Microprocessor and Computer Architecture Laboratory

UE19CS256

4th Semester, Academic Year 2020-21

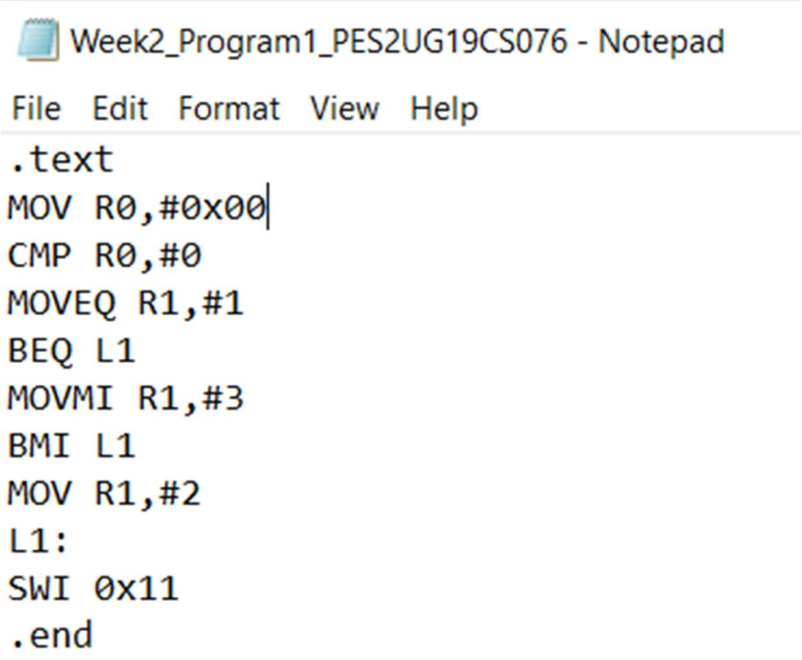
Date: 1/2/20

Name: B.Pravena	SRN: PES2UG19CS076	Section: B

Week#____2____ Program Number: ___1___

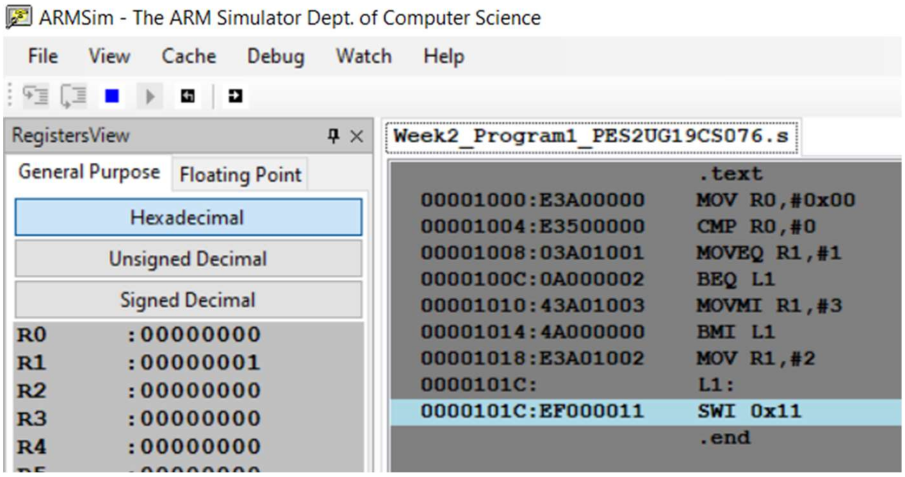
Title - Based on the value of the number in R0, Write an ALP to store 1 in R1 if R0 is zero, Store 2 in R1 if R0 is positive, Store 3 in R1 if R0 is negative.

I. ARM Assembly Code


Week2_Program1_PES2UG19CS076 - Notepad
File Edit Format View Help

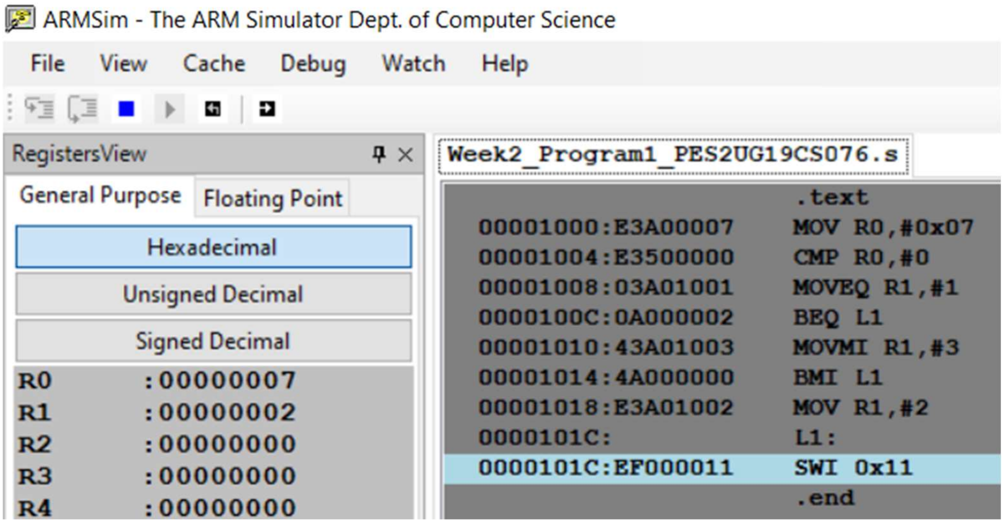
```
.text
MOV R0,#0x00
CMP R0,#0
MOVEQ R1,#1
BEQ L1
MOVMI R1,#3
BMI L1
MOV R1,#2
L1:
SWI 0x11
.end
```

II. Output Screen Shot



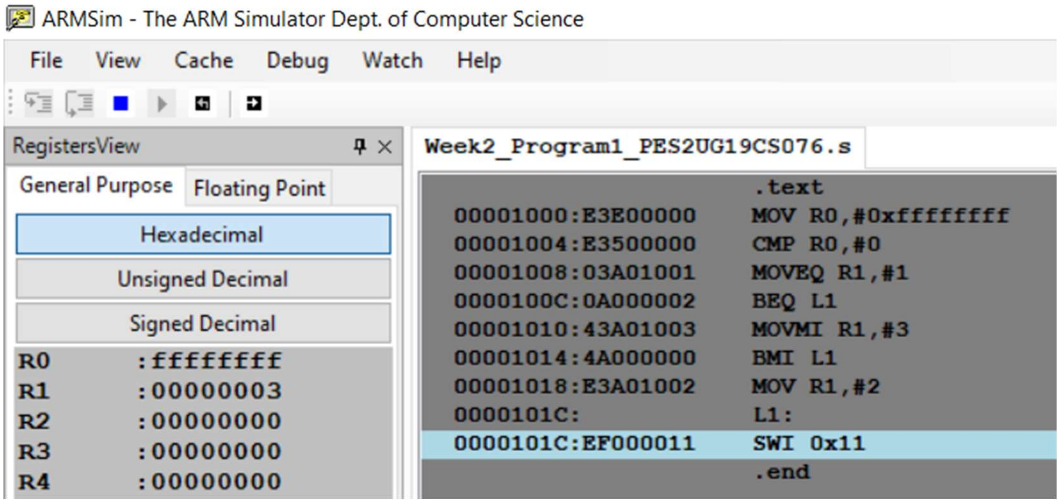
```
R0          :00000000
R1          :00000001
R2          :00000000
R3          :00000000
R4          :00000000
```

```
                            .text
00001000:E3A00000           MOV R0,#0x00
00001004:E3500000           CMP R0,#0
00001008:03A01001           MOVEQ R1,#1
0000100C:0A000002           BEQ L1
00001010:43A01003           MOVMI R1,#3
00001014:4A000000           BMI L1
00001018:E3A01002           MOV R1,#2
0000101C:                   L1:
0000101C:EF000011           SWI 0x11
                            .end
```



```
R0          :00000007
R1          :00000002
R2          :00000000
R3          :00000000
R4          :00000000
```

```
                            .text
00001000:E3A00007           MOV R0,#0x07
00001004:E3500000           CMP R0,#0
00001008:03A01001           MOVEQ R1,#1
0000100C:0A000002           BEQ L1
00001010:43A01003           MOVMI R1,#3
00001014:4A000000           BMI L1
00001018:E3A01002           MOV R1,#2
0000101C:                   L1:
0000101C:EF000011           SWI 0x11
                            .end
```



```
R0          :ffffffff
R1          :00000003
R2          :00000000
R3          :00000000
R4          :00000000
```

```
                            .text
00001000:E3E00000           MOV R0,#0xffffffff
00001004:E3500000           CMP R0,#0
00001008:03A01001           MOVEQ R1,#1
0000100C:0A000002           BEQ L1
00001010:43A01003           MOVMI R1,#3
00001014:4A000000           BMI L1
00001018:E3A01002           MOV R1,#2
0000101C:                   L1:
0000101C:EF000011           SWI 0x11
                            .end
```

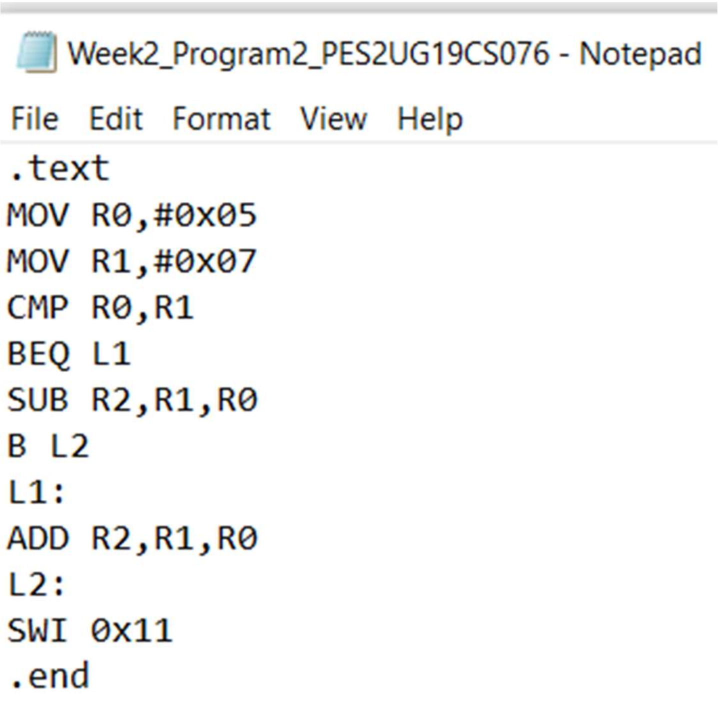
III. Input-Output Table

CASE 1	R0	AFTER COMPARE	0x00
	R1		1
CASE 2	R0	AFTER COMPARE	0X07
	R1		2
CASE 3	R0	AFTER COMPARE	0XFFFFFFFF
	R1		3

Week#____2____ Program Number: ___2___

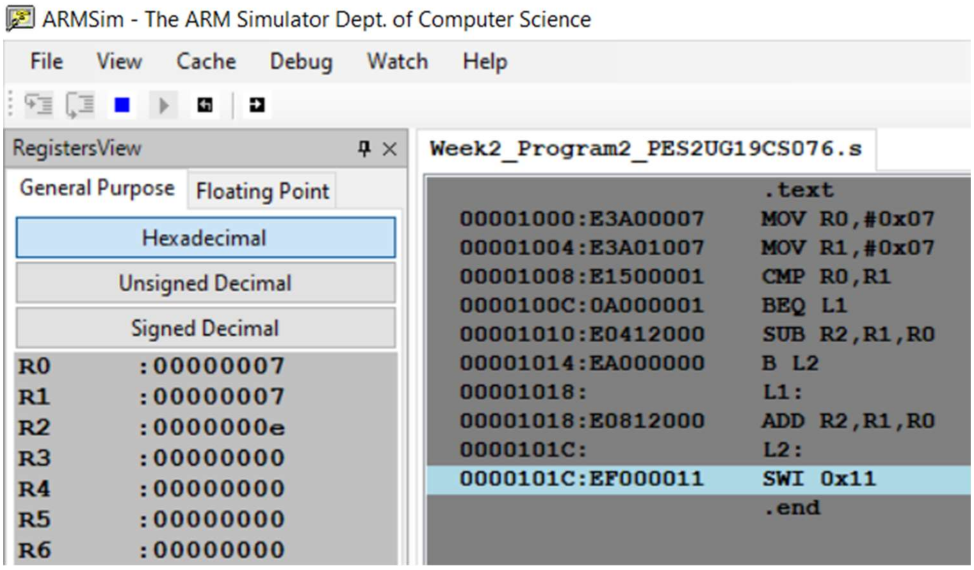
Title - Write an ALP to compare the value of R0 and R1, add if R0 = R1, else subtract.

I. ARM Assembly Code


Week2_Program2_PES2UG19CS076 - Notepad
File Edit Format View Help

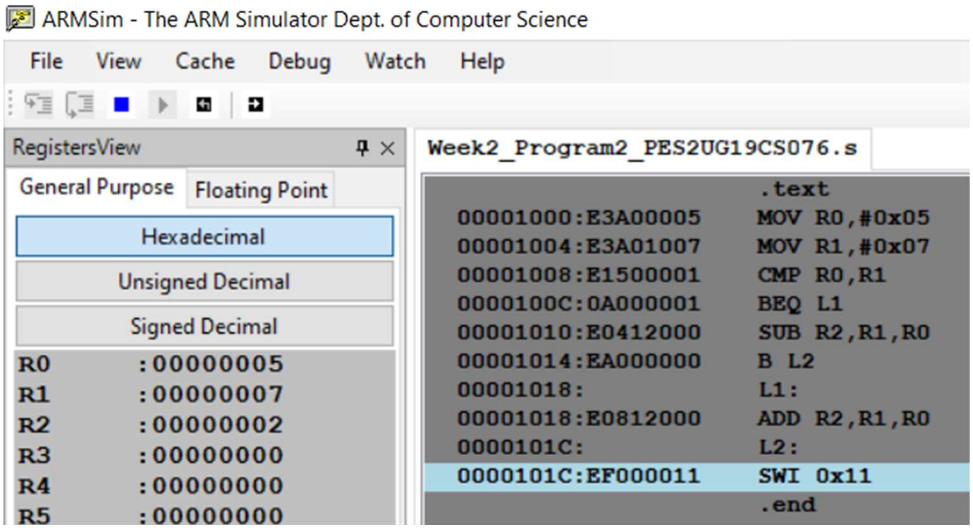
```
.text
MOV R0,#0x05
MOV R1,#0x07
CMP R0,R1
BEQ L1
SUB R2,R1,R0
B L2
L1:
ADD R2,R1,R0
L2:
SWI 0x11
.end
```

II. Output Screen Shot



```
R0      :00000007
R1      :00000007
R2      :0000000e
R3      :00000000
R4      :00000000
R5      :00000000
R6      :00000000
```

```
                        .text
00001000:E3A00007       MOV R0,#0x07
00001004:E3A01007       MOV R1,#0x07
00001008:E1500001       CMP R0,R1
0000100C:0A000001       BEQ L1
00001010:E0412000       SUB R2,R1,R0
00001014:EA000000       B L2
00001018:               L1:
00001018:E0812000       ADD R2,R1,R0
0000101C:               L2:
0000101C:EF000011       SWI 0x11
                        .end
```



```
R0      :00000005
R1      :00000007
R2      :00000002
R3      :00000000
R4      :00000000
R5      :00000000
```

```
                        .text
00001000:E3A00005       MOV R0,#0x05
00001004:E3A01007       MOV R1,#0x07
00001008:E1500001       CMP R0,R1
0000100C:0A000001       BEQ L1
00001010:E0412000       SUB R2,R1,R0
00001014:EA000000       B L2
00001018:               L1:
00001018:E0812000       ADD R2,R1,R0
0000101C:               L2:
0000101C:EF000011       SWI 0x11
                        .end
```

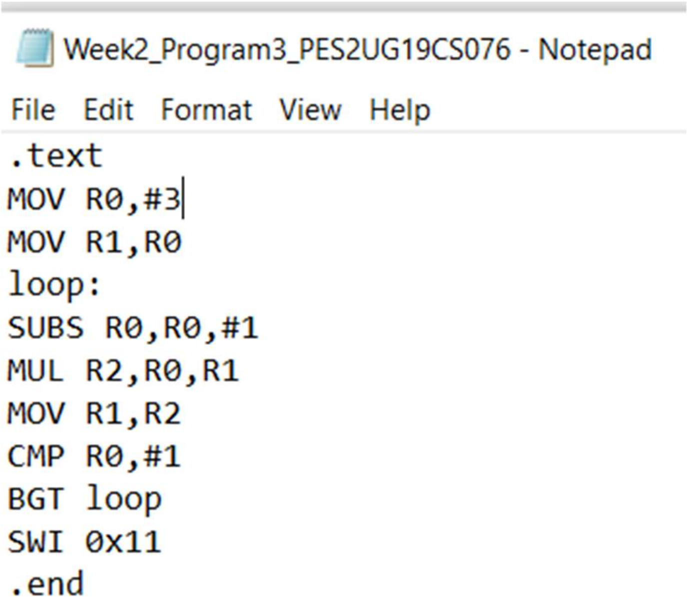
III. Input-Output Table

CASE 1	R1=0x07, R0=0x07, R2=R1+R0=0x0E
CASE 2	R1=0x07, R0=0x05, R2=R1-R0=0x02

Week#____2____ Program Number: ___3___

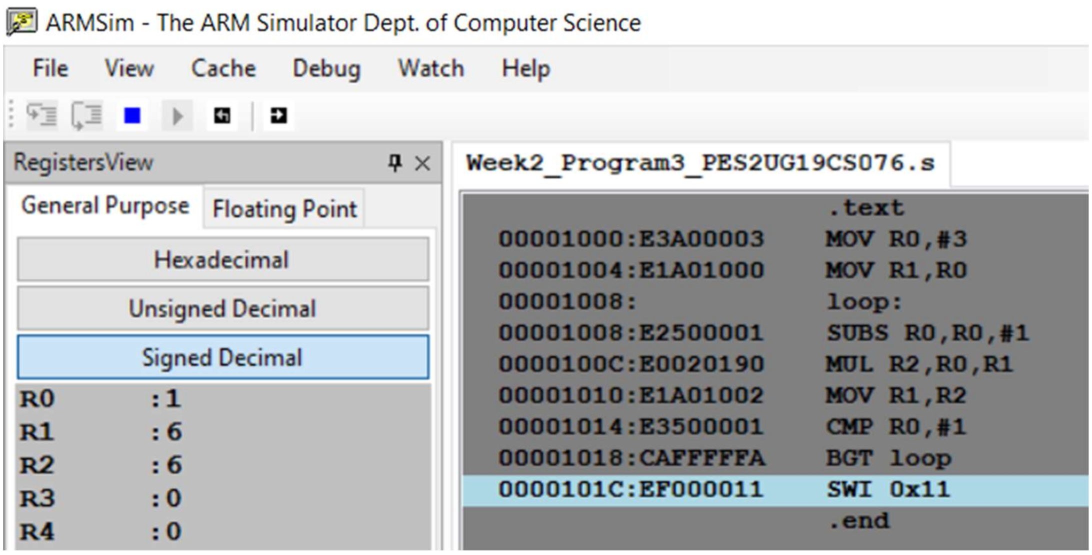
Title - Write an ALP to find the factorial of a number stored in R0. Store the value in R1 (without using LDR and STR instructions). Use only registers.

I. ARM Assembly Code


Week2_Program3_PES2UG19CS076 - Notepad
File Edit Format View Help

```
.text
MOV R0,#3
MOV R1,R0
loop:
SUBS R0,R0,#1
MUL R2,R0,R1
MOV R1,R2
CMP R0,#1
BGT loop
SWI 0x11
.end
```

II. 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
Week2_Program3_PES2UG19CS076.s

```
R0        :1
R1        :6
R2        :6
R3        :0
R4        :0
```

```
                          .text
00001000:E3A00003         MOV R0,#3
00001004:E1A01000         MOV R1,R0
00001008:                 loop:
00001008:E2500001         SUBS R0,R0,#1
0000100C:E0020190         MUL R2,R0,R1
00001010:E1A01002         MOV R1,R2
00001014:E3500001         CMP R0,#1
00001018:CAFFFFFA         BGT loop
0000101C:EF000011         SWI 0x11
                          .end
```

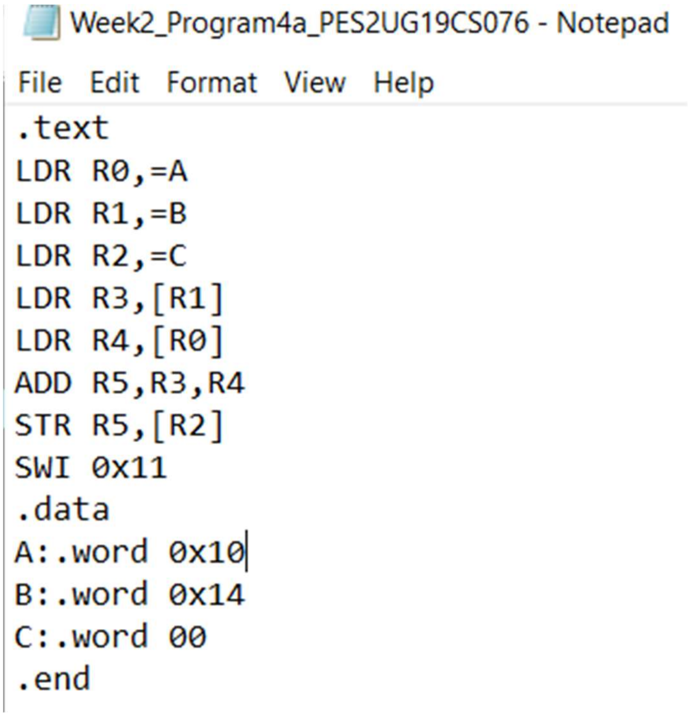
III. Input-Output Table

1st ITERATION	R0 =0x02
	R1 = 0x03
	R2 =0x06
2ND ITERATION	R0 = 0x01
	R1 = 0x06
	R2 = 0x06

Week#____2____ Program Number: __4a___

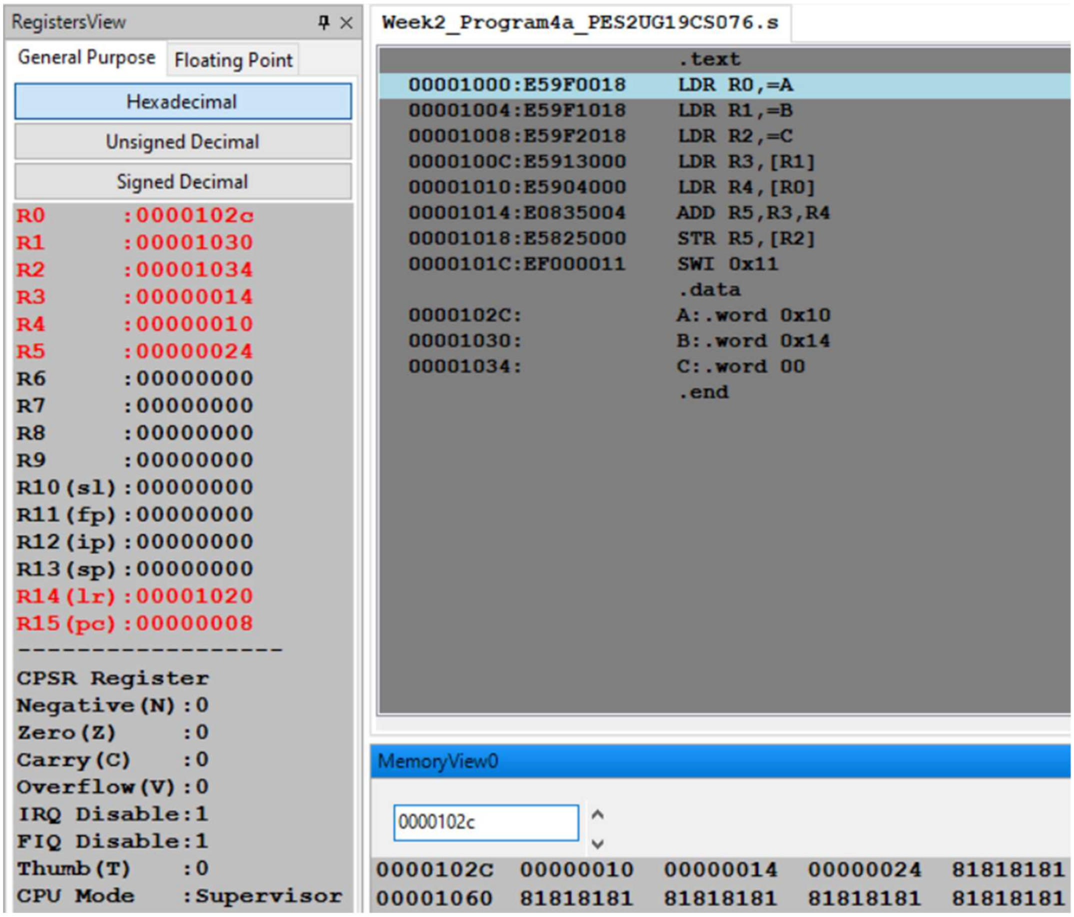
Title - Write an ALP to add two 32 bit numbers loaded from memory and store the result in memory.

I. ARM Assembly Code


Week2_Program4a_PES2UG19CS076 - Notepad
File Edit Format View Help

```
.text
LDR R0,=A
LDR R1,=B
LDR R2,=C
LDR R3,[R1]
LDR R4,[R0]
ADD R5,R3,R4
STR R5,[R2]
SWI 0x11
.data
A:.word 0x10
B:.word 0x14
C:.word 00
.end
```

II. Output Screen Shot



```
R0        :0000102c
R1        :00001030
R2        :00001034
R3        :00000014
R4        :00000010
R5        :00000024
R6        :00000000
R7        :00000000
R8        :00000000
R9        :00000000
R10(sl):00000000
R11(fp):00000000
R12(ip):00000000
R13(sp):00000000
R14(lr):00001020
R15(pc):00000008
------------------
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   :Supervisor
```

```
                     .text
00001000:E59F0018    LDR R0,=A
00001004:E59F1018    LDR R1,=B
00001008:E59F2018    LDR R2,=C
0000100C:E5913000    LDR R3,[R1]
00001010:E5904000    LDR R4,[R0]
00001014:E0835004    ADD R5,R3,R4
00001018:E5825000    STR R5,[R2]
0000101C:EF000011    SWI 0x11
                     .data
0000102C:            A:.word 0x10
00001030:            B:.word 0x14
00001034:            C:.word 00
                     .end
```

```
0000102c
0000102c  00000010  00000014  00000024  81818181
00001060  81818181  81818181  81818181  81818181
```

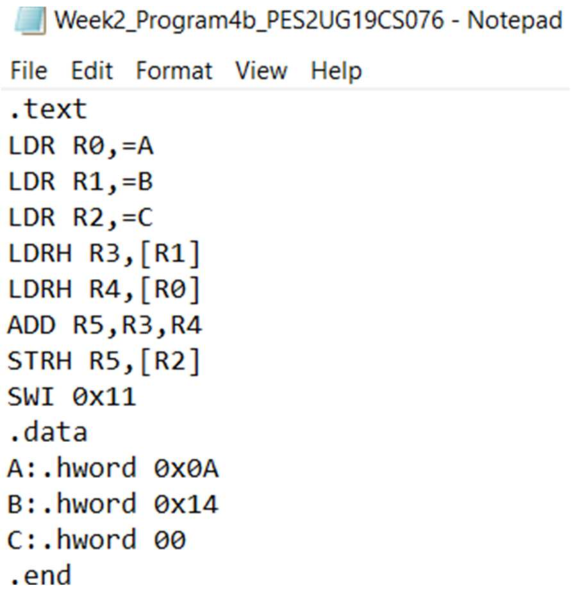
III. Input-Output Table

	A=0x10, B=0x14
R0	Address of A
R1	Address of B
R2	Address of C
R3	0x14 = decimal 20
R4	0x10 = decimal 16
R5	0x24 = decimal 36
Location C	0x24 = decimal 36

Week#____2____ Program Number: __4b___

Title - Write an ALP to add two 16 bit numbers loaded from memory and store the result in memory.

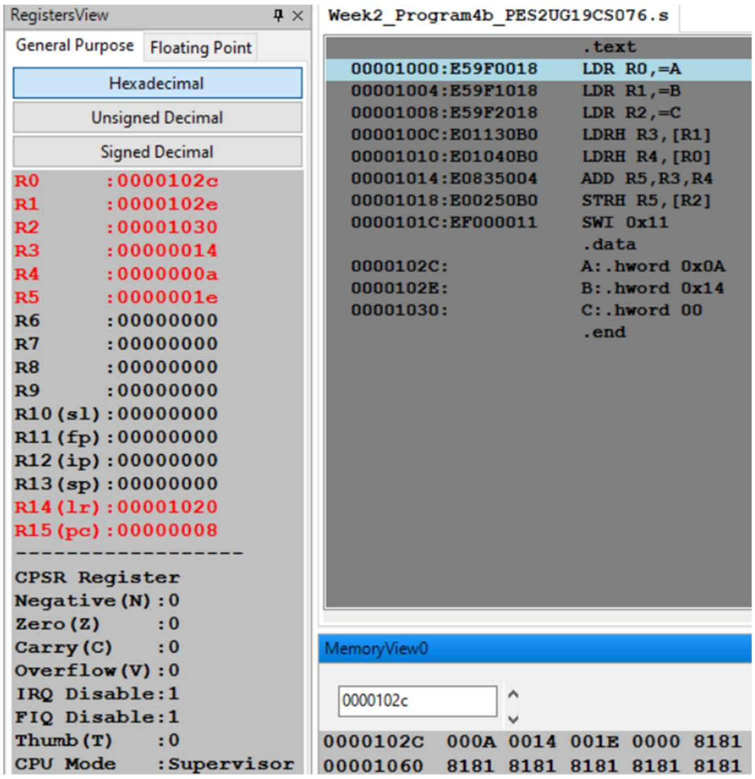
I. ARM Assembly Code



Week2_Program4b_PES2UG19CS076 - Notepad
File Edit Format View Help

```
.text
LDR R0,=A
LDR R1,=B
LDR R2,=C
LDRH R3,[R1]
LDRH R4,[R0]
ADD R5,R3,R4
STRH R5,[R2]
SWI 0x11
.data
A:.hword 0x0A
B:.hword 0x14
C:.hword 00
.end
```

II. Output Screen Shot



```
R0        :0000102c
R1        :0000102e
R2        :00001030
R3        :00000014
R4        :0000000a
R5        :0000001e
R6        :00000000
R7        :00000000
R8        :00000000
R9        :00000000
R10(sl):00000000
R11(fp):00000000
R12(ip):00000000
R13(sp):00000000
R14(lr):00001020
R15(pc):00000008
------------------
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    :Supervisor
```

```
                  .text
00001000:E59F0018 LDR R0,=A
00001004:E59F1018 LDR R1,=B
00001008:E59F2018 LDR R2,=C
0000100C:E01130B0 LDRH R3,[R1]
00001010:E01040B0 LDRH R4,[R0]
00001014:E0835004 ADD R5,R3,R4
00001018:E00250B0 STRH R5,[R2]
0000101C:EF000011 SWI 0x11
                  .data
0000102C:         A:.hword 0x0A
0000102E:         B:.hword 0x14
00001030:         C:.hword 00
                  .end
```

```
0000102c
0000102C  000A 0014 001E 0000 8181
00001060  8181 8181 8181 8181 8181
```

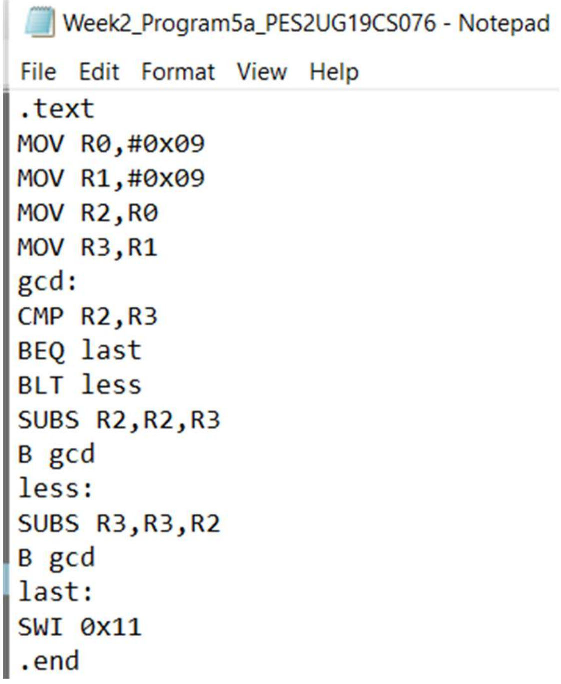
III. Input-Output Table

	A=0x0A, B=0x14
R0	Address of A
R1	Address of B
R2	Address of C
R3	0x14 = decimal 20
R4	0x0A = decimal 10
R5	0x1E = decimal 30
Location C	0x1E = decimal 30

Week#____2____ Program Number: __5a___

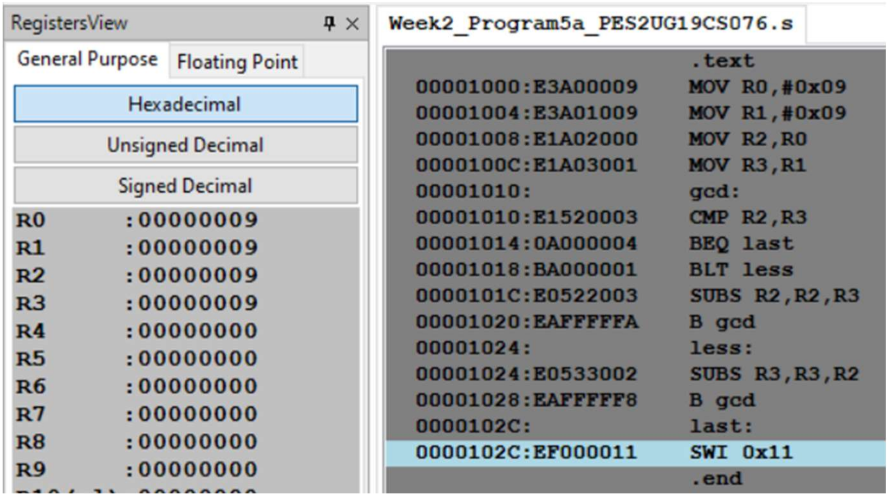
Title - Write an ALP to find GCD of two numbers (without using LDR and STR instructions).Both numbers are in registers.

I. ARM Assembly Code


Week2_Program5a_PES2UG19CS076 - Notepad
File Edit Format View Help

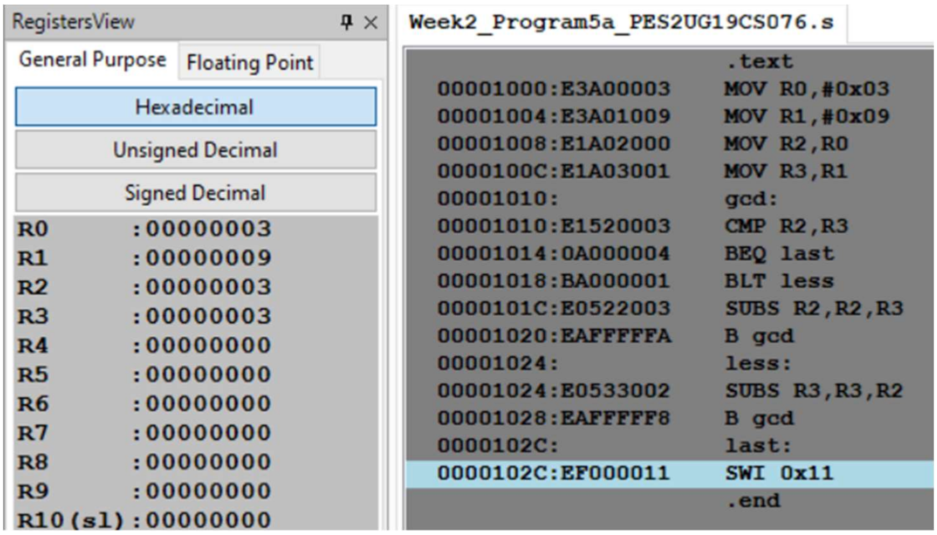
```
.text
MOV R0,#0x09
MOV R1,#0x09
MOV R2,R0
MOV R3,R1
gcd:
CMP R2,R3
BEQ last
BLT less
SUBS R2,R2,R3
B gcd
less:
SUBS R3,R3,R2
B gcd
last:
SWI 0x11
.end
```

II. Output Screen Shot



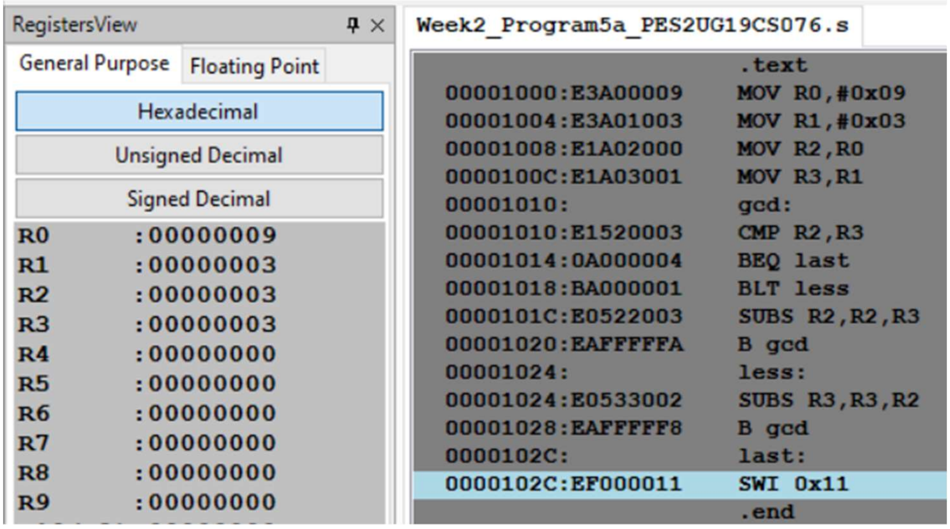
```
R0      :00000009
R1      :00000009
R2      :00000009
R3      :00000009
R4      :00000000
R5      :00000000
R6      :00000000
R7      :00000000
R8      :00000000
R9      :00000000
```

```
                        .text
00001000:E3A00009       MOV R0,#0x09
00001004:E3A01009       MOV R1,#0x09
00001008:E1A02000       MOV R2,R0
0000100C:E1A03001       MOV R3,R1
00001010:               gcd:
00001010:E1520003       CMP R2,R3
00001014:0A000004       BEQ last
00001018:BA000001       BLT less
0000101C:E0522003       SUBS R2,R2,R3
00001020:EAFFFFFA       B gcd
00001024:               less:
00001024:E0533002       SUBS R3,R3,R2
00001028:EAFFFFF8       B gcd
0000102C:               last:
0000102C:EF000011       SWI 0x11
                        .end
```



```
R0      :00000003
R1      :00000009
R2      :00000003
R3      :00000003
R4      :00000000
R5      :00000000
R6      :00000000
R7      :00000000
R8      :00000000
R9      :00000000
R10(sl) :00000000
```

```
                        .text
00001000:E3A00003       MOV R0,#0x03
00001004:E3A01009       MOV R1,#0x09
00001008:E1A02000       MOV R2,R0
0000100C:E1A03001       MOV R3,R1
00001010:               gcd:
00001010:E1520003       CMP R2,R3
00001014:0A000004       BEQ last
00001018:BA000001       BLT less
0000101C:E0522003       SUBS R2,R2,R3
00001020:EAFFFFFA       B gcd
00001024:               less:
00001024:E0533002       SUBS R3,R3,R2
00001028:EAFFFFF8       B gcd
0000102C:               last:
0000102C:EF000011       SWI 0x11
                        .end
```



```
R0      :00000009
R1      :00000003
R2      :00000003
R3      :00000003
R4      :00000000
R5      :00000000
R6      :00000000
R7      :00000000
R8      :00000000
R9      :00000000
```

```
                        .text
00001000:E3A00009       MOV R0,#0x09
00001004:E3A01003       MOV R1,#0x03
00001008:E1A02000       MOV R2,R0
0000100C:E1A03001       MOV R3,R1
00001010:               gcd:
00001010:E1520003       CMP R2,R3
00001014:0A000004       BEQ last
00001018:BA000001       BLT less
0000101C:E0522003       SUBS R2,R2,R3
00001020:EAFFFFFA       B gcd
00001024:               less:
00001024:E0533002       SUBS R3,R3,R2
00001028:EAFFFFF8       B gcd
0000102C:               last:
0000102C:EF000011       SWI 0x11
                        .end
```

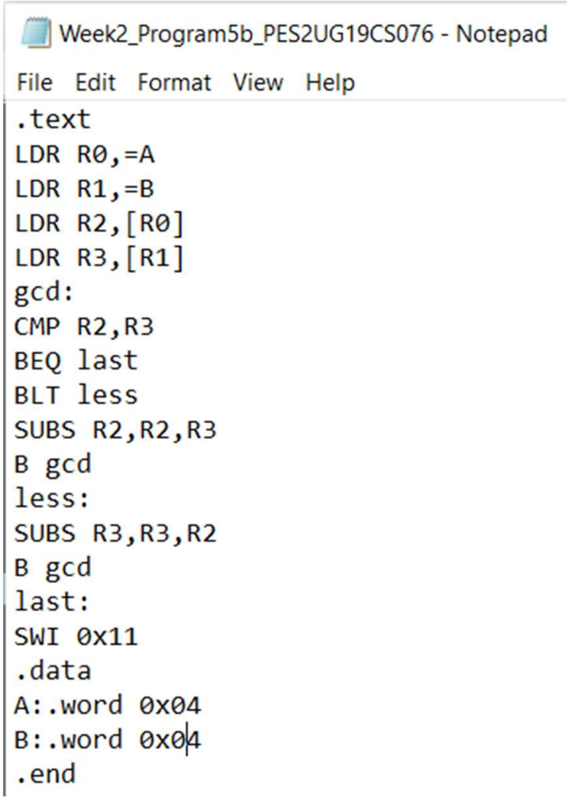
III. Input-Output Table

CASE 1	R0	0x09
	R1	0x09
	R2	0x09
	R3	0x09
CASE 2	R0	0x03
	R1	0x09
	R2	0x03
	R3	0x03
CASE 3	R0	0x09
	R1	0x03
	R2	0x03
	R3	0x03

Week#____2____ Program Number: __5b___

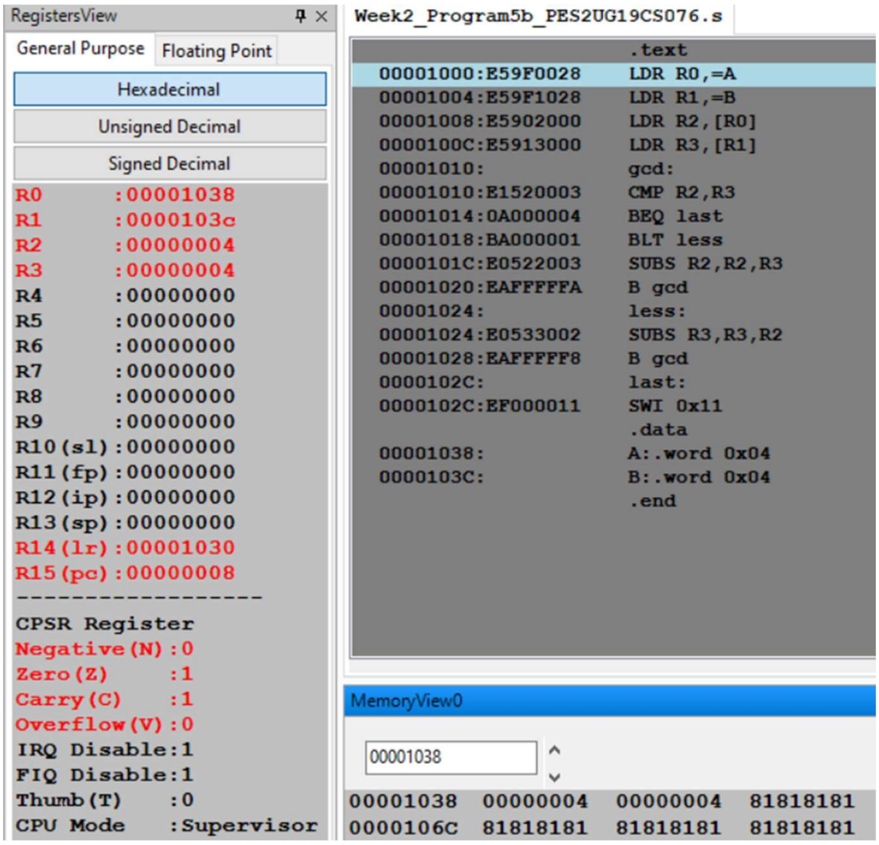
Title - Write an ALP to find the GCD of given numbers (both numbers in memory). Store result in memory.

I. ARM Assembly Code


Week2_Program5b_PES2UG19CS076 - Notepad
File Edit Format View Help

```
.text
LDR R0,=A
LDR R1,=B
LDR R2,[R0]
LDR R3,[R1]
gcd:
CMP R2,R3
BEQ last
BLT less
SUBS R2,R2,R3
B gcd
less:
SUBS R3,R3,R2
B gcd
last:
SWI 0x11
.data
A:.word 0x04
B:.word 0x04
.end
```

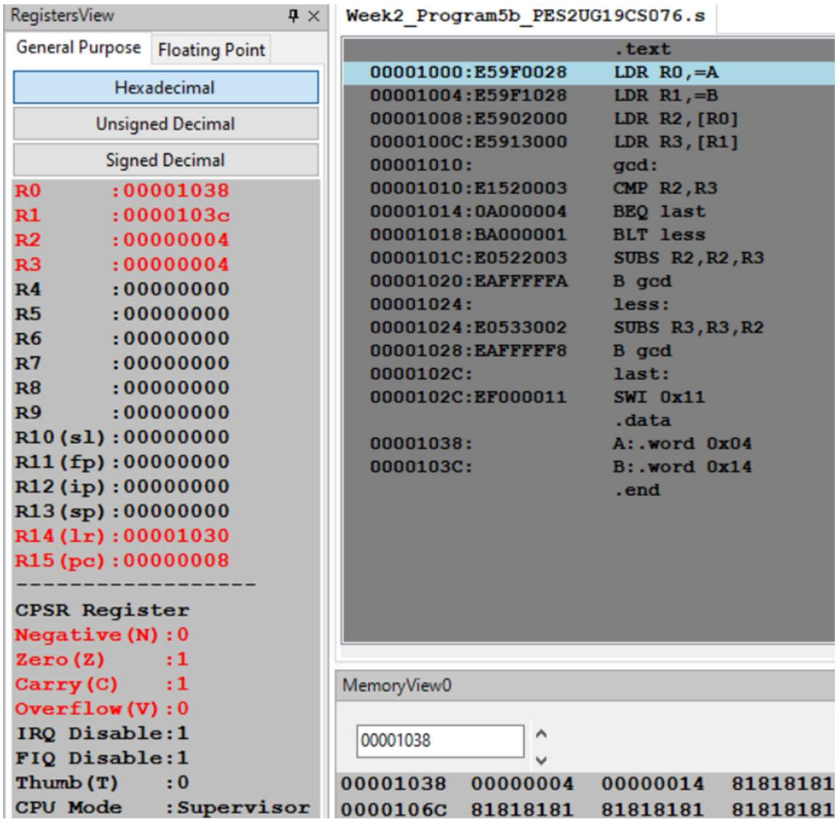
II. Output Screen Shot



```
R0       :00001038
R1       :0000103c
R2       :00000004
R3       :00000004
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)  :00001030
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
```

```
                         .text
00001000:E59F0028        LDR R0,=A
00001004:E59F1028        LDR R1,=B
00001008:E5902000        LDR R2,[R0]
0000100C:E5913000        LDR R3,[R1]
00001010:                gcd:
00001010:E1520003        CMP R2,R3
00001014:0A000004        BEQ last
00001018:BA000001        BLT less
0000101C:E0522003        SUBS R2,R2,R3
00001020:EAFFFFFA        B gcd
00001024:                less:
00001024:E0533002        SUBS R3,R3,R2
00001028:EAFFFFF8        B gcd
0000102C:                last:
0000102C:EF000011        SWI 0x11
                         .data
00001038:                A:.word 0x04
0000103C:                B:.word 0x04
                         .end
```

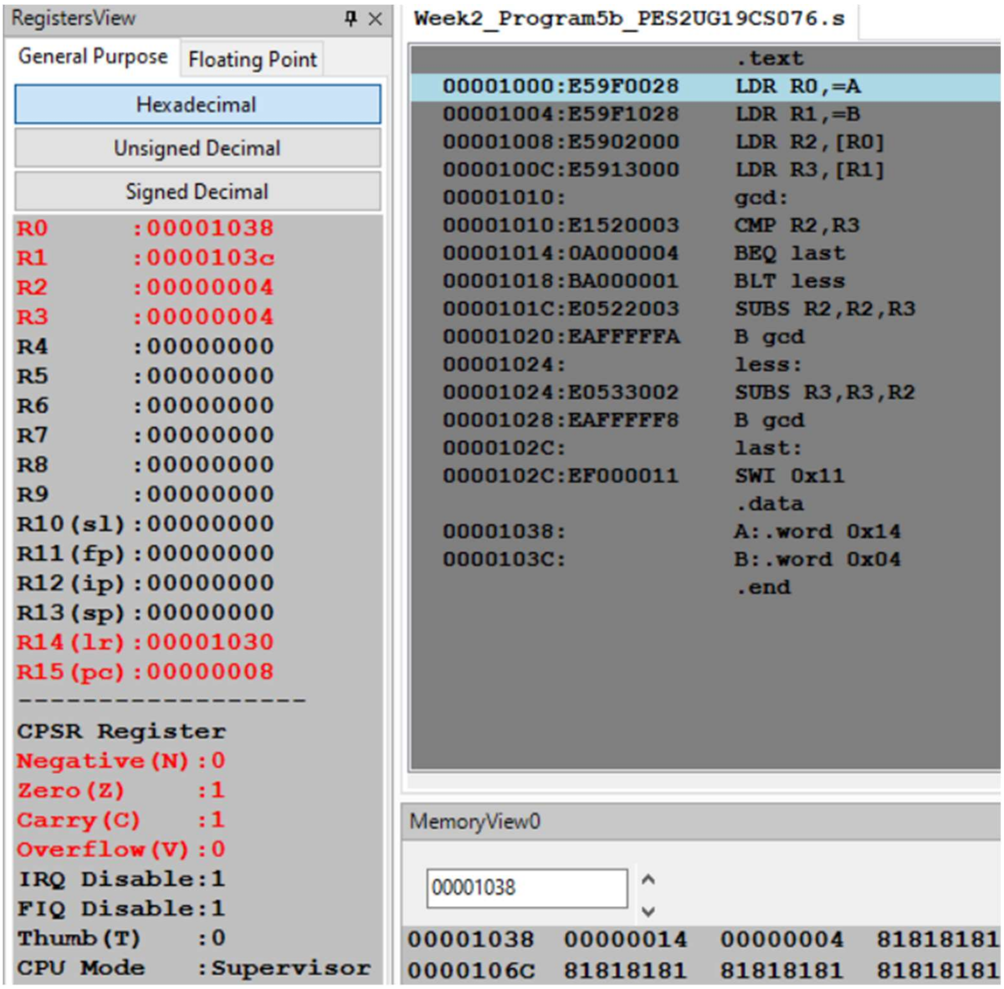
```
00001038  00000004  00000004  81818181
0000106c  81818181  81818181  81818181
```



```
R0       :00001038
R1       :0000103c
R2       :00000004
R3       :00000004
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)  :00001030
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
```

```
                         .text
00001000:E59F0028        LDR R0,=A
00001004:E59F1028        LDR R1,=B
00001008:E5902000        LDR R2,[R0]
0000100C:E5913000        LDR R3,[R1]
00001010:                gcd:
00001010:E1520003        CMP R2,R3
00001014:0A000004        BEQ last
00001018:BA000001        BLT less
0000101C:E0522003        SUBS R2,R2,R3
00001020:EAFFFFFA        B gcd
00001024:                less:
00001024:E0533002        SUBS R3,R3,R2
00001028:EAFFFFF8        B gcd
0000102C:                last:
0000102C:EF000011        SWI 0x11
                         .data
00001038:                A:.word 0x04
0000103C:                B:.word 0x14
                         .end
```

```
00001038  00000004  00000014  81818181
0000106c  81818181  81818181  81818181
```



```
R0       :00001038
R1       :0000103c
R2       :00000004
R3       :00000004
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)  :00001030
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
```

```
                        .text
00001000:E59F0028       LDR R0,=A
00001004:E59F1028       LDR R1,=B
00001008:E5902000       LDR R2,[R0]
0000100C:E5913000       LDR R3,[R1]
00001010:               gcd:
00001010:E1520003       CMP R2,R3
00001014:0A000004       BEQ last
00001018:BA000001       BLT less
0000101C:E0522003       SUBS R2,R2,R3
00001020:EAFFFFFA       B gcd
00001024:               less:
00001024:E0533002       SUBS R3,R3,R2
00001028:EAFFFFF8       B gcd
0000102C:               last:
0000102C:EF000011       SWI 0x11
                        .data
00001038:               A:.word 0x14
0000103C:               B:.word 0x04
                        .end
```

```
00001038
00001038  00000014  00000004  81818181
0000106C  81818181  81818181  81818181
```

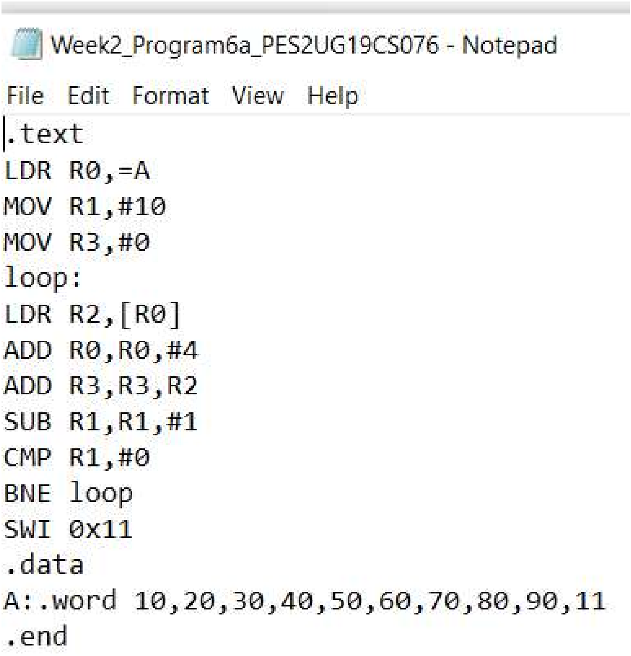
III. Input-Output Table

		Hexadecimal	Decimal
CASE 1	R2	0x04	4
	R3	0x04	4
CASE 2	R2	0x04	4
	R3	0x14	20
CASE 3	R2	0x14	20
	R3	0x04	4

Week#____2____ Program Number: __6a___

Title - Write an ALP to add an array of ten 32 bit numbers from memory.

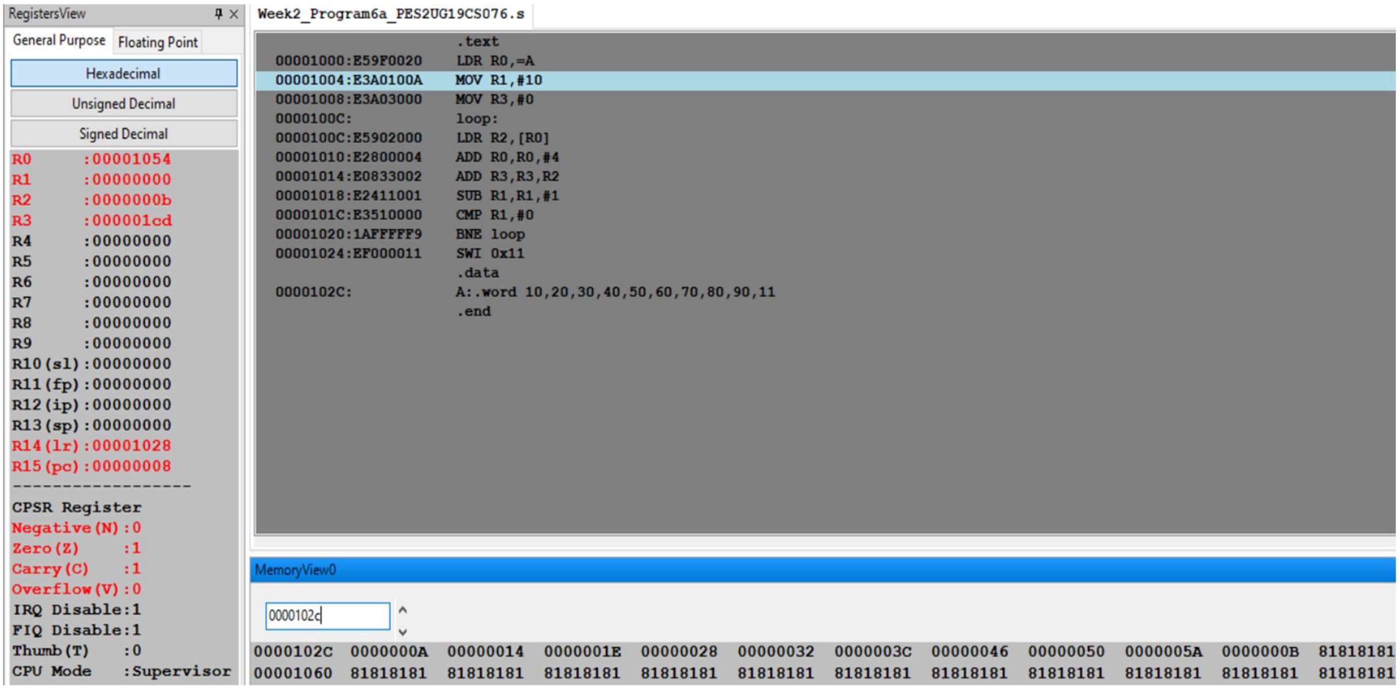
I. ARM Assembly Code



Week2_Program6a_PES2UG19CS076 - Notepad
File Edit Format View Help

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

II. Output Screen Shot

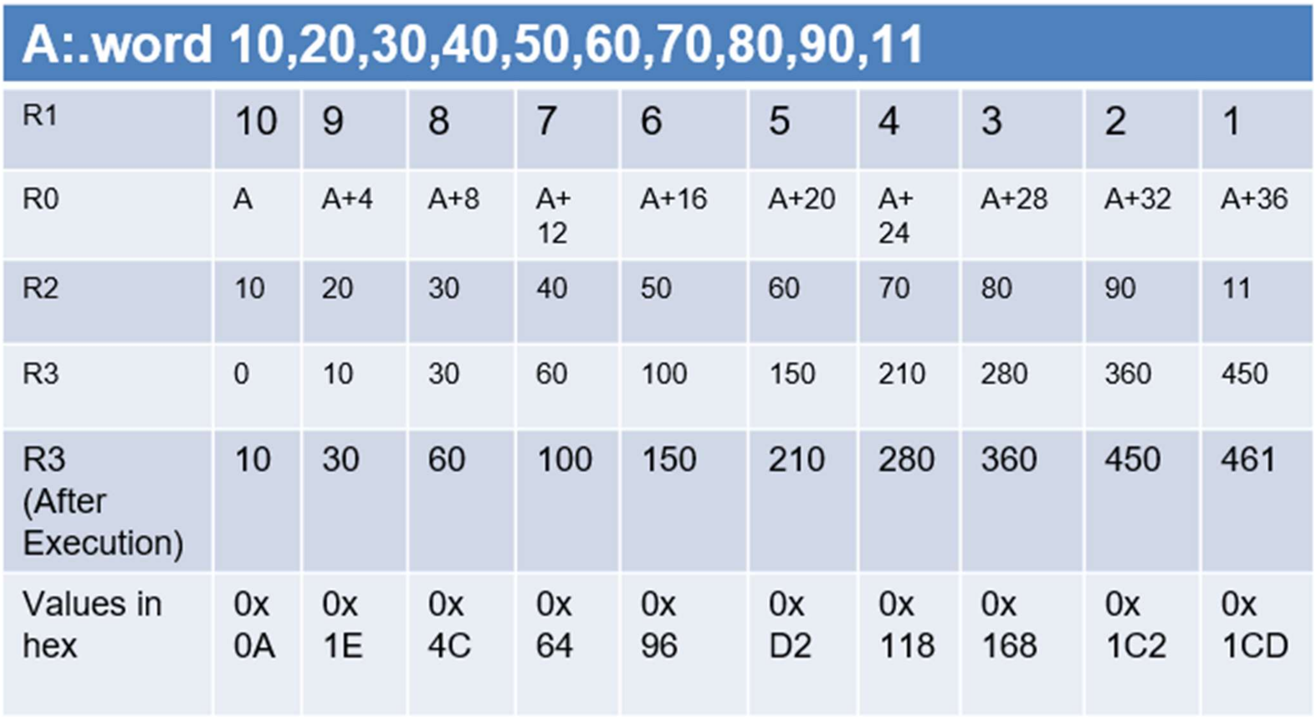


```
R0        :00001054
R1        :00000000
R2        :0000000b
R3        :000001cd
R4        :00000000
R5        :00000000
R6        :00000000
R7        :00000000
R8        :00000000
R9        :00000000
R10(sl):00000000
R11(fp):00000000
R12(ip):00000000
R13(sp):00000000
R14(lr):00001028
R15(pc):00000008
------------------
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
```

```
                        .text
00001000:E59F0020       LDR R0,=A
00001004:E3A0100A       MOV R1,#10
00001008:E3A03000       MOV R3,#0
0000100C:               loop:
0000100C:E5902000       LDR R2,[R0]
00001010:E2800004       ADD R0,R0,#4
00001014:E0833002       ADD R3,R3,R2
00001018:E2411001       SUB R1,R1,#1
0000101C:E3510000       CMP R1,#0
00001020:1AFFFFF9       BNE loop
00001024:EF000011       SWI 0x11
                        .data
0000102C:               A:.word 10,20,30,40,50,60,70,80,90,11
                        .end
```

```
0000102c
0000102C  0000000A  00000014  0000001E  00000028  00000032  0000003C  00000046  00000050  0000005A  0000000B  81818181
00001060  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181
```

III. Input-Output Table

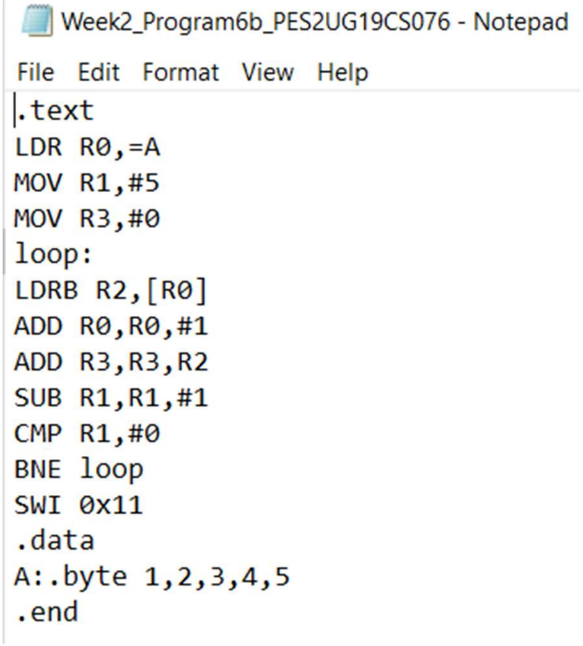


A:.word 10,20,30,40,50,60,70,80,90,11										
R1	10	9	8	7	6	5	4	3	2	1
R0	A	A+4	A+8	A+12	A+16	A+20	A+24	A+28	A+32	A+36
R2	10	20	30	40	50	60	70	80	90	11
R3	0	10	30	60	100	150	210	280	360	450
R3 (After Execution)	10	30	60	100	150	210	280	360	450	461
Values in hex	0x 0A	0x 1E	0x 4C	0x 64	0x 96	0x D2	0x 118	0x 168	0x 1C2	0x 1CD

Week#____2____ Program Number: __6b___

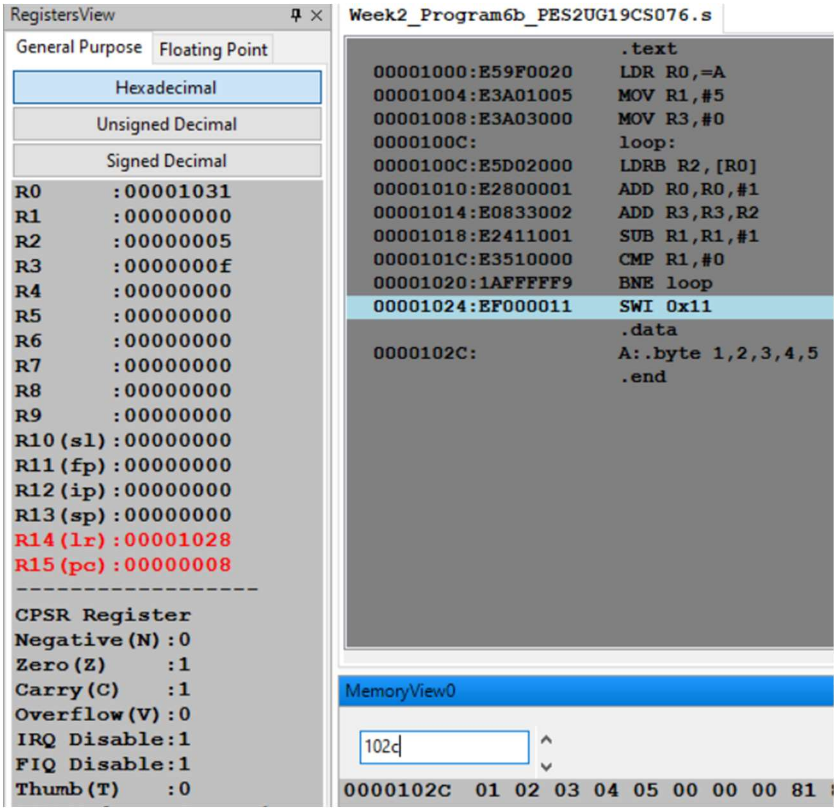
Title - Add array of five 8 bit numbers taking data from memory location (use .byte to store the data instead of .word)

I. ARM Assembly Code


Week2_Program6b_PES2UG19CS076 - Notepad
File Edit Format View Help

```
.text
LDR R0,=A
MOV R1,#5
MOV R3,#0
loop:
LDRB R2,[R0]
ADD R0,R0,#1
ADD R3,R3,R2
SUB R1,R1,#1
CMP R1,#0
BNE loop
SWI 0x11
.data
A:.byte 1,2,3,4,5
.end
```

II. Output Screen Shot

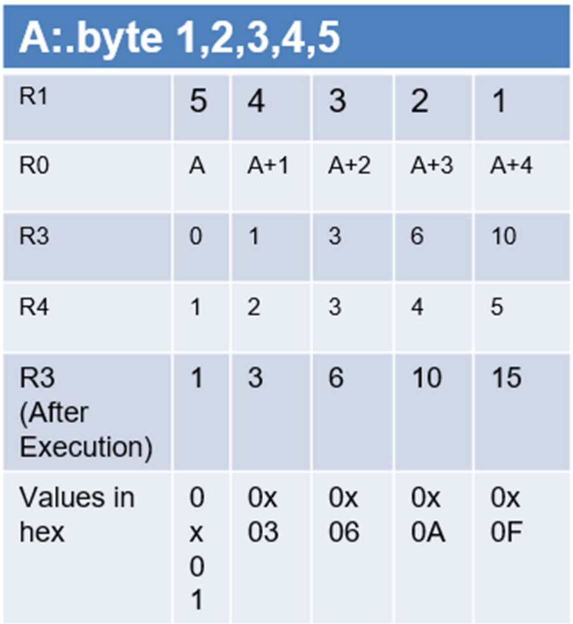


```
R0        :00001031
R1        :00000000
R2        :00000005
R3        :0000000f
R4        :00000000
R5        :00000000
R6        :00000000
R7        :00000000
R8        :00000000
R9        :00000000
R10(sl):00000000
R11(fp):00000000
R12(ip):00000000
R13(sp):00000000
R14(lr):00001028
R15(pc):00000008
------------------
CPSR Register
Negative(N):0
Zero(Z)     :1
Carry(C)    :1
Overflow(V):0
IRQ Disable:1
FIQ Disable:1
Thumb(T)    :0
```

```
                            .text
00001000:E59F0020        LDR R0,=A
00001004:E3A01005        MOV R1,#5
00001008:E3A03000        MOV R3,#0
0000100C:                loop:
0000100C:E5D02000        LDRB R2,[R0]
00001010:E2800001        ADD R0,R0,#1
00001014:E0833002        ADD R3,R3,R2
00001018:E2411001        SUB R1,R1,#1
0000101C:E3510000        CMP R1,#0
00001020:1AFFFFF9        BNE loop
00001024:EF000011        SWI 0x11
                         .data
0000102C:                A:.byte 1,2,3,4,5
                         .end
```

```
0000102C  01 02 03 04 05 00 00 00 81
```

III. Input-Output Table

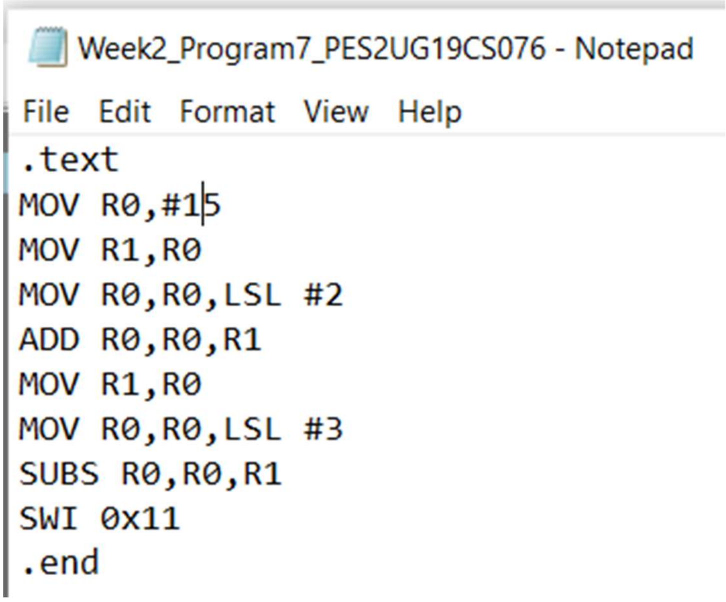


A:.byte 1,2,3,4,5					
R1	5	4	3	2	1
R0	A	A+1	A+2	A+3	A+4
R3	0	1	3	6	10
R4	1	2	3	4	5
R3 (After Execution)	1	3	6	10	15
Values in hex	0x01	0x03	0x06	0x0A	0x0F

Week#____2____   Program Number: __7__

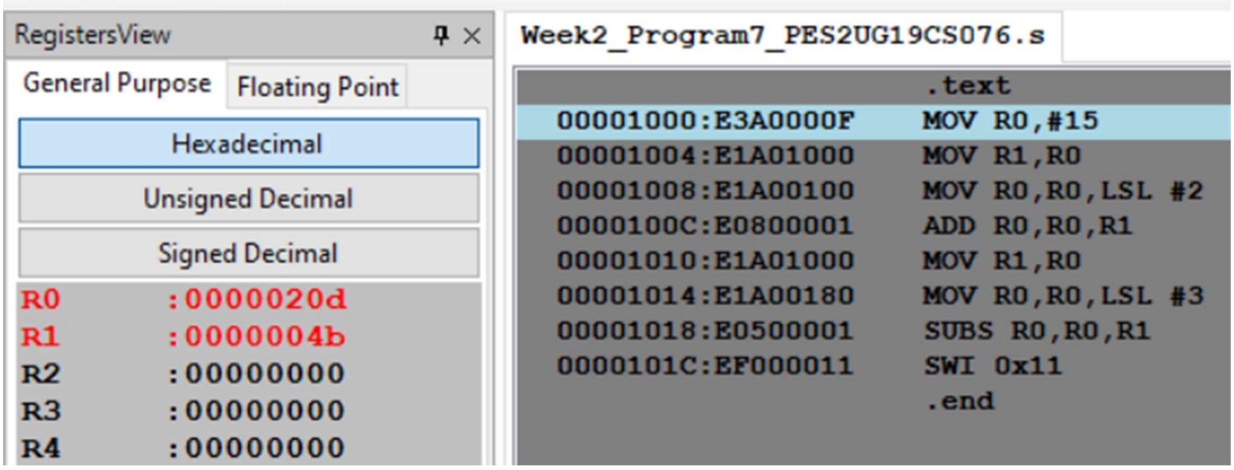
Title - Write an ALP to multiply 35*R0. *Use LSL instruction for multiplication.

I. ARM Assembly Code


Week2_Program7_PES2UG19CS076 - Notepad
File Edit Format View Help

```
.text
MOV R0,#15
MOV R1,R0
MOV R0,R0,LSL #2
ADD R0,R0,R1
MOV R1,R0
MOV R0,R0,LSL #3
SUBS R0,R0,R1
SWI 0x11
.end
```

II. Output Screen Shot



```
R0          :0000020d
R1          :0000004b
R2          :00000000
R3          :00000000
R4          :00000000
```

```
                          .text
00001000:E3A0000F         MOV R0,#15
00001004:E1A01000         MOV R1,R0
00001008:E1A00100         MOV R0,R0,LSL #2
0000100C:E0800001         ADD R0,R0,R1
00001010:E1A01000         MOV R1,R0
00001014:E1A00180         MOV R0,R0,LSL #3
00001018:E0500001         SUBS R0,R0,R1
0000101C:EF000011         SWI 0x11
                          .end
```

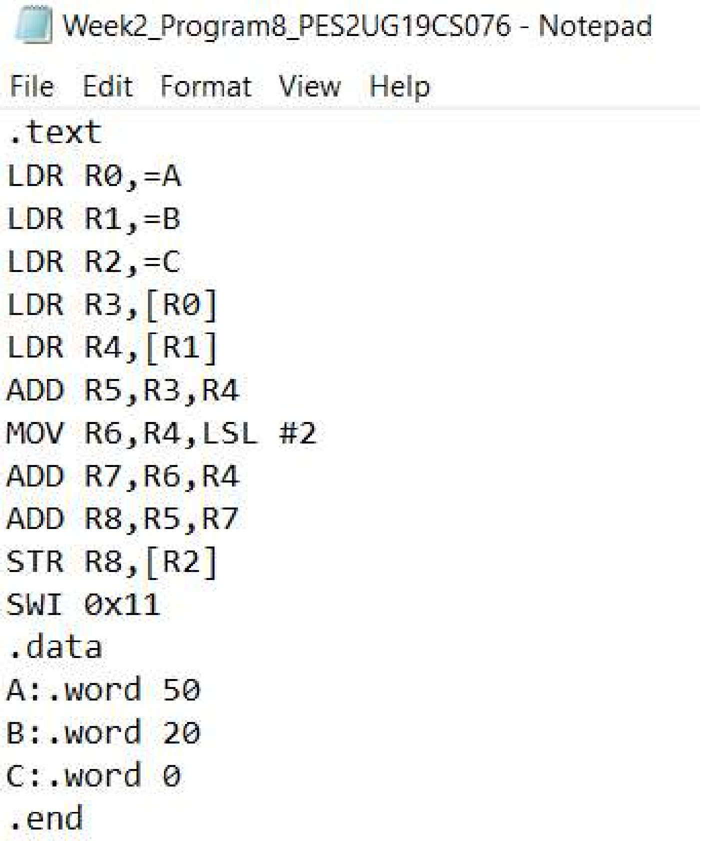
III. Input-Output Table

	Hexadecimal	Decimal
R0	0x0000020d	525
R1	0x4b	75

Week#____2____ Program Number: __8__

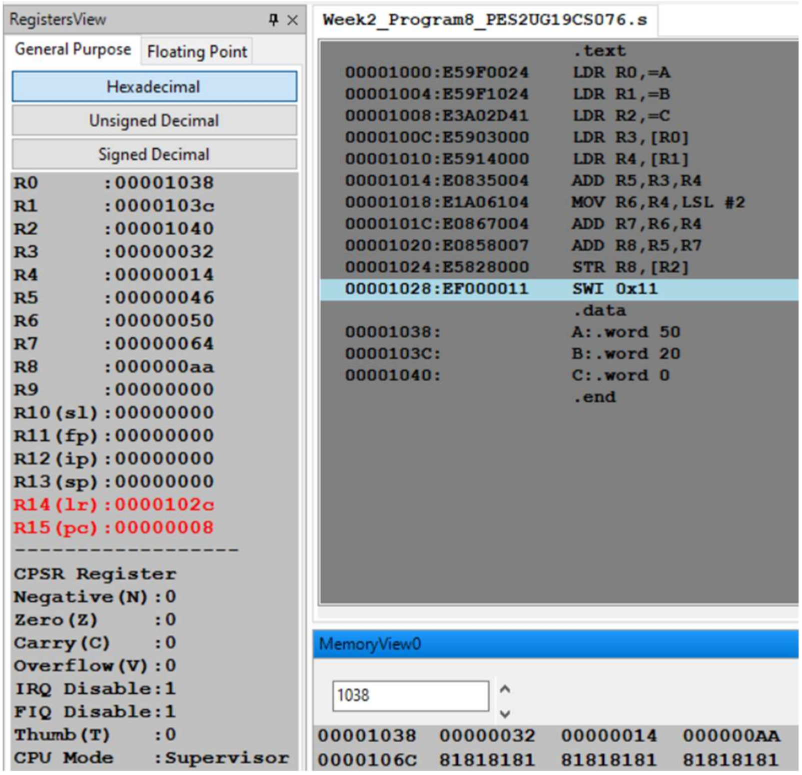
Title - Write an ALP to evaluate the expression (A+B) + (5*B), where A and B are available in memory location.

I. ARM Assembly Code


Week2_Program8_PES2UG19CS076 - Notepad
File Edit Format View Help

```
.text
LDR R0,=A
LDR R1,=B
LDR R2,=C
LDR R3,[R0]
LDR R4,[R1]
ADD R5,R3,R4
MOV R6,R4,LSL #2
ADD R7,R6,R4
ADD R8,R5,R7
STR R8,[R2]
SWI 0x11
.data
A:.word 50
B:.word 20
C:.word 0
.end
```

II. Output Screen Shot



```
R0         :00001038
R1         :0000103c
R2         :00001040
R3         :00000032
R4         :00000014
R5         :00000046
R6         :00000050
R7         :00000064
R8         :000000aa
R9         :00000000
R10(sl):00000000
R11(fp):00000000
R12(ip):00000000
R13(sp):00000000
R14(lr):0000102c
R15(pc):00000008
-------------------
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   :Supervisor
```

```
                        .text
00001000:E59F0024       LDR R0,=A
00001004:E59F1024       LDR R1,=B
00001008:E3A02D41       LDR R2,=C
0000100C:E5903000       LDR R3,[R0]
00001010:E5914000       LDR R4,[R1]
00001014:E0835004       ADD R5,R3,R4
00001018:E1A06104       MOV R6,R4,LSL #2
0000101C:E0867004       ADD R7,R6,R4
00001020:E0858007       ADD R8,R5,R7
00001024:E5828000       STR R8,[R2]
00001028:EF000011       SWI 0x11
                        .data
00001038:               A:.word 50
0000103C:               B:.word 20
00001040:               C:.word 0
                        .end
```

```
00001038  00000032  00000014  000000AA
0000106C  81818181  81818181  81818181
```

III. Input-Output Table

A = Decimal 50, B = Decimal 20		
	Hexadecimal	Decimal
R0	Address of A	
R1	Address of B	
R2	Address of C	
R3	0x32	50
R4	0x14	20
R5	0x46	70
R6	0x50	80
R7	0x64	100
R8	0xaa	170