Microprocessors LAB

EC210

Assignment 2

Submitted By:

Suhas S G

201EC263

Nikhil P Reddy

201EC241

2.1 Write an assembly program to take two 32 bit unsigned numbers in to the registers using MOV or MVN instructions and perform the following

(a) Add using ADD, ADDS, ADC

ADD R3, R1, R2 => addition of R2 and R1 stored in R3 without updating the cpsr/(flags Z,C,V,N)

R3 : R1 + R2

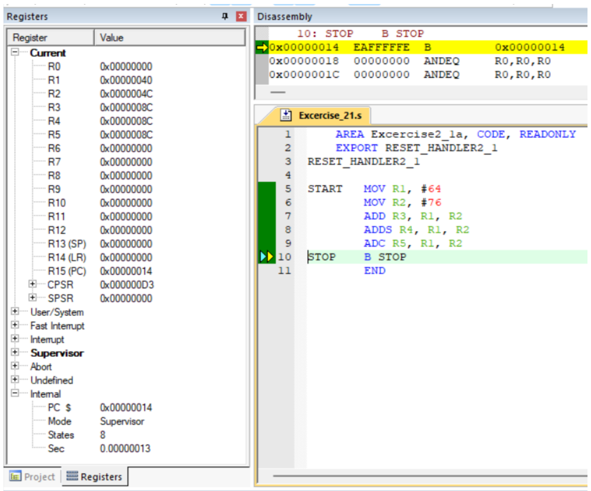
ADDS R3, R1, R2 =>addition of R2 and R1 stored in R3 and updates the cpsr/(flags Z,C,V,N)

R3: R1 + R2

ADDC R3, R1, R2 =>addition of R2, R1 and carry C stored in R3

R3: R1 + R2 + C

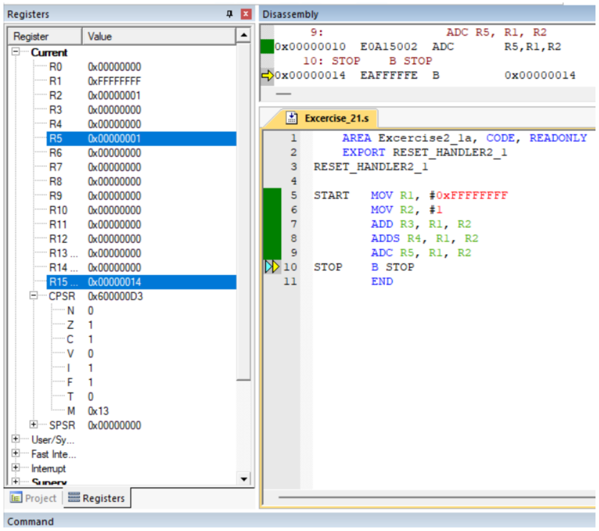
<u>OUTPUTS</u>



Register	Value
Current	
R0	0x00000000
R1	0x00000040
R2	0x0000004C
R3	0x0000008C
R4	0x0000008C
R5	0x0000008C
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x00000014
CPSR	0x000000D3
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x00000014
Mode	Supervisor
States	8
Sec	0.00000013

```
    10: STOP     B STOP
0x00000014  EAFFFFFE  B         0x00000014
0x00000018  00000000  ANDEQ     R0,R0,R0
0x0000001C  00000000  ANDEQ     R0,R0,R0
```

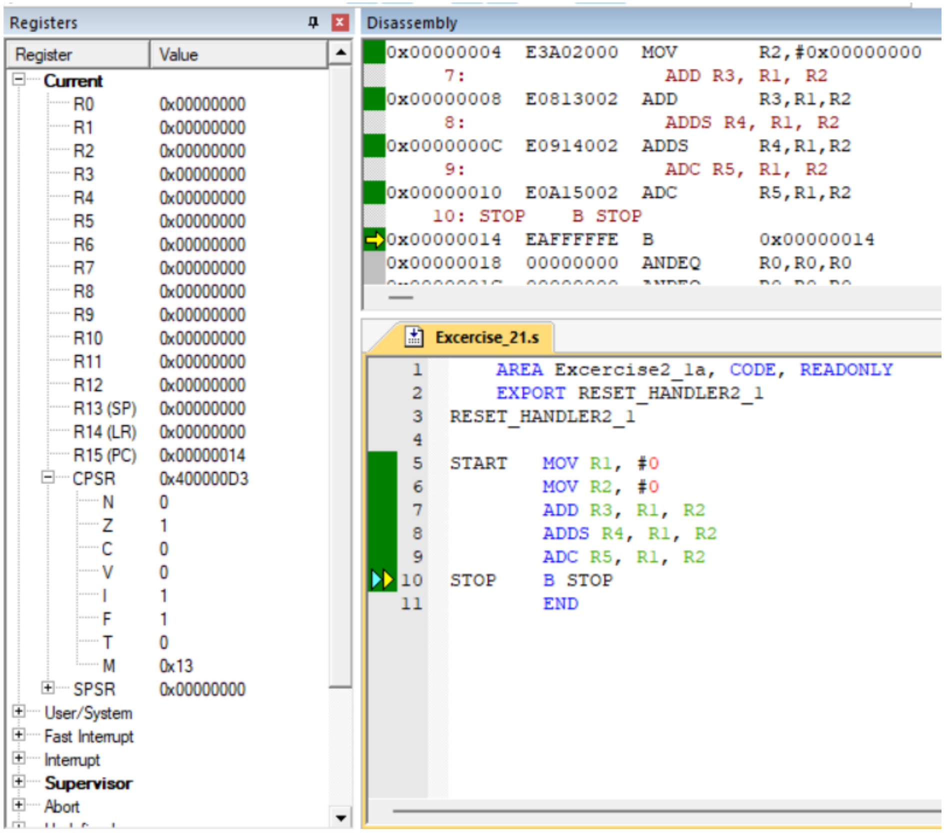
```
    AREA Excercise2_1a, CODE, READONLY
    EXPORT RESET_HANDLER2_1
RESET_HANDLER2_1

START   MOV R1, #64
        MOV R2, #76
        ADD R3, R1, R2
        ADDS R4, R1, R2
        ADC R5, R1, R2
STOP    B STOP
        END
```



Register	Value
Current	
R0	0x00000000
R1	0xFFFFFFFF
R2	0x00000001
R3	0x00000000
R4	0x00000000
R5	0x00000001
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 ...	0x00000000
R14 ...	0x00000000
R15 ...	0x00000014
CPSR	0x600000D3
N	0
Z	1
C	1
V	0
I	1
F	1
T	0
M	0x13
SPSR	0x00000000
User/Sy...	
Fast Inte...	
Interrupt	
Superv	

```
     9:                 ADC R5, R1, R2
0x00000010  E0A15002  ADC       R5,R1,R2
    10: STOP     B STOP
0x00000014  EAFFFFFE  B         0x00000014
```

```
    AREA Excercise2_1a, CODE, READONLY
    EXPORT RESET_HANDLER2_1
RESET_HANDLER2_1

START   MOV R1, #0xFFFFFFFF
        MOV R2, #1
        ADD R3, R1, R2
        ADDS R4, R1, R2
        ADC R5, R1, R2
STOP    B STOP
        END
```



Register	Value
Current	
R0	0x00000000
R1	0x00000000
R2	0x00000000
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x00000014
CPSR	0x400000D3
N	0
Z	1
C	0
V	0
I	1
F	1
T	0
M	0x13
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	

```
0x00000004  E3A02000  MOV       R2,#0x00000000
     7:                 ADD R3, R1, R2
0x00000008  E0813002  ADD       R3,R1,R2
     8:                 ADDS R4, R1, R2
0x0000000C  E0914002  ADDS      R4,R1,R2
     9:                 ADC R5, R1, R2
0x00000010  E0A15002  ADC       R5,R1,R2
    10: STOP     B STOP
0x00000014  EAFFFFFE  B         0x00000014
0x00000018  00000000  ANDEQ     R0,R0,R0
```

```
    AREA Excercise2_1a, CODE, READONLY
    EXPORT RESET_HANDLER2_1
RESET_HANDLER2_1

START   MOV R1, #0
        MOV R2, #0
        ADD R3, R1, R2
        ADDS R4, R1, R2
        ADC R5, R1, R2
STOP    B STOP
        END
```

As you can notice form the above examples the flags are only updated while using ADDS, and the carry is added to the sum using ADC

(b) Subtract using SUB, SUBS, SBC

SUB R3, R1, R2 => subtraction of R1 from R2 stored in R3 without updating the cpsr/(flags Z,C,V,N)

R3 : R1 - R2

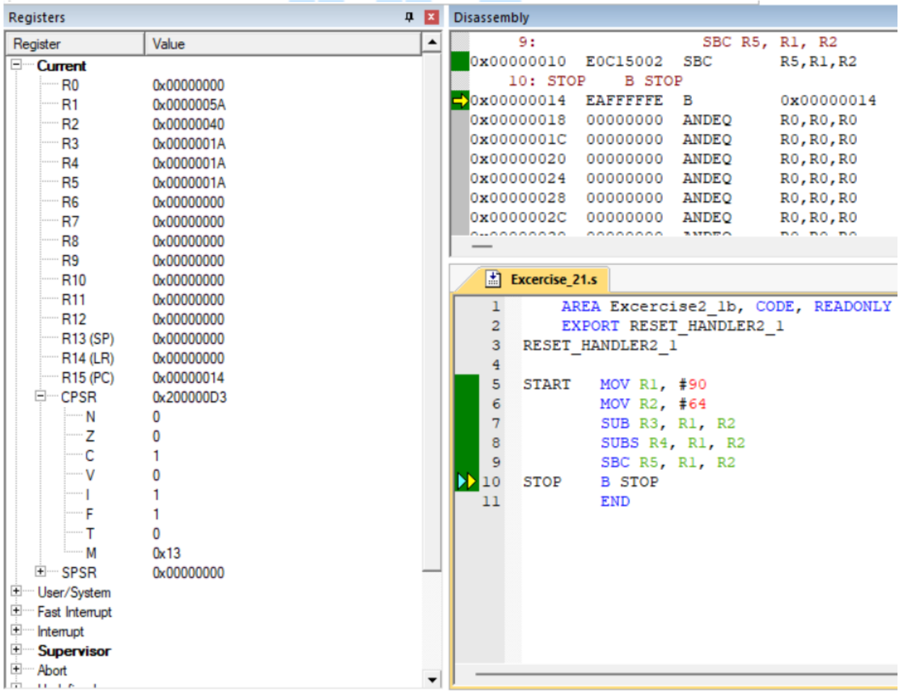
SUBS R3, R1, R2 =>subtraction of R1 from R2 stored in R3 and updates the cpsr/(flags Z,C,V,N)

R3: R1 - R2

SBC R3, R1, R2 =>subtraction of R1 and carry C from R2 stored in R3

R3: R1 - R2 - C

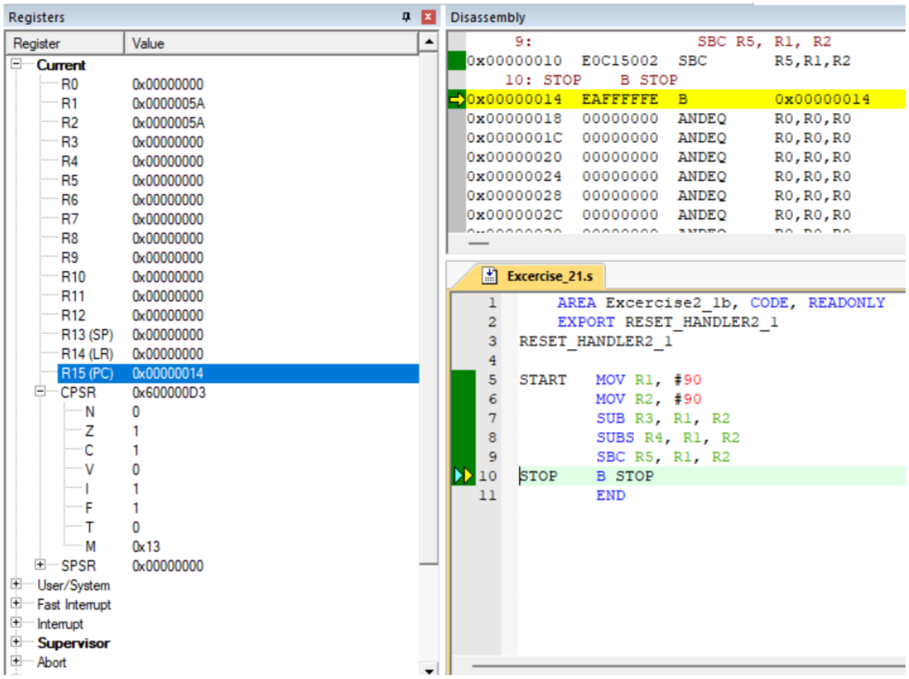
OUTPUTS



```
Register        Value
Current
  R0            0x00000000
  R1            0x0000005A
  R2            0x00000040
  R3            0x0000001A
  R4            0x0000001A
  R5            0x0000001A
  R6            0x00000000
  R7            0x00000000
  R8            0x00000000
  R9            0x00000000
  R10           0x00000000
  R11           0x00000000
  R12           0x00000000
  R13 (SP)      0x00000000
  R14 (LR)      0x00000000
  R15 (PC)      0x00000014
  CPSR          0x200000D3
    N           0
    Z           0
    C           1
    V           0
    I           1
    F           1
    T           0
    M           0x13
  SPSR          0x00000000
User/System
Fast Interrupt
Interrupt
Supervisor
Abort
```

```
     9:                 SBC R5, R1, R2
0x00000010  E0C15002  SBC       R5,R1,R2
    10: STOP     B STOP
0x00000014  EAFFFFFE  B         0x00000014
0x00000018  00000000  ANDEQ     R0,R0,R0
0x0000001C  00000000  ANDEQ     R0,R0,R0
0x00000020  00000000  ANDEQ     R0,R0,R0
0x00000024  00000000  ANDEQ     R0,R0,R0
0x00000028  00000000  ANDEQ     R0,R0,R0
0x0000002C  00000000  ANDEQ     R0,R0,R0
```

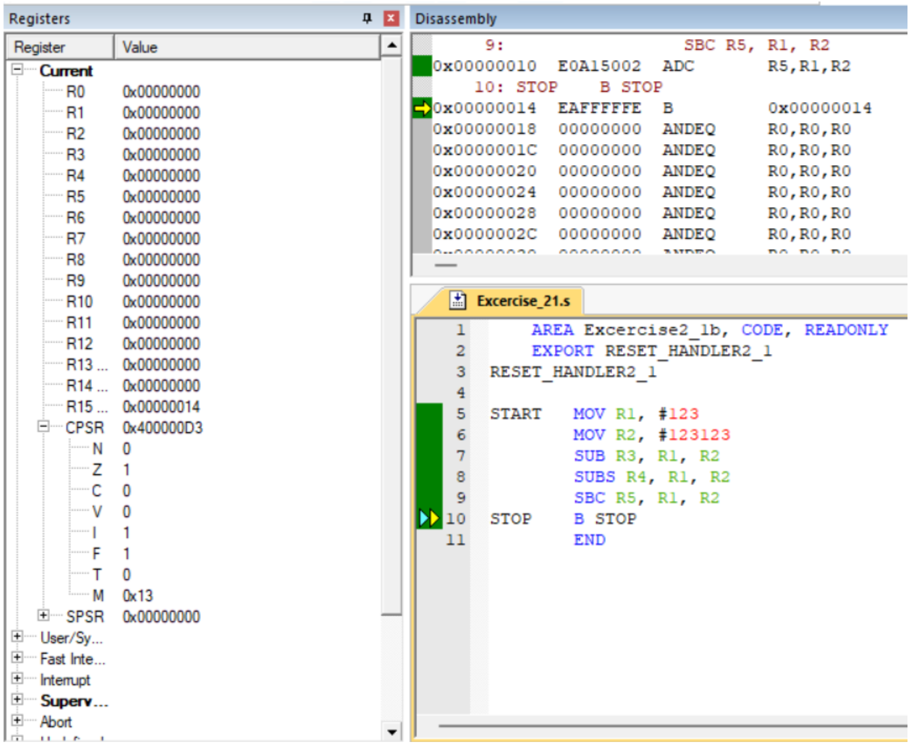
```
    AREA Excercise2_1b, CODE, READONLY
    EXPORT RESET_HANDLER2_1
RESET_HANDLER2_1

START   MOV R1, #90
        MOV R2, #64
        SUB R3, R1, R2
        SUBS R4, R1, R2
        SBC R5, R1, R2
STOP    B STOP
        END
```



```
Register        Value
Current
  R0            0x00000000
  R1            0x0000005A
  R2            0x0000005A
  R3            0x00000000
  R4            0x00000000
  R5            0x00000000
  R6            0x00000000
  R7            0x00000000
  R8            0x00000000
  R9            0x00000000
  R10           0x00000000
  R11           0x00000000
  R12           0x00000000
  R13 (SP)      0x00000000
  R14 (LR)      0x00000000
  R15 (PC)      0x00000014
  CPSR          0x600000D3
    N           0
    Z           1
    C           1
    V           0
    I           1
    F           1
    T           0
    M           0x13
  SPSR          0x00000000
User/System
Fast Interrupt
Interrupt
Supervisor
Abort
```

```
     9:                 SBC R5, R1, R2
0x00000010  E0C15002  SBC       R5,R1,R2
    10: STOP     B STOP
0x00000014  EAFFFFFE  B         0x00000014
0x00000018  00000000  ANDEQ     R0,R0,R0
0x0000001C  00000000  ANDEQ     R0,R0,R0
0x00000020  00000000  ANDEQ     R0,R0,R0
0x00000024  00000000  ANDEQ     R0,R0,R0
0x00000028  00000000  ANDEQ     R0,R0,R0
0x0000002C  00000000  ANDEQ     R0,R0,R0
```

```
    AREA Excercise2_1b, CODE, READONLY
    EXPORT RESET_HANDLER2_1
RESET_HANDLER2_1

START   MOV R1, #90
        MOV R2, #90
        SUB R3, R1, R2
        SUBS R4, R1, R2
        SBC R5, R1, R2
STOP    B STOP
        END
```

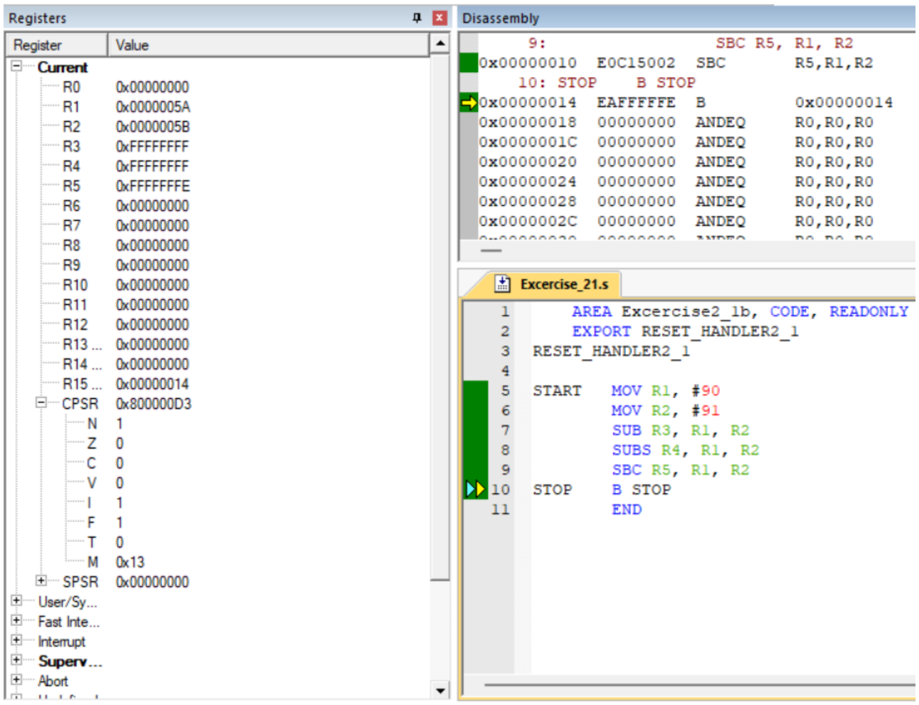


Register	Value
Current	
R0	0x00000000
R1	0x00000000
R2	0x00000000
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 ...	0x00000000
R14 ...	0x00000000
R15 ...	0x00000014
CPSR	0x400000D3
N	0
Z	1
C	0
V	0
I	1
F	1
T	0
M	0x13
SPSR	0x00000000
User/Sy...	
Fast Inte...	
Interrupt	
Superv...	
Abort	

```
       9:                     SBC R5, R1, R2
0x00000010  E0A15002  ADC       R5,R1,R2
      10: STOP     B STOP
0x00000014  EAFFFFFE  B         0x00000014
0x00000018  00000000  ANDEQ     R0,R0,R0
0x0000001C  00000000  ANDEQ     R0,R0,R0
0x00000020  00000000  ANDEQ     R0,R0,R0
0x00000024  00000000  ANDEQ     R0,R0,R0
0x00000028  00000000  ANDEQ     R0,R0,R0
0x0000002C  00000000  ANDEQ     R0,R0,R0
```

Excercise_21.s

```
        AREA Excercise2_1b, CODE, READONLY
        EXPORT RESET_HANDLER2_1
RESET_HANDLER2_1

START   MOV R1, #123
        MOV R2, #123123
        SUB R3, R1, R2
        SUBS R4, R1, R2
        SBC R5, R1, R2
STOP    B STOP
        END
```



Register	Value
Current	
R0	0x00000000
R1	0x0000005A
R2	0x0000005B
R3	0xFFFFFFFF
R4	0xFFFFFFFF
R5	0xFFFFFFFE
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 ...	0x00000000
R14 ...	0x00000000
R15 ...	0x00000014
CPSR	0x800000D3
N	1
Z	0
C	0
V	0
I	1
F	1
T	0
M	0x13
SPSR	0x00000000
User/Sy...	
Fast Inte...	
Interrupt	
Superv...	
Abort	

```
       9:                     SBC R5, R1, R2
0x00000010  E0C15002  SBC       R5,R1,R2
      10: STOP     B STOP
0x00000014  EAFFFFFE  B         0x00000014
0x00000018  00000000  ANDEQ     R0,R0,R0
0x0000001C  00000000  ANDEQ     R0,R0,R0
0x00000020  00000000  ANDEQ     R0,R0,R0
0x00000024  00000000  ANDEQ     R0,R0,R0
0x00000028  00000000  ANDEQ     R0,R0,R0
0x0000002C  00000000  ANDEQ     R0,R0,R0
```

Excercise_21.s

```
        AREA Excercise2_1b, CODE, READONLY
        EXPORT RESET_HANDLER2_1
RESET_HANDLER2_1

START   MOV R1, #90
        MOV R2, #91
        SUB R3, R1, R2
        SUBS R4, R1, R2
        SBC R5, R1, R2
STOP    B STOP
        END
```

(c) Reverse subtract using RSB, RSC

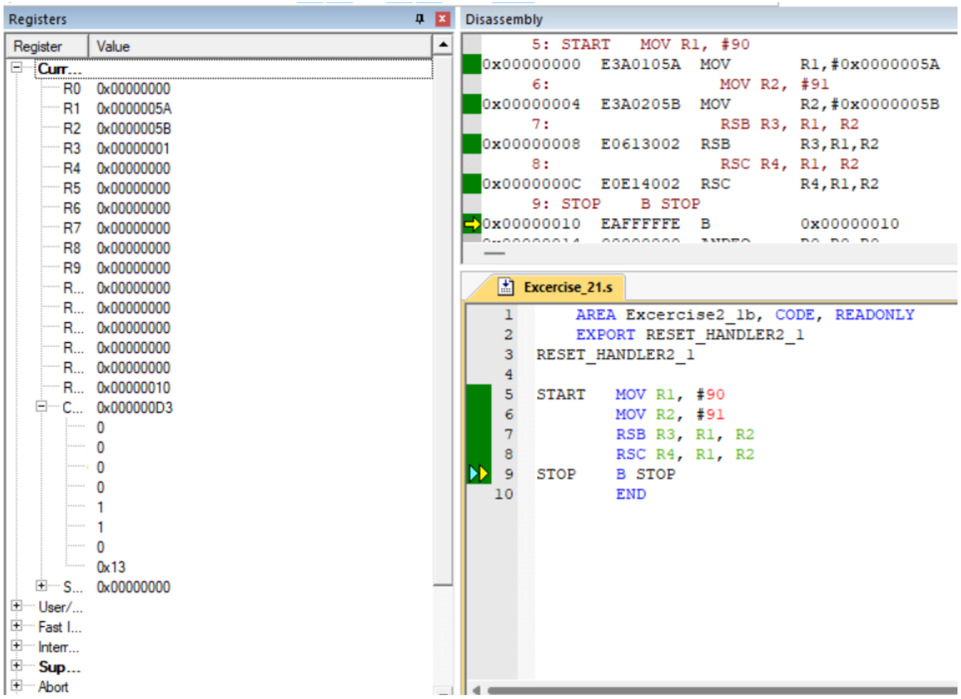
RSB R3, R1, R2 => subtraction of R2 from R1 stored in R3 without updating the cpsr/(flags Z,C,V,N)

R3 : R2-R1

RSC R3, R1, R2 =>subtraction of R2 and carry C from R1 stored in R3

R3: R1 - R2 - C

OUTPUTS

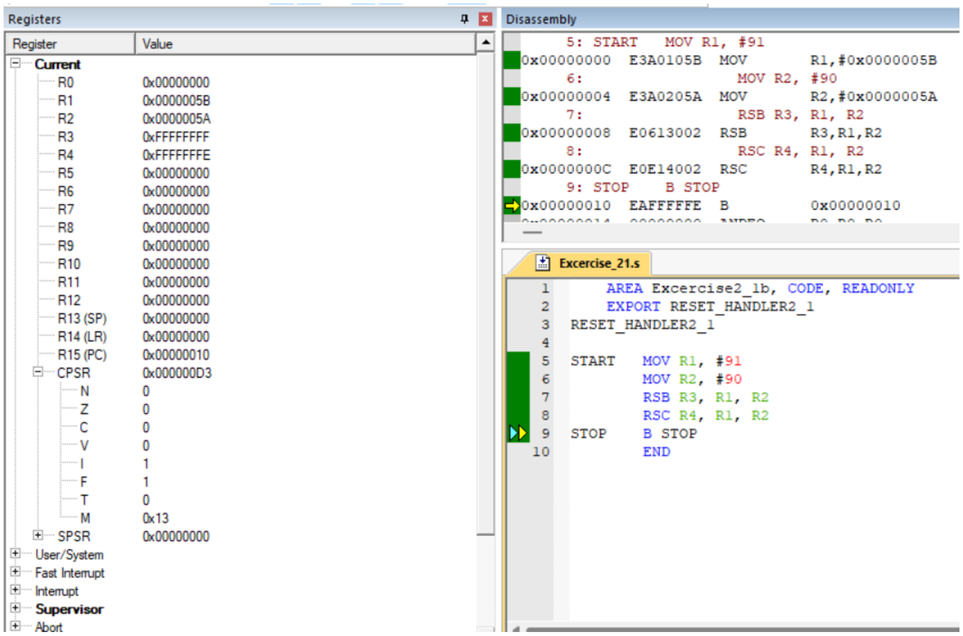


```
Registers
Register  Value
Curr...
  R0   0x00000000
  R1   0x0000005A
  R2   0x0000005B
  R3   0x00000001
  R4   0x00000000
  R5   0x00000000
  R6   0x00000000
  R7   0x00000000
  R8   0x00000000
  R9   0x00000000
  R... 0x00000000
  R... 0x00000000
  R... 0x00000000
  R... 0x00000000
  R... 0x00000000
  R... 0x00000010
  C... 0x000000D3
       0
       0
       0
       0
       1
       1
       0
       0x13
  S... 0x00000000
User/...
Fast I...
Interr...
Sup...
Abort
```

```
Disassembly
     5: START   MOV R1, #90
0x00000000  E3A0105A  MOV       R1,#0x0000005A
     6:                 MOV R2, #91
0x00000004  E3A0205B  MOV       R2,#0x0000005B
     7:                 RSB R3, R1, R2
0x00000008  E0613002  RSB       R3,R1,R2
     8:                 RSC R4, R1, R2
0x0000000C  E0E14002  RSC       R4,R1,R2
     9: STOP    B STOP
0x00000010  EAFFFFFE  B         0x00000010
```

Excercise_21.s

```
        AREA Excercise2_1b, CODE, READONLY
        EXPORT RESET_HANDLER2_1
RESET_HANDLER2_1

START   MOV R1, #90
        MOV R2, #91
        RSB R3, R1, R2
        RSC R4, R1, R2
STOP    B STOP
        END
```



```
Registers
Register    Value
Current
  R0        0x00000000
  R1        0x0000005B
  R2        0x0000005A
  R3        0xFFFFFFFF
  R4        0xFFFFFFFE
  R5        0x00000000
  R6        0x00000000
  R7        0x00000000
  R8        0x00000000
  R9        0x00000000
  R10       0x00000000
  R11       0x00000000
  R12       0x00000000
  R13 (SP)  0x00000000
  R14 (LR)  0x00000000
  R15 (PC)  0x00000010
  CPSR      0x000000D3
    N       0
    Z       0
    C       0
    V       0
    I       1
    F       1
    T       0
    M       0x13
  SPSR      0x00000000
User/System
Fast Interrupt
Interrupt
Supervisor
Abort
```

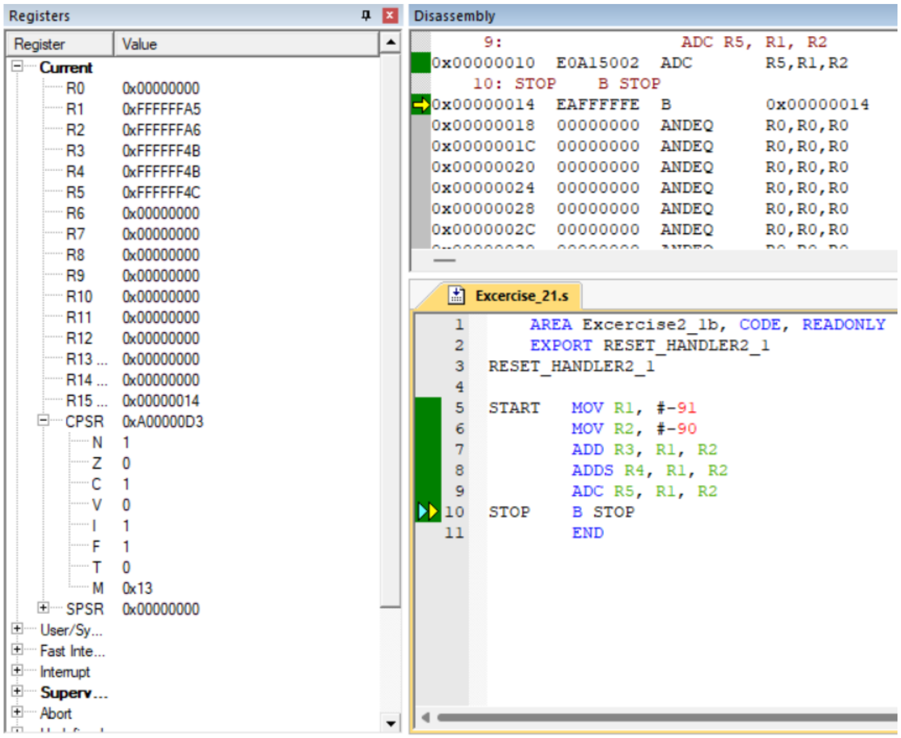
```
Disassembly
     5: START   MOV R1, #91
0x00000000  E3A0105B  MOV       R1,#0x0000005B
     6:                 MOV R2, #90
0x00000004  E3A0205A  MOV       R2,#0x0000005A
     7:                 RSB R3, R1, R2
0x00000008  E0613002  RSB       R3,R1,R2
     8:                 RSC R4, R1, R2
0x0000000C  E0E14002  RSC       R4,R1,R2
     9: STOP    B STOP
0x00000010  EAFFFFFE  B         0x00000010
```

Excercise_21.s

```
        AREA Excercise2_1b, CODE, READONLY
        EXPORT RESET_HANDLER2_1
RESET_HANDLER2_1

START   MOV R1, #91
        MOV R2, #90
        RSB R3, R1, R2
        RSC R4, R1, R2
STOP    B STOP
        END
```

2.2 (a) Repeat ex 2.1 a,b,c for 32 bit signed numbers

Add using ADD, ADDS, ADC



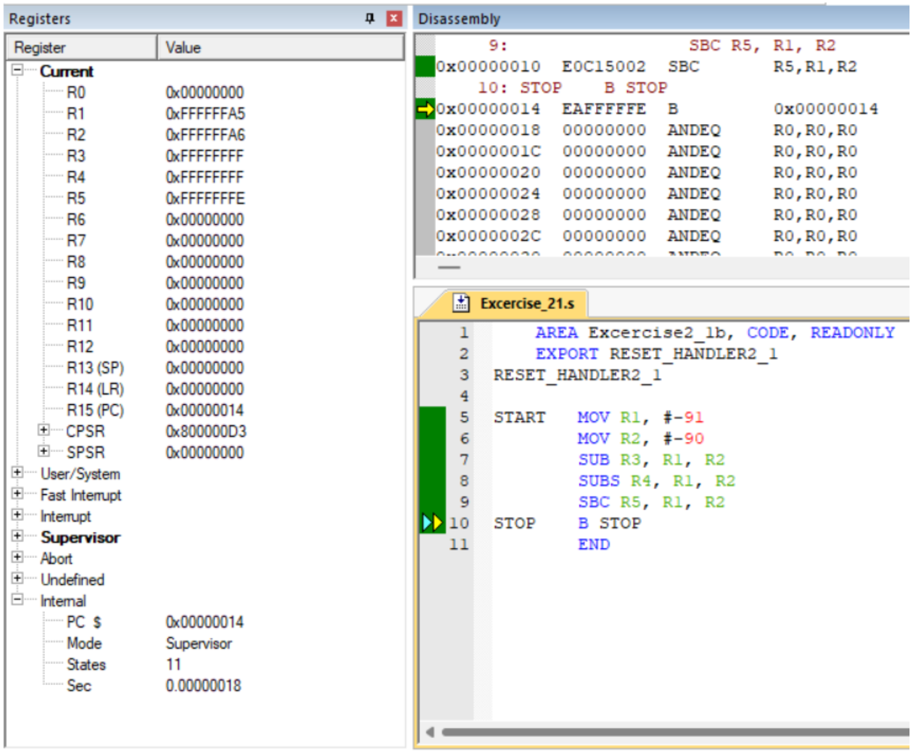
```
Registers
Register    Value
Current
  R0        0x00000000
  R1        0xFFFFFFA5
  R2        0xFFFFFFA6
  R3        0xFFFFFF4B
  R4        0xFFFFFF4B
  R5        0xFFFFFF4C
  R6        0x00000000
  R7        0x00000000
  R8        0x00000000
  R9        0x00000000
  R10       0x00000000
  R11       0x00000000
  R12       0x00000000
  R13 ...   0x00000000
  R14 ...   0x00000000
  R15 ...   0x00000014
  CPSR      0xA00000D3
    N       1
    Z       0
    C       1
    V       0
    I       1
    F       1
    T       0
    M       0x13
  SPSR      0x00000000
User/Sy...
Fast Inte...
Interrupt
Superv...
Abort
```

```
Disassembly
     9:                      ADC R5, R1, R2
0x00000010  E0A15002  ADC        R5,R1,R2
    10: STOP     B STOP
0x00000014  EAFFFFFE  B          0x00000014
0x00000018  00000000  ANDEQ      R0,R0,R0
0x0000001C  00000000  ANDEQ      R0,R0,R0
0x00000020  00000000  ANDEQ      R0,R0,R0
0x00000024  00000000  ANDEQ      R0,R0,R0
0x00000028  00000000  ANDEQ      R0,R0,R0
0x0000002C  00000000  ANDEQ      R0,R0,R0
```

Excercise_21.s

```
    AREA Excercise2_1b, CODE, READONLY
    EXPORT RESET_HANDLER2_1
RESET_HANDLER2_1

START   MOV R1, #-91
        MOV R2, #-90
        ADD R3, R1, R2
        ADDS R4, R1, R2
        ADC R5, R1, R2
STOP    B STOP
        END
```

Subtract using SUB, SUBS, SBC



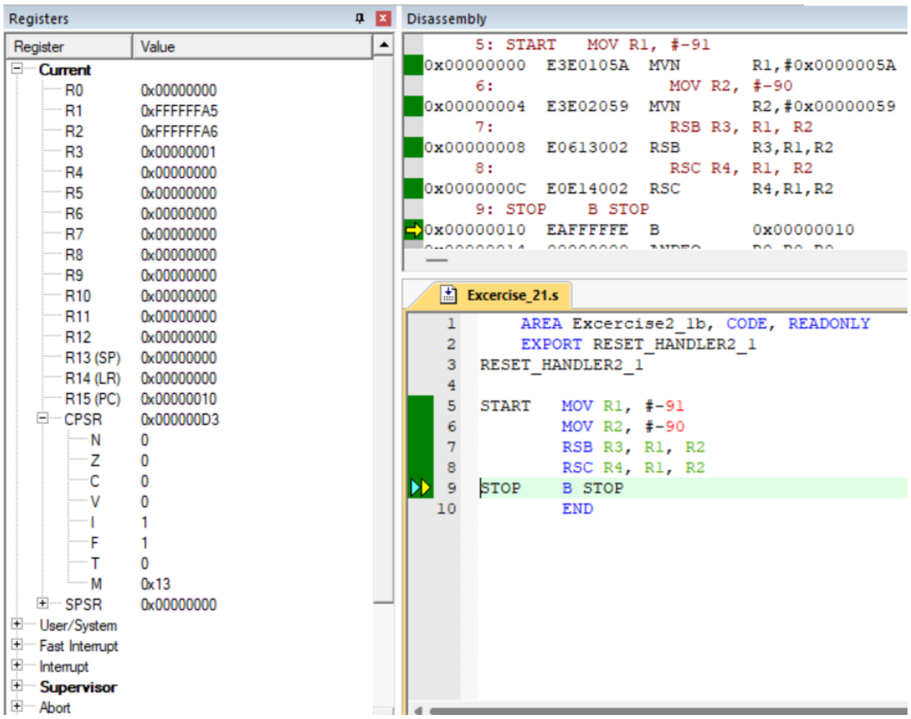
```
Registers
Register    Value
Current
  R0        0x00000000
  R1        0xFFFFFFA5
  R2        0xFFFFFFA6
  R3        0xFFFFFFFF
  R4        0xFFFFFFFF
  R5        0xFFFFFFFE
  R6        0x00000000
  R7        0x00000000
  R8        0x00000000
  R9        0x00000000
  R10       0x00000000
  R11       0x00000000
  R12       0x00000000
  R13 (SP)  0x00000000
  R14 (LR)  0x00000000
  R15 (PC)  0x00000014
  CPSR      0x800000D3
  SPSR      0x00000000
User/System
Fast Interrupt
Interrupt
Supervisor
Abort
Undefined
Internal
  PC $      0x00000014
  Mode      Supervisor
  States    11
  Sec       0.00000018
```

```
Disassembly
     9:                      SBC R5, R1, R2
0x00000010  E0C15002  SBC        R5,R1,R2
    10: STOP     B STOP
0x00000014  EAFFFFFE  B          0x00000014
0x00000018  00000000  ANDEQ      R0,R0,R0
0x0000001C  00000000  ANDEQ      R0,R0,R0
0x00000020  00000000  ANDEQ      R0,R0,R0
0x00000024  00000000  ANDEQ      R0,R0,R0
0x00000028  00000000  ANDEQ      R0,R0,R0
0x0000002C  00000000  ANDEQ      R0,R0,R0
```

Excercise_21.s

```
    AREA Excercise2_1b, CODE, READONLY
    EXPORT RESET_HANDLER2_1
RESET_HANDLER2_1

START   MOV R1, #-91
        MOV R2, #-90
        SUB R3, R1, R2
        SUBS R4, R1, R2
        SBC R5, R1, R2
STOP    B STOP
        END
```

Reverse subtract using RSB, RSC



```
Registers
Register        Value
Current
  R0            0x00000000
  R1            0xFFFFFFA5
  R2            0xFFFFFFA6
  R3            0x00000001
  R4            0x00000000
  R5            0x00000000
  R6            0x00000000
  R7            0x00000000
  R8            0x00000000
  R9            0x00000000
  R10           0x00000000
  R11           0x00000000
  R12           0x00000000
  R13 (SP)      0x00000000
  R14 (LR)      0x00000000
  R15 (PC)      0x00000010
  CPSR          0x000000D3
    N           0
    Z           0
    C           0
    V           0
    I           1
    F           1
    T           0
    M           0x13
  SPSR          0x00000000
User/System
Fast Interrupt
Interrupt
Supervisor
Abort
```

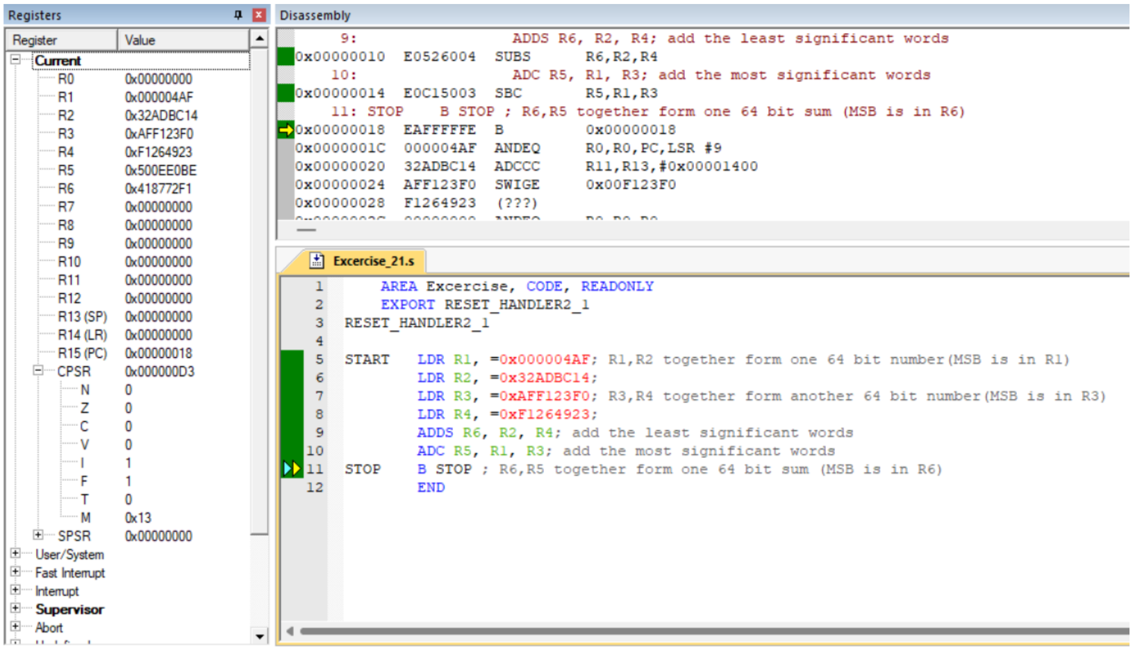
```
Disassembly
     5: START   MOV R1, #-91
0x00000000  E3E0105A  MVN        R1,#0x0000005A
     6:                  MOV R2, #-90
0x00000004  E3E02059  MVN        R2,#0x00000059
     7:                  RSB R3, R1, R2
0x00000008  E0613002  RSB        R3,R1,R2
     8:                  RSC R4, R1, R2
0x0000000C  E0E14002  RSC        R4,R1,R2
     9: STOP     B STOP
0x00000010  EAFFFFFE  B          0x00000010
```

```
Excercise_21.s
        AREA Excercise2_1b, CODE, READONLY
        EXPORT RESET_HANDLER2_1
RESET_HANDLER2_1

START       MOV R1, #-91
            MOV R2, #-90
            RSB R3, R1, R2
            RSC R4, R1, R2
STOP        B STOP
            END
```

(c) Repeat ex 2.1 a,b,c for 64 bit unsigned numbers.

For 64 bit unsigned numbers we use two different registers to store each number and perform operations by using the carry and overflow flag in the CPSR

Add using ADD, ADDS, ADC

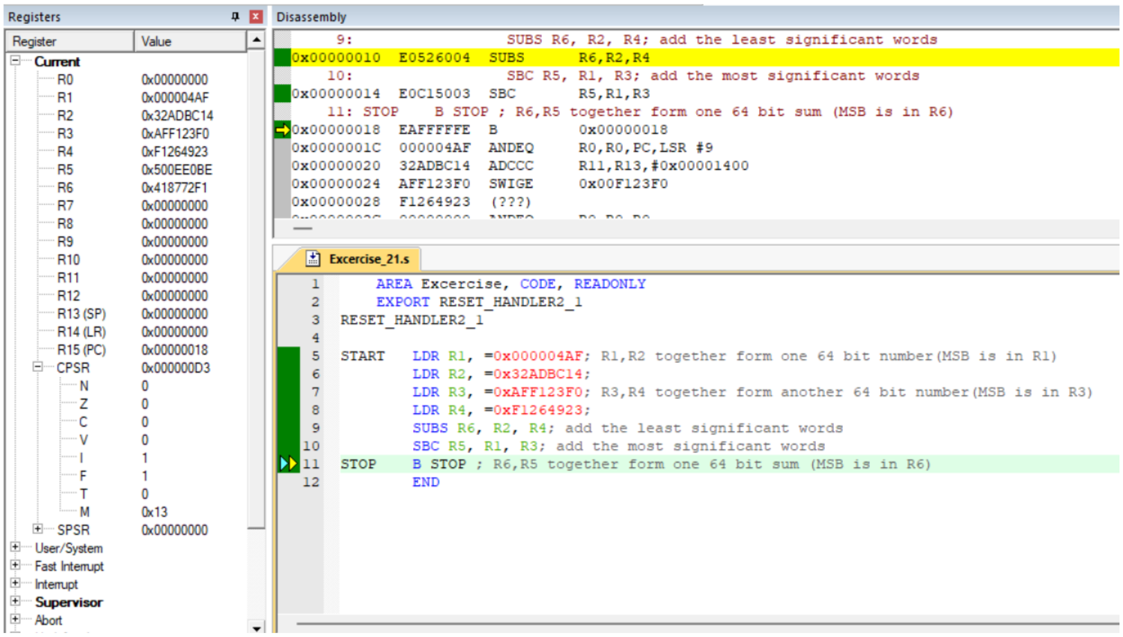


```
Registers
Register        Value
Current
  R0            0x00000000
  R1            0x000004AF
  R2            0x32ADBC14
  R3            0xAFF123F0
  R4            0xF1264923
  R5            0x500EE0BE
  R6            0x418772F1
  R7            0x00000000
  R8            0x00000000
  R9            0x00000000
  R10           0x00000000
  R11           0x00000000
  R12           0x00000000
  R13 (SP)      0x00000000
  R14 (LR)      0x00000000
  R15 (PC)      0x00000018
  CPSR          0x000000D3
    N           0
    Z           0
    C           0
    V           0
    I           1
    F           1
    T           0
    M           0x13
  SPSR          0x00000000
User/System
Fast Interrupt
Interrupt
Supervisor
Abort
```

```
Disassembly
     9:                  ADDS R6, R2, R4; add the least significant words
0x00000010  E0526004  SUBS       R6,R2,R4
    10:                  ADC R5, R1, R3; add the most significant words
0x00000014  E0C15003  SBC        R5,R1,R3
    11: STOP     B STOP ; R6,R5 together form one 64 bit sum (MSB is in R6)
0x00000018  EAFFFFFE  B          0x00000018
0x0000001C  000004AF  ANDEQ      R0,R0,PC,LSR #9
0x00000020  32ADBC14  ADCCC      R11,R13,#0x00001400
0x00000024  AFF123F0  SWIGE      0x00F123F0
0x00000028  F1264923  (???)
```

```
Excercise_21.s
        AREA Excercise, CODE, READONLY
        EXPORT RESET_HANDLER2_1
RESET_HANDLER2_1

START       LDR R1, =0x000004AF; R1,R2 together form one 64 bit number(MSB is in R1)
            LDR R2, =0x32ADBC14;
            LDR R3, =0xAFF123F0; R3,R4 together form another 64 bit number(MSB is in R3)
            LDR R4, =0xF1264923;
            ADDS R6, R2, R4; add the least significant words
            ADC R5, R1, R3; add the most significant words
STOP        B STOP ; R6,R5 together form one 64 bit sum (MSB is in R6)
            END
```

Subtract using SUB, SUBS, SBC



Registers

Register	Value
Current	
R0	0x00000000
R1	0x000004AF
R2	0x32ADBC14
R3	0xAFF123F0
R4	0xF1264923
R5	0x500EE0BE
R6	0x418772F1
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x00000018
CPSR	0x000000D3
N	0
Z	0
C	0
V	0
I	1
F	1
T	0
M	0x13
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	

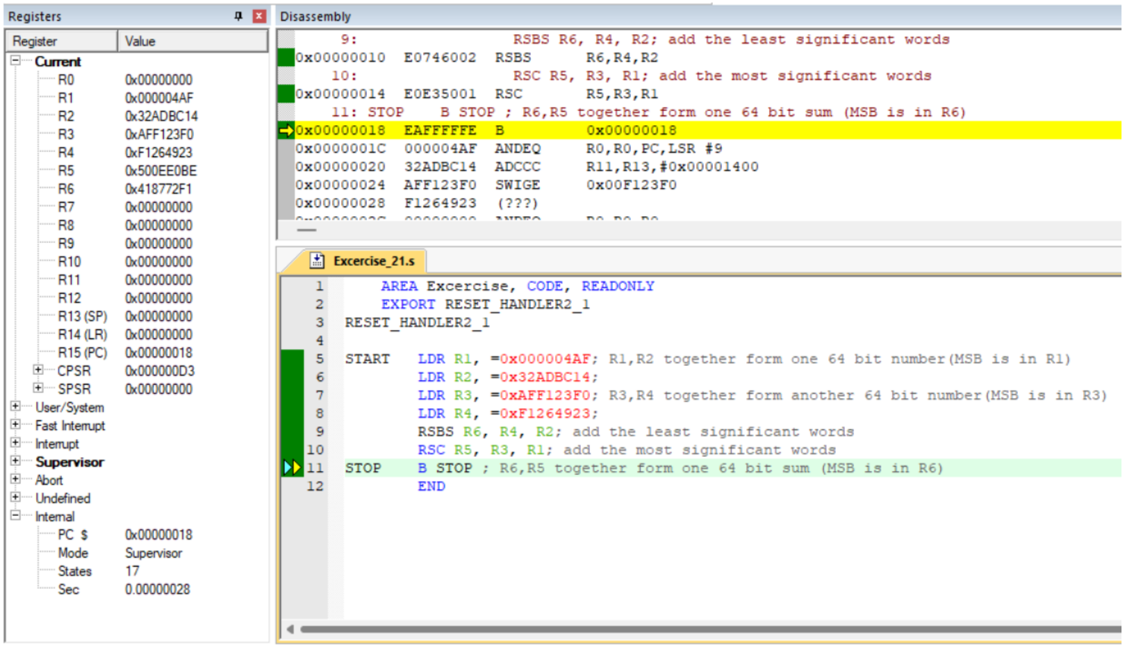
Disassembly

```
     9:                  SUBS R6, R2, R4; add the least significant words
0x00000010  E0526004  SUBS      R6,R2,R4
    10:                  SBC R5, R1, R3; add the most significant words
0x00000014  E0C15003  SBC       R5,R1,R3
    11: STOP     B STOP ; R6,R5 together form one 64 bit sum (MSB is in R6)
0x00000018  EAFFFFFE  B         0x00000018
0x0000001C  000004AF  ANDEQ     R0,R0,PC,LSR #9
0x00000020  32ADBC14  ADCCC     R11,R13,#0x00001400
0x00000024  AFF123F0  SWIGE     0x00F123F0
0x00000028  F1264923  (???)
```

Excercise_21.s

```
    AREA Excercise, CODE, READONLY
    EXPORT RESET_HANDLER2_1
RESET_HANDLER2_1

START   LDR R1, =0x000004AF; R1,R2 together form one 64 bit number(MSB is in R1)
        LDR R2, =0x32ADBC14;
        LDR R3, =0xAFF123F0; R3,R4 together form another 64 bit number(MSB is in R3)
        LDR R4, =0xF1264923;
        SUBS R6, R2, R4; add the least significant words
        SBC R5, R1, R3; add the most significant words
STOP    B STOP ; R6,R5 together form one 64 bit sum (MSB is in R6)
        END
```

Reverse subtract using RSB, RSC



Registers

Register	Value
Current	
R0	0x00000000
R1	0x000004AF
R2	0x32ADBC14
R3	0xAFF123F0
R4	0xF1264923
R5	0x500EE0BE
R6	0x418772F1
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x00000018
CPSR	0x000000D3
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x00000018
Mode	Supervisor
States	17
Sec	0.00000028

Disassembly

```
     9:                  RSBS R6, R4, R2; add the least significant words
0x00000010  E0746002  RSBS      R6,R4,R2
    10:                  RSC R5, R3, R1; add the most significant words
0x00000014  E0E35001  RSC       R5,R3,R1
    11: STOP     B STOP ; R6,R5 together form one 64 bit sum (MSB is in R6)
0x00000018  EAFFFFFE  B         0x00000018
0x0000001C  000004AF  ANDEQ     R0,R0,PC,LSR #9
0x00000020  32ADBC14  ADCCC     R11,R13,#0x00001400
0x00000024  AFF123F0  SWIGE     0x00F123F0
0x00000028  F1264923  (???)
```

Excercise_21.s

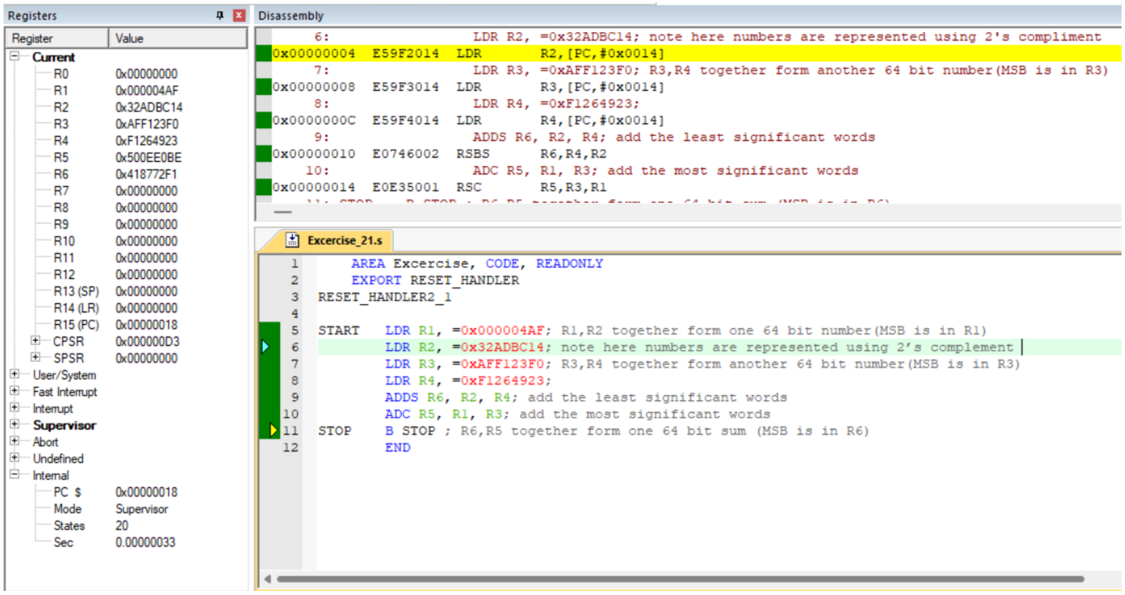
```
    AREA Excercise, CODE, READONLY
    EXPORT RESET_HANDLER2_1
RESET_HANDLER2_1

START   LDR R1, =0x000004AF; R1,R2 together form one 64 bit number(MSB is in R1)
        LDR R2, =0x32ADBC14;
        LDR R3, =0xAFF123F0; R3,R4 together form another 64 bit number(MSB is in R3)
        LDR R4, =0xF1264923;
        RSBS R6, R4, R2; add the least significant words
        RSC R5, R3, R1; add the most significant words
STOP    B STOP ; R6,R5 together form one 64 bit sum (MSB is in R6)
        END
```

(b) Repeat ex 2.1 a,b,c for 64 bit signed numbers.

For 64 bit signed numbers we use two different registers to store each number and perform operations by using the carry and overflow flag in the CPSR.

And the numbers are represented using 2's complement method otherwise the process is same as the signed numbers

Add using ADD, ADDS, ADC



Registers

Register	Value
Current	
R0	0x00000000
R1	0x000004AF
R2	0x32ADBC14
R3	0xAFF123F0
R4	0xF1264923
R5	0x500EE0BE
R6	0x418772F1
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x00000018
CPSR	0x000000D3
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x00000018
Mode	Supervisor
States	20
Sec	0.00000033

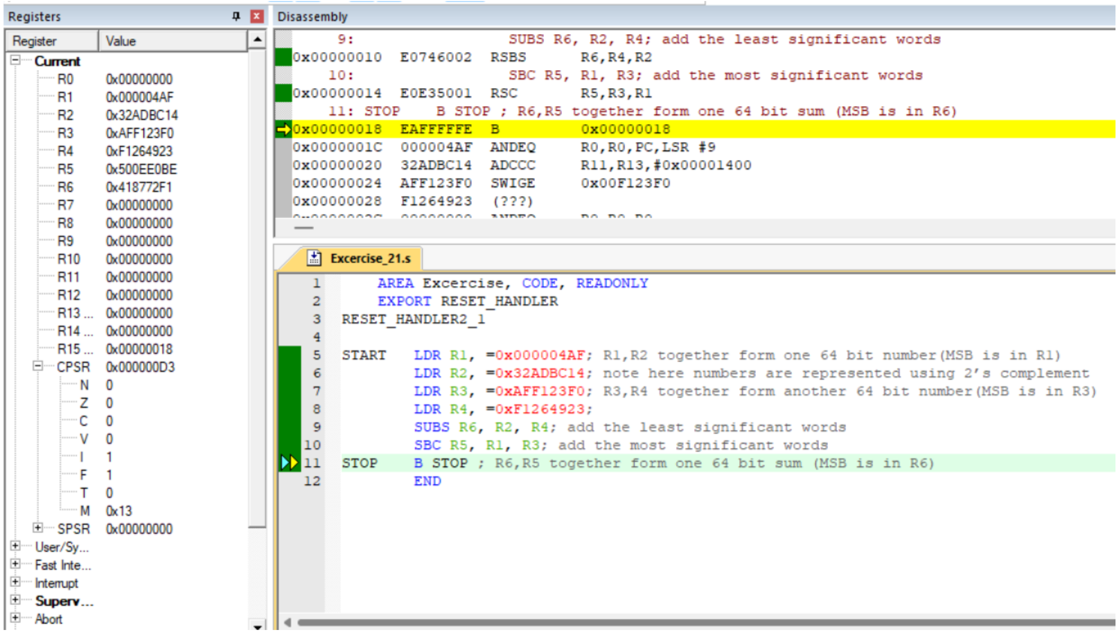
Disassembly

```
      6:                LDR R2, =0x32ADBC14; note here numbers are represented using 2's compliment
0x00000004  E59F2014  LDR       R2,[PC,#0x0014]
      7:                LDR R3, =0xAFF123F0; R3,R4 together form another 64 bit number(MSB is in R3)
0x00000008  E59F3014  LDR       R3,[PC,#0x0014]
      8:                LDR R4, =0xF1264923;
0x0000000C  E59F4014  LDR       R4,[PC,#0x0014]
      9:                ADDS R6, R2, R4; add the least significant words
0x00000010  E0746002  RSBS      R6,R4,R2
     10:                ADC R5, R1, R3; add the most significant words
0x00000014  E0E35001  RSC       R5,R3,R1
```

Excercise_21.s

```
    AREA Excercise, CODE, READONLY
    EXPORT RESET_HANDLER
RESET_HANDLER2_1

START   LDR R1, =0x000004AF; R1,R2 together form one 64 bit number(MSB is in R1)
        LDR R2, =0x32ADBC14; note here numbers are represented using 2's complement
        LDR R3, =0xAFF123F0; R3,R4 together form another 64 bit number(MSB is in R3)
        LDR R4, =0xF1264923;
        ADDS R6, R2, R4; add the least significant words
        ADC R5, R1, R3; add the most significant words
STOP    B STOP ; R6,R5 together form one 64 bit sum (MSB is in R6)
        END
```

Subtract using SUB, SUBS, SBC



Registers

Register	Value
Current	
R0	0x00000000
R1	0x000004AF
R2	0x32ADBC14
R3	0xAFF123F0
R4	0xF1264923
R5	0x500EE0BE
R6	0x418772F1
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 ...	0x00000000
R14 ...	0x00000000
R15 ...	0x00000018
CPSR	0x000000D3
N	0
Z	0
C	0
V	0
I	1
F	1
T	0
M	0x13
SPSR	0x00000000
User/Sy...	
Fast Inte...	
Interrupt	
Superv...	
Abort	

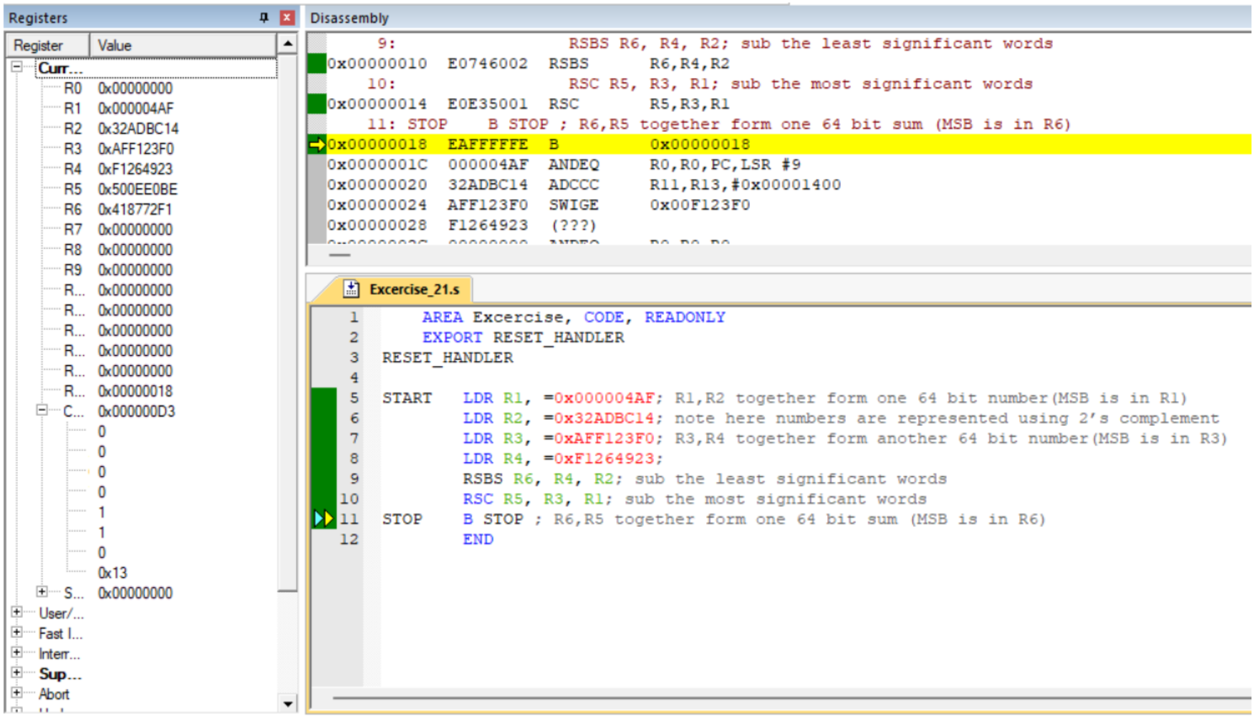
Disassembly

```
      9:                SUBS R6, R2, R4; add the least significant words
0x00000010  E0746002  RSBS      R6,R4,R2
     10:                SBC R5, R1, R3; add the most significant words
0x00000014  E0E35001  RSC       R5,R3,R1
     11: STOP     B STOP ; R6,R5 together form one 64 bit sum (MSB is in R6)
0x00000018  EAFFFFFE  B         0x00000018
0x0000001C  000004AF  ANDEQ     R0,R0,PC,LSR #9
0x00000020  32ADBC14  ADCCC     R11,R13,#0x00001400
0x00000024  AFF123F0  SWIGE     0x00F123F0
0x00000028  F1264923  (???)
```

Excercise_21.s

```
    AREA Excercise, CODE, READONLY
    EXPORT RESET_HANDLER
RESET_HANDLER2_1

START   LDR R1, =0x000004AF; R1,R2 together form one 64 bit number(MSB is in R1)
        LDR R2, =0x32ADBC14; note here numbers are represented using 2's complement
        LDR R3, =0xAFF123F0; R3,R4 together form another 64 bit number(MSB is in R3)
        LDR R4, =0xF1264923;
        SUBS R6, R2, R4; add the least significant words
        SBC R5, R1, R3; add the most significant words
STOP    B STOP ; R6,R5 together form one 64 bit sum (MSB is in R6)
        END
```

Reverse subtract using RSB, RSC



Register	Value
Curr...	
R0	0x00000000
R1	0x000004AF
R2	0x32ADBC14
R3	0xAFF123F0
R4	0xF1264923
R5	0x500EE0BE
R6	0x418772F1
R7	0x00000000
R8	0x00000000
R9	0x00000000
R...	0x00000000
R...	0x00000000
R...	0x00000000
R...	0x00000000
R...	0x00000000
R...	0x00000018
C...	0x000000D3
	0
	0
	0
	0
	1
	1
	0
	0x13
S...	0x00000000
User/...	
Fast I...	
Interr...	
Sup...	
Abort	

```
     9:                  RSBS R6, R4, R2; sub the least significant words
0x00000010  E0746002  RSBS      R6,R4,R2
    10:                  RSC R5, R3, R1; sub the most significant words
0x00000014  E0E35001  RSC       R5,R3,R1
    11: STOP     B STOP ; R6,R5 together form one 64 bit sum (MSB is in R6)
0x00000018  EAFFFFFE  B         0x00000018
0x0000001C  000004AF  ANDEQ     R0,R0,PC,LSR #9
0x00000020  32ADBC14  ADCCC     R11,R13,#0x00001400
0x00000024  AFF123F0  SWIGE     0x00F123F0
0x00000028  F1264923  (???)
```

```
    AREA Excercise, CODE, READONLY
    EXPORT RESET_HANDLER
RESET_HANDLER

START   LDR R1, =0x000004AF; R1,R2 together form one 64 bit number(MSB is in R1)
        LDR R2, =0x32ADBC14; note here numbers are represented using 2's complement
        LDR R3, =0xAFF123F0; R3,R4 together form another 64 bit number(MSB is in R3)
        LDR R4, =0xF1264923;
        RSBS R6, R4, R2; sub the least significant words
        RSC R5, R3, R1; sub the most significant words
STOP    B STOP ; R6,R5 together form one 64 bit sum (MSB is in R6)
        END
```