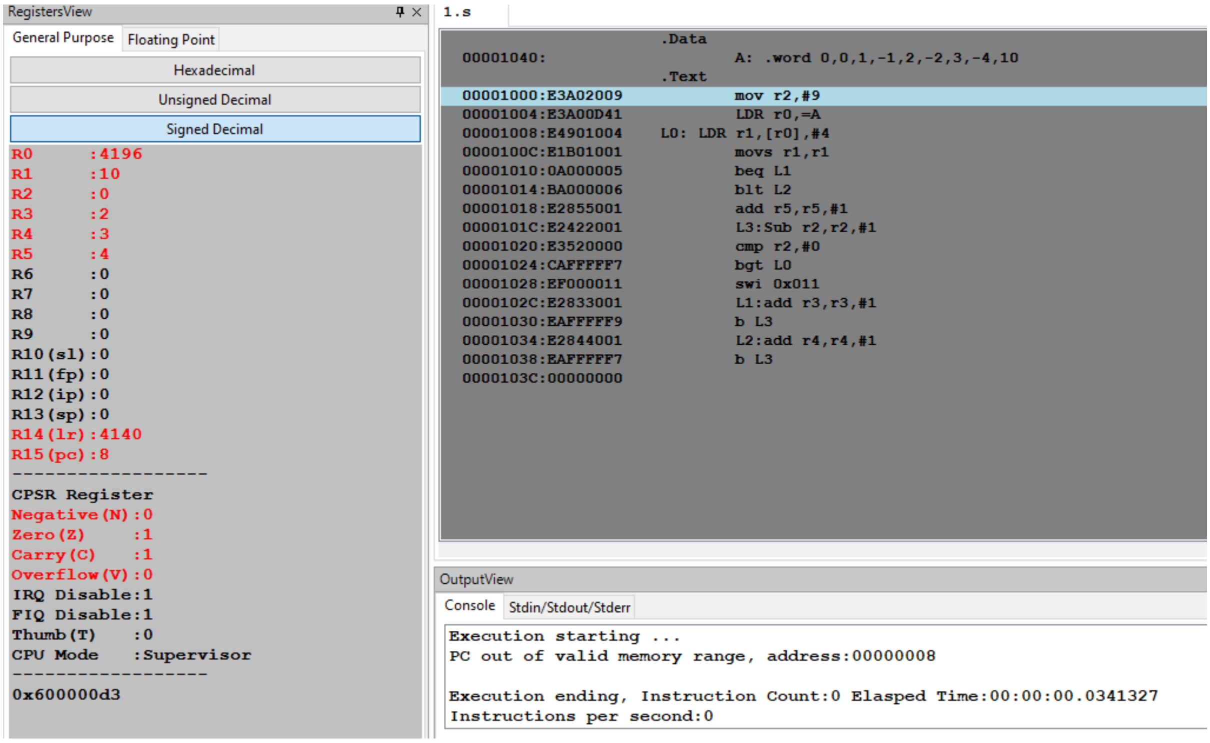
Week#____3_______Program Number: ____4b___

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

I.ARM Assembly Code

```
.Data
        A: .word 0,0,1,-1,2,-2,3,-4,10
.Text
        mov r2,#9
        LDR r0,=A
L0: LDR r1,[r0],#4
        movs r1,r1
        beq L1
        blt L2
        add r5,r5,#1
        L3:Sub r2,r2,#1
        cmp r2,#0
        bgt L0
        swi 0x011
        L1:add r3,r3,#1
        b L3
        L2:add r4,r4,#1
        b L3
```

II. Output



```
R0       :4196
R1       :10
R2       :0
R3       :2
R4       :3
R5       :4
R6       :0
R7       :0
R8       :0
R9       :0
R10(sl):0
R11(fp):0
R12(ip):0
R13(sp):0
R14(lr):4140
R15(pc):8
------------------
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    :Supervisor
------------------
0x600000d3
```

```
                    .Data
00001040:                   A: .word 0,0,1,-1,2,-2,3,-4,10
                    .Text
00001000:E3A02009           mov r2,#9
00001004:E3A00D41           LDR r0,=A
00001008:E4901004   L0: LDR r1,[r0],#4
0000100C:E1B01001           movs r1,r1
00001010:0A000005           beq L1
00001014:BA000006           blt L2
00001018:E2855001           add r5,r5,#1
0000101C:E2422001           L3:Sub r2,r2,#1
00001020:E3520000           cmp r2,#0
00001024:CAFFFFF7           bgt L0
00001028:EF000011           swi 0x011
0000102C:E2833001           L1:add r3,r3,#1
00001030:EAFFFFF9           b L3
00001034:E2844001           L2:add r4,r4,#1
00001038:EAFFFFF7           b L3
0000103C:00000000
```

```
Execution starting ...
PC out of valid memory range, address:00000008

Execution ending, Instruction Count:0 Elasped Time:00:00:00.0341327
Instructions per second:0
```

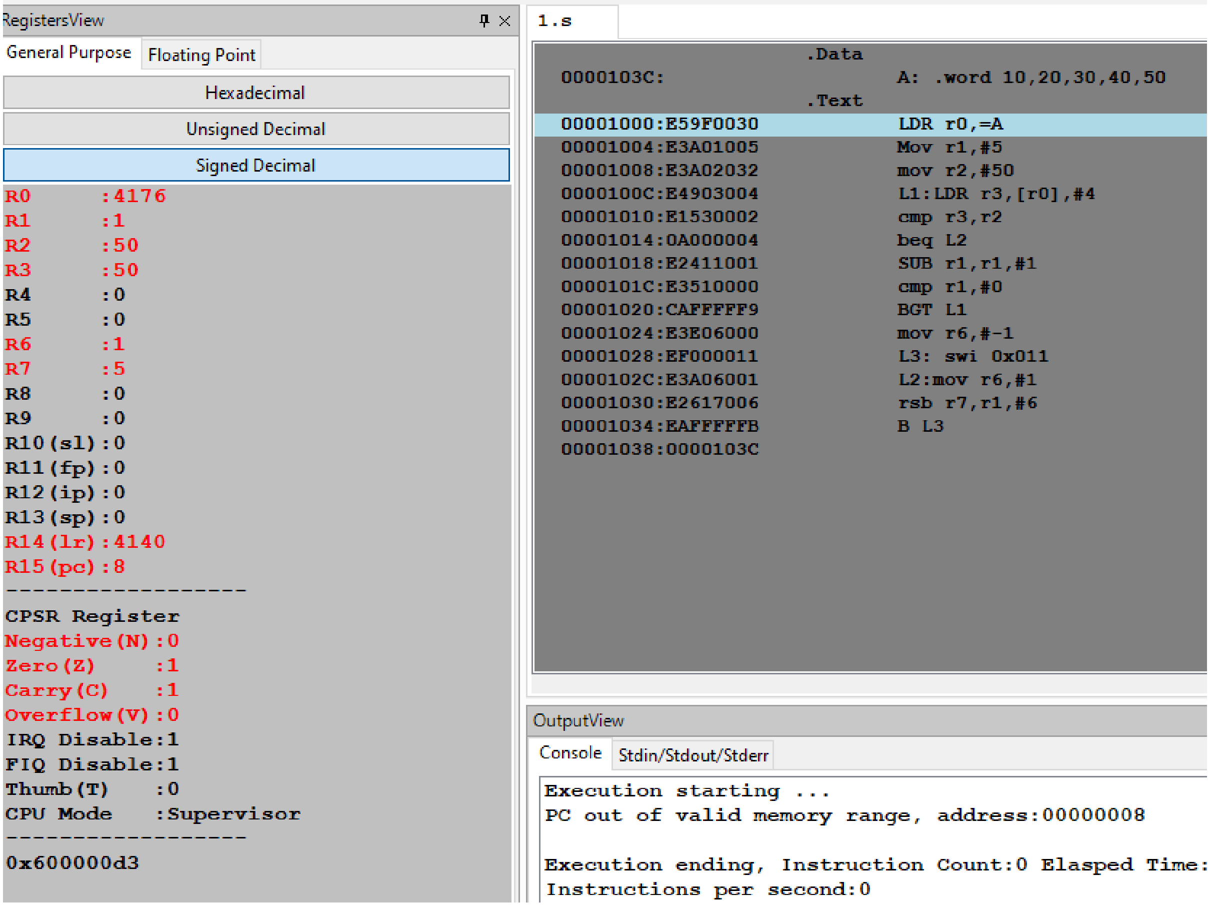
Week#____3_______Program Number: ____5___

Write an ALP to check whether a given number is present in array using Linear Search (Without SWI 0x02), if found move +1 to R6 and key position to R7 else move -1 to R6 (if number not found)

I.ARM Assembly Code

```
.Data
        A: .word 10,20,30,40,50
.Text
        LDR r0,=A
        Mov r1,#5
        mov r2,#50
        L1:LDR r3,[r0],#4
        cmp r3,r2
        beq L2
        SUB r1,r1,#1
        cmp r1,#0
        BGT L1
        mov r6,#-1
        L3: swi 0x011
        L2:mov r6,#1
        rsb r7,r1,#6
        B L3
```

II. Output



```
R0        :4176
R1        :1
R2        :50
R3        :50
R4        :0
R5        :0
R6        :1
R7        :5
R8        :0
R9        :0
R10(sl):0
R11(fp):0
R12(ip):0
R13(sp):0
R14(lr):4140
R15(pc):8
------------------
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    :Supervisor
------------------
0x600000d3
```

```
                       .Data
0000103C:                          A: .word 10,20,30,40,50
                       .Text
00001000:E59F0030                  LDR r0,=A
00001004:E3A01005                  Mov r1,#5
00001008:E3A02032                  mov r2,#50
0000100C:E4903004                  L1:LDR r3,[r0],#4
00001010:E1530002                  cmp r3,r2
00001014:0A000004                  beq L2
00001018:E2411001                  SUB r1,r1,#1
0000101C:E3510000                  cmp r1,#0
00001020:CAFFFFF9                  BGT L1
00001024:E3E06000                  mov r6,#-1
00001028:EF000011                  L3: swi 0x011
0000102C:E3A06001                  L2:mov r6,#1
00001030:E2617006                  rsb r7,r1,#6
00001034:EAFFFFFB                  B L3
00001038:0000103C
```

```
Execution starting ...
PC out of valid memory range, address:00000008

Execution ending, Instruction Count:0 Elasped Time:
Instructions per second:0
```

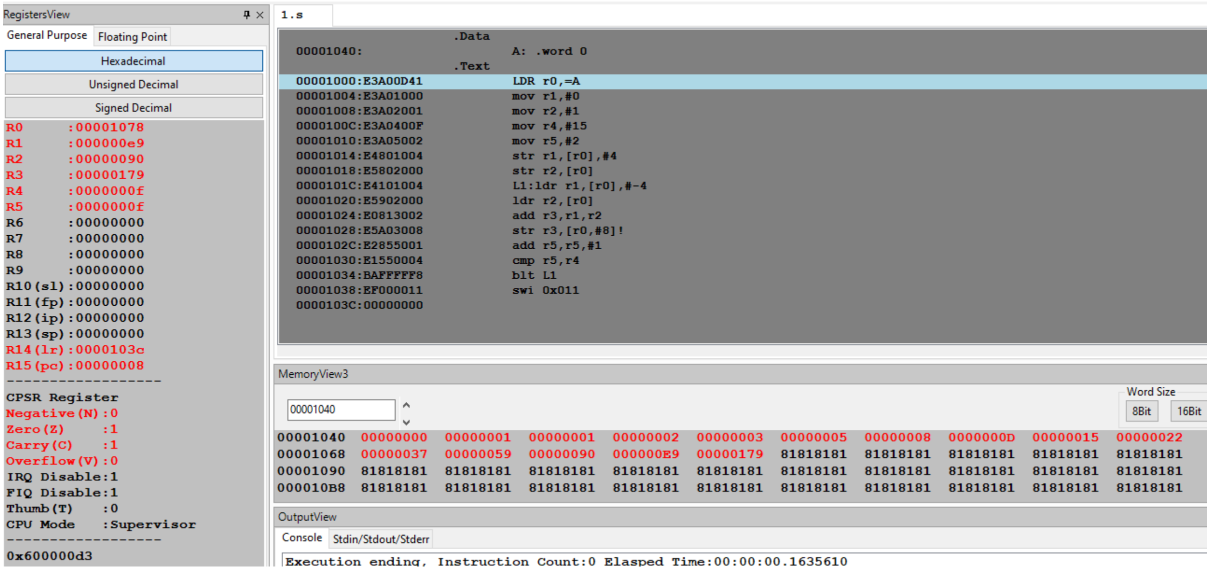
Week#____3_______Program Number: ____6___

Write an ALP to generate Fibonacci Series and store them in an array

I.ARM Assembly Code

```
.Data
        A: .word 0
.Text
        LDR r0,=A
        mov r1,#0
        mov r2,#1
        mov r4,#15
        mov r5,#2
        str r1,[r0],#4
        str r2,[r0]
        L1:ldr r1,[r0],#-4
        ldr r2,[r0]
        add r3,r1,r2
        str r3,[r0,#8]!
        add r5,r5,#1
        cmp r5,r4
        blt L1
        swi 0x011
```

II. Output



```
R0        :00001078
R1        :000000e9
R2        :00000090
R3        :00000179
R4        :0000000f
R5        :0000000f
R6        :00000000
R7        :00000000
R8        :00000000
R9        :00000000
R10(sl)   :00000000
R11(fp)   :00000000
R12(ip)   :00000000
R13(sp)   :00000000
R14(lr)   :0000103c
R15(pc)   :00000008
--------------------
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   :Supervisor
--------------------
0x600000d3
```

```
                    .Data
00001040:                   A:  .word 0
                    .Text
00001000:E3A00D41           LDR r0,=A
00001004:E3A01000           mov r1,#0
00001008:E3A02001           mov r2,#1
0000100C:E3A0400F           mov r4,#15
00001010:E3A05002           mov r5,#2
00001014:E4801004           str r1,[r0],#4
00001018:E5802000           str r2,[r0]
0000101C:E4101004           L1:ldr r1,[r0],#-4
00001020:E5902000           ldr r2,[r0]
00001024:E0813002           add r3,r1,r2
00001028:E5A03008           str r3,[r0,#8]!
0000102C:E2855001           add r5,r5,#1
00001030:E1550004           cmp r5,r4
00001034:BAFFFFF8           blt L1
00001038:EF000011           swi 0x011
0000103C:00000000
```

```
00001040  00000000  00000001  00000001  00000002  00000003  00000005  00000008  0000000D  00000015  00000022
00001068  00000037  00000059  00000090  000000E9  00000179  81818181  81818181  81818181  81818181  81818181
00001090  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181
000010B8  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181
```

```
Execution ending, Instruction Count:0 Elasped Time:00:00:00.1635610
```