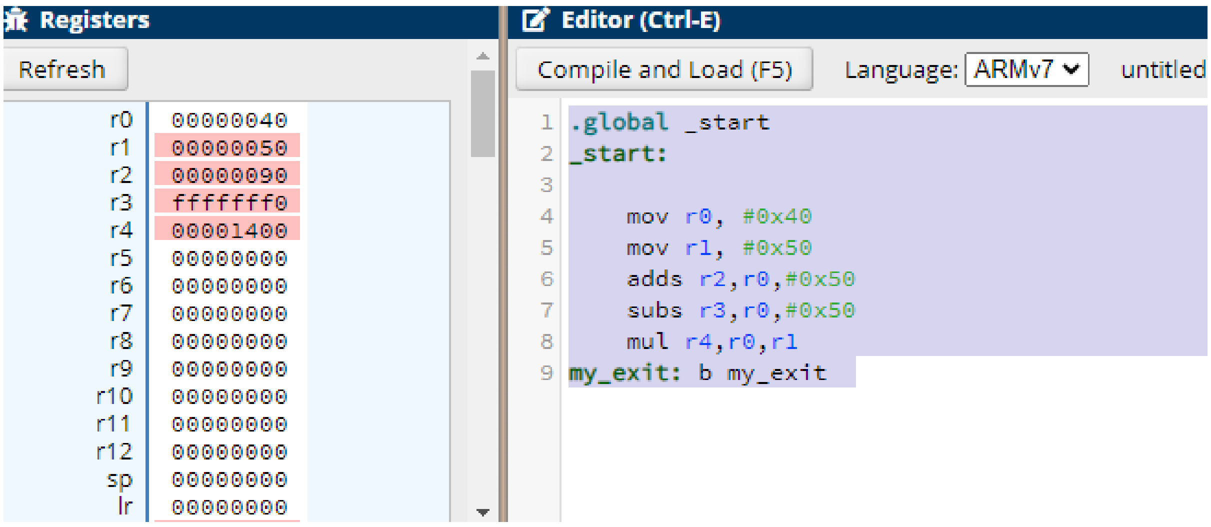
Lab-4

Obj-1: Perform Addition and Subtraction of two 32-bit numbers using data processing addressing mode (with 8-bit immediate data).

Program:

```
.global _start
_start:
        mov r0, #0x40
        mov r1, #0x50
        adds r2,r0,#0x50
        subs r3,r0,#0x50
        mul r4,r0,r1
my_exit: b my_exit
```

RESULT:



Language: ARMv7

```
r0   00000040
r1   00000050
r2   00000090
r3   fffffff0
r4   00001400
r5   00000000
r6   00000000
r7   00000000
r8   00000000
r9   00000000
r10  00000000
r11  00000000
r12  00000000
sp   00000000
lr   00000000
```

```
.global _start
_start:

    mov r0, #0x40
    mov r1, #0x50
    adds r2,r0,#0x50
    subs r3,r0,#0x50
    mul r4,r0,r1
my_exit: b my_exit
```

INPUT / **OUTPUT**

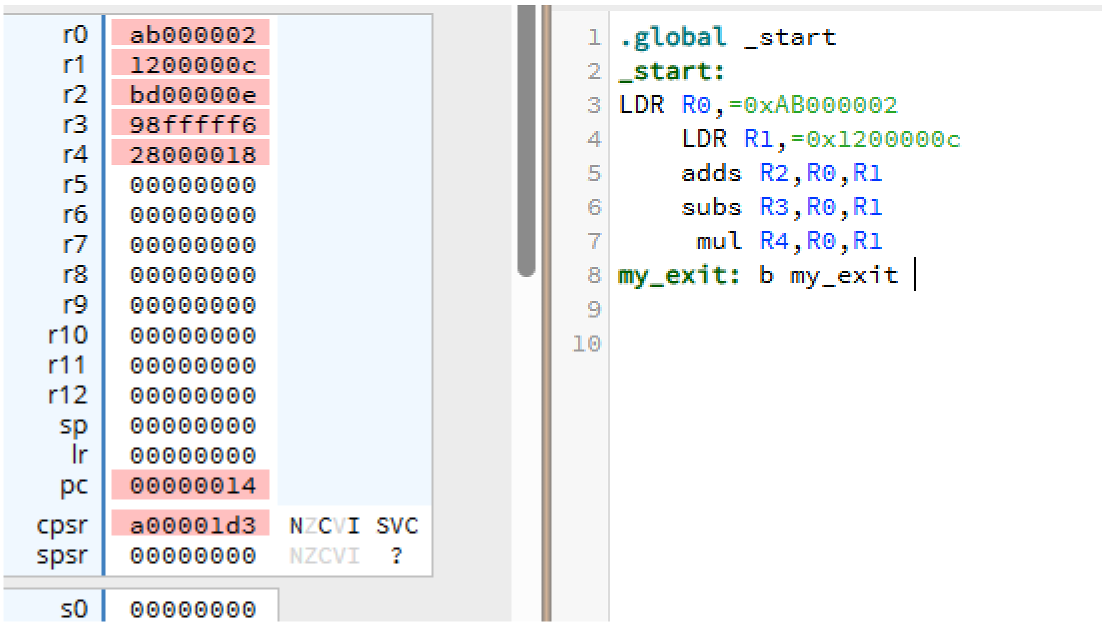
ML	**DATA**	**ML**	**DATA**
-	**0X40 (R0)**	**-**	**0X90 (R2)**
-	**0X50 (R1)**	**-**	**0Xfffffff0 (R3)**
		-	**0X1400 (R4)**

OR

Objective-1: (with 32-bit immediate data).

```
.global _start
_start:
        LDR R0,=0xAB000002
        LDR R1,=0x1200000c
        adds R2,R0,R1
        subs R3,R0,R1
        mul R4,R0,R1
my_exit: b my_exit
```

RESULT:



```
  r0   ab000002
  r1   1200000c
  r2   bd00000e
  r3   98fffff6
  r4   28000018
  r5   00000000
  r6   00000000
  r7   00000000
  r8   00000000
  r9   00000000
 r10   00000000
 r11   00000000
 r12   00000000
  sp   00000000
  lr   00000000
  pc   00000014
cpsr   a00001d3   NZCVI SVC
spsr   00000000   NZCVI  ?
  s0   00000000
```

```
.global _start
_start:
LDR R0,=0xAB000002
    LDR R1,=0x1200000c
    adds R2,R0,R1
    subs R3,R0,R1
     mul R4,R0,R1
my_exit: b my_exit
```

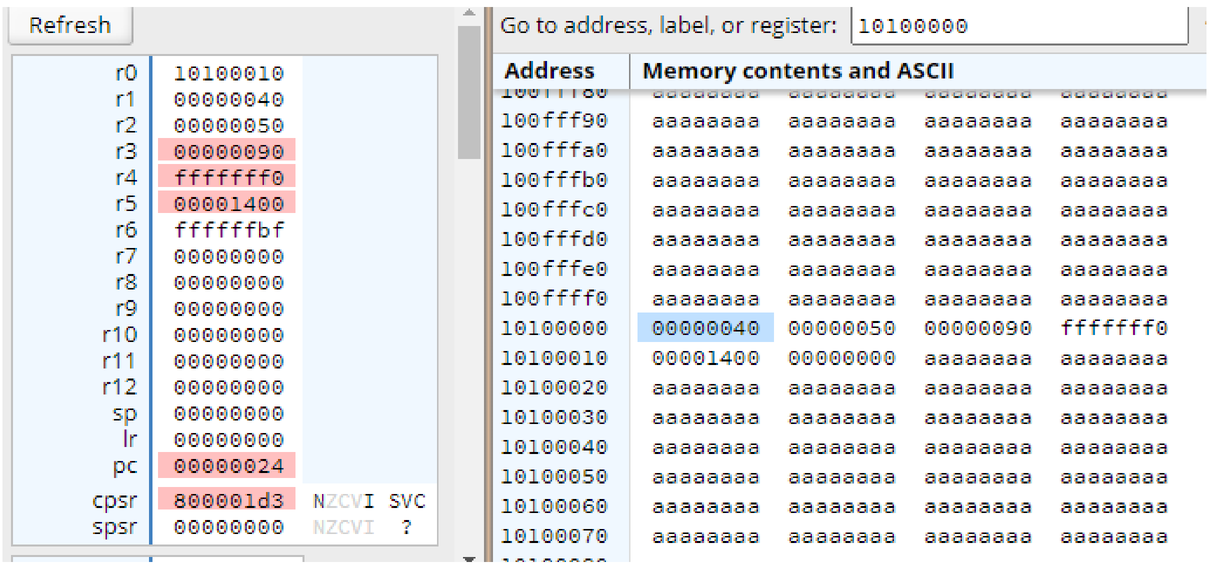
INPUT		**OUTPUT**	
ML	**DATA**	**ML**	**DATA**
-	**0XAB000002**	**-**	**0Xbd00000e**
-	**0X1200000C**	**-**	**0X98fffff6**
		-	**0X28000018**

Objective 2: Perform Addition, Subtraction, and Multiplication of two 32-bit numbers using load/store addressing mode.

Program:

```
.global _start
_start:
        LDR R0,=0X10100000
        LDR R1,[R0],#4
        LDR R2,[R0],#4
        ADDS R3,R1,R2
        STR R3,[R0],#4
        SUBS R4,R1,R2
        STR R4,[R0],#4
        MUL R5,R1,R2
        STR R5,[R0]
my_exit: b my_exit
```

RESULT


Refresh
Go to address, label, or register: 10100000
Address
Memory contents and ASCII

```
  r0  10100010
  r1  00000040
  r2  00000050
  r3  00000090
  r4  fffffff0
  r5  00001400
  r6  ffffffbf
  r7  00000000
  r8  00000000
  r9  00000000
 r10  00000000
 r11  00000000
 r12  00000000
  sp  00000000
  lr  00000000
  pc  00000024
cpsr  800001d3  NZCVI SVC
spsr  00000000  NZCVI  ?
```

```
100fff90  aaaaaaaa  aaaaaaaa  aaaaaaaa  aaaaaaaa
100fffa0  aaaaaaaa  aaaaaaaa  aaaaaaaa  aaaaaaaa
100fffb0  aaaaaaaa  aaaaaaaa  aaaaaaaa  aaaaaaaa
100fffc0  aaaaaaaa  aaaaaaaa  aaaaaaaa  aaaaaaaa
100fffd0  aaaaaaaa  aaaaaaaa  aaaaaaaa  aaaaaaaa
100fffe0  aaaaaaaa  aaaaaaaa  aaaaaaaa  aaaaaaaa
100ffff0  aaaaaaaa  aaaaaaaa  aaaaaaaa  aaaaaaaa
10100000  00000040  00000050  00000090  fffffff0
10100010  00001400  00000000  aaaaaaaa  aaaaaaaa
10100020  aaaaaaaa  aaaaaaaa  aaaaaaaa  aaaaaaaa
10100030  aaaaaaaa  aaaaaaaa  aaaaaaaa  aaaaaaaa
10100040  aaaaaaaa  aaaaaaaa  aaaaaaaa  aaaaaaaa
10100050  aaaaaaaa  aaaaaaaa  aaaaaaaa  aaaaaaaa
10100060  aaaaaaaa  aaaaaaaa  aaaaaaaa  aaaaaaaa
10100070  aaaaaaaa  aaaaaaaa  aaaaaaaa  aaaaaaaa
```

INPUT

ML	DATA
0X10100000	0X40
0X10100004	0X50

OUTPUT

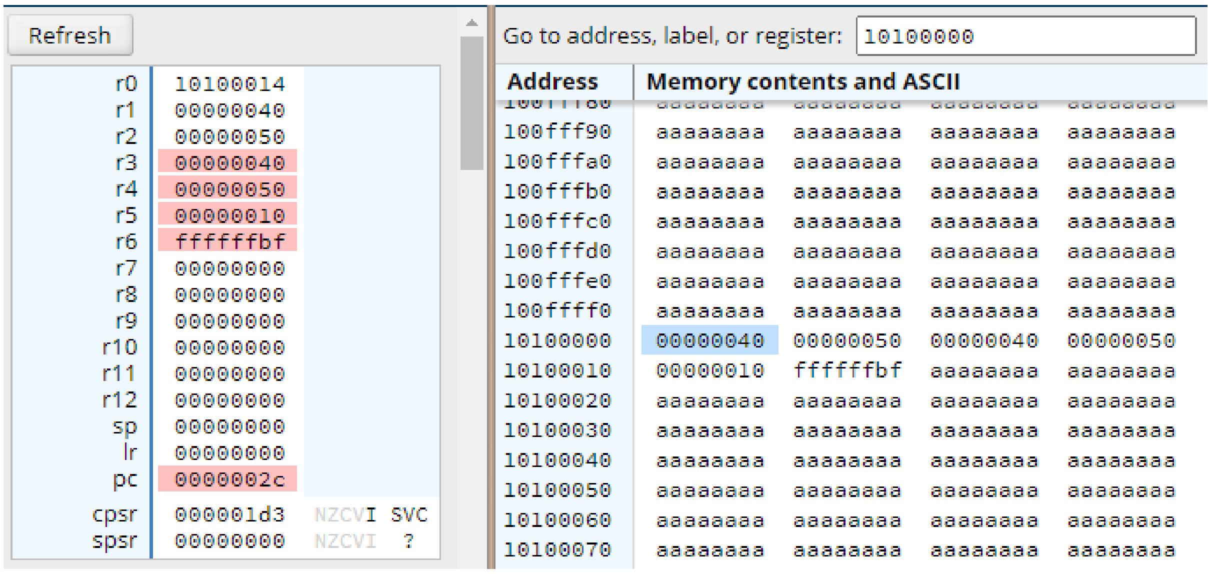
ML	DATA
0X10100008	0X90
0X1010000C	0Xfffffff0
0X10100010	0X10

Objective-3: Perform the logical operations (AND, OR, XOR, and NOT) on two 32-bit numbers using load/store addressing mode

Program

```
.global _start
_start:
        LDR R0,=0X10100000
        LDR R1,[R0],#4
        LDR R2,[R0],#4
        ANDS R3,R2,R1
        STR R3,[R0],#4
        ORR R4,R2,R1
        STR R4,[R0],#4
        EOR R5,R2,R1
        STR R5,[R0],#4
        MVN R6, R1
        STR R6,[R0]
my_exit: b my_exit
```

RESULT:



```
Refresh

  r0   10100014
  r1   00000040
  r2   00000050
  r3   00000040
  r4   00000050
  r5   00000010
  r6   ffffffbf
  r7   00000000
  r8   00000000
  r9   00000000
 r10   00000000
 r11   00000000
 r12   00000000
  sp   00000000
  lr   00000000
  pc   0000002c

cpsr   000001d3   NZCVI SVC
spsr   00000000   NZCVI  ?
```

```
Go to address, label, or register: 10100000

Address    Memory contents and ASCII
100fff90   aaaaaaaa  aaaaaaaa  aaaaaaaa  aaaaaaaa
100fffa0   aaaaaaaa  aaaaaaaa  aaaaaaaa  aaaaaaaa
100fffb0   aaaaaaaa  aaaaaaaa  aaaaaaaa  aaaaaaaa
100fffc0   aaaaaaaa  aaaaaaaa  aaaaaaaa  aaaaaaaa
100fffd0   aaaaaaaa  aaaaaaaa  aaaaaaaa  aaaaaaaa
100fffe0   aaaaaaaa  aaaaaaaa  aaaaaaaa  aaaaaaaa
100ffff0   aaaaaaaa  aaaaaaaa  aaaaaaaa  aaaaaaaa
10100000   00000040  00000050  00000040  00000050
10100010   00000010  ffffffbf  aaaaaaaa  aaaaaaaa
10100020   aaaaaaaa  aaaaaaaa  aaaaaaaa  aaaaaaaa
10100030   aaaaaaaa  aaaaaaaa  aaaaaaaa  aaaaaaaa
10100040   aaaaaaaa  aaaaaaaa  aaaaaaaa  aaaaaaaa
10100050   aaaaaaaa  aaaaaaaa  aaaaaaaa  aaaaaaaa
10100060   aaaaaaaa  aaaaaaaa  aaaaaaaa  aaaaaaaa
10100070   aaaaaaaa  aaaaaaaa  aaaaaaaa  aaaaaaaa
```

INPUT		**OUTPUT**	
ML	**DATA**	**ML**	**DATA**
0X10100000	**0X40**	**0X10100008**	**0X40**
0X10100004	**0X50**	**0X1010000C**	**0X50**
		0X10100010	**0X10**
		0X10100014	**0Xffffffbf**