

LAB 2Solved Exercise:

Write a program to add two 32-bit numbers available in the code memory. Store the result in the data memory.

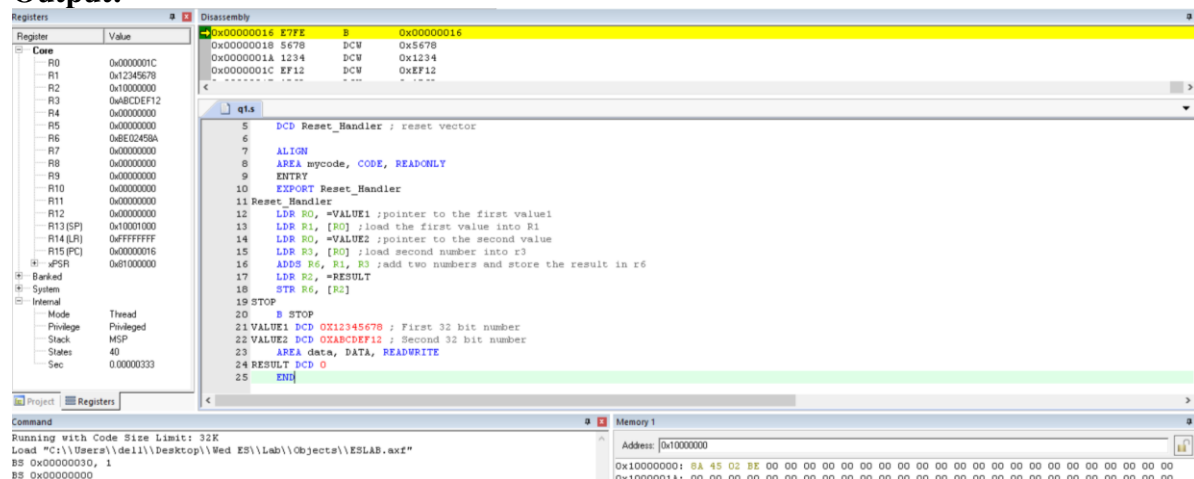
Program:

```

AREA RESET, DATA, READONLY
EXPORT __Vectors
__Vectors
    DCD 0x10001000 ; stack pointer value when stack is empty
    DCD Reset_Handler ; reset vector
    ALIGN
    AREA mycode, CODE, READONLY
    ENTRY
    EXPORT Reset_Handler
Reset_Handler
    LDR R0, =VALUE1 ;pointer to the first value1
    LDR R1, [R0] ;load the first value into R1
    LDR R0, =VALUE2 ;pointer to the second value
    LDR R3, [R0] ;load second number into r3
    ADDS R6, R1, R3 ;add two numbers and store the result in r6
    LDR R2, =RESULT
    STR R6, [R2]

STOP
    B STOP
VALUE1 DCD 0X12345678 ; First 32 bit number
VALUE2 DCD 0XABCDEF12 ; Second 32 bit number
    AREA data, DATA, READWRITE
RESULT DCD 0
    END

```

Output:

Lab Exercises:

1. Write a program to add ten 32-bit numbers available in code memory and store the result in data memory.

Program:

```
        AREA RESET,DATA,READONLY
        EXPORT __Vectors

__Vectors

        DCD 0X10001000
        DCD Reset_Handler

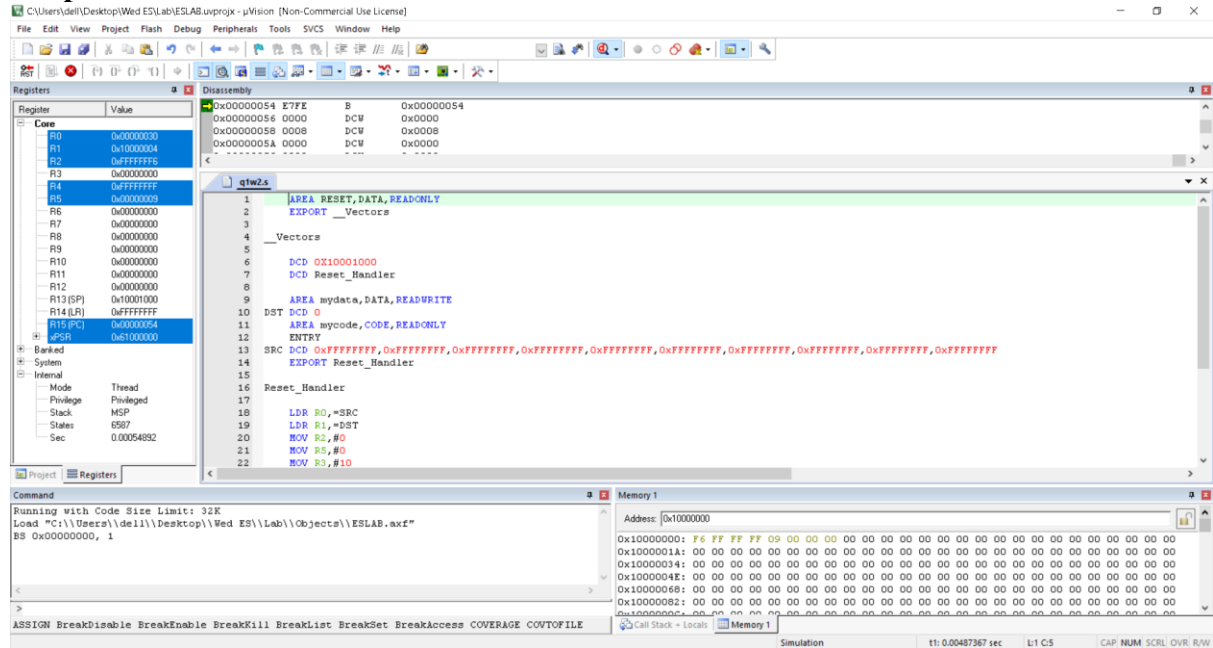
        AREA mydata,DATA,READWRITE
DST DCD 0
        AREA mycode,CODE,READONLY
        ENTRY
SRC DCD 0xFFFFFFFF,0xFFFFFFFF,0xFFFFFFFF,0xFFFFFFFF,0xFFFFFFFF,
        0xFFFFFFFF, 0xFFFFFFFF,0xFFFFFFFF,0xFFFFFFFF,0xFFFFFFFF
        EXPORT Reset_Handler

Reset_Handler

        LDR R0,=SRC
        LDR R1,=DST
        MOV R2,#0
        MOV R5,#0
        MOV R3,#10

UP      LDR R4,[R0],#4
        ADDS R2,R4
        ADC R5,#0
        SUBS R3,#1
        BNE UP
        STR R2,[R1],#4
        STR R5,[R1]
STOP B STOP
        END
```

Output:



2. Write a program to add two 128-bit numbers available in code memory and store the result in data memory.

Hint: Use indexed addressing mode.

Program:

```

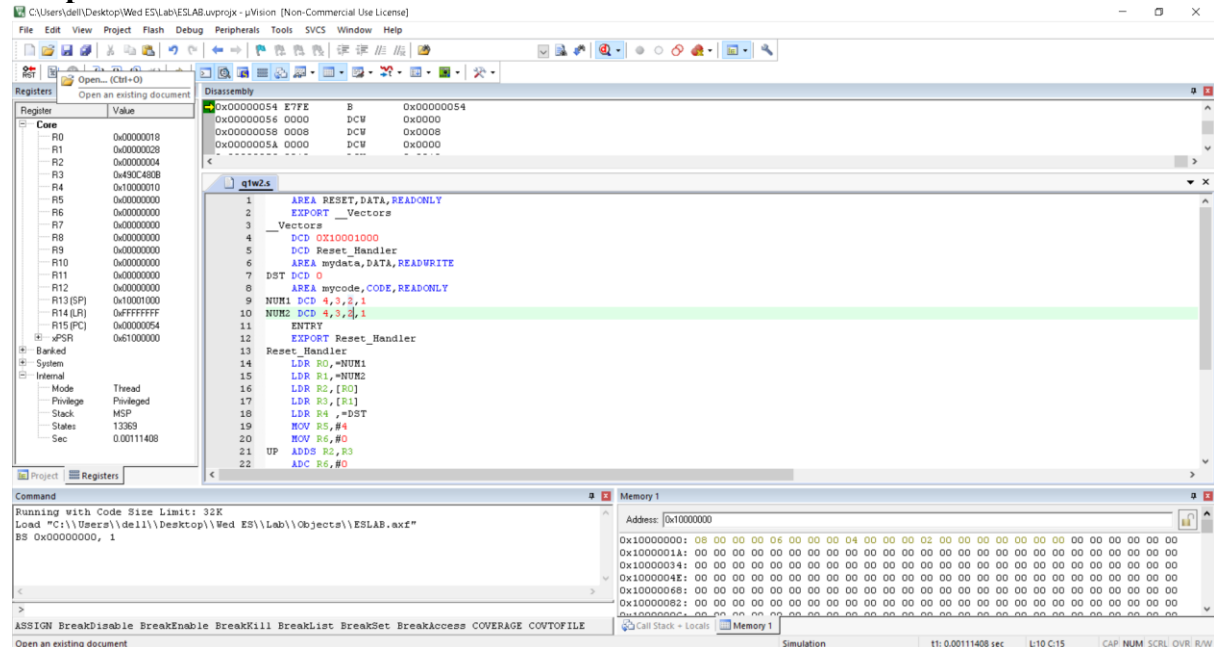
        AREA RESET,DATA,READONLY
        EXPORT __Vectors
__Vectors
        DCD 0x10001000
        DCD Reset_Handler
        AREA mydata,DATA,READWRITE
DST DCD 0
        AREA mycode,CODE,READONLY
NUM1 DCD 4,3,2,1
NUM2 DCD 4,3,2,1
        ENTRY
        EXPORT Reset_Handler
Reset_Handler
        LDR R0,=NUM1
        LDR R1,=NUM2
        LDR R2,[R0]
        LDR R3,[R1]
        LDR R4,=DST
        MOV R5,#4
        MOV R6,#0
UP      ADDS R2,R3
        ADC R6,#0
        STR R2,[R4],#4
        LDR R2,[R0,#4]!
        LDR R3,[R1,#4]!
        SUBS R5,#1
    
```

```

BNE UP
STR R6,[R4]
ALIGN
STOP
B STOP
END

```

Output:



3. Write a program to subtract two 32-bit numbers available in the code memory and store the result in the data memory.

Program:

```

AREA RESET ,DATA, READONLY
EXPORT __Vectors

```

__Vectors

```

DCD 0X10001000
DCD Reset_Handler
AREA mydata,DATA,READWRITE
DST DCD 0
AREA mycode,CODE,READONLY
ENTRY
EXPORT Reset_Handler

```

Reset_Handler

```

LDR R0,=NUM1
LDR R1,[R0]
LDR R0,=NUM2
LDR R2,[R0]
MOV R3,#0
SUBS R3,R2,R1

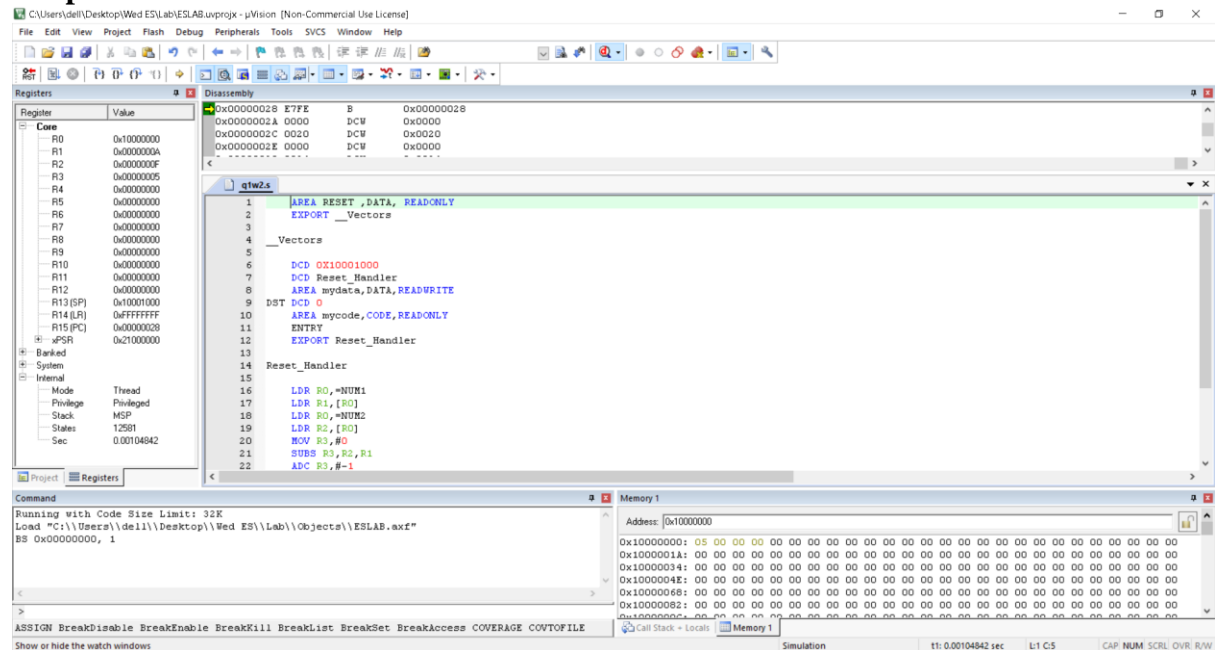
```

```

ADC R3,#-1
LDR R0,=DST
STR R3,[R0]
ALIGN
NUM1 DCD 10
NUM2 DCD 15
STOP B STOP
END

```

Output:



4. Write a program to subtract two 128-bit numbers available in the code memory and store the result in the data memory.

Program:

```

AREA RESET,DATA,READONLY
EXPORT __Vectors
__Vectors
DCD 0X10001000
DCD Reset_Handler
AREA mydata,DATA,READWRITE
DST DCD 0
AREA mycode,CODE,READONLY
NUM1 DCD 3,2,2,3
NUM2 DCD 1,0,0,1
ENTRY
EXPORT Reset_Handler
Reset_Handler
LDR R0,=NUM1
LDR R1,=NUM2
LDR R2,[R0]
LDR R3,[R1]
LDR R4,=DST
MOV R5,#4

```

```

MOV R6,#0
UP
SUBS R2,R2,R3
ADC R6,#-1
STR R2,[R4],#4
LDR R2,[R0,#4]!
LDR R3,[R1,#4]!
SUBS R5,#1
BNE UP
STR R6,[R4]
ALIGN
STOP
B STOP
END

```

Output:

