UE23CS251B

MPCA_LAB -2	
NAME	*Chirag K M*
SEC	*C*
SRN	*PES1UG23CS167*
SEM	*4*

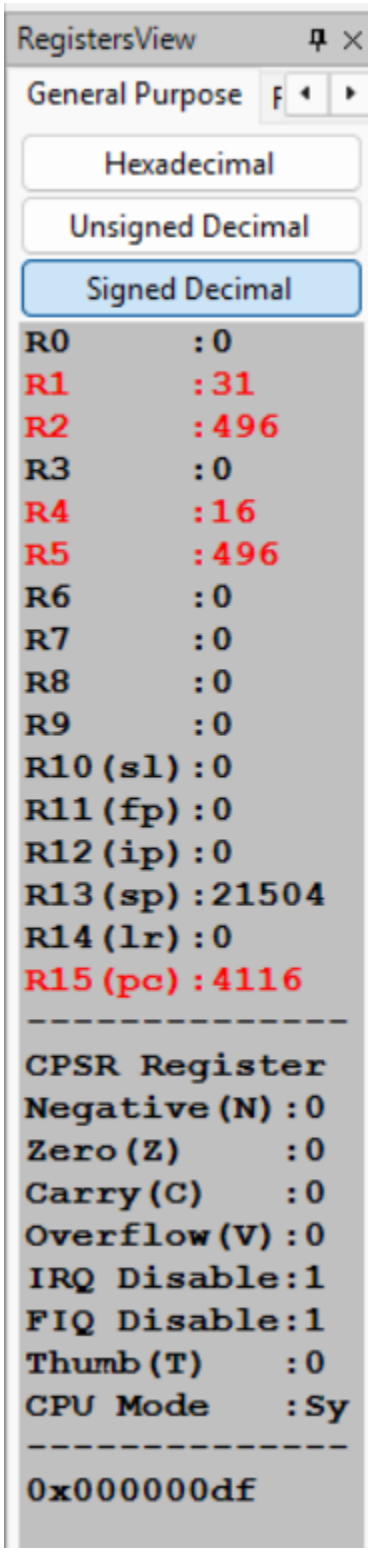
Q1. Write an ALP using ARM7TDMI to perform to multiplication of 16X31 without using mul instructions.
(Hint: barrel shifter instructions.)

Program screen shot:

q1.s

```
                    .text
00001000:E3A0101F           MOV R1,#31          ;31*16
00001004:E1A02201           MOV R2,R1, LSL #4

00001008:E3A04010           MOV R4,#16          ;16*31
0000100C:E1A05284           MOV R5,R4, LSL #5
00001010:E0455004           SUB R5,R5,R4
00001014:EF000011    SWI 0X011
```

Screenshot of Register set output:



Q2.Write an ALP using ARM7TDMI to Classify the given set of numbers as positive OR negative and also store them in different memory locations.

.data

A: .word 1,2,3,4,-1,5,-2,-3,6,0

POS: .word 0,0,0,0,0,0,0,0,0,0

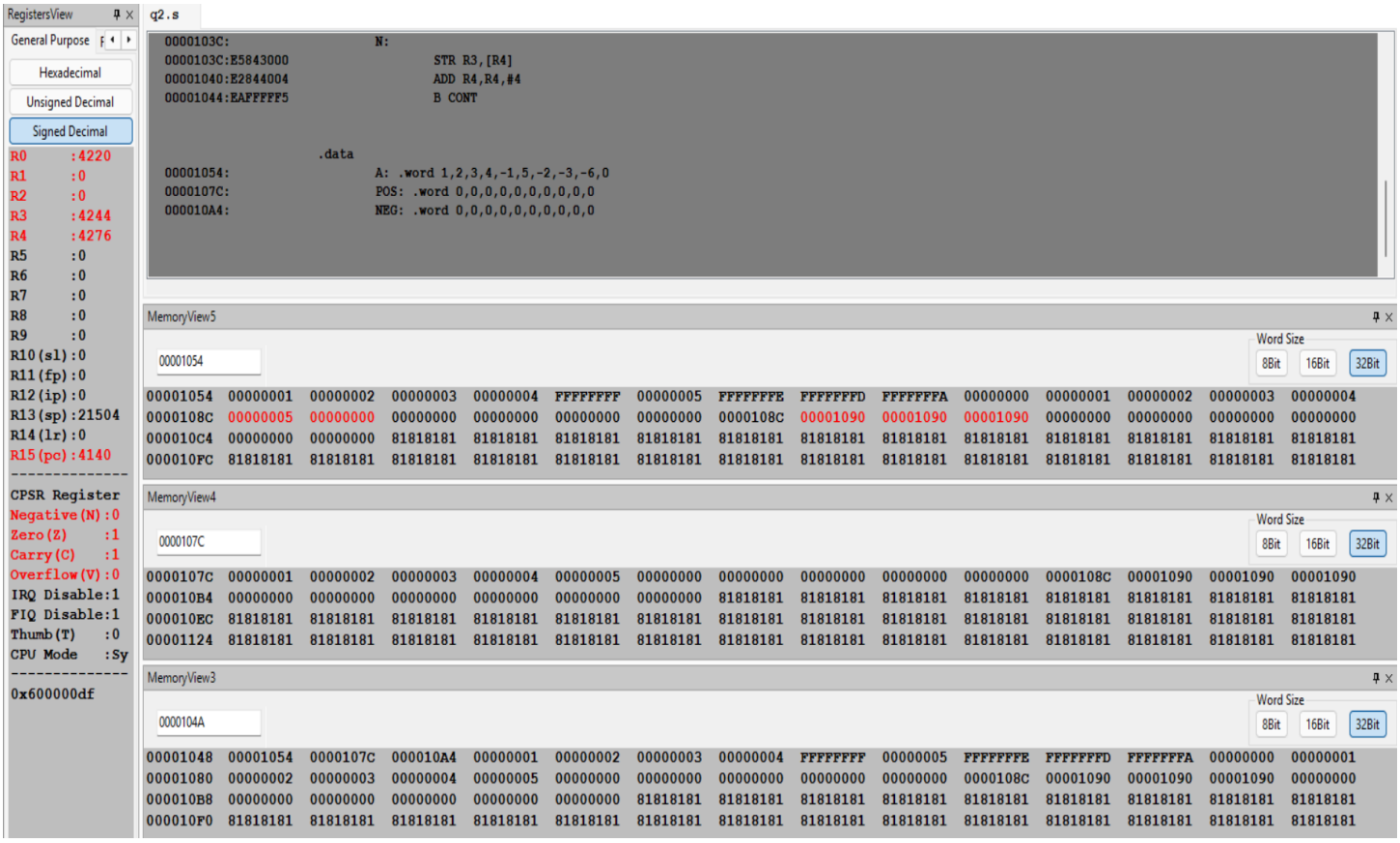
NEG: .word 0,0,0,0,0,0,0,0,0,0

Program screen shot:

q2.s

```
                        .text
00001000:E59F0040               LDR R0,=A
00001004:E3A0100A               MOV R1,#10
00001008:E59F303C               LDR R3,=POS
0000100C:E59F403C               LDR R4,=NEG

00001010:                       LOOP:
00001010:E5902000                       LDR R2,[R0]
00001014:E3520000                       CMP R2,#0
00001018:AA000004                       BGE P
0000101C:EA000006                       B   N
00001020:                       CONT:
00001020:E2800004                       ADD R0,R0,#4
00001024:E2511001                       SUBS R1,R1,#1
00001028:1AFFFFF8                       BNE LOOP
0000102C:EF000011               SWI 0X011


00001030:                       P:
00001030:E5832000                       STR R2,[R3]
00001034:E2833004                       ADD R3,R3,#4
00001038:EAFFFFF8                       B CONT
0000103C:                       N:
0000103C:E5843000                       STR R3,[R4]
00001040:E2844004                       ADD R4,R4,#4
00001044:EAFFFFF5                       B CONT


                        .data
00001054:                       A: .word 1,2,3,4,-1,5,-2,-3,-6,0
0000107C:                       POS: .word 0,0,0,0,0,0,0,0,0,0
000010A4:                       NEG: .word 0,0,0,0,0,0,0,0,0,0
```

Screenshot of Register set and Memory output:



Q3.Write an ALP using ARM7TDMI to add only negative numbers stored in memory location for a given set of numbers(having both positive and negative numbers) and store the sum of negative numbers in the memory location.

Array:.WORD 1,2,3,4,-1,5,-2,-3,6,0

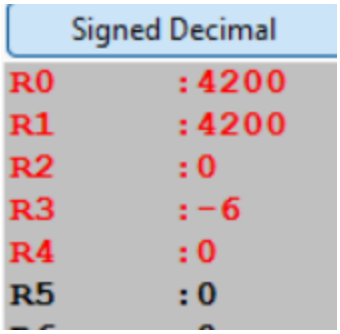
negsum:.WORD

Program screen shot:

q3.s

```
                        .text
00001000:E3A00D41               LDR R0,=Array
00001004:E59F102C               LDR R1,=negsum
00001008:E3A0200A               MOV R2,#10
0000100C:E3A03000               MOV R3,#0

00001010:                       LOOP:
00001010:E5904000                       LDR R4,[R0]
00001014:E3540000                       CMP R4,#0
00001018:BA000004                       BLT Sum
0000101C:                       CONT:
0000101C:E2522001                       SUBS R2,R2,#1
00001020:E2800004                       ADD R0,R0,#4
00001024:1AFFFFF9                       BNE  LOOP
00001028:E5813000               STR R3,[R1]
0000102C:EF000011               SWI 0X011

00001030:                       Sum:
00001030:E0833004                       ADD R3,R3,R4
00001034:EAFFFFF8                       B CONT


                        .data
00001040:                       Array: .word 1,2,3,4,-1,5,-2,-3,6,0
00001068:                       negsum: .word 0
```

Screenshot of Register set and Memory output:




```
                .data
00001040:               Array: .word 1,2,3,4,-1,5,-2,-3,6,0
00001068:               negsum: .word 0
```

MemoryView9

00001040

00001040	00000001	00000002	00000003	00000004	FFFFFFFF	00000005	FFFFFFFE	FFFFFFFD	00000006
00001078	81818181	81818181	81818181	81818181	81818181	81818181	81818181	81818181	81818181
000010B0	81818181	81818181	81818181	81818181	81818181	81818181	81818181	81818181	81818181
000010E8	81818181	81818181	81818181	81818181	81818181	81818181	81818181	81818181	81818181

MemoryView6

00001068

00001068	FFFFFFFA	81818181	81818181	81818181	81818181	81818181	81818181	81818181	81818181
000010A0	81818181	81818181	81818181	81818181	81818181	81818181	81818181	81818181	81818181
000010D8	81818181	81818181	81818181	81818181	81818181	81818181	81818181	81818181	81818181

Q4.Write an ALP using ARM7TDMI to find the smallest number from a given set of numbers:

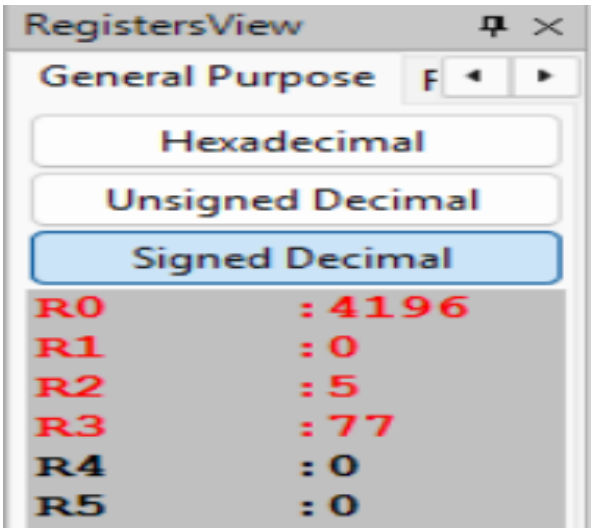
A: .word 10,50,41,55,30,20,11,5,100,77

Program screen shot:

q4.s

```
                    .text
00001000:E59F0030           LDR R0,=A
00001004:E3A01009           MOV R1,#9
00001008:E5902000           LDR R2,[R0]
0000100C:E3A03000           MOV R3,#0
00001010:E2800004           ADD R0,R0,#4

00001014:                   LOOP:
00001014:E5903000                   LDR R3,[R0]
00001018:E1520003                   CMP R2,R3
0000101C:CA000003                   BGT Move
00001020:                   CONT:
00001020:E2800004                   ADD R0,R0,#4
00001024:E2511001                   SUBS R1,R1,#1
00001028:1AFFFFF9                   BNE LOOP
0000102C:EF000011           SWI 0X011

00001030:                   Move:
00001030:E1A02003                   MOV R2,R3
00001034:EAFFFFF9                   B CONT


                    .data
0000103C:                   A: .word 10,50,41,55,30,20,11,5,100,77
```

Screenshot of Register set and Memory output:



ASSIGNMENT QUESTIONS :

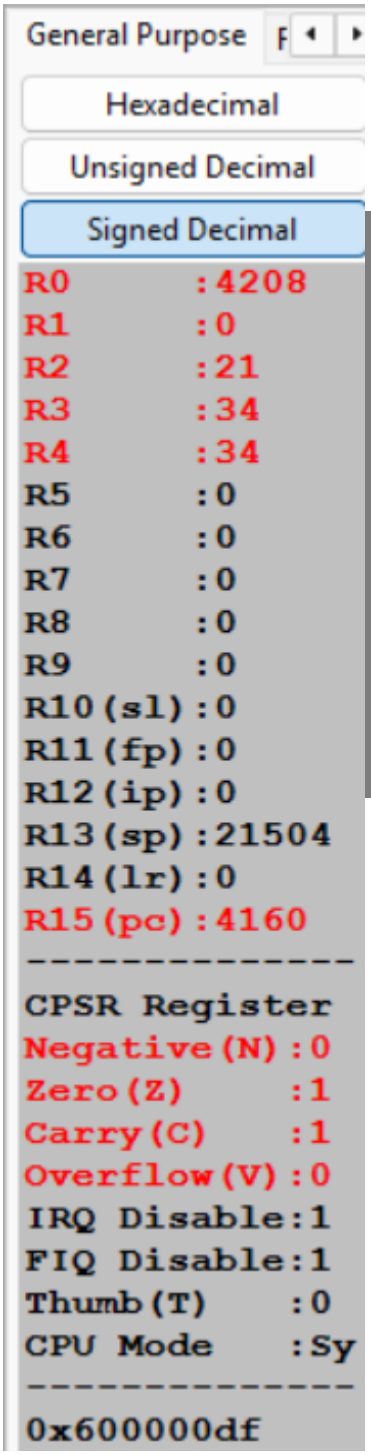
Q6. Write an ALP using ARM7TDMI to generate Fibonacci series of n numbers and store it in the memory location

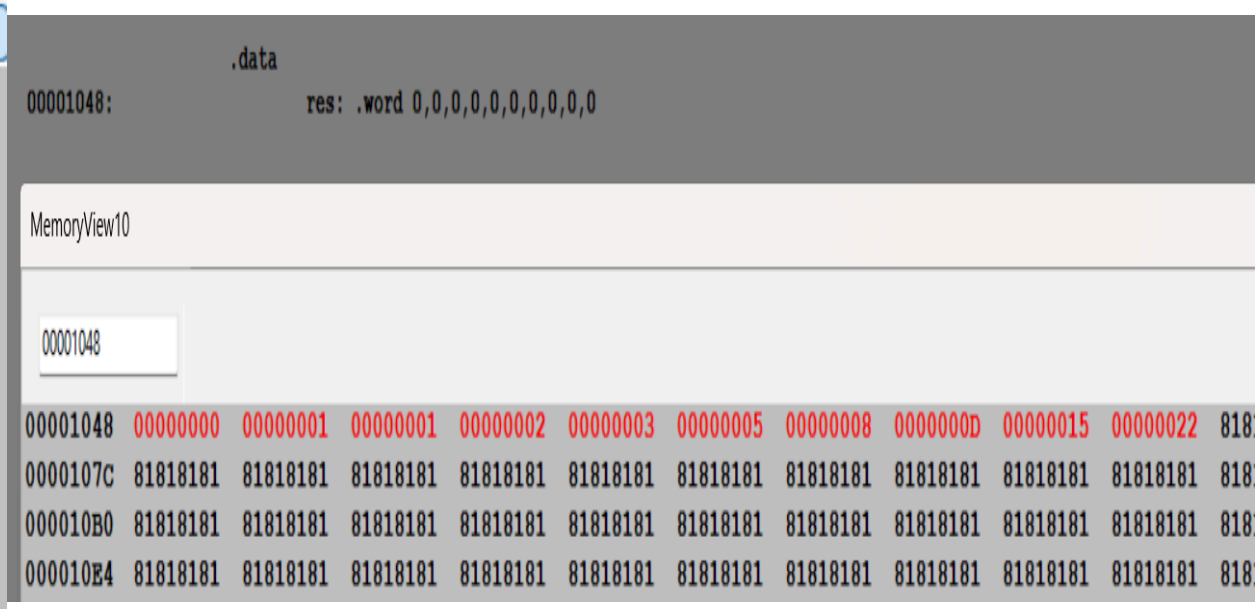
Program screen shot:

q6.s

```
                    .text
00001000:E59F003C           LDR R0,=res
00001004:E3A01008           MOV R1,#8

00001008:E3A02000           MOV R2,#0
0000100C:E3A03001           MOV R3,#1
00001010:E3A04000           MOV R4,#0

00001014:                   STORING:
00001014:E5802000                   STR R2,[R0]
00001018:E2800004                   ADD R0,R0,#4
0000101C:E5803000                   STR R3,[R0]
00001020:E2800004                   ADD R0,R0,#4

00001024:                   LOOP:
00001024:E0834002                   ADD R4,R3,R2
00001028:E5804000                   STR R4,[R0]
0000102C:E2800004                   ADD R0,R0,#4

00001030:E1A02003                   MOV R2,R3
00001034:E1A03004                   MOV R3,R4

00001038:E2511001                   SUBS R1,R1,#1
0000103C:1AFFFFF8                   BNE LOOP
00001040:EF000011   SWI 0X011


                    .data
00001048:                   res: .word 0,0,0,0,0,0,0,0,0,0
```

Screenshot of Register set and Memory output:


General Purpose
Hexadecimal
Unsigned Decimal
Signed Decimal
R0 :4208
R1 :0
R2 :21
R3 :34
R4 :34
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):4160
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 :Sy
0x600000df


.data
00001048: res: .word 0,0,0,0,0,0,0,0,0,0
MemoryView10
00001048
00001048 00000000 00000001 00000001 00000002 00000003 00000005 00000008 0000000D 00000015 00000022
0000107C 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
000010B0 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
000010E4 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181

Q7.Write an ALP using ARM7TDMI to multiplication of 32X50 without using mul instructions.

(Hint: barrel shifter instructions.)

(Note :any number can be considered as multiplier)

Program screen shot:

q7.s

```
                        .text
00001000:E3A01020               MOV R1,#32          ;32*50    50 = 2^5 + 2^4 + 2
00001004:E1A02281               MOV R2,R1,LSL #5
00001008:E0822201               ADD R2,R2,R1, LSL #4
0000100C:E0822081               ADD R2,R2,R1,LSL #1

00001010:EF000011     SWI 0X011
```

Screenshot of Register set and Memory output: