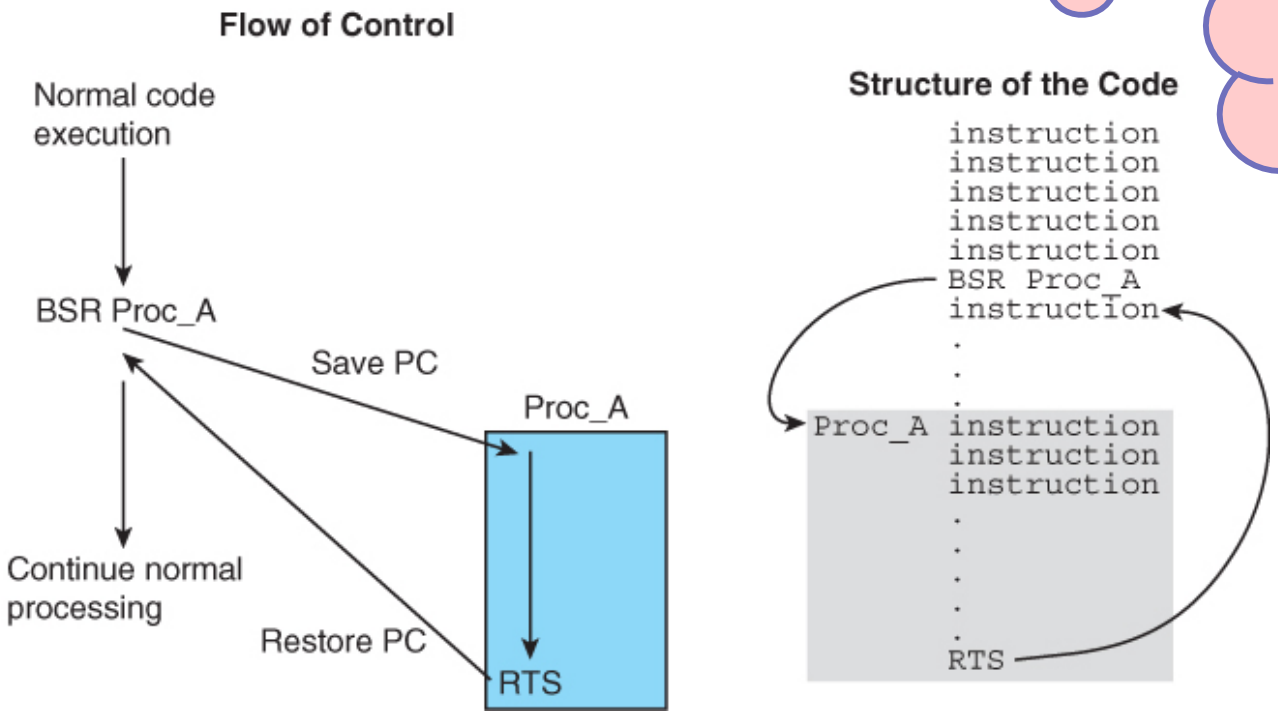
Tutorial 07: ARM Subroutine Call and Return

Computer Science Department
CS2208b: Introduction to Computer Organization and Architecture
Winter 2019
Instructor: Mahmoud R. El-Sakka
Office: MC-419
Email: elsakka@csd.uwo.ca
Phone: 519-661-2111 x86996

Subroutine call and Return

FIGURE 3.40 The subroutine call and return



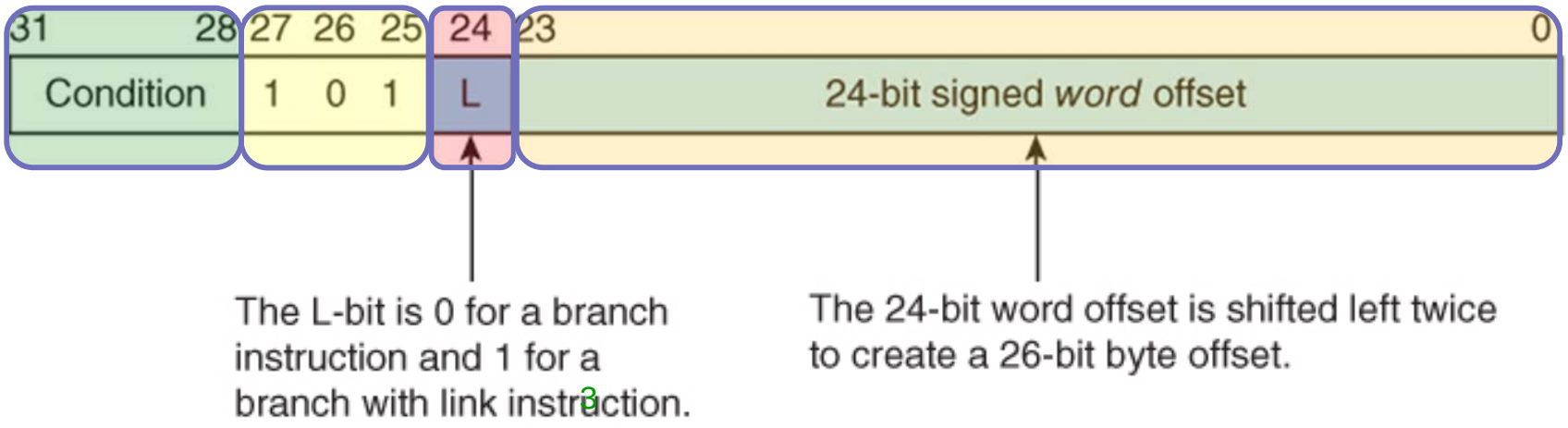
This is not used by ARM processors

Subroutine call and Return

- ARM's *branch with link* instruction, **BL**,
  - automatically saves the return address in register **r14**.
- The branch instruction (Figure 3.41) has an 8-bit op-code with a 24-bit ***signed*** program counter relative offset (*word address offset*).
- The 24-bit offset is
  - shifted left twice to convert the word-offset address to a byte address,
    sign-extended to 32 bits,
  - added it to the program counter (***the result is PC ± 32 MBytes***).

Do not forget the pipelining effect

FIGURE 3.41 Encoding ARM's branch and branch-with-link instructions



Example 1: Subroutine call and Return

Calling a leaf subroutine using **BL** instruction

```
;-------------------------------------------------
Main      MOV r1,#0x10        ;just a dummy operation
          BL  Function1       ;jump to Function1
          MOV r3,#0x30        ;just a dummy operation
          BL  Function2       ;jump to Function2
          MOV r5,#0x50        ;just a dummy operation
loop      B   loop
;-------------------------------------------------
```

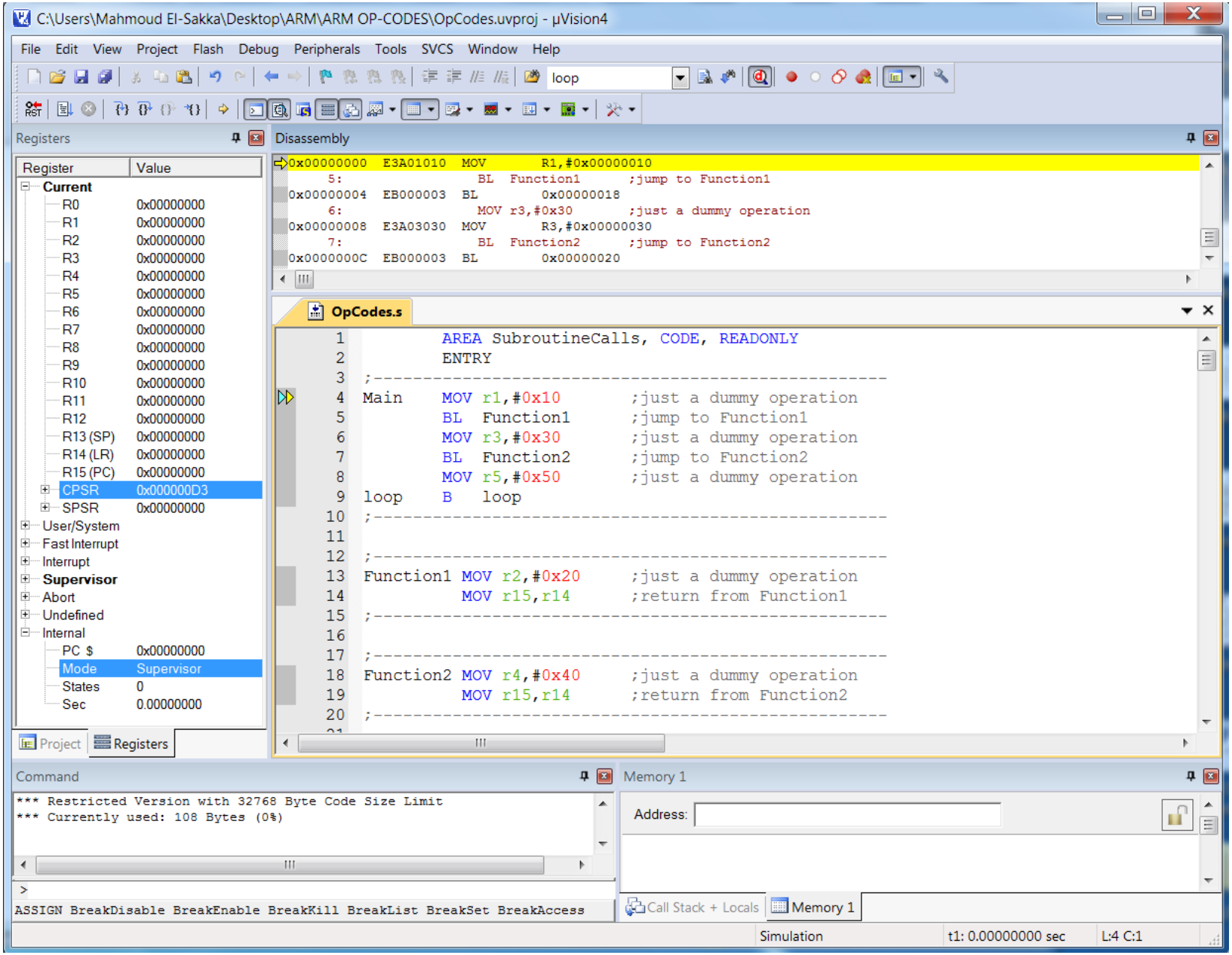
Leaf subroutine

```
;-------------------------------------------------
Function1 MOV r2,#0x20        ;just a dummy operation
          MOV r15,r14         ;return from Function1
;-------------------------------------------------
```

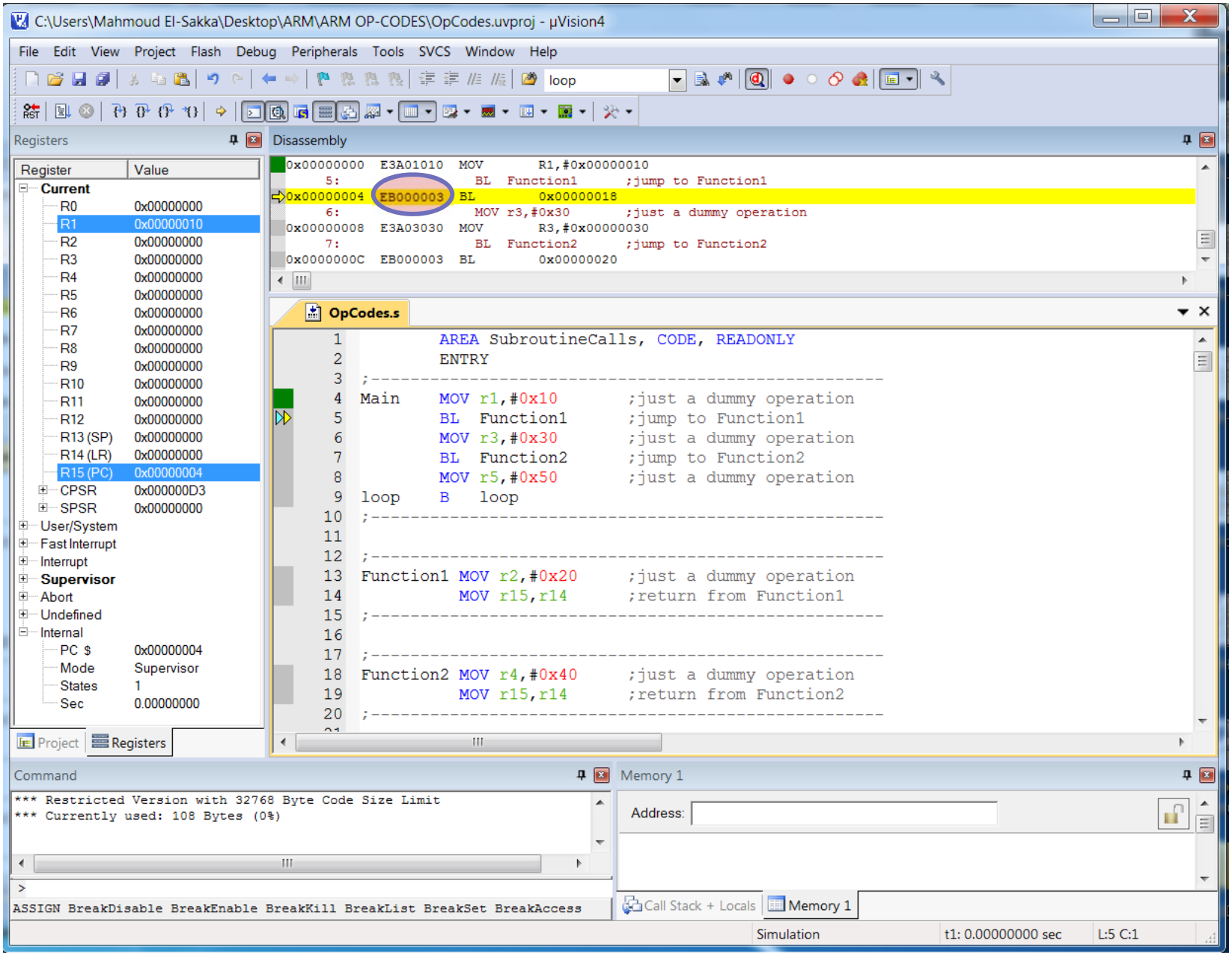
Leaf subroutine

```
;-------------------------------------------------
Function2 MOV r4,#0x40        ;just a dummy operation
          MOV r15,r14         ;return from Function2
;-------------------------------------------------
```

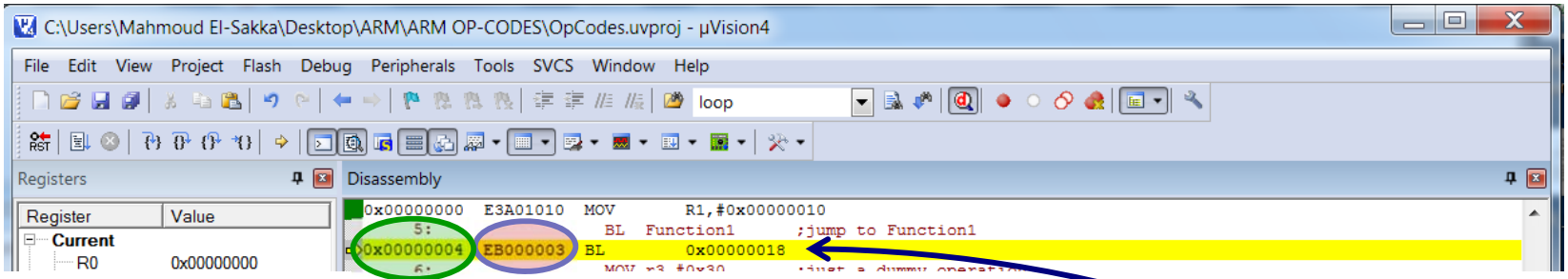
Example 1: Subroutine call and Return



Example 1: Subroutine call and Return

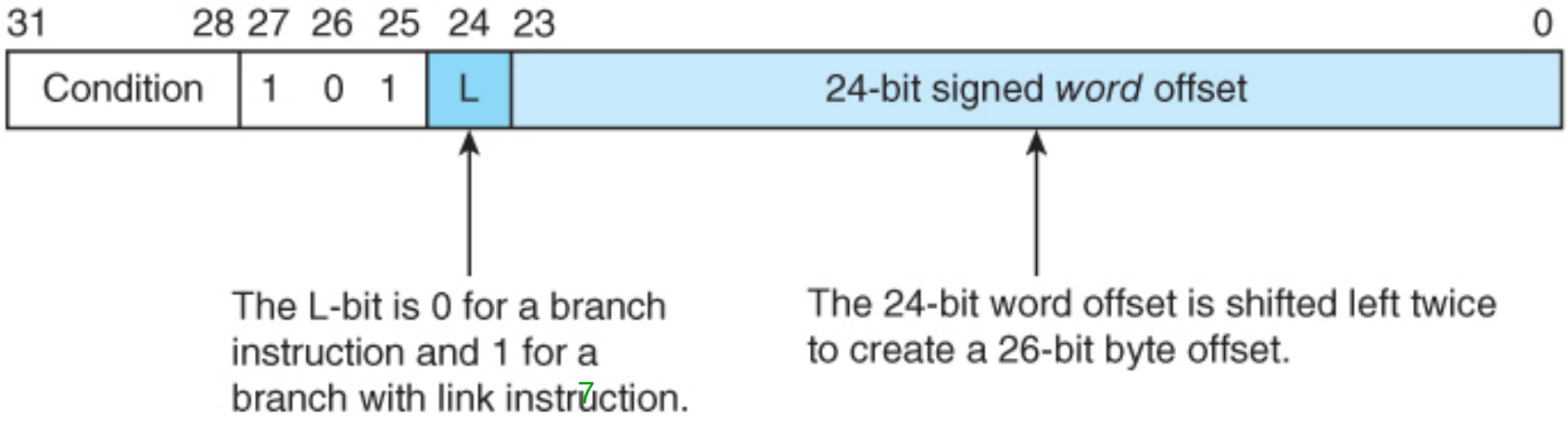


Example 1: Subroutine call and Return

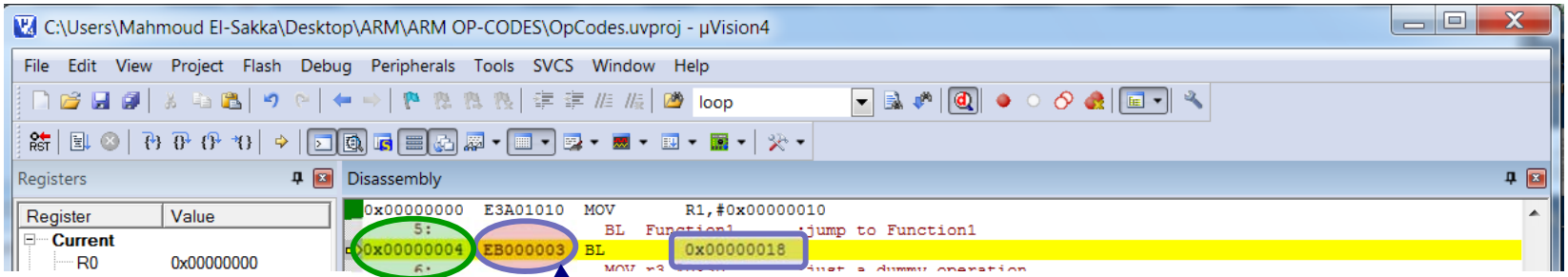


- 24-bit signed *word* offset = `0x000003`
- 24-bit signed ***byte*** offset = `0x000003 × 0x4 = 0x00000C`
- Effective address = 24-bit signed ***byte*** offset + *PC value* + *pipeline effect*
  = `0x00000C + 0x00000004 + 0x8 = 0x00000018`

FIGURE 3.41 Encoding ARM's branch and branch-with-link instructions



Example 1: Subroutine call and Return

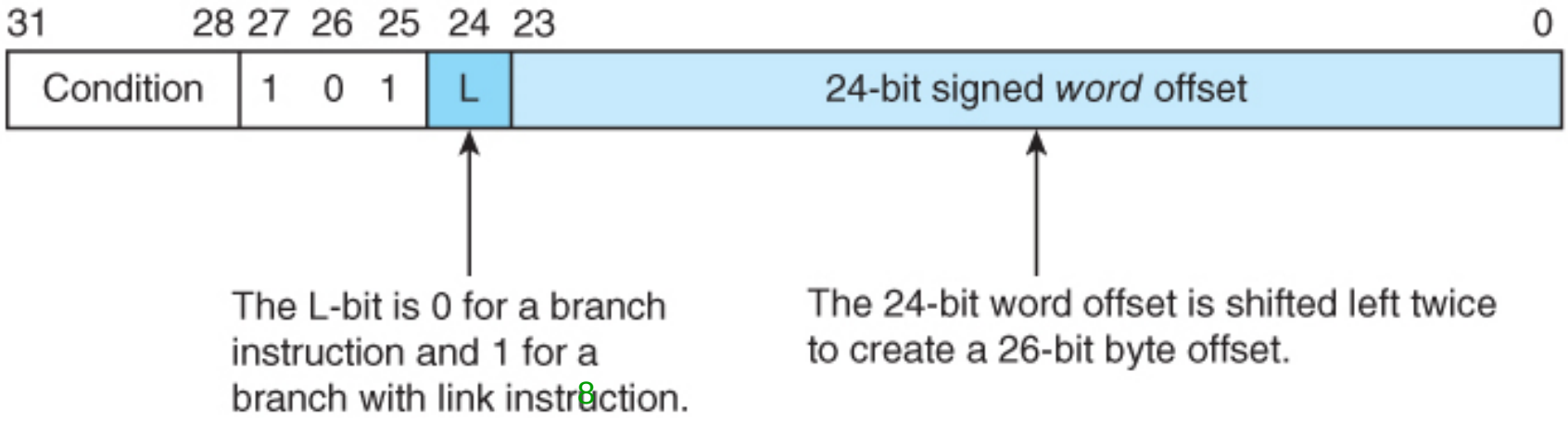


- *PC value* = **0x00000004** Effective address = **0x00000018**
- 24-bit signed ***byte*** offset =

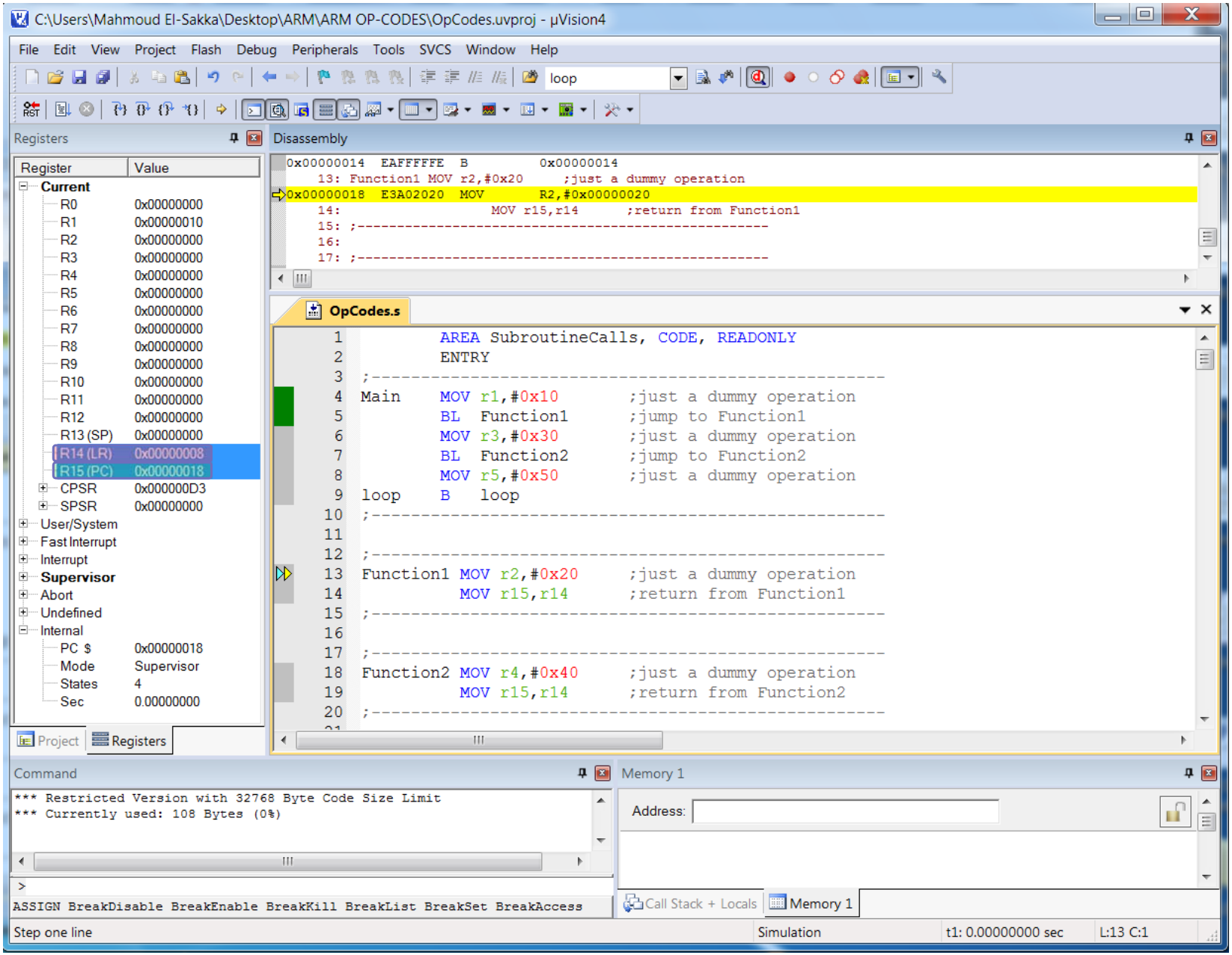
  Effective address − *PC value* − *pipeline effect* = **0x18** − **0x4** − **0x8** = **0xC**
- 24-bit signed *word* offset = 24-bit signed ***byte*** offset ÷ 4

  = **0x0000000C** ÷ 4 = 0x000003

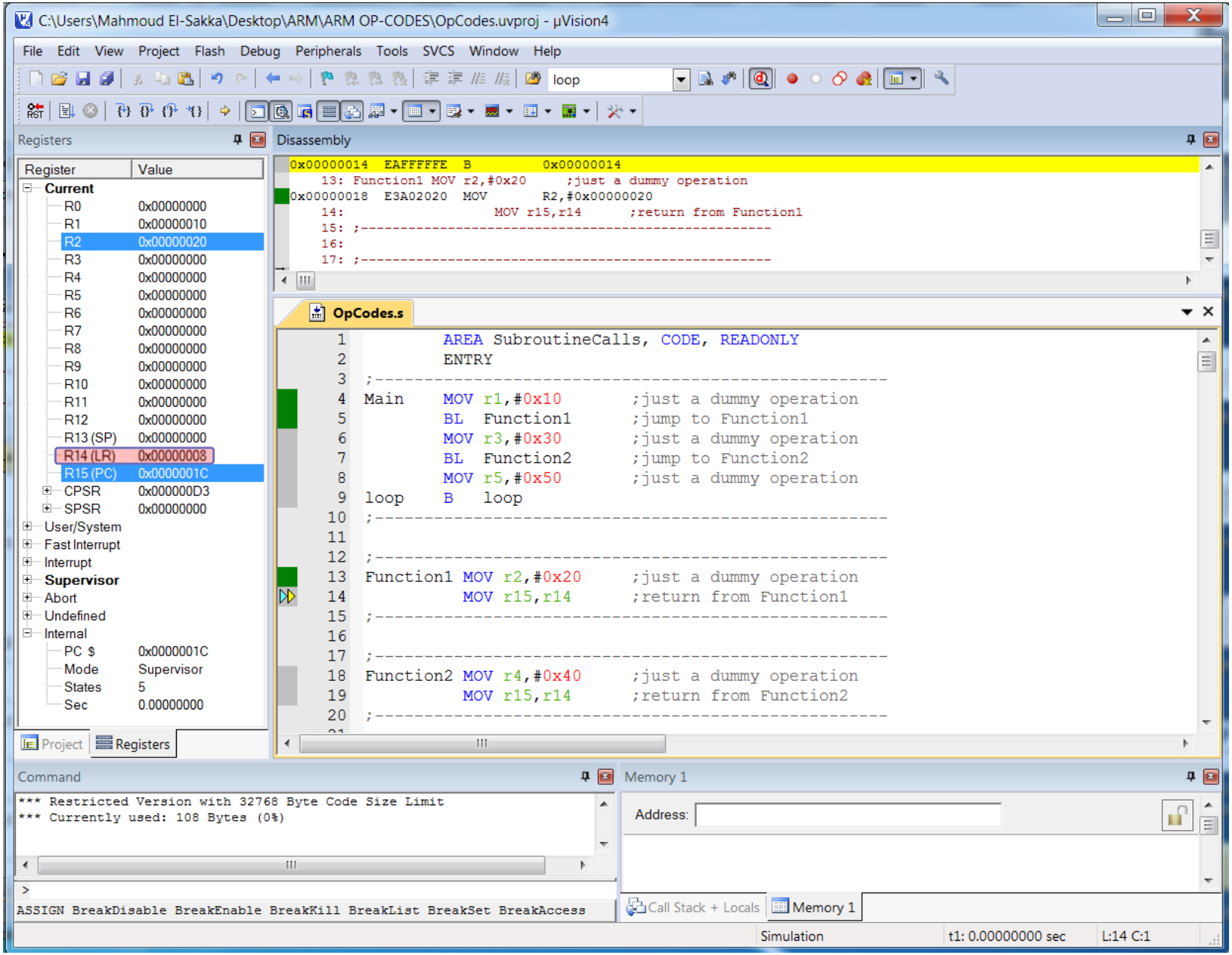
FIGURE 3.41 Encoding ARM's branch and branch-with-link instructions



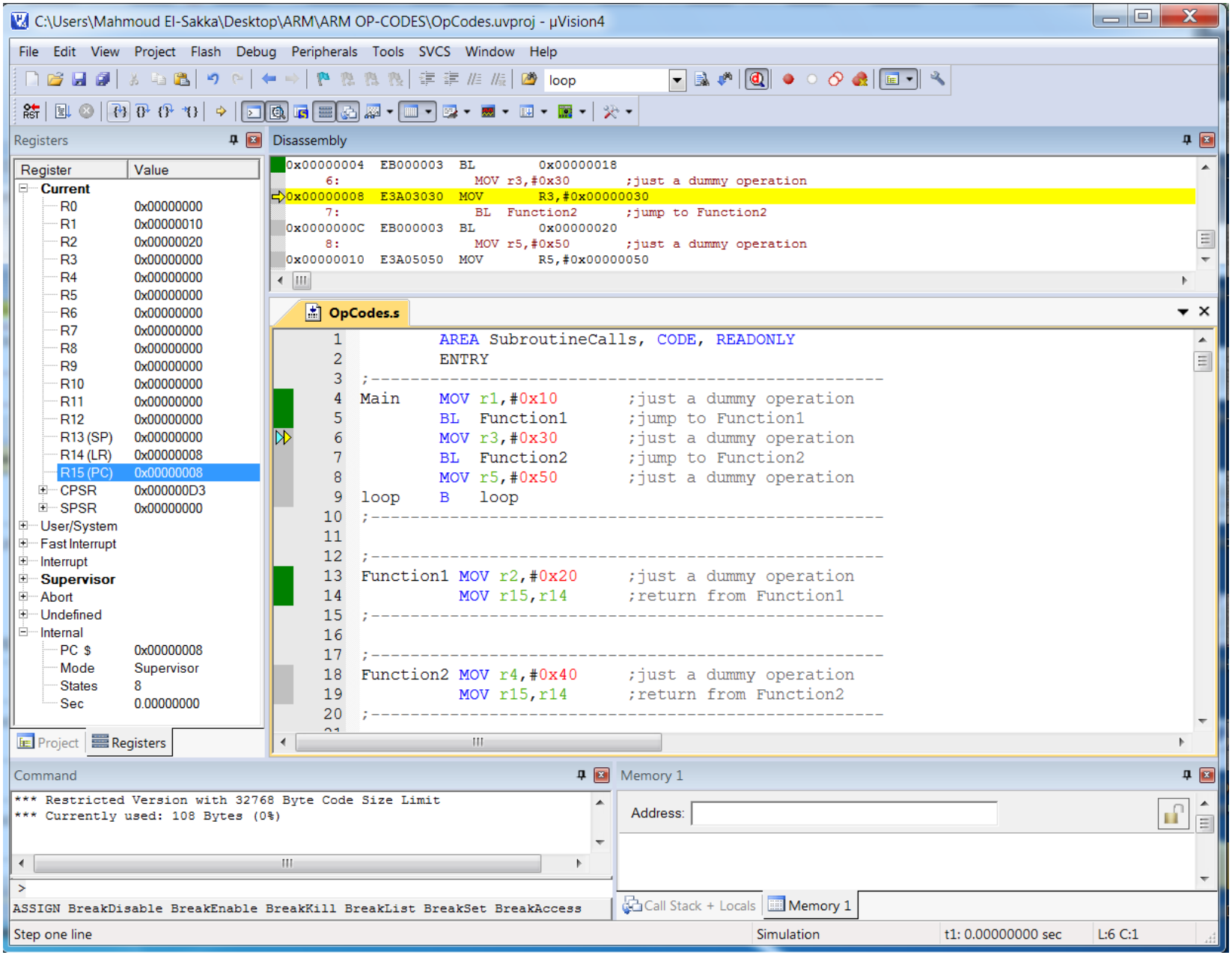
Example 1: Subroutine call and Return



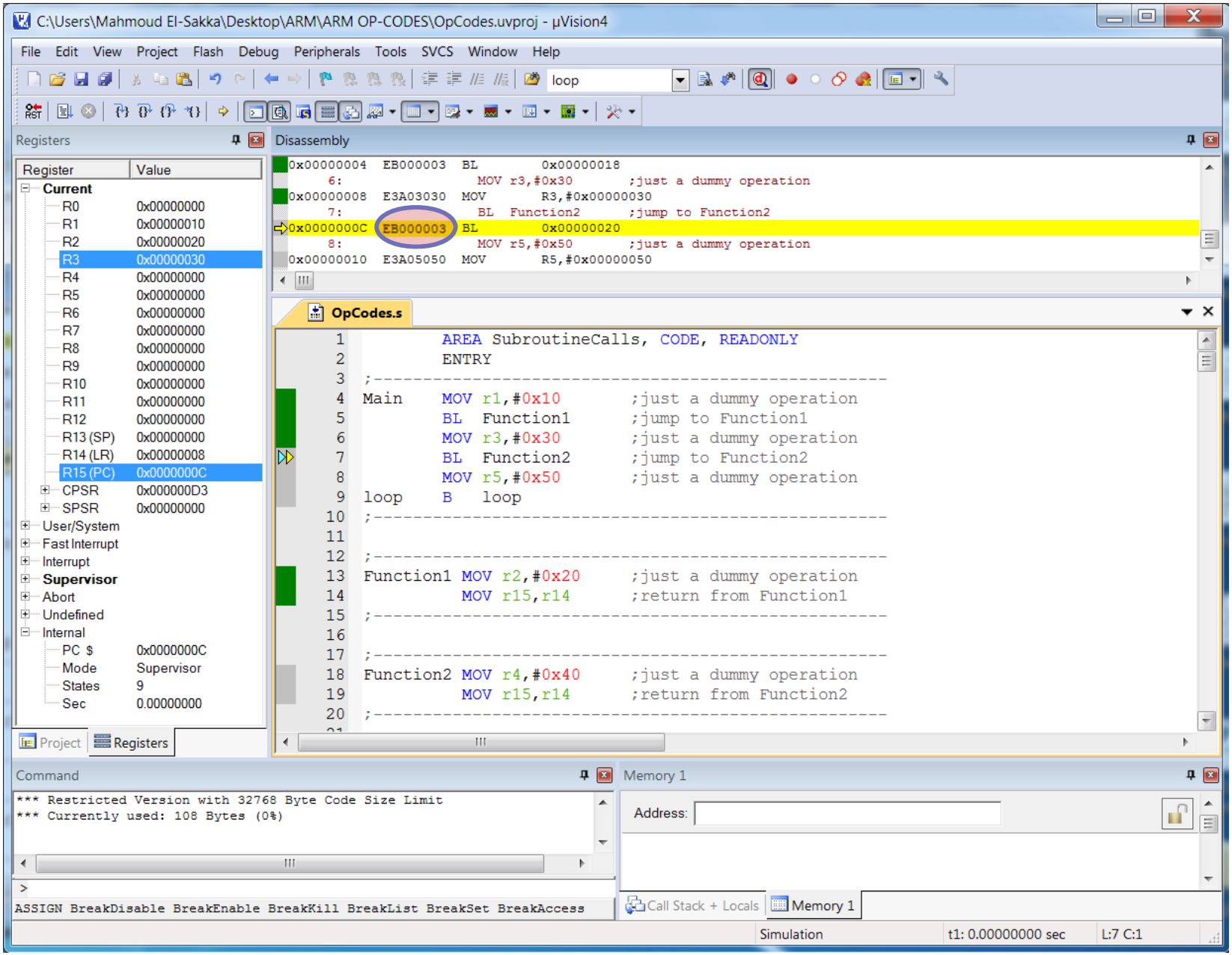
Example 1: Subroutine call and Return



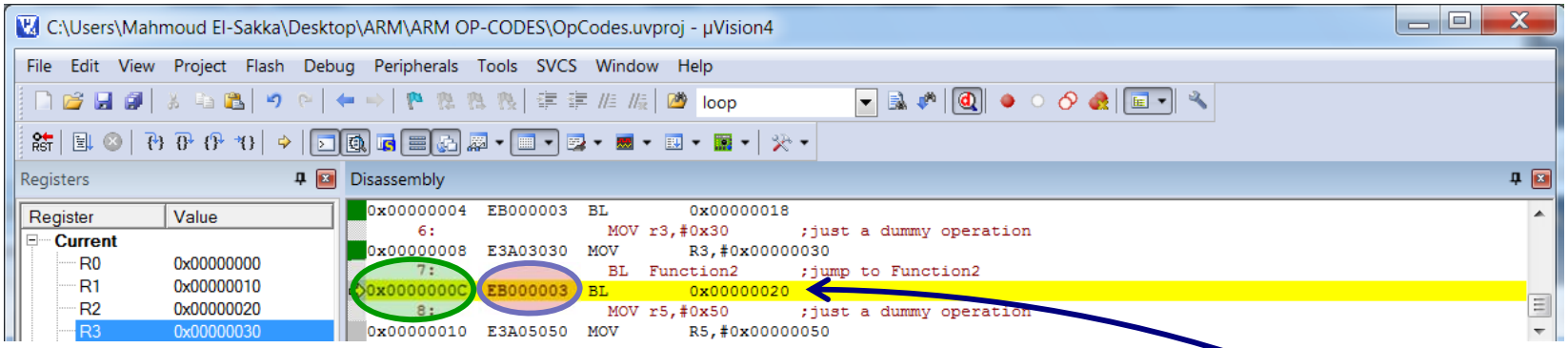
Example 1: Subroutine call and Return



Example 1: Subroutine call and Return

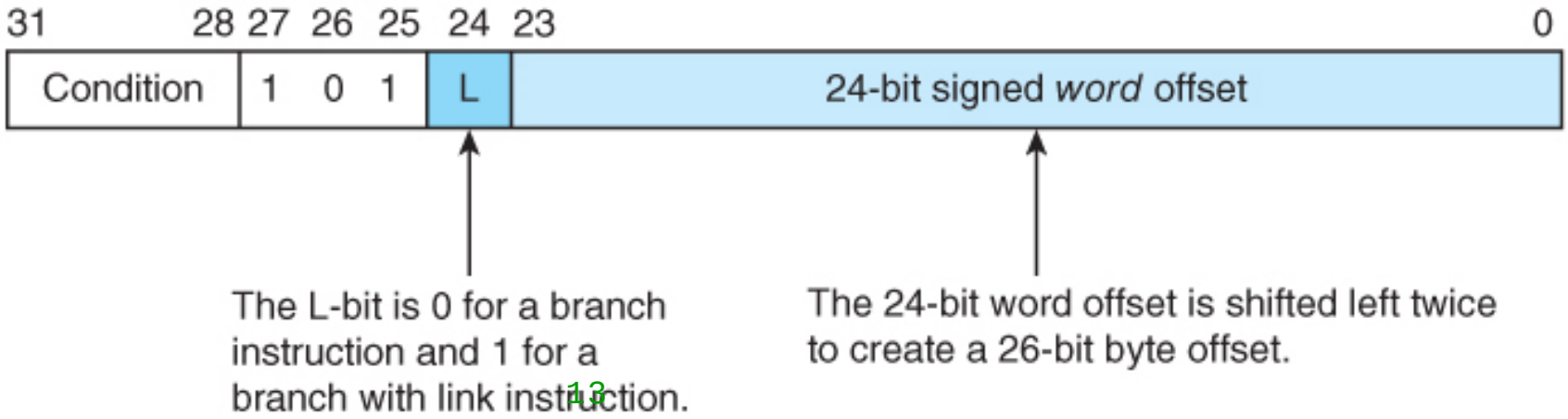


Example 1: Subroutine call and Return

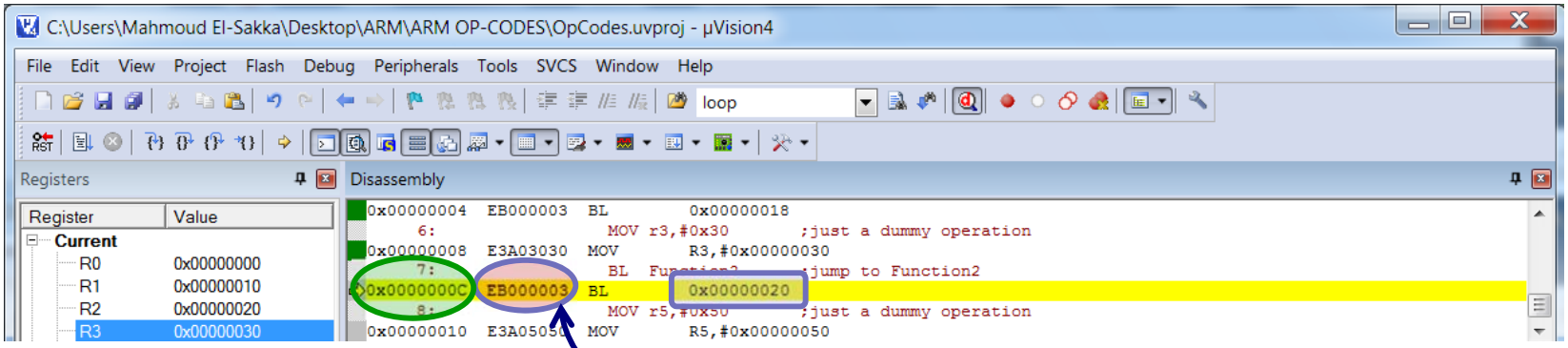


- 24-bit signed *word* offset = `0x000003`
- 24-bit signed ***byte*** offset = `0x000003` × `0x4` = **`0x00000C`**
- Effective address = 24-bit signed ***byte*** offset + *PC value* + *pipeline effect*
  = **`0x00000C`** + **`0x0000000C`** + **`0x8`** = `0x00000020`

FIGURE 3.41 Encoding ARM's branch and branch-with-link instructions



Example 1: Subroutine call and Return

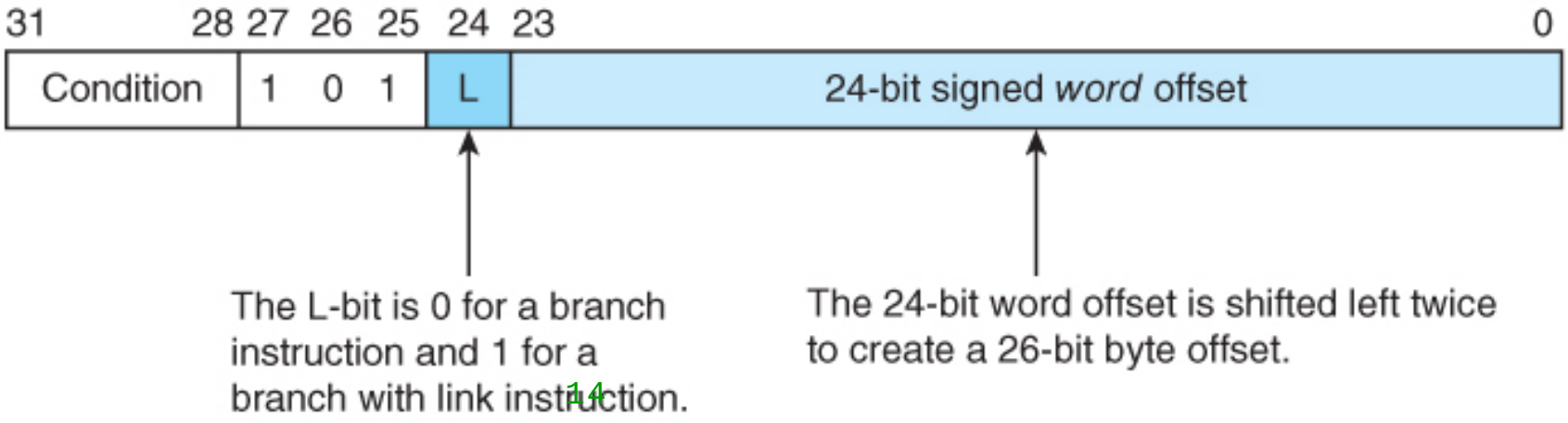


- *PC value* = **0x0000000C** Effective address = **0x00000020**
- 24-bit signed ***byte*** offset =

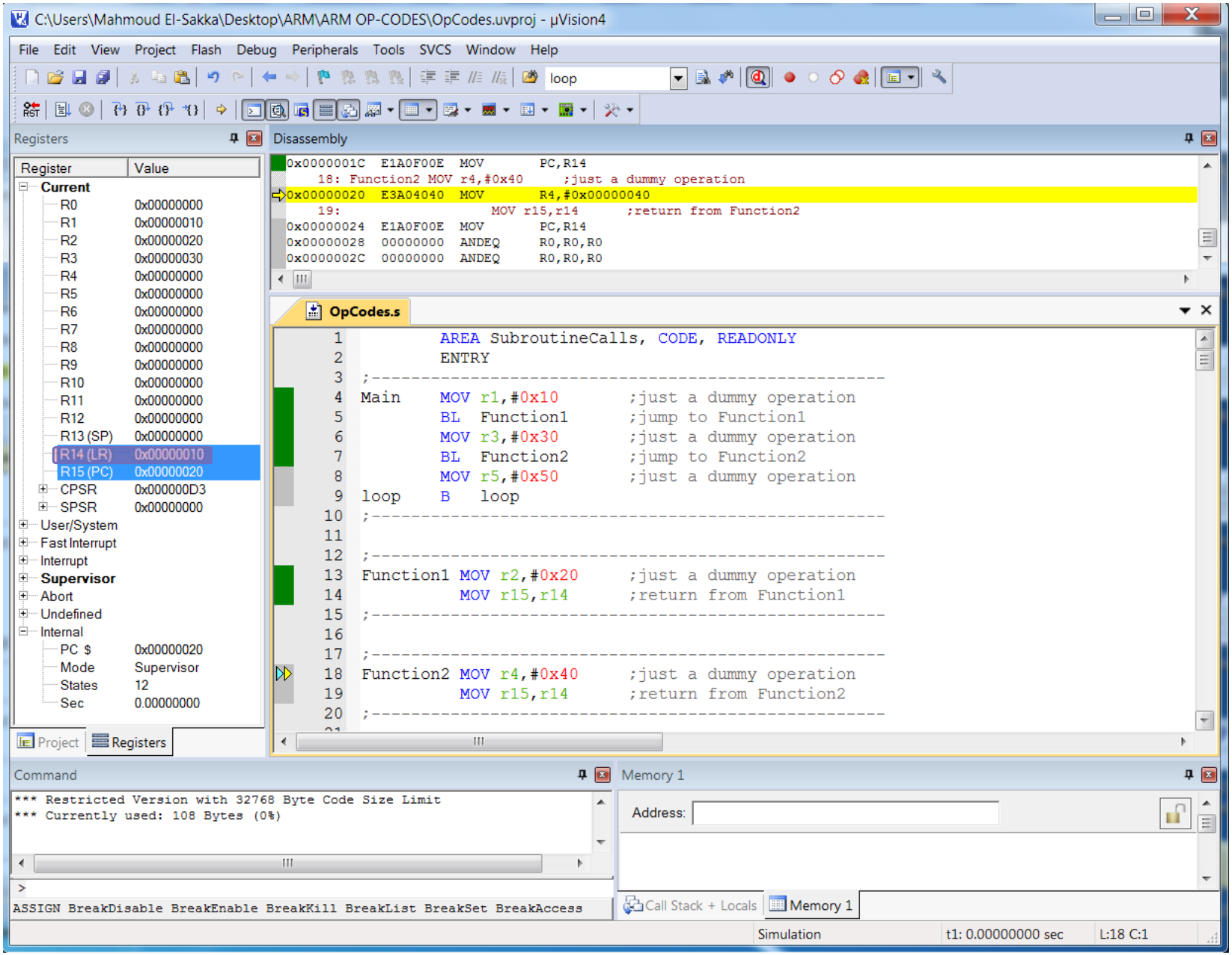
  Effective address − *PC value* − *pipeline effect* = **0x20** − **0xC** − **0x8** = **0xC**
- 24-bit signed *word* offset = 24-bit signed ***byte*** offset ÷ 4

  = **0x0000000C** ÷ 4 = 0x000003

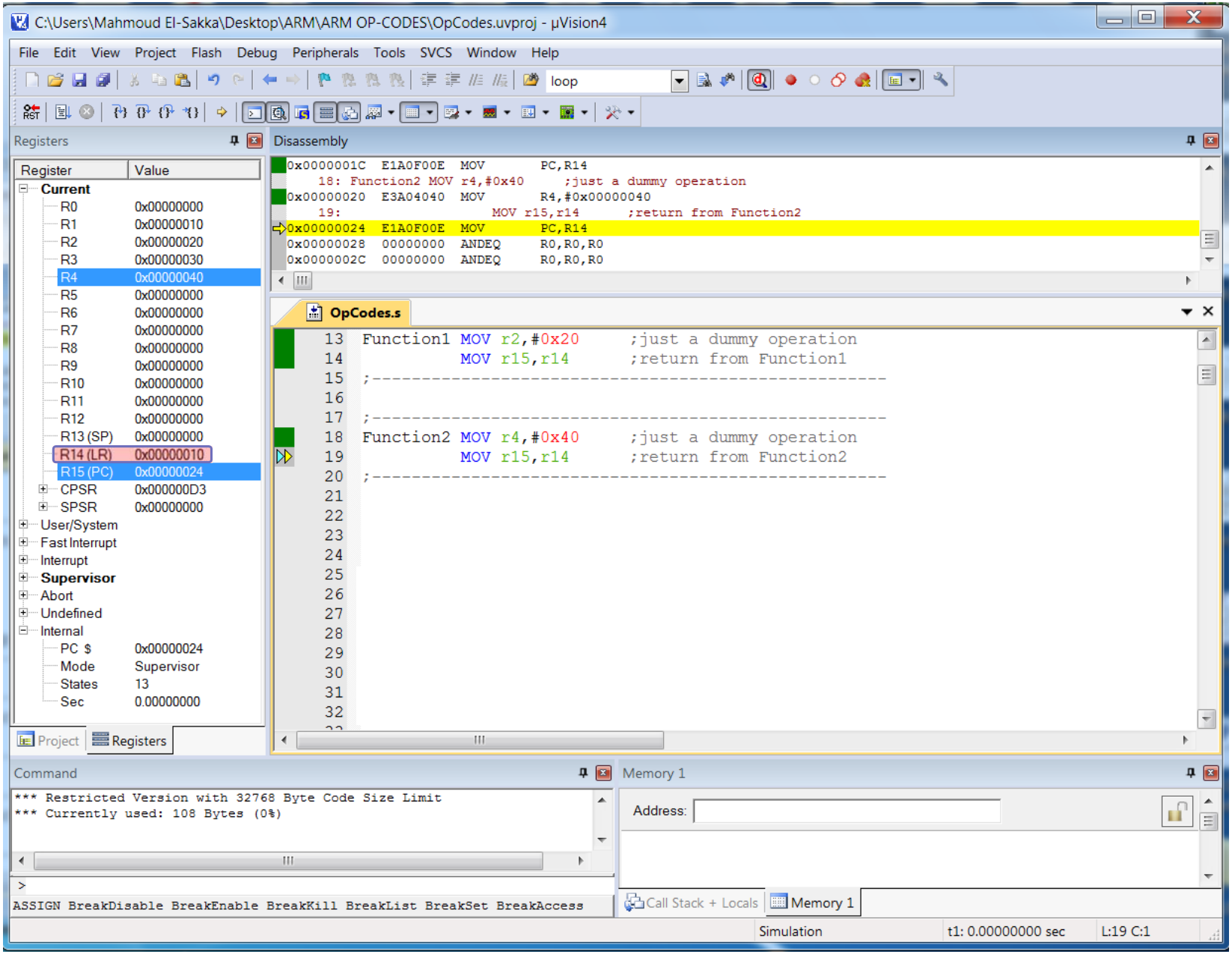
FIGURE 3.41 Encoding ARM's branch and branch-with-link instructions



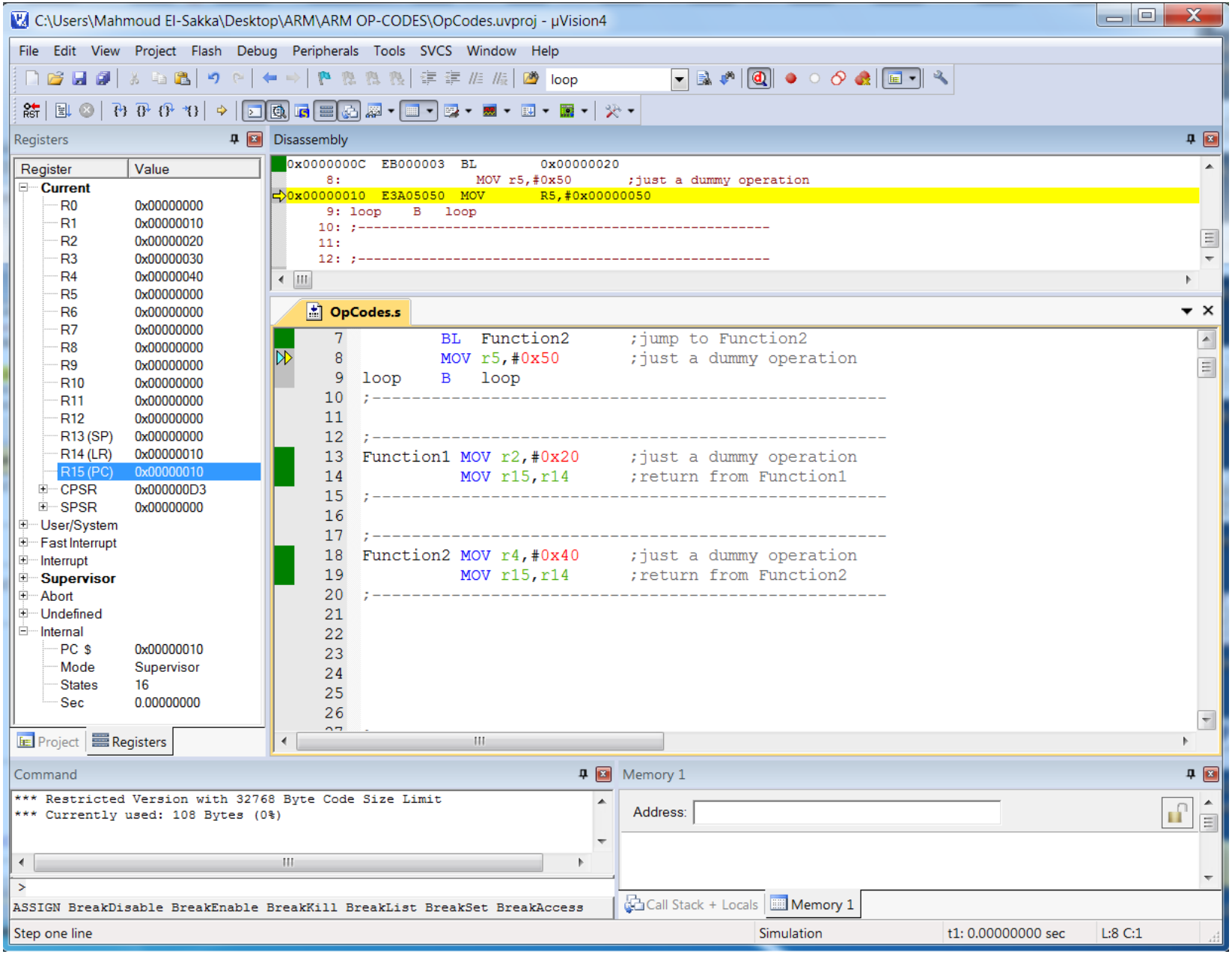
Example 1: Subroutine call and Return



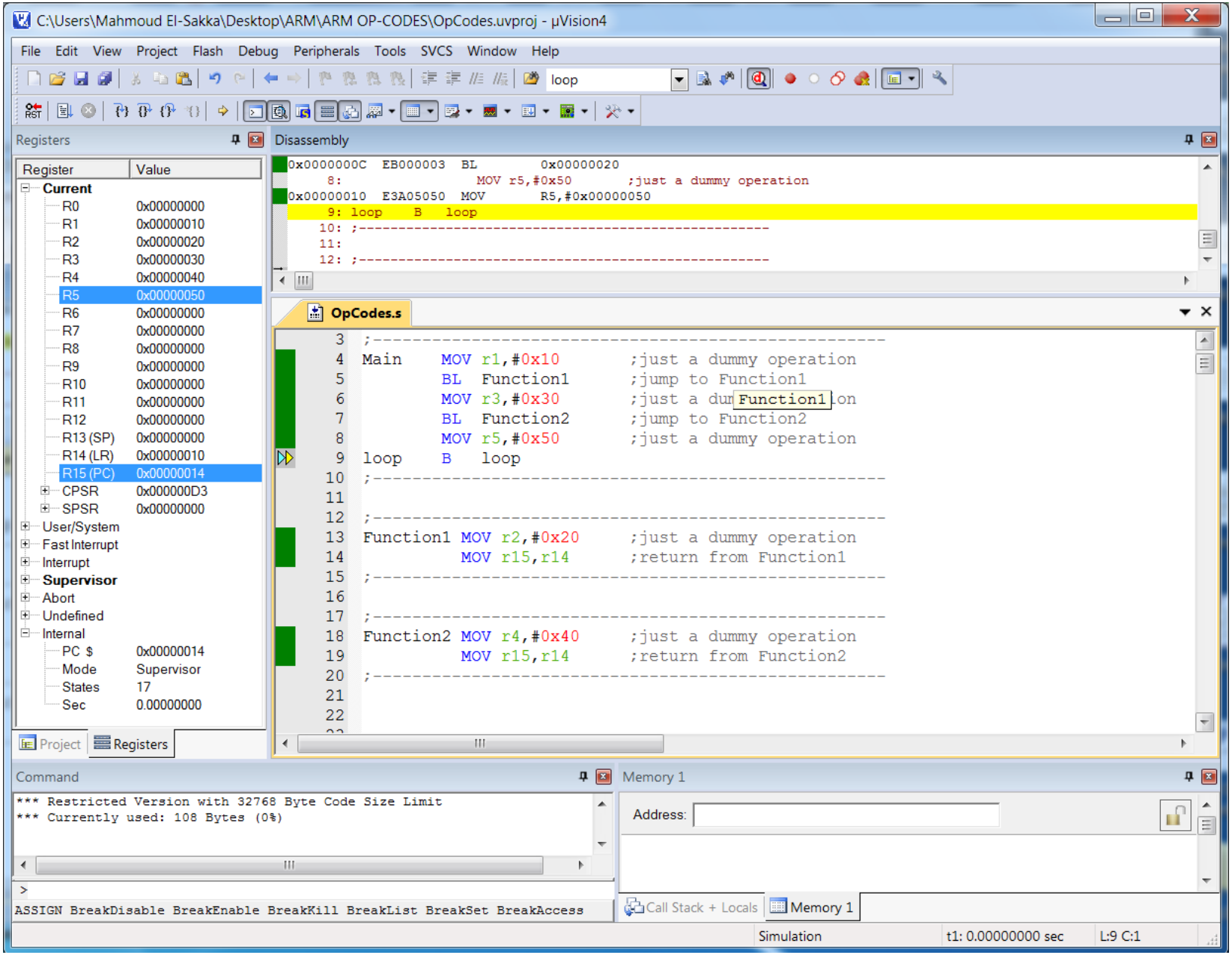
Example 1: Subroutine call and Return



Example 1: Subroutine call and Return



Example 1: Subroutine call and Return



Example 2: Subroutine call and Return

Calling a none leaf subroutine using **BL** instruction

```
;-------------------------------------------------------
Main     MOV r1,#0x10          ;just a dummy operation
         BL  Function1         ;jump to Function1
         MOV r3,#0x30          ;just a dummy operation
         BL  Function2         ;jump to Function2
         MOV r5,#0x50          ;just a dummy operation
loop     B   loop
;-------------------------------------------------------
```

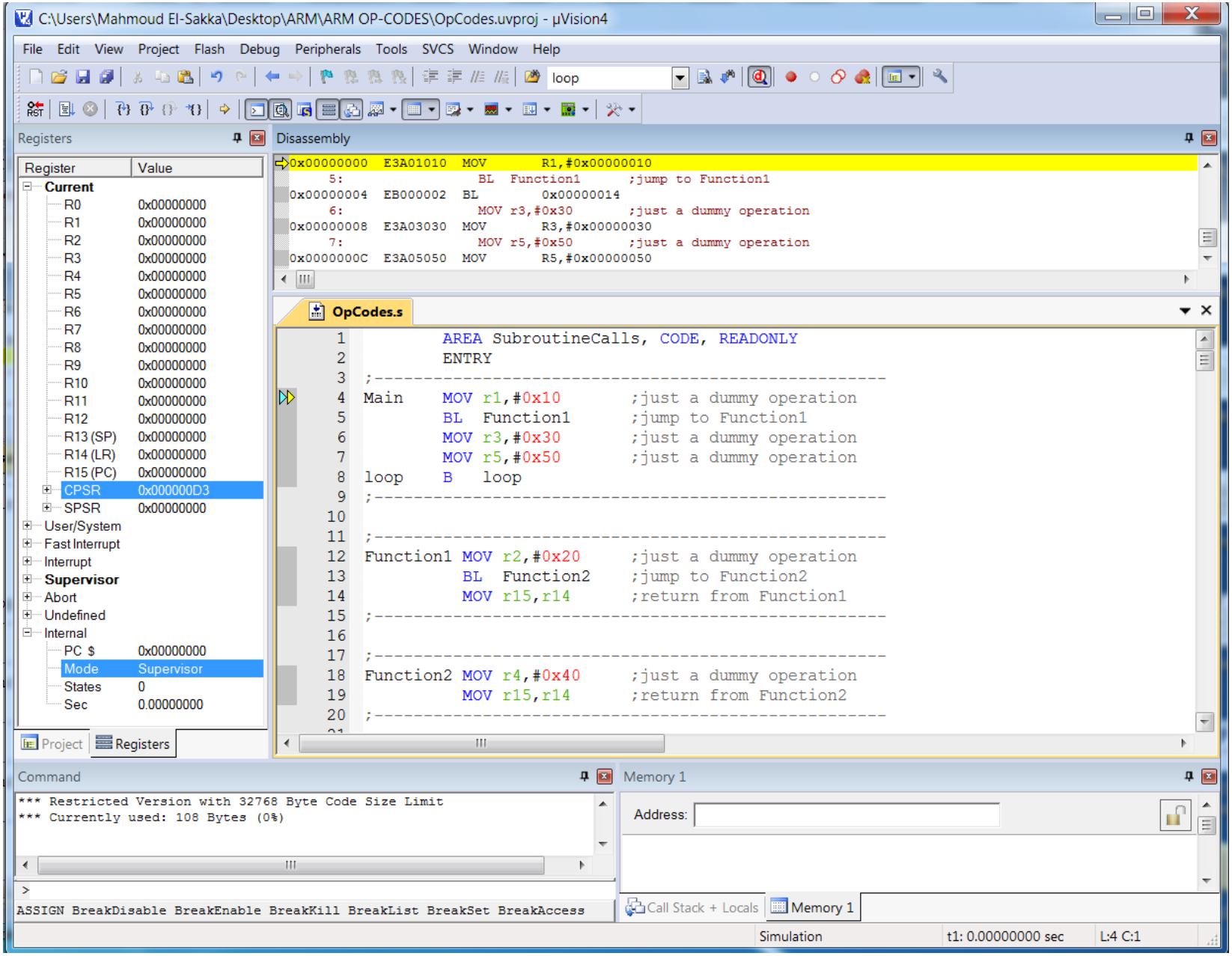
None leaf subroutine

```
;-------------------------------------------------------
Function1 MOV r2,#0x20         ;just a dummy operation
          BL  Function2        ;jump to Function2
          MOV r15,r14          ;return from Function1
;-------------------------------------------------------
```

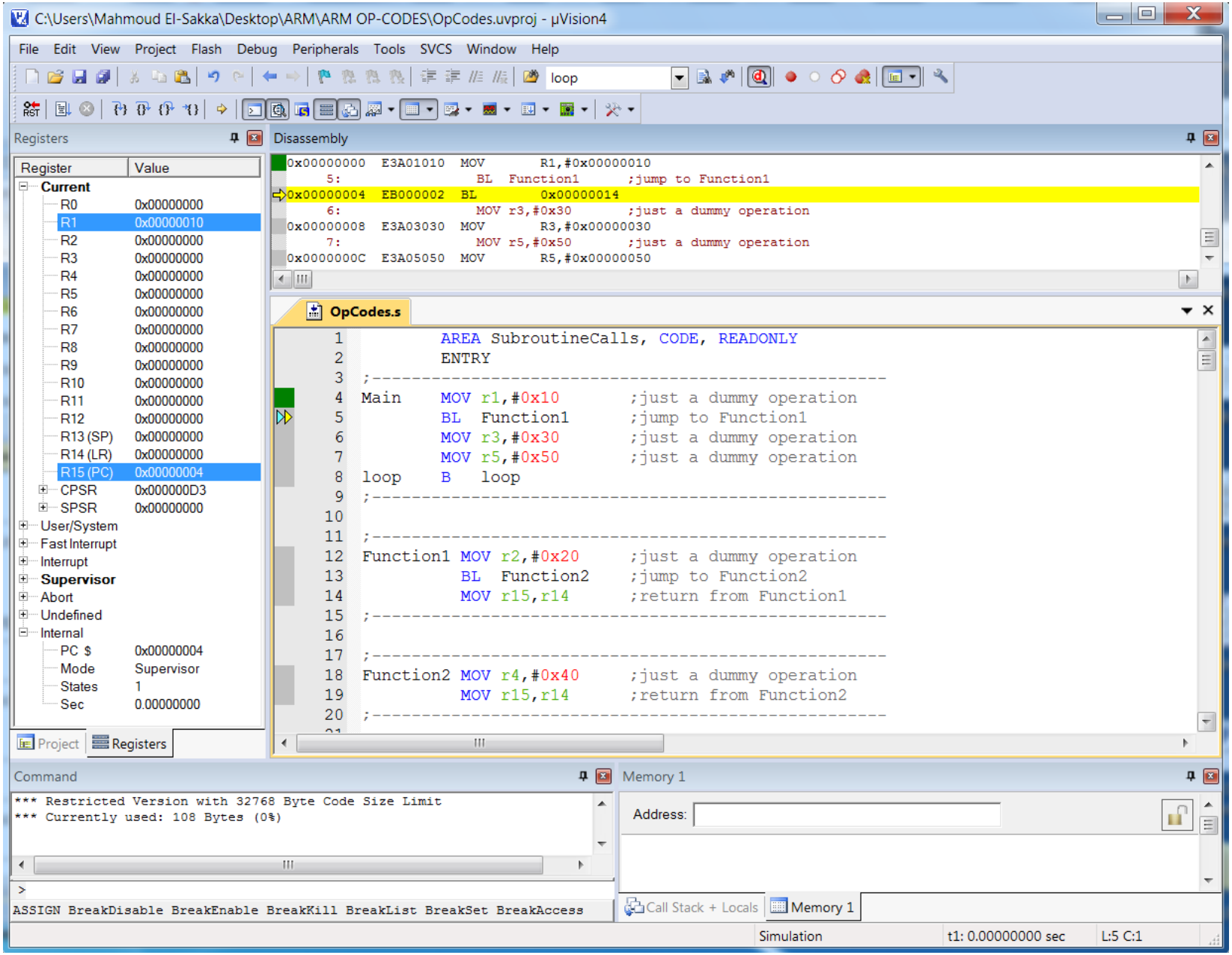
Leaf subroutine

```
;-------------------------------------------------------
Function2 MOV r4,#0x40         ;just a dummy operation
          MOV r15,r14          ;return from Function2
;-------------------------------------------------------
```

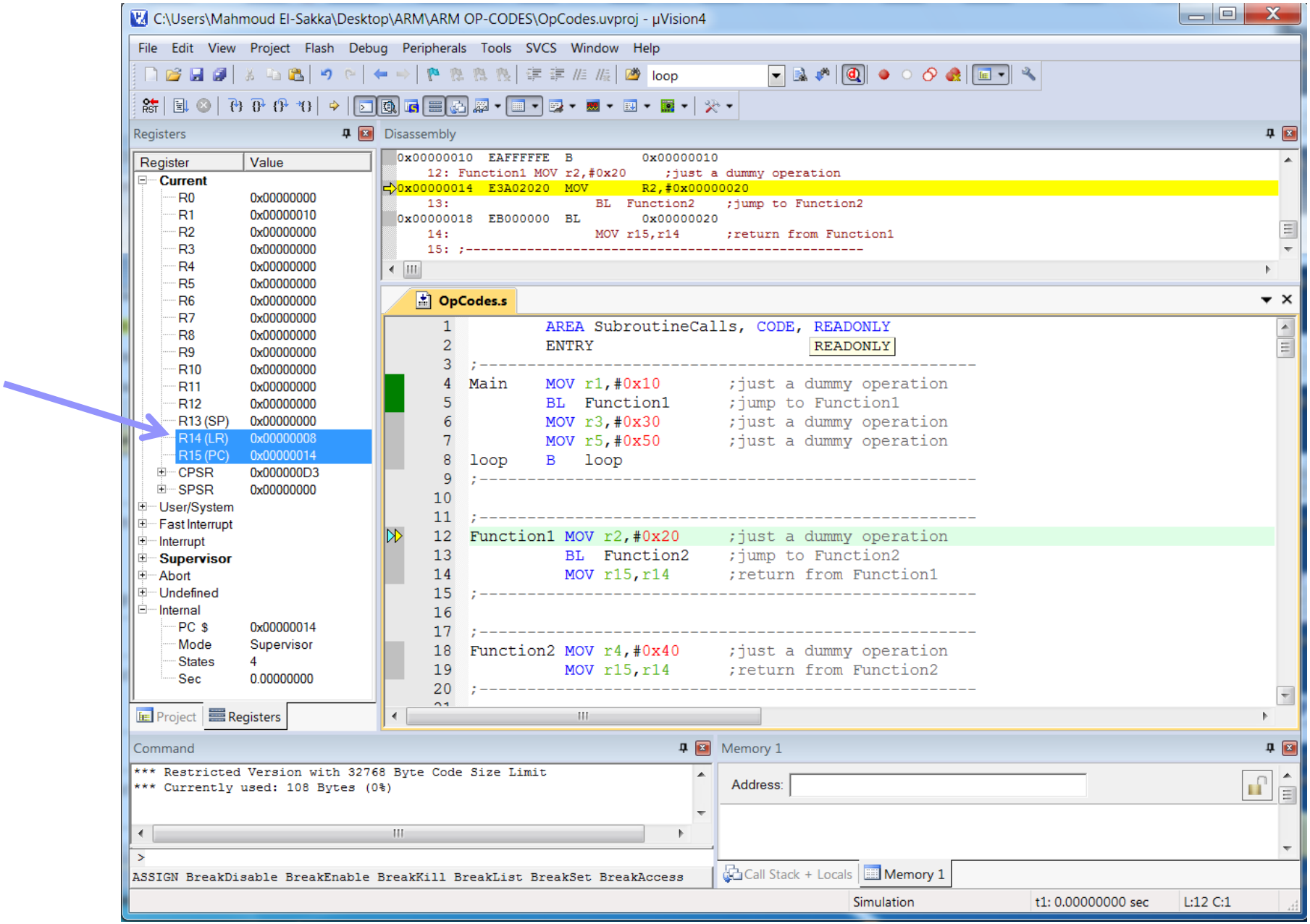
Example 2: Subroutine call and Return



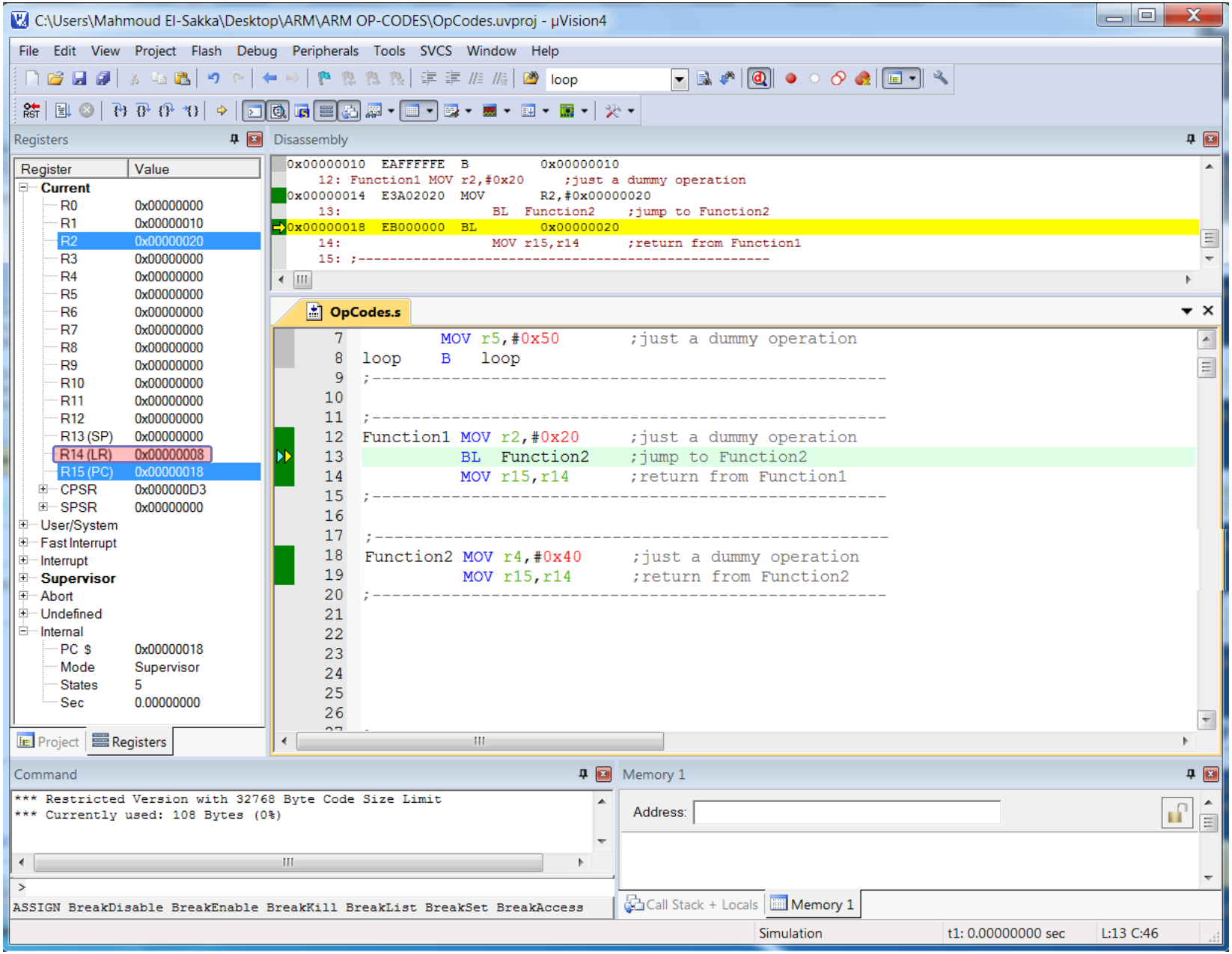
Example 2: Subroutine call and Return



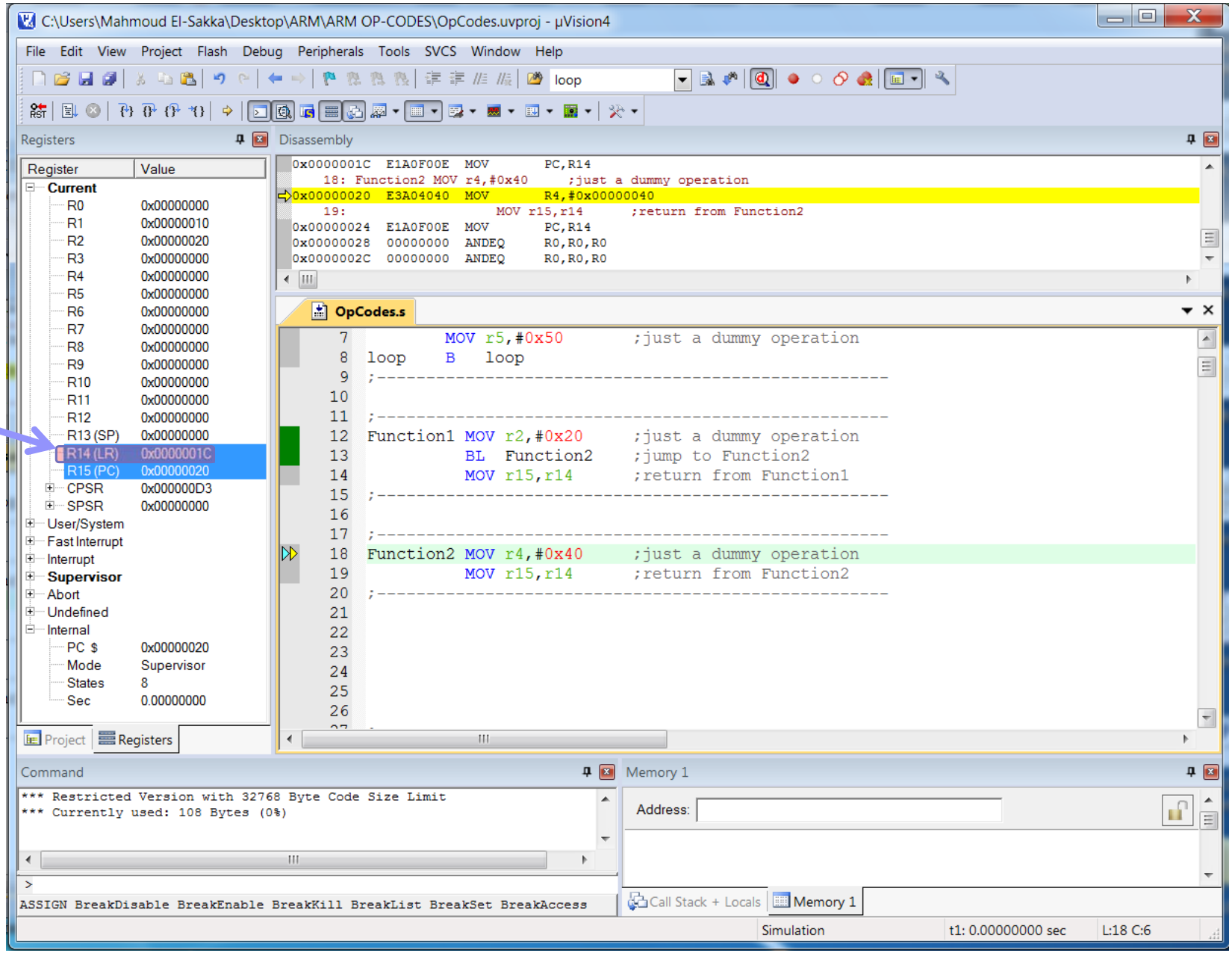
Example 2: Subroutine call and Return



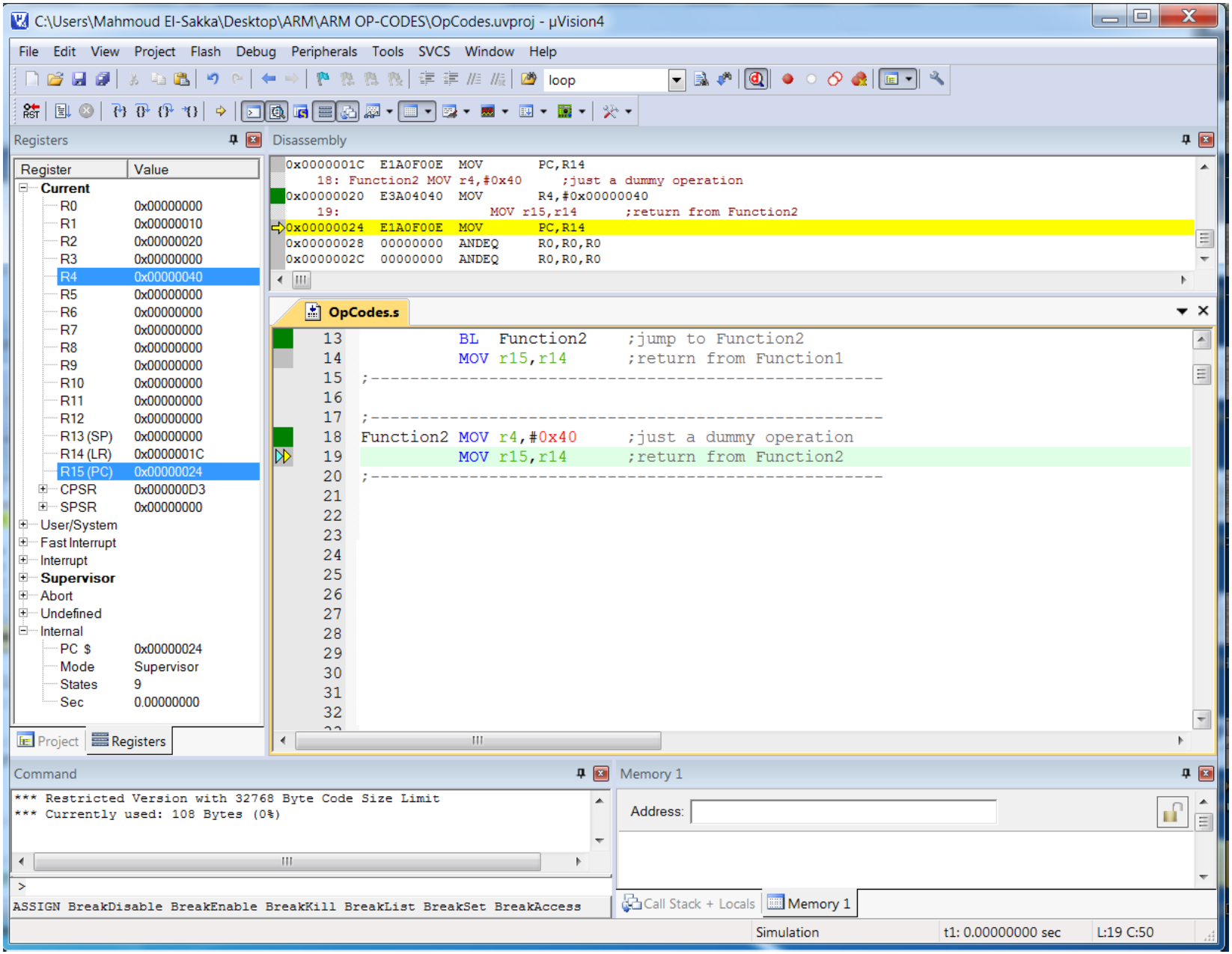
Example 2: Subroutine call and Return



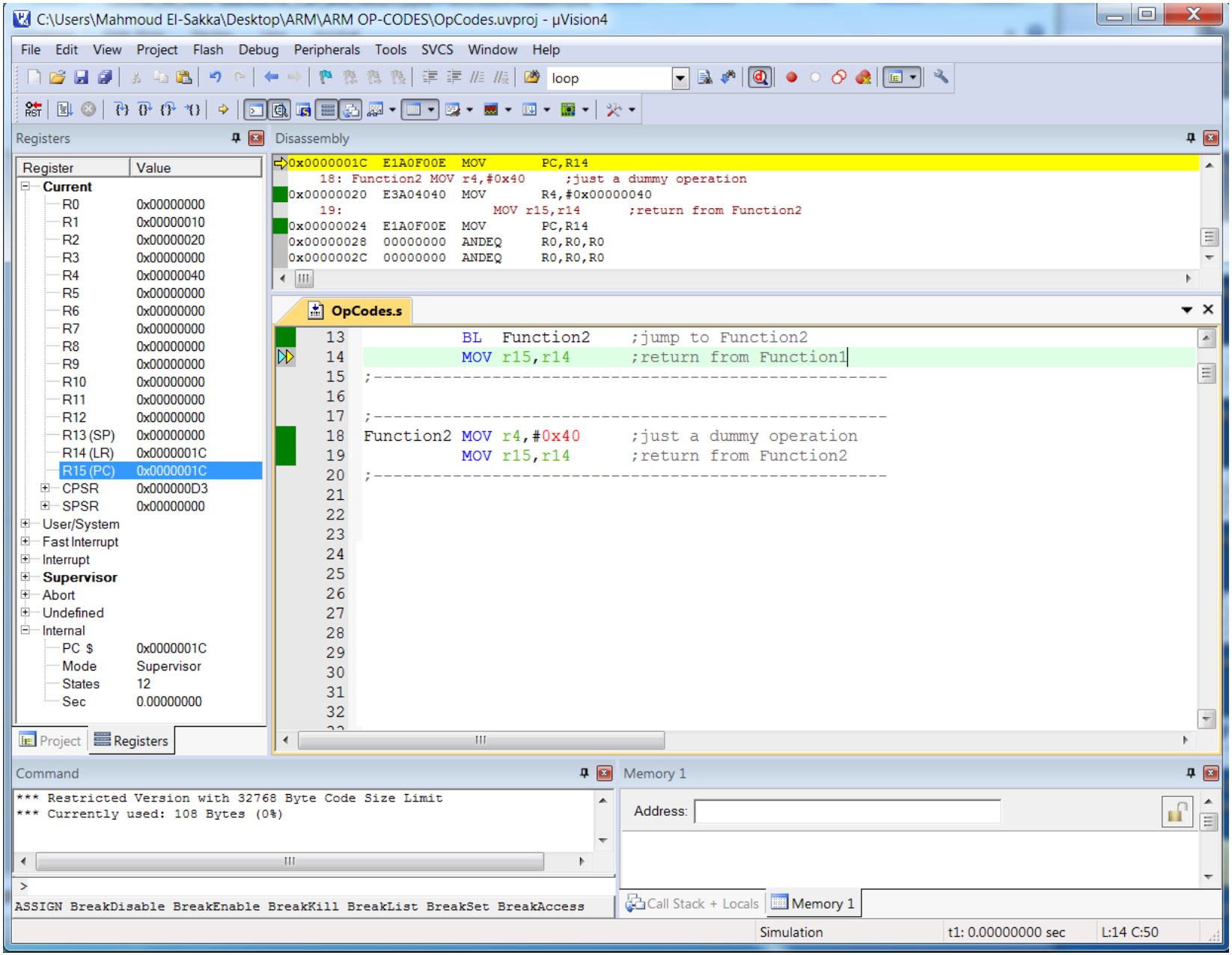
Example 2: Subroutine call and Return



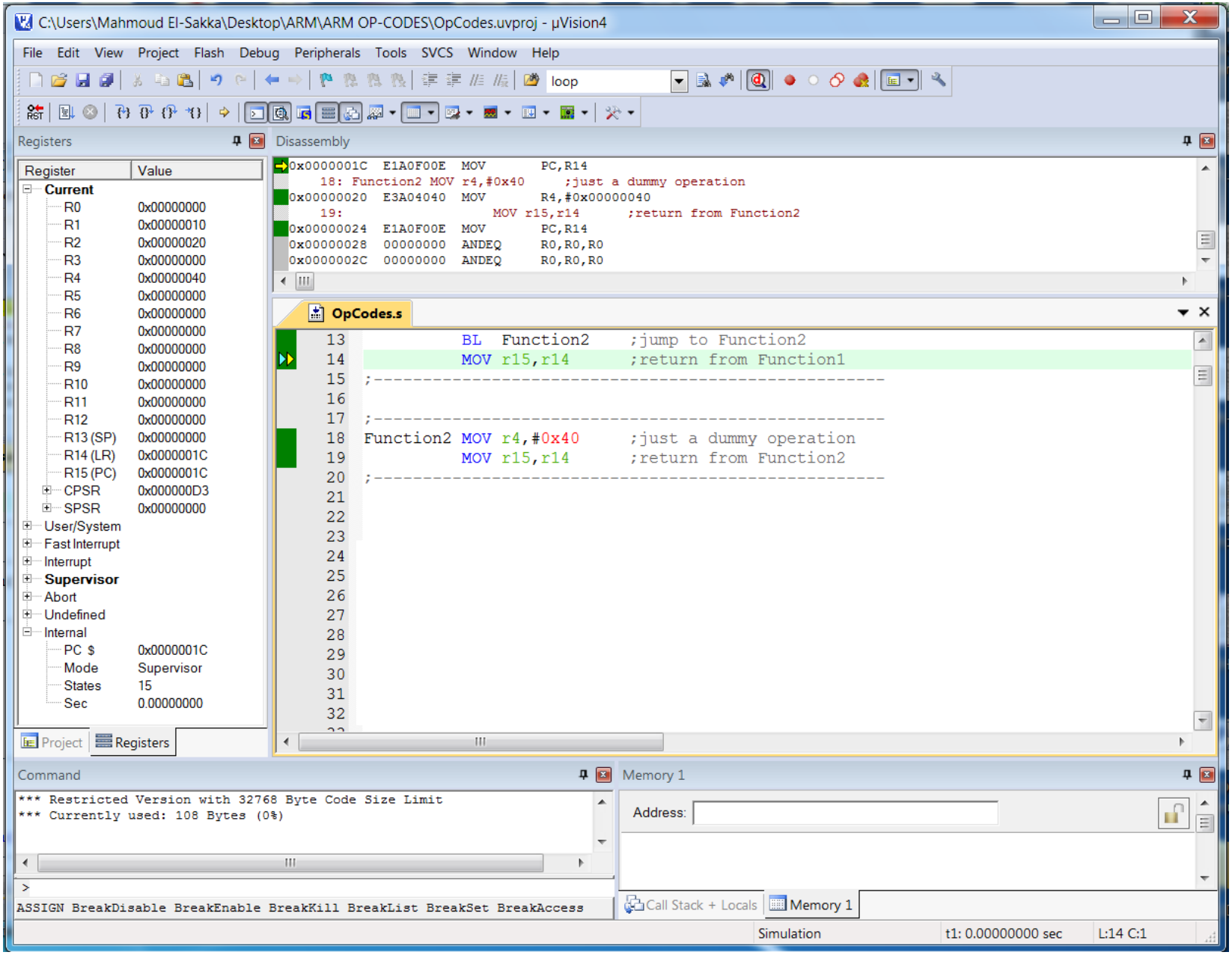
Example 2: Subroutine call and Return



Example 2: Subroutine call and Return



Example 2: Subroutine call and Return



Example 3: Subroutine call and Return

Calling a none leaf subroutine using **B** instruction and a stack without using **r14**

```
;-----------------------------------------------------
Main     MOV r1,#0x10          ;just a dummy operation
         BL  Function1         ;jump to Function1
         MOV r3,#0x30          ;just a dummy operation
loop     B   loop
;-----------------------------------------------------

;-----------------------------------------------------
Function1 MOV r2,#0x20         ;just a dummy operation
          BL  Function2        ;jump to Function2
          MOV r5,#0x50         ;just a dummy operation
          MOV r15,r14          ;return from Function1
;-----------------------------------------------------

;-----------------------------------------------------
Function2 MOV r4,#0x40         ;just a dummy operation
          MOV r15,r14          ;return from Function2
;-----------------------------------------------------
```

```
ADR r13,stack
```

```
STR r15,[r13,#-4]!
B   Function1
```

Our stack grows up toward low memory addresses and the SP is pointing to the top item on the stack, i.e., pointing to a filled item.

```
STR r15,[r13,#-4]!
B   Function1
```

```
LDR r12,[r13],#4
SUB r15,r12,#4
```

```
LDR r12,[r13],#4
SUB r15,r12,#4
```

You need to re-map the memory to make the stack space read/write enabled (Debug/Memory Map)

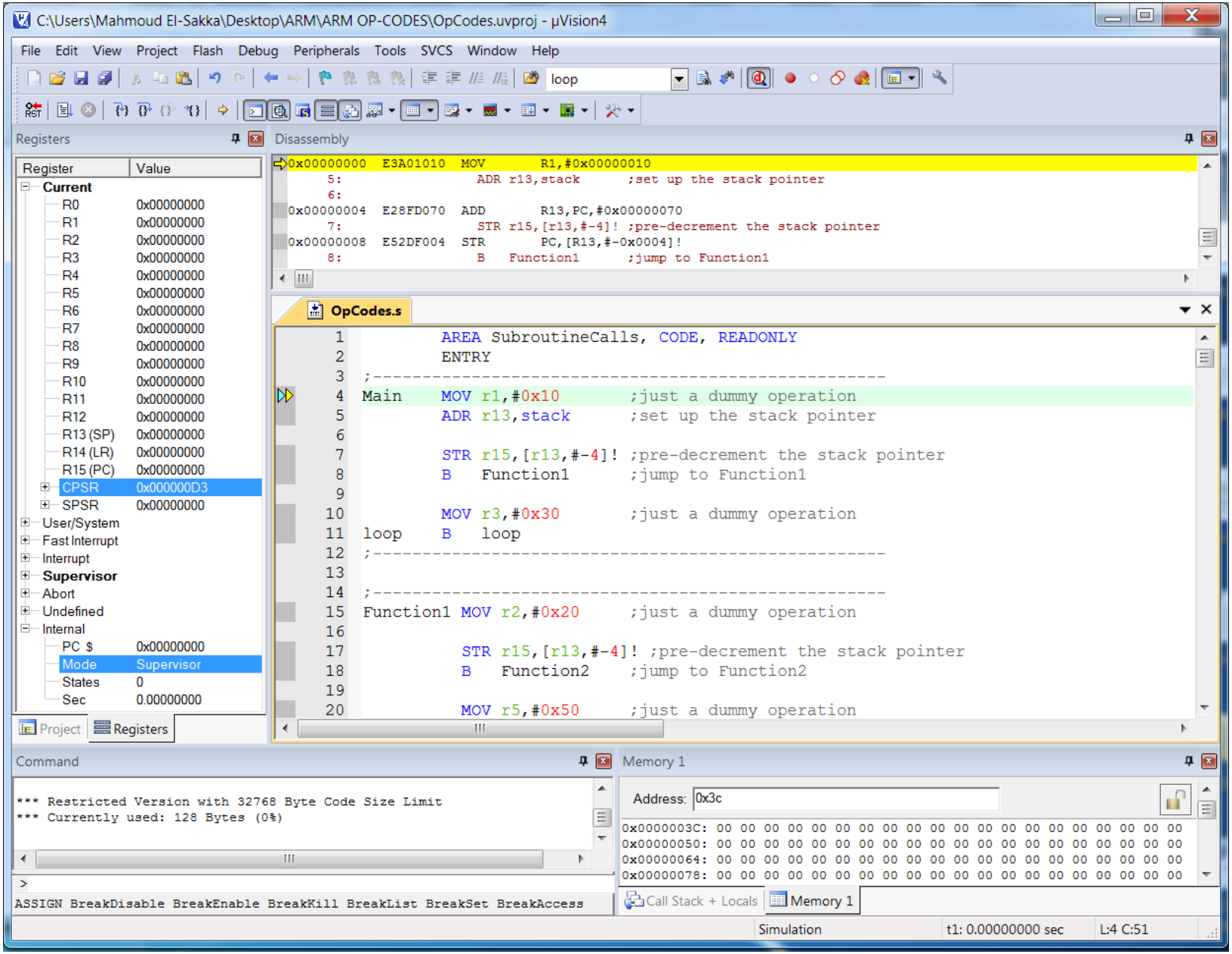
```
        SPACE 0x40
stack   DCD   0x0
```

Example 3: Subroutine call and Return

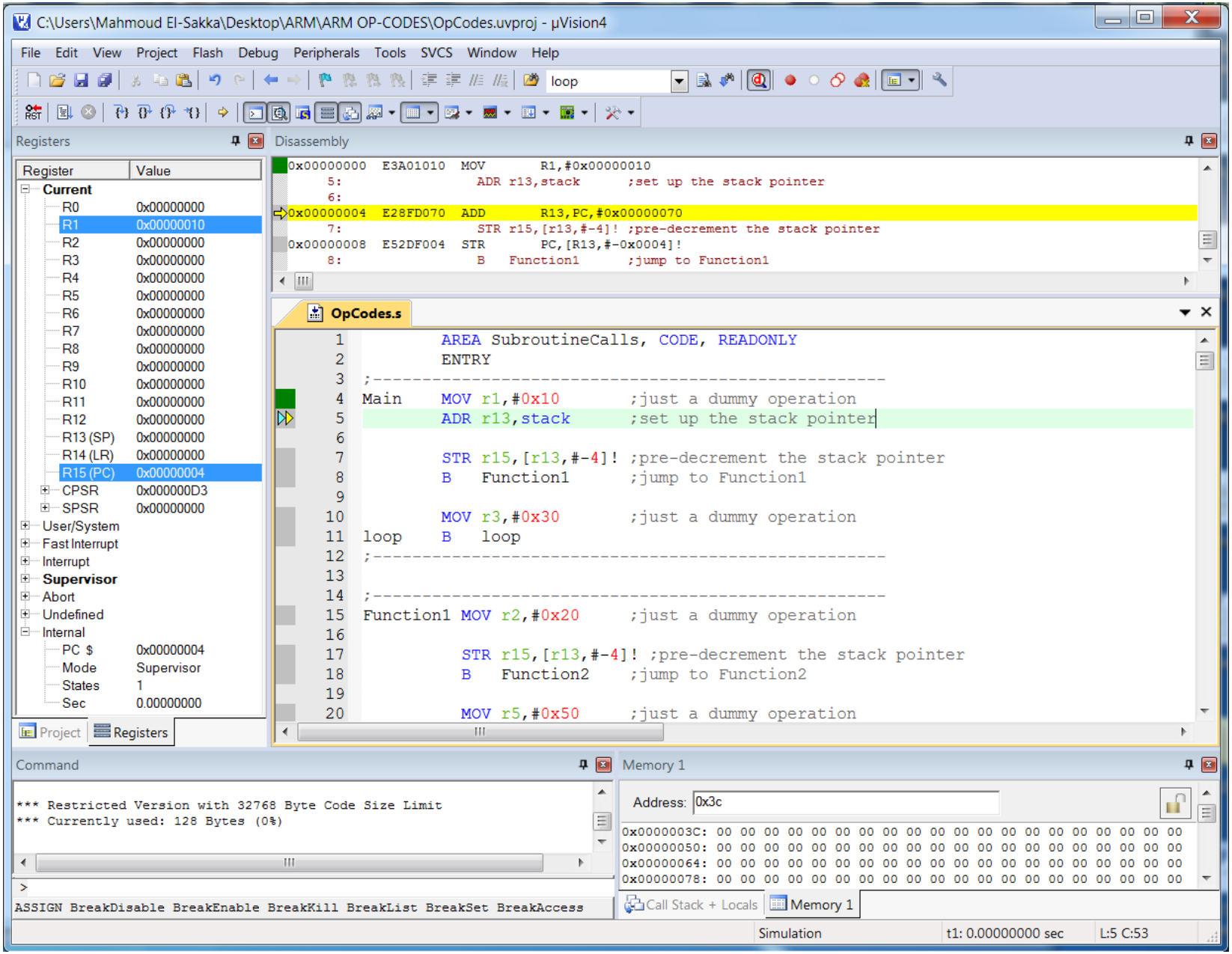
Calling a none leaf subroutine using **B** instruction and a stack without using **r14**

```
;-------------------------------------------------
Main     MOV r1,#0x10          ;just a dummy operation
         ADR r13,stack         ;set up the stack pointer
         STR r15,[r13,#-4]! ;pre-decrement the stack pointer
         B   Function1         ;jump to Function1 (B not BL)
         MOV r3,#0x30          ;just a dummy operation
loop     B   loop
;-------------------------------------------------
;-------------------------------------------------
Function1 MOV r2,#0x20      ;just a dummy operation
          STR r15,[r13,#-4]! ;pre-decrement the stack pointer
          B   Function2      ;jump to Function2 (B not BL)
          MOV r5,#0x50       ;just a dummy operation
          LDR r12,[r13],#4   ;get the PC and post-increment the stack pointer
          SUB r15,r12,#4     ;modify the restored PC (because it is 4 too big)
                             ;and return from Function1
;-------------------------------------------------
;-------------------------------------------------
Function2 MOV r4,#0x40        ;just a dummy operation
          LDR r12,[r13],#4    ;get the PC and post-increment the stack pointer
          SUB r15,r12,#4      ;modify the restored PC (because it is 4 too big)
                              ;and return from Function2
;-------------------------------------------------
;-------------------------------------------------
          SPACE 0x40          ;reserved room for the stack to grow up
stack     DCD   0x0           ;the base of the stack
;-------------------------------------------------
```

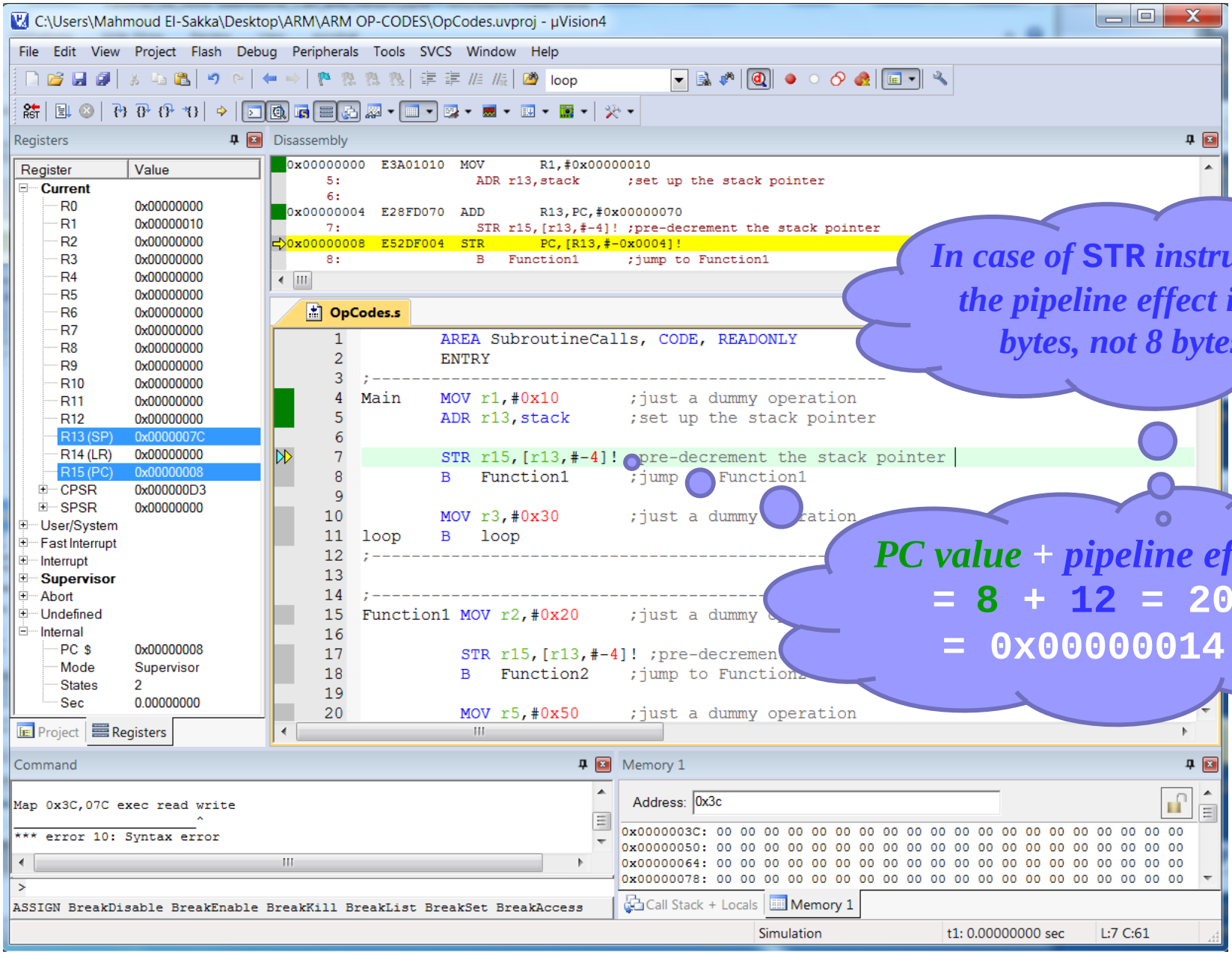
Example 3: Subroutine call and Return



Example 3: Subroutine call and Return



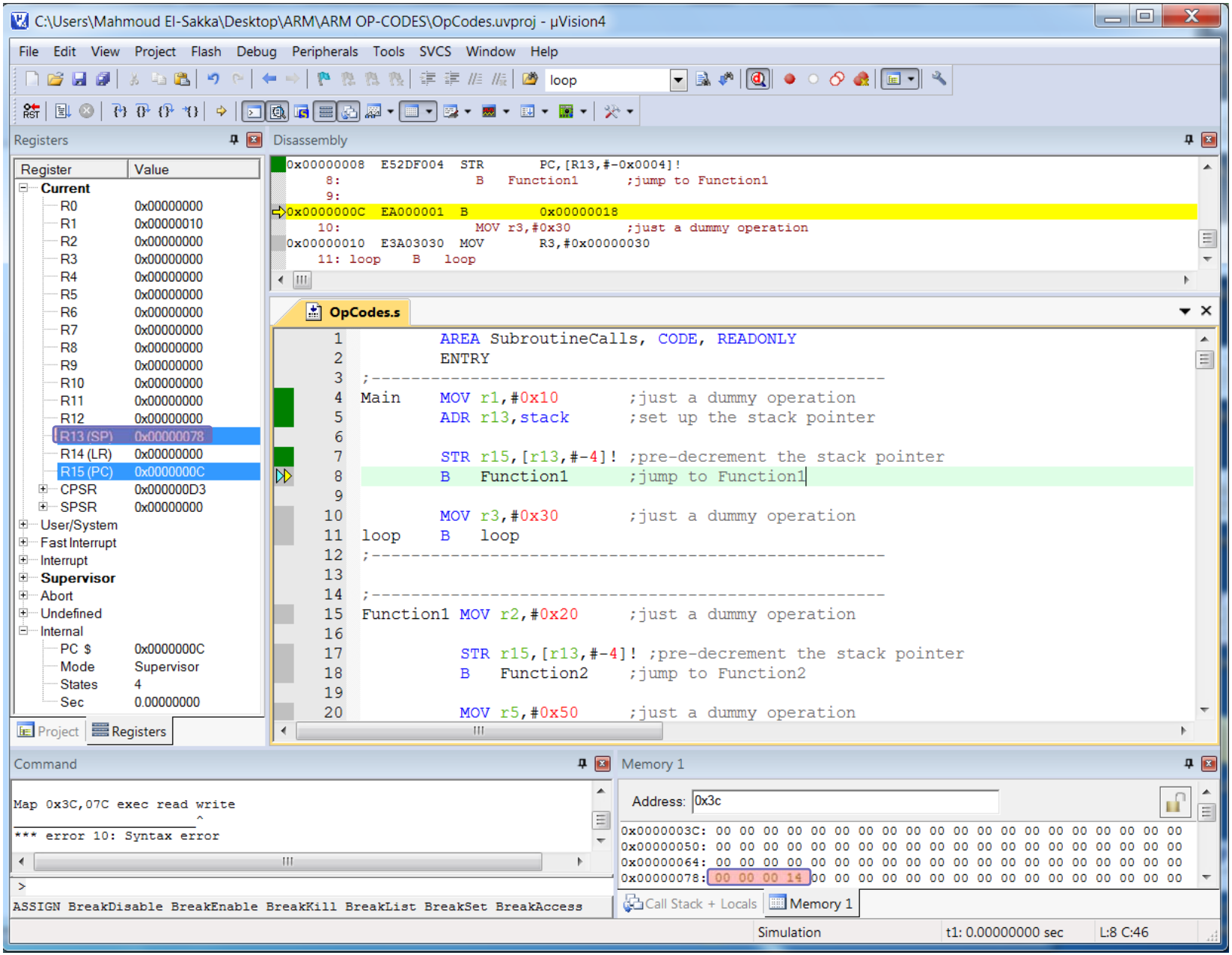
Example 3: Subroutine call and Return



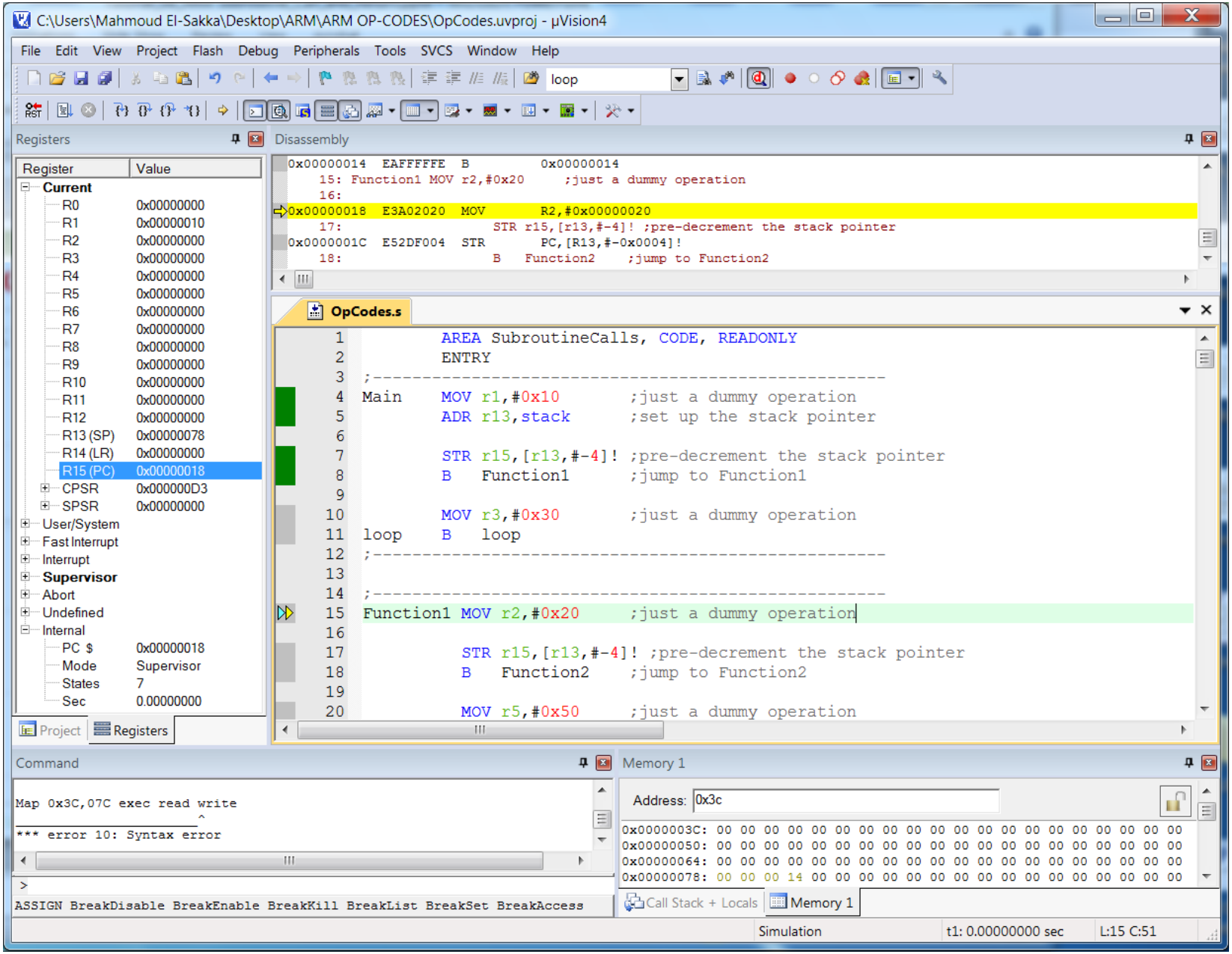
In case of **STR** *instruction, the pipeline effect is 12 bytes, not 8 bytes.*

PC value + *pipeline effect*
= 8 + 12 = 20
= 0x00000014

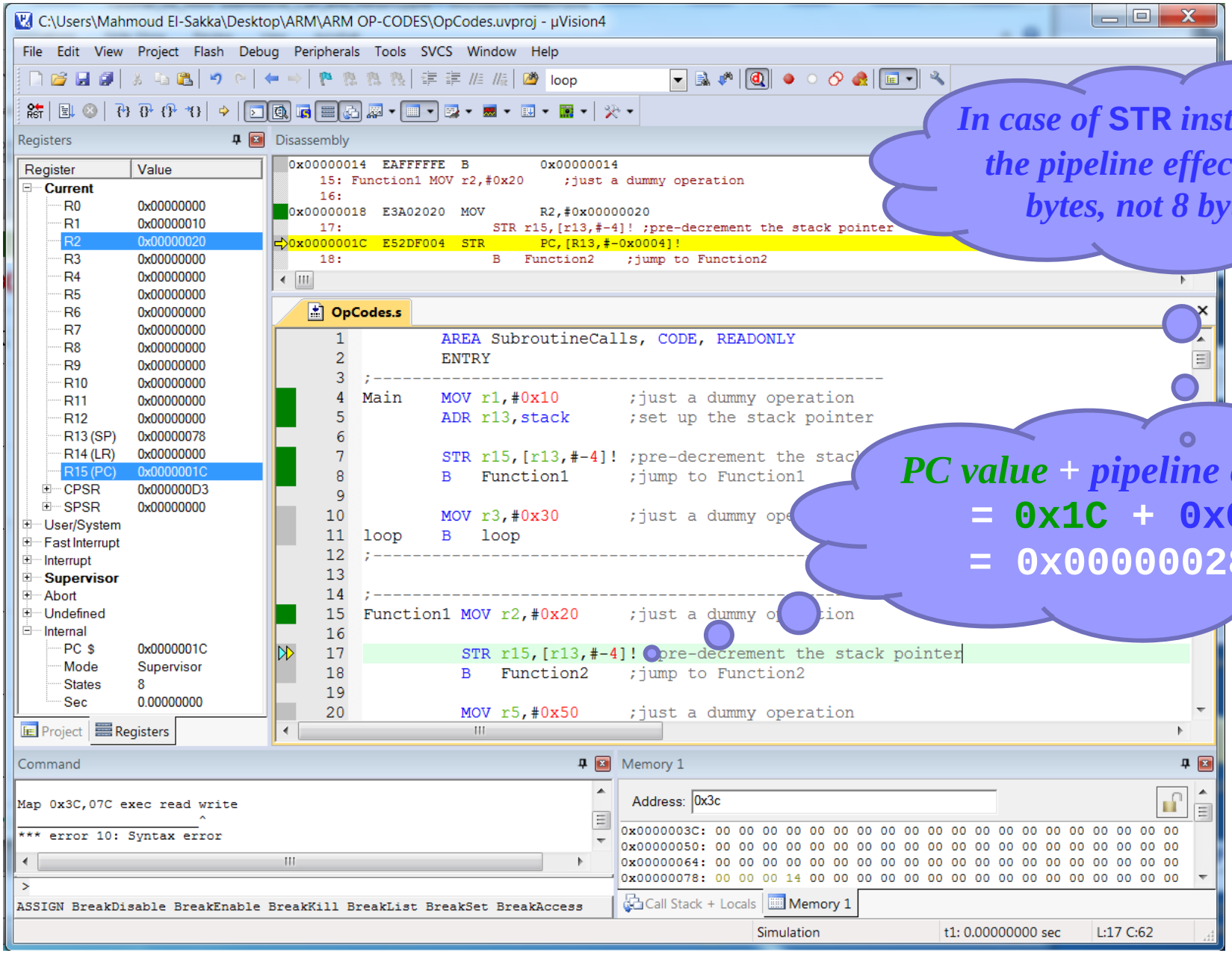
Example 3: Subroutine call and Return



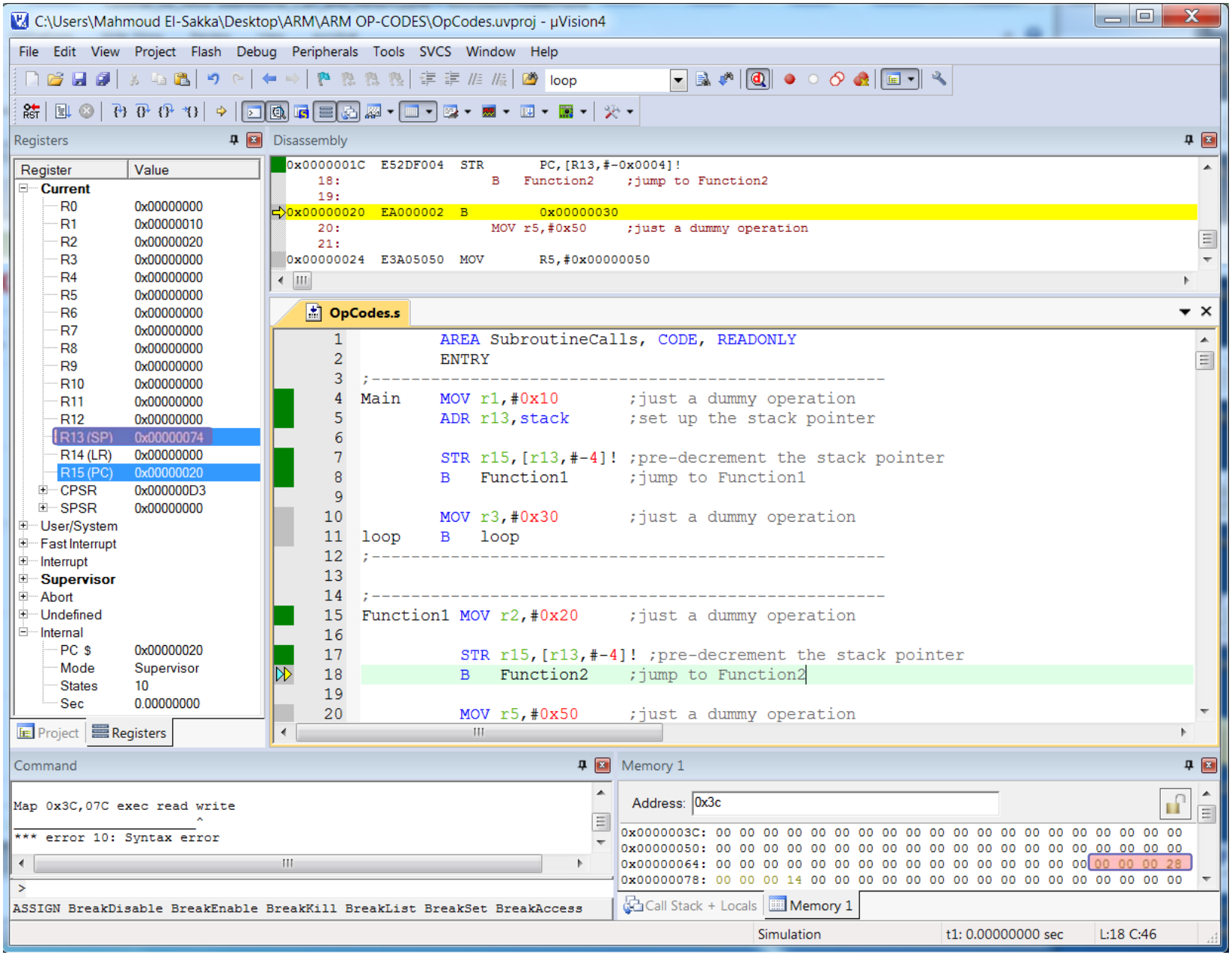
Example 3: Subroutine call and Return



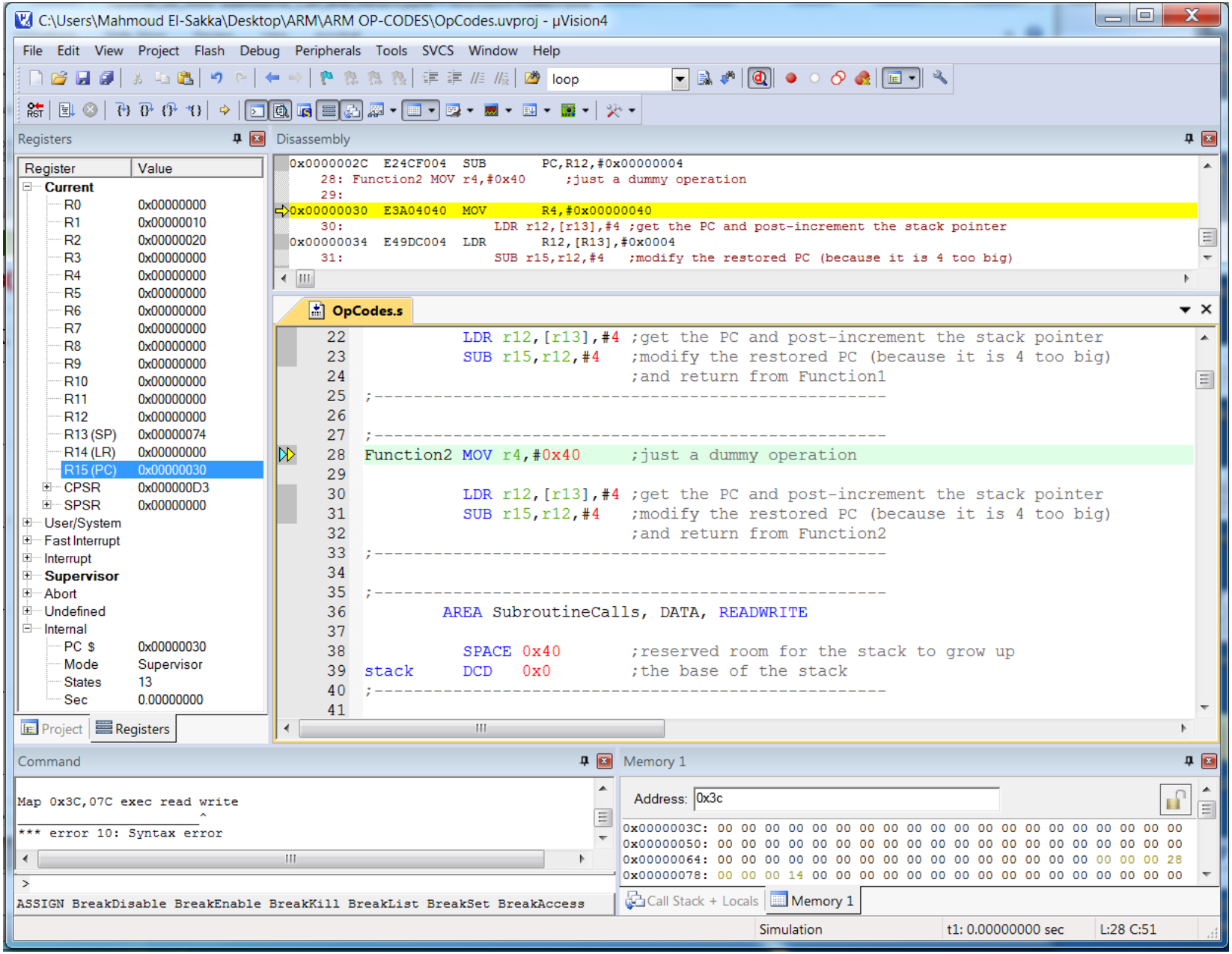
Example 3: Subroutine call and Return



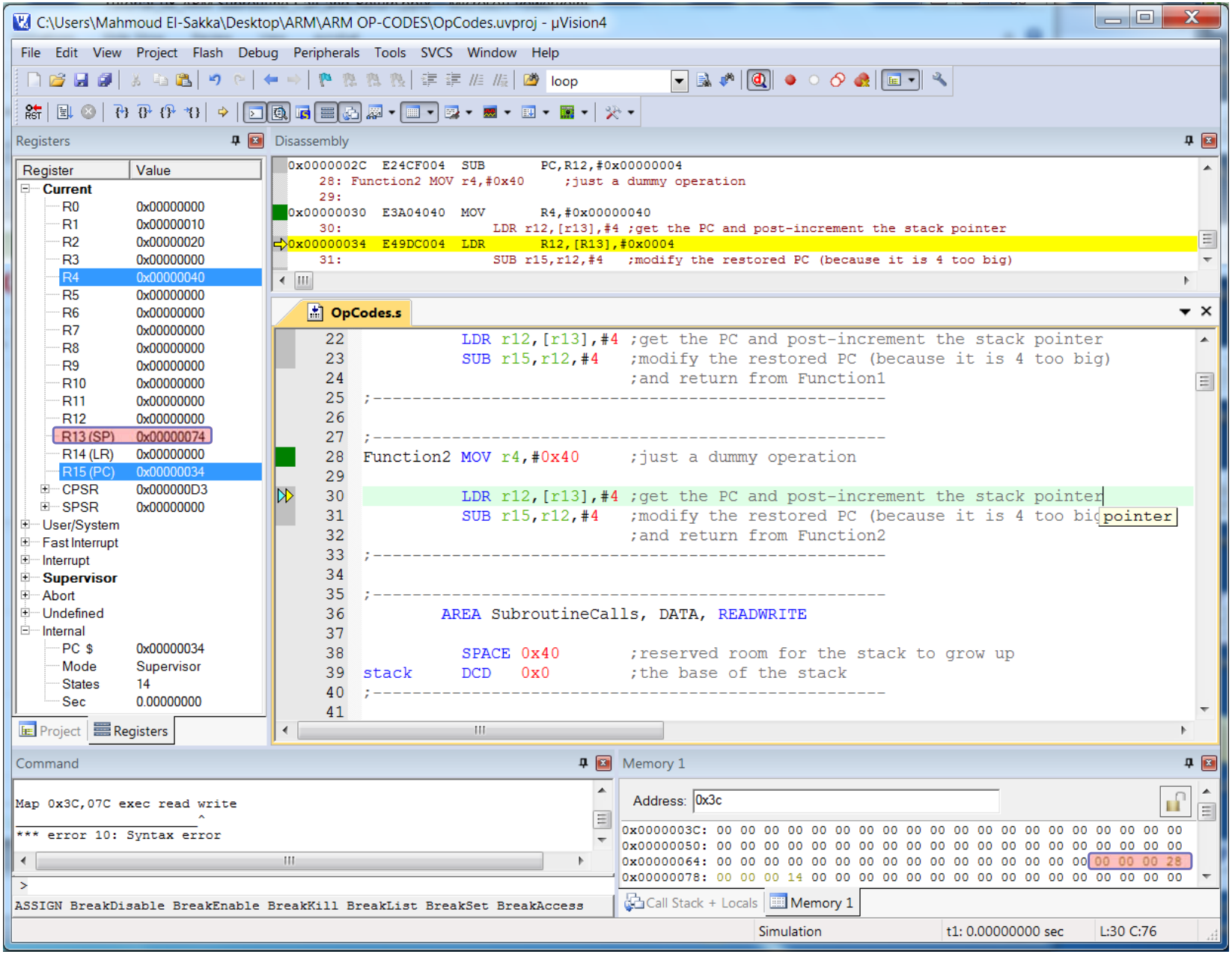
Example 3: Subroutine call and Return



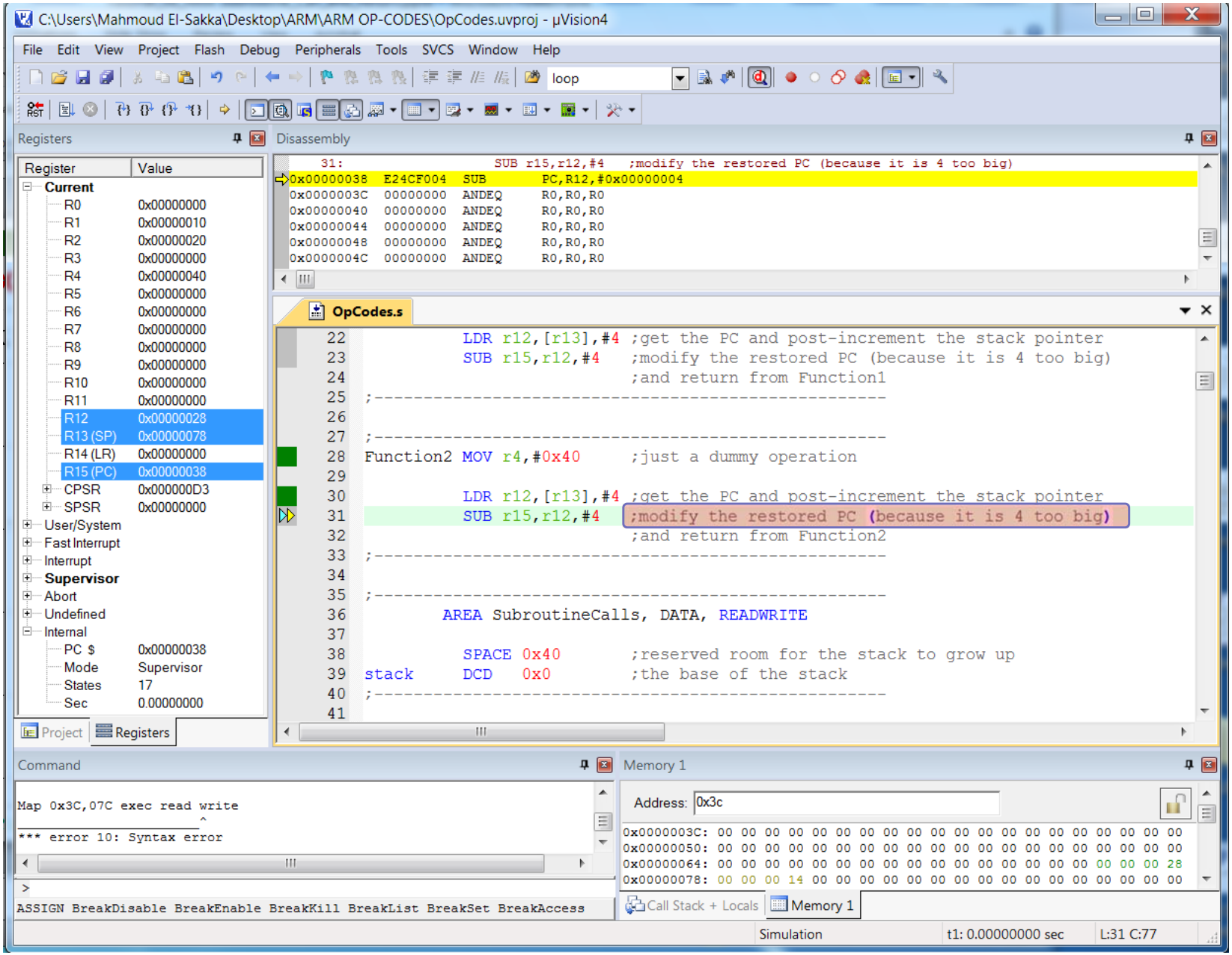
Example 3: Subroutine call and Return



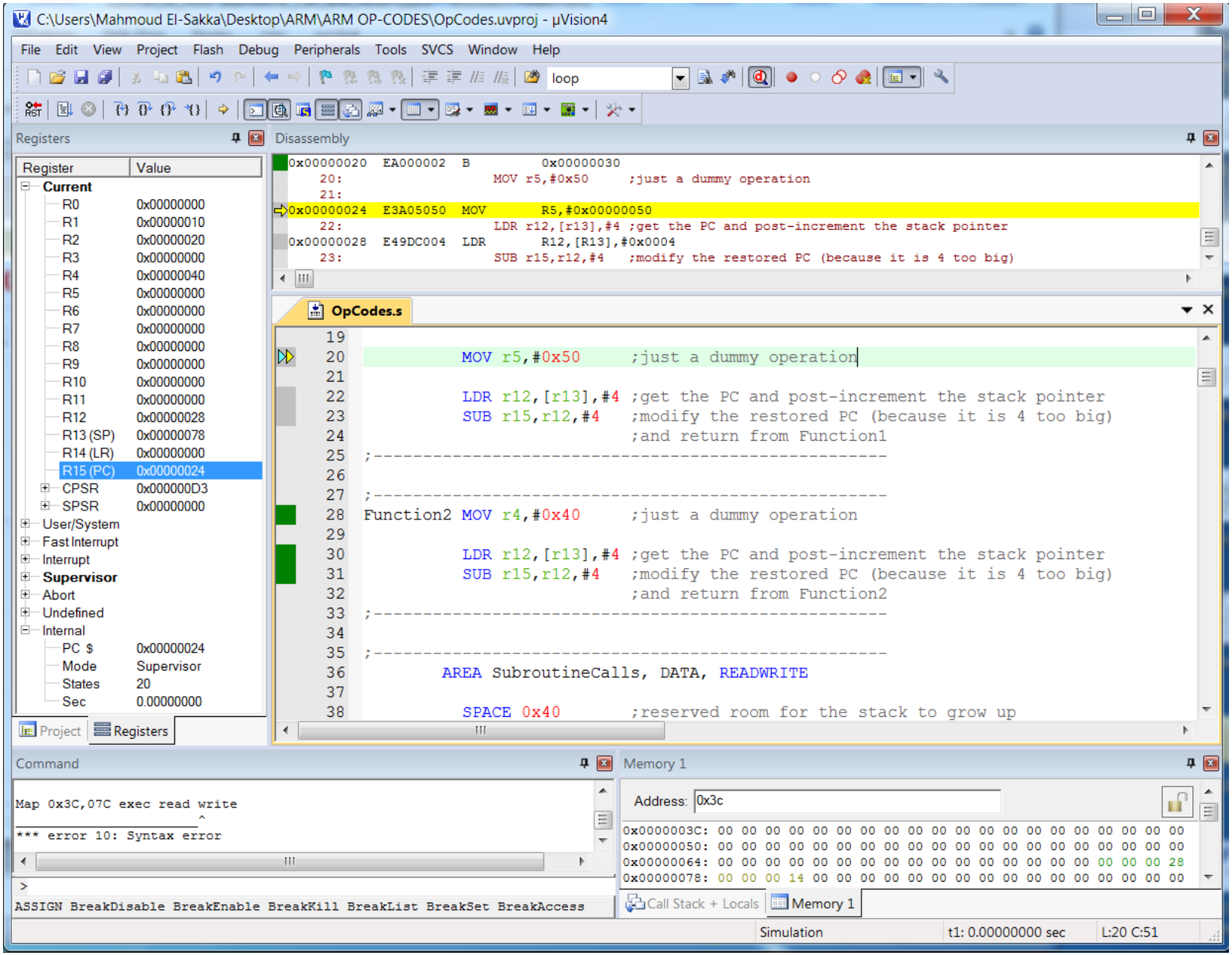
Example 3: Subroutine call and Return



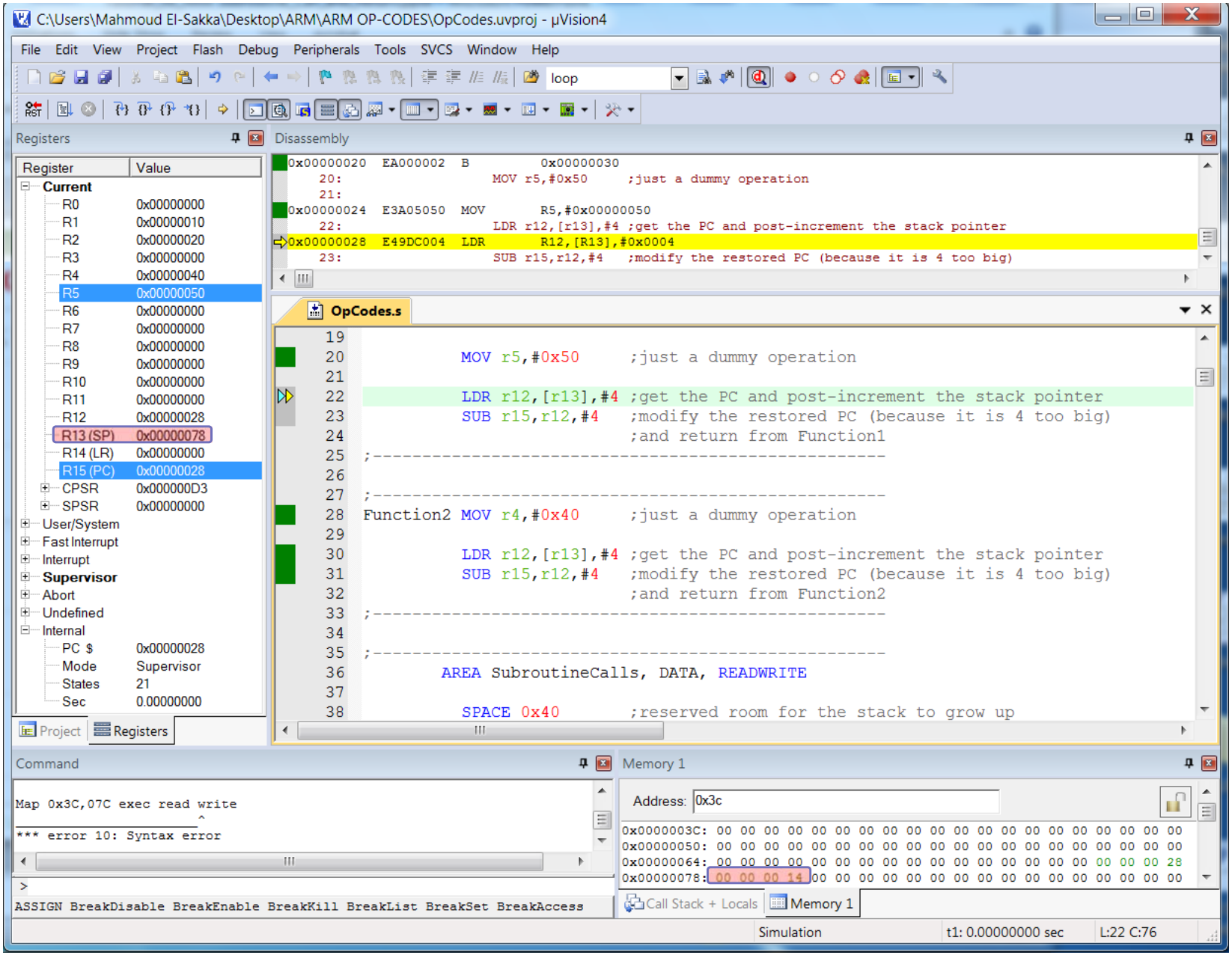
Example 3: Subroutine call and Return



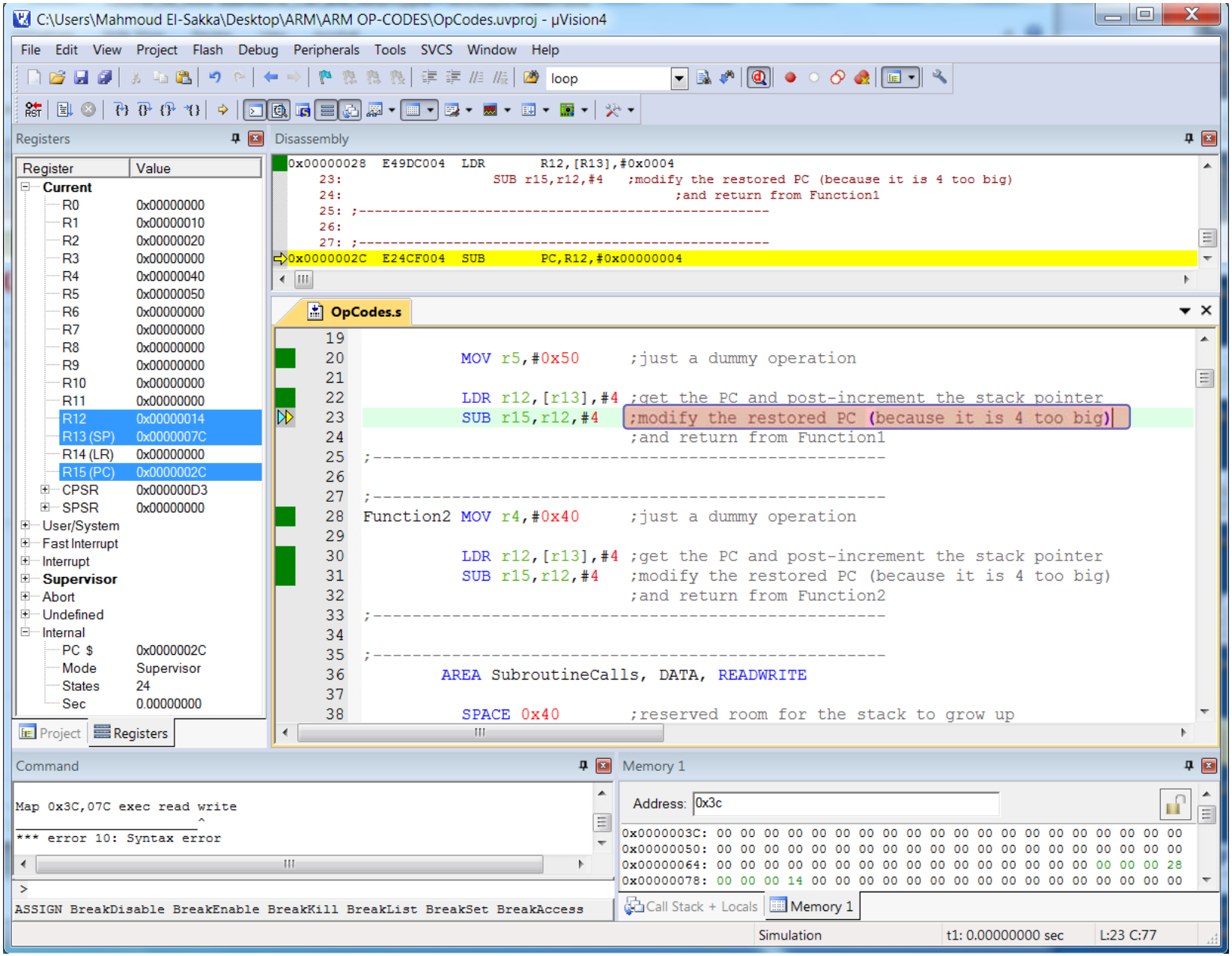
Example 3: Subroutine call and Return



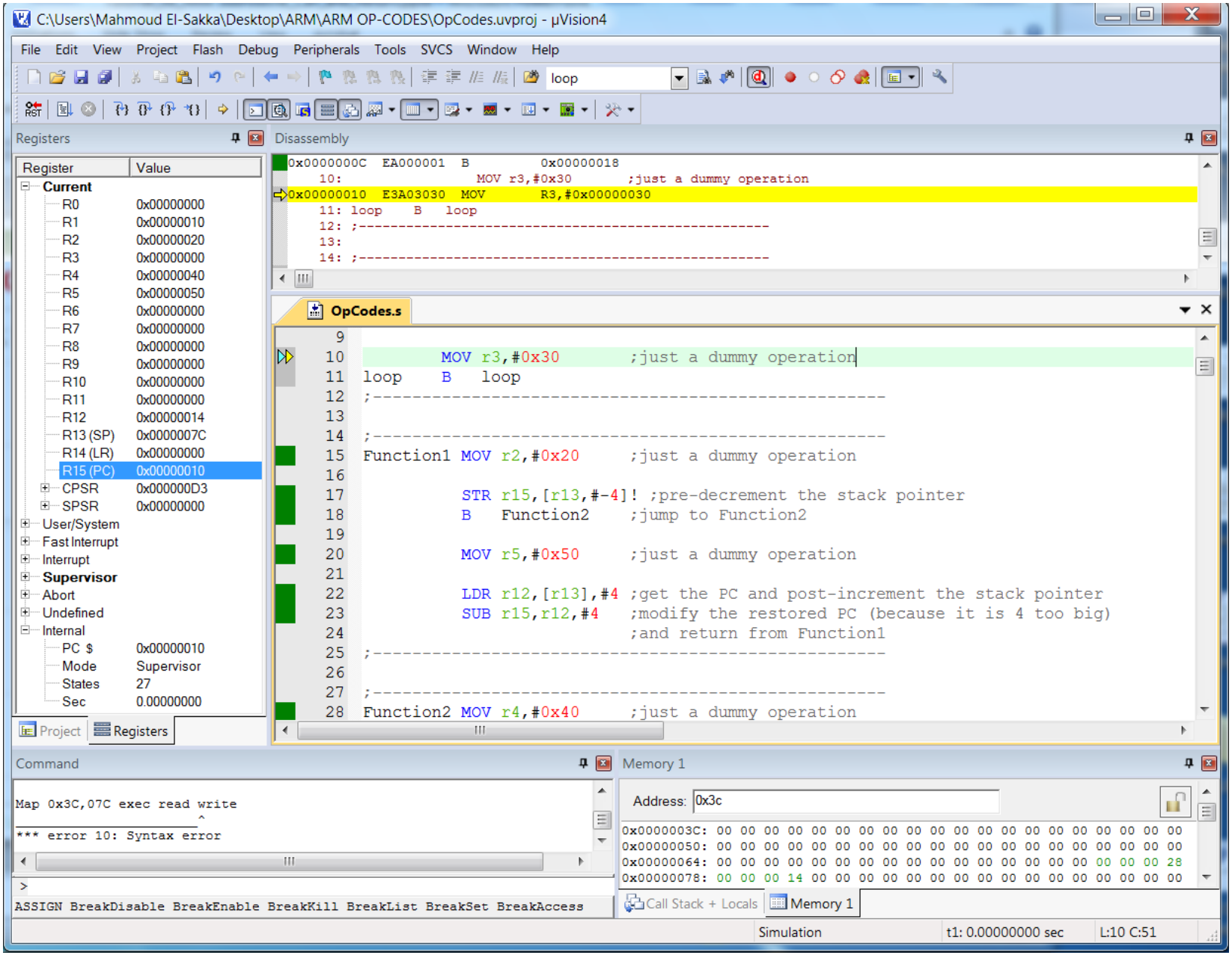
Example 3: Subroutine call and Return



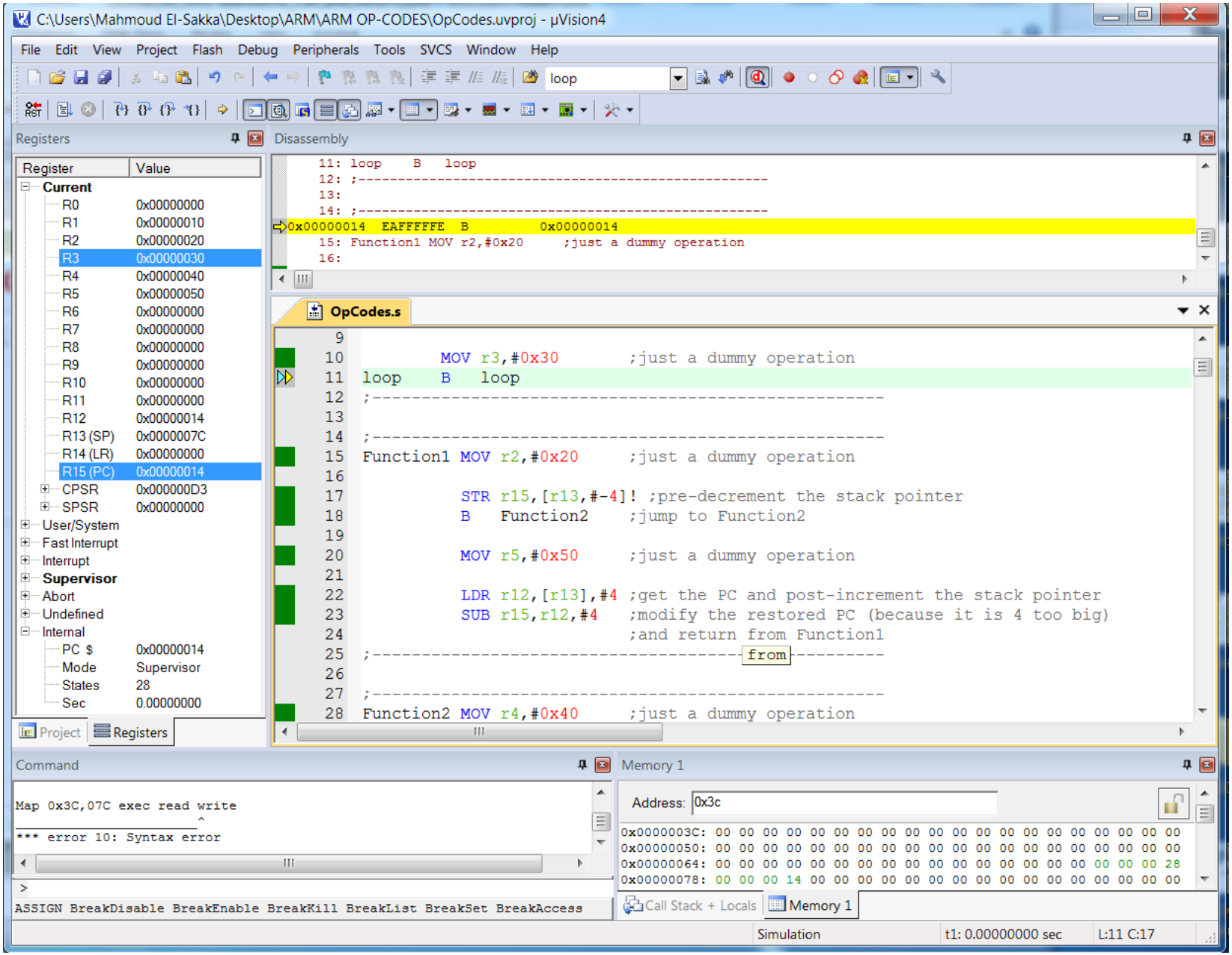
Example 3: Subroutine call and Return



Example 3: Subroutine call and Return



Example 3: Subroutine call and Return



Example 4: Subroutine call and Return

Calling a none leaf subroutine using **B** instruction and a stack without using **r14**

```
;-----------------------------------------------------
Main     MOV r1,#0x10          ;just a dummy operation
         ADR r13,stack         ;set up the stack pointer
         STR r15,[r13,#-4]! ;pre-                nt the stack pointer
         B   Function1         ;jum              unction1 (B not BL)
         MOV r3,#0x30          ;just a dummy operation
loop     B   loop
;-----------------------------------------------------
;-----------------------------------------------------
Function1 MOV r2,#0x20      ;just a dummy operation
          STR r15,[r13,#-4]! ;pre-                nt the stack pointer
          B   Function2     ;jump                 ction2 (B not BL)
          MOV r5,#0x50      ;just a dummy operation
          LDR r12,[r13],#4  ;get the PC and post-increment the stack pointer
          SUB r15,r12,#4    ;modify the restored PC (because it is 4 too big)
                            ;and return from Function1
;-----------------------------------------------------
;-----------------------------------------------------
Function2 MOV r4,#0x40        ;just a dummy operation
          LDR r12,[r13],#4    ;get the PC and post-increment the stack pointer
          SUB r15,r12,#4      ;modify the restored PC (because it is 4 too big)
                              ;and return from Function2
;-----------------------------------------------------
;-----------------------------------------------------
          SPACE 0x40          ;reserved room for the stack to grow up
stack     DCD   0x0           ;the base of the stack
;-----------------------------------------------------
```

NOP

NOP

What is the effect of adding these two NOP instructions as shown?

What does it need to be done here?

What does it need to be done here?

Example 4: Subroutine call and Return

Calling a none leaf subroutine using **B** instruction and a stack without using **r14**

```
;-------------------------------------------------------
Main     MOV r1,#0x10          ;just a dummy operation
         ADR r13,stack         ;set up the stack pointer
         STR r15,[r13,#-4]! ;pre-decrement the stack pointer
         NOP
         B   Function1         ;jump to Function1 (B not BL)
         MOV r3,#0x30          ;just a dummy operation
loop     B   loop
;-------------------------------------------------------
;-------------------------------------------------------
Function1 MOV r2,#0x20      ;just a dummy operation
          STR r15,[r13,#-4]! ;pre-decrement the stack pointer
          NOP
          B   Function2      ;jump to Function2 (B not BL)
          MOV r5,#0x50       ;just a dummy operation
          LDR r15,[r13],#4   ;get the PC and post-increment the stack pointer
                             ;and return from Function1
;-------------------------------------------------------
;-------------------------------------------------------
Function2 MOV r4,#0x40       ;just a dummy operation
          LDR r15,[r13],#4   ;get the PC and post-increment the stack pointer
                             ;and return from Function2
;-------------------------------------------------------
;-------------------------------------------------------
          SPACE 0x40         ;reserved room for the
stack     DCD   0x0          ;the base of the stack
;-------------------------------------------------------
```

Try this example by yourself!!

You need to re-map the memory to make the stack space read/write enabled (Debug/Memory Map)

Example 5: Subroutine call and Return

Calling a none leaf subroutine using **BL** instruction, **r14** and a stack

```
;-------------------------------------------------
Main    MOV r1,#0x10          ;just a dummy op...
        BL  Function1         ;jump to Function1
        MOV r3,#0x30          ;just a dummy operation
loop    B   loop
;-------------------------------------------------
;                 None leaf subroutine
;-------------------------------------------------
Function1 MOV r2,#0x20        ;just a dummy operation
          BL  Function2       ;jump to Function2
          MOV r5,#0x50        ;just a dummy operation
          MOV r15,r14         ;return from Function1
;-------------------------------------------------
;                   Leaf subroutine
;-------------------------------------------------
Function2 MOV r4,#0x40        ;just a dummy operation
          MOV r15,r14         ;return from Function2
;-------------------------------------------------
```

ADR r13,stack

STR r14,[r13,#-4]!

You do this ONLY with None leaf subroutine

LDR r14,[r13],#4

```
        SPACE 0x40
stack   DCD   0x0
```

You need to re-map the memory to make the stack space read/write enabled (Debug/Memory Map)

Example 5: Subroutine call and Return

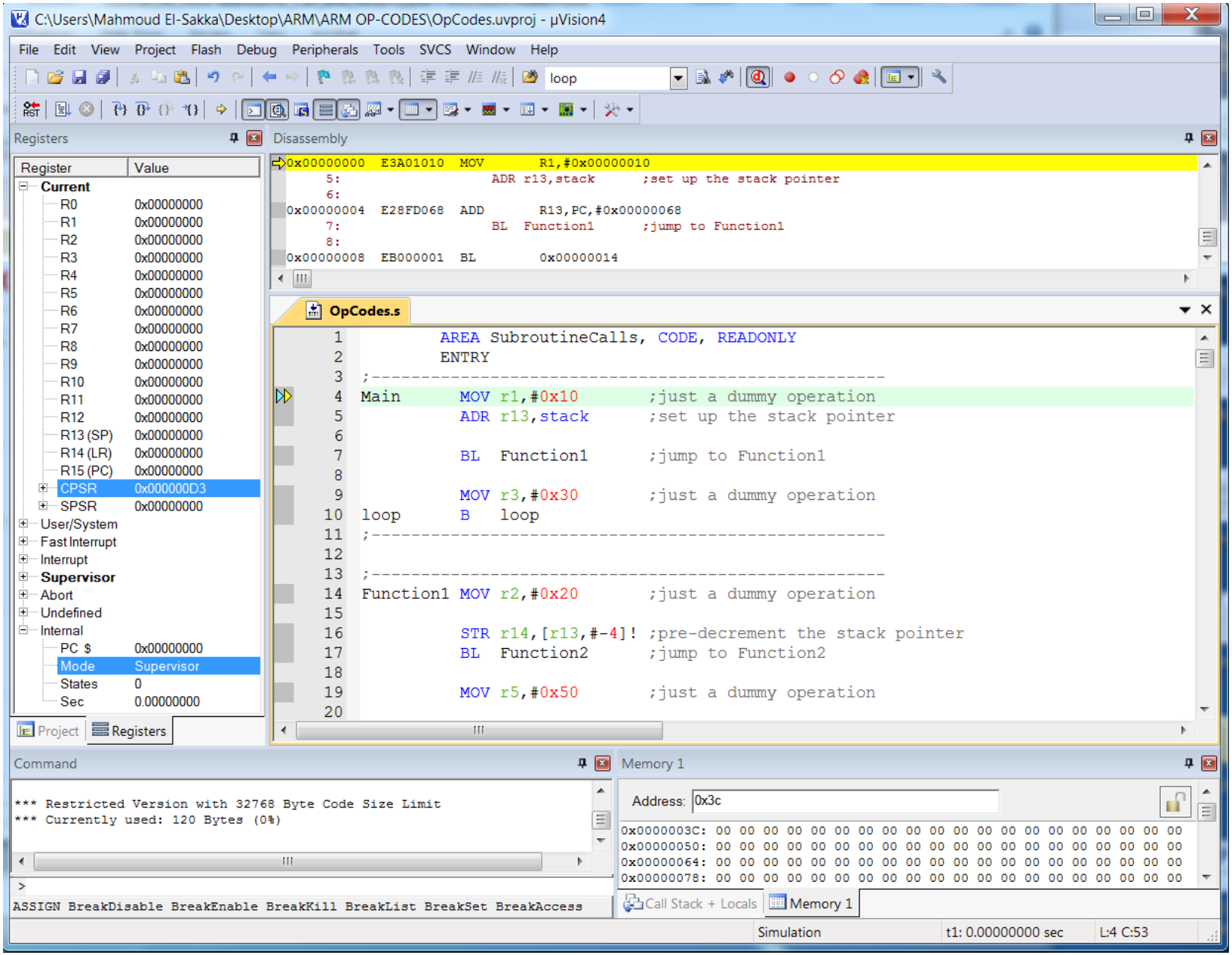
Calling a none leaf subroutine using **BL** instruction, **r14** and a stack

```
;-----------------------------------------------------
Main      MOV r1,#0x10          ;just a dummy operation
          ADR r13,stack         ;set up the stack pointer

          BL   Function1        ;jump to Function1

          MOV r3,#0x30          ;just a dummy operation
loop      B   loop
;-----------------------------------------------------
;-----------------------------------------------------
Function1 MOV r2,#0x20      ;just a dummy operation
          STR r14,[r13,#-4]! ;pre-decrement the stack pointer
          BL   Function2    ;jump to Function2

          MOV r5,#0x50          ;just a dummy oper

          LDR r14,[r13],#4  ;get the PC and post-increment the stack pointer
          MOV r15,r14       ;return from Function1
;-----------------------------------------------------
;-----------------------------------------------------
Function2 MOV r4,#0x40          ;just a dummy operation
          MOV r15,r14           ;return from Function2
;-----------------------------------------------------
;-----------------------------------------------------
          SPACE 0x40            ;reserved room for the stack to grow up
stack     DCD   0x0             ;the base of the stack
;-----------------------------------------------------
```

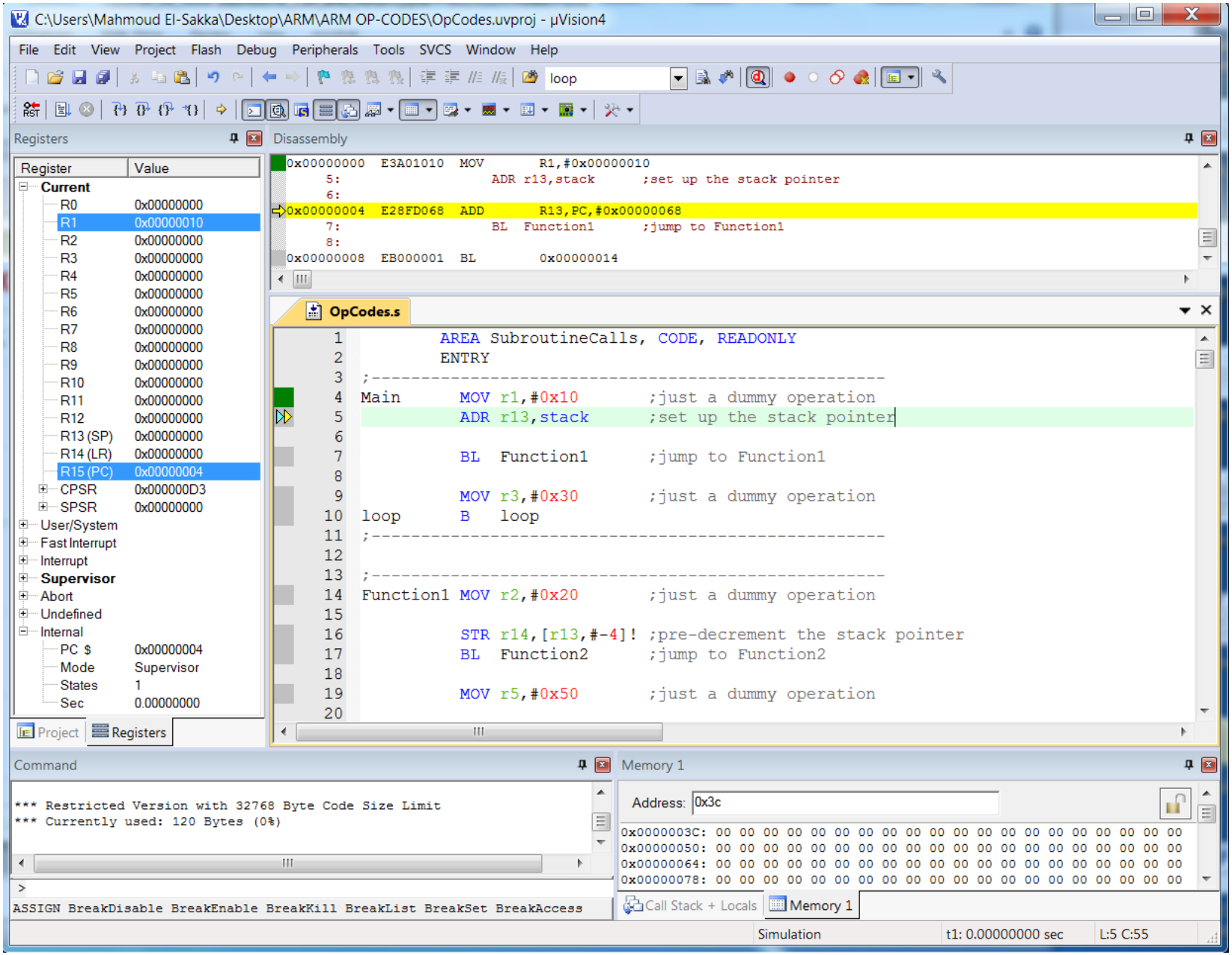
These two instructions can be combined in one instruction:

```
LDR r15,[r13],#4
```

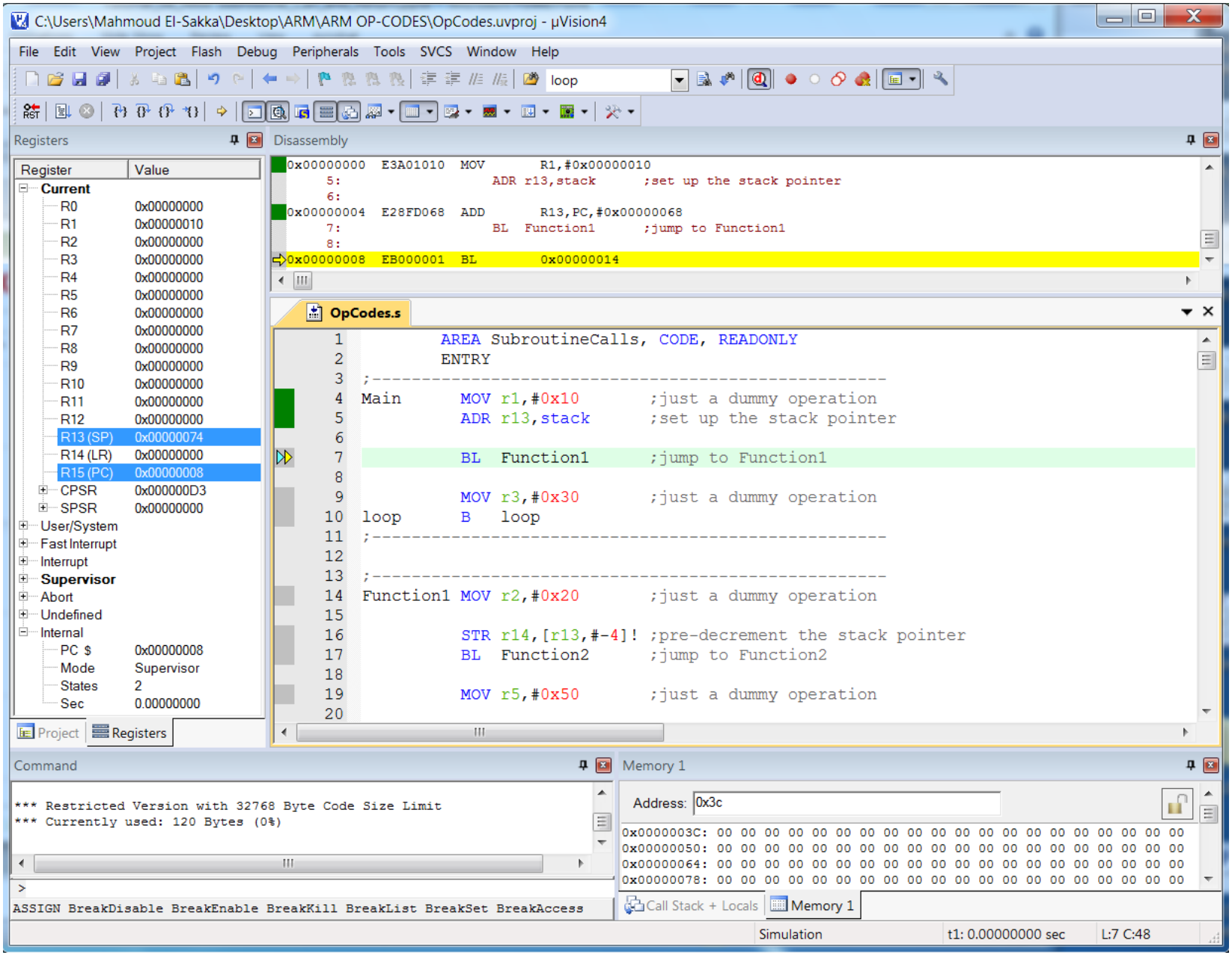
Example 5: Subroutine call and Return



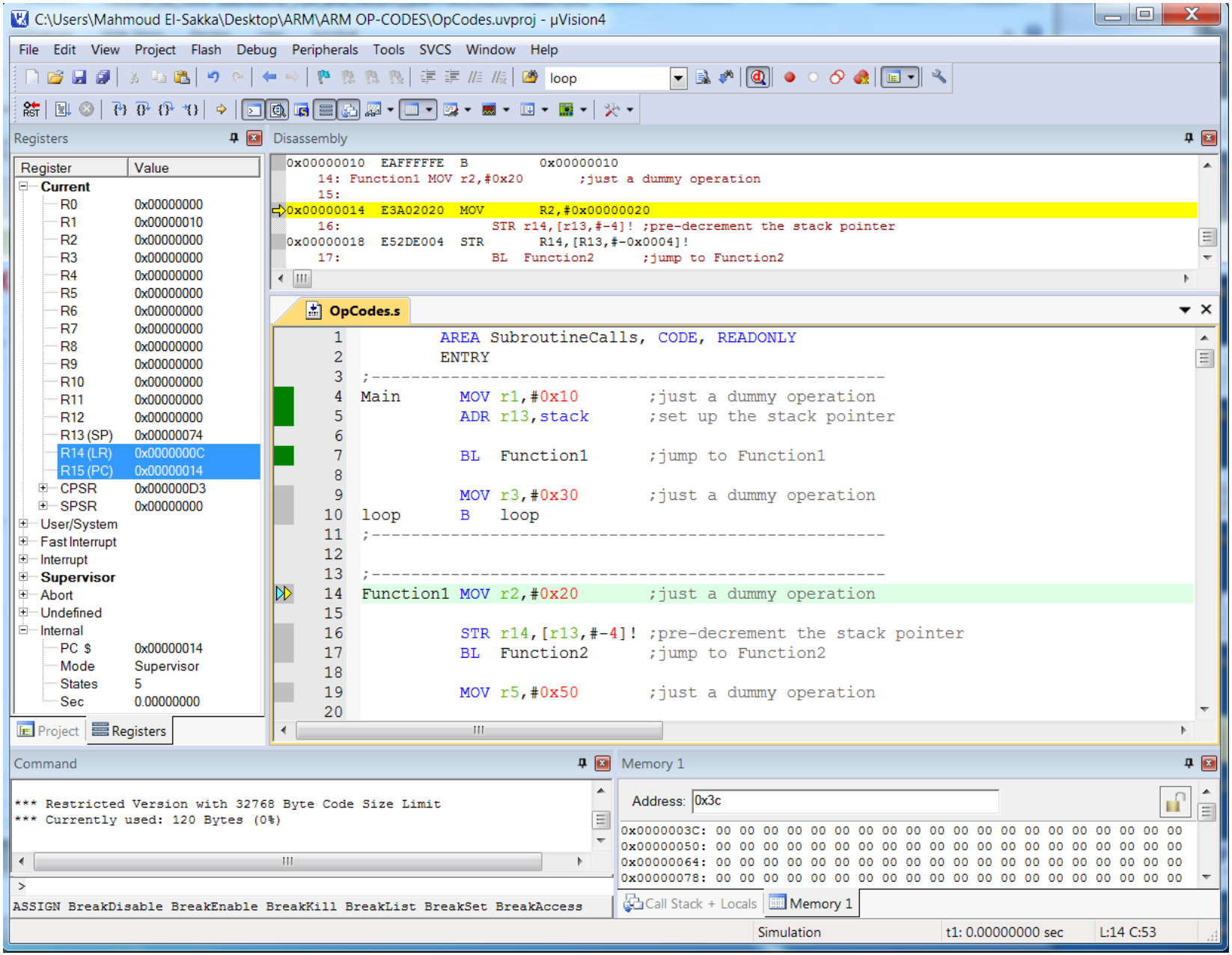
Example 5: Subroutine call and Return



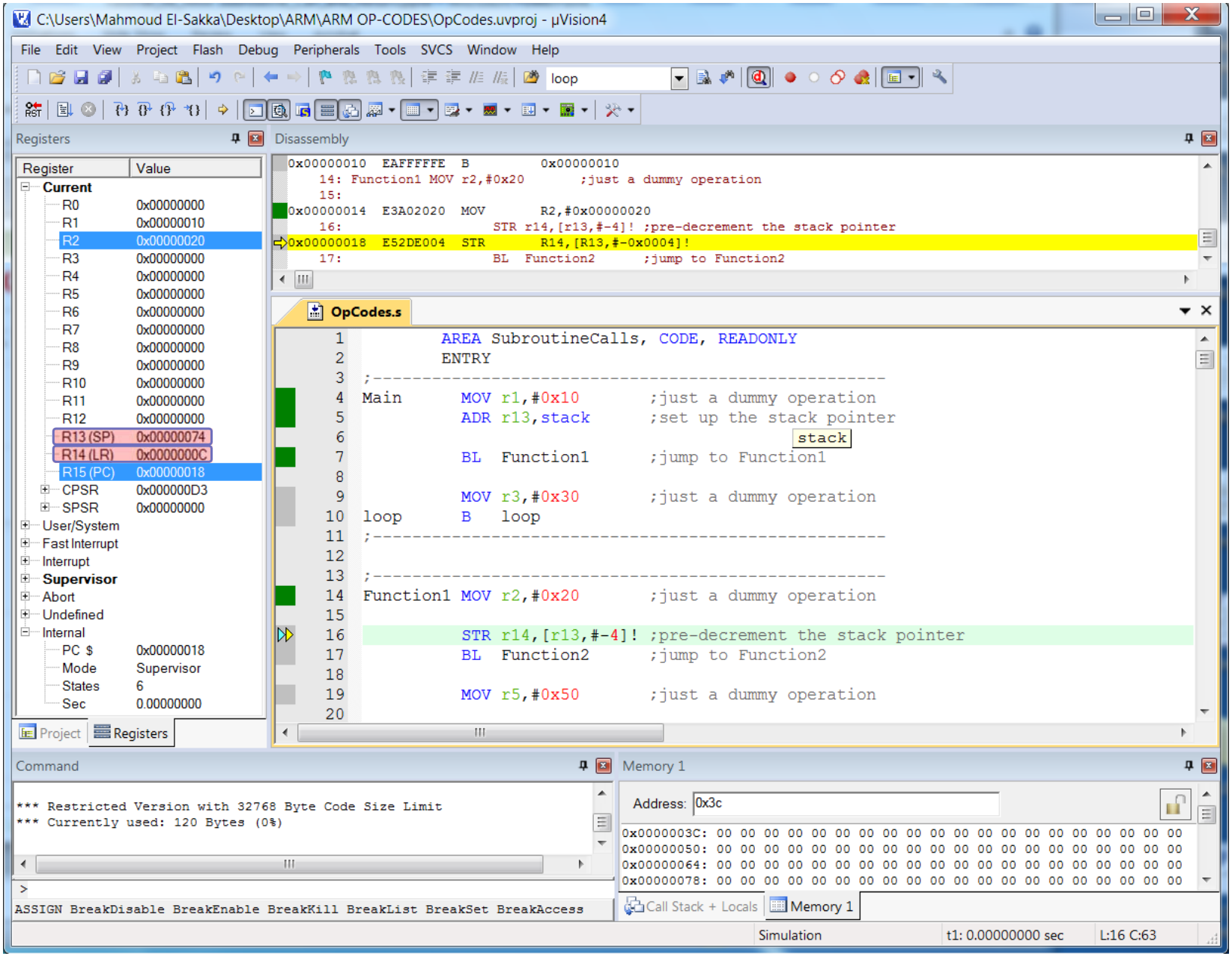
Example 5: Subroutine call and Return



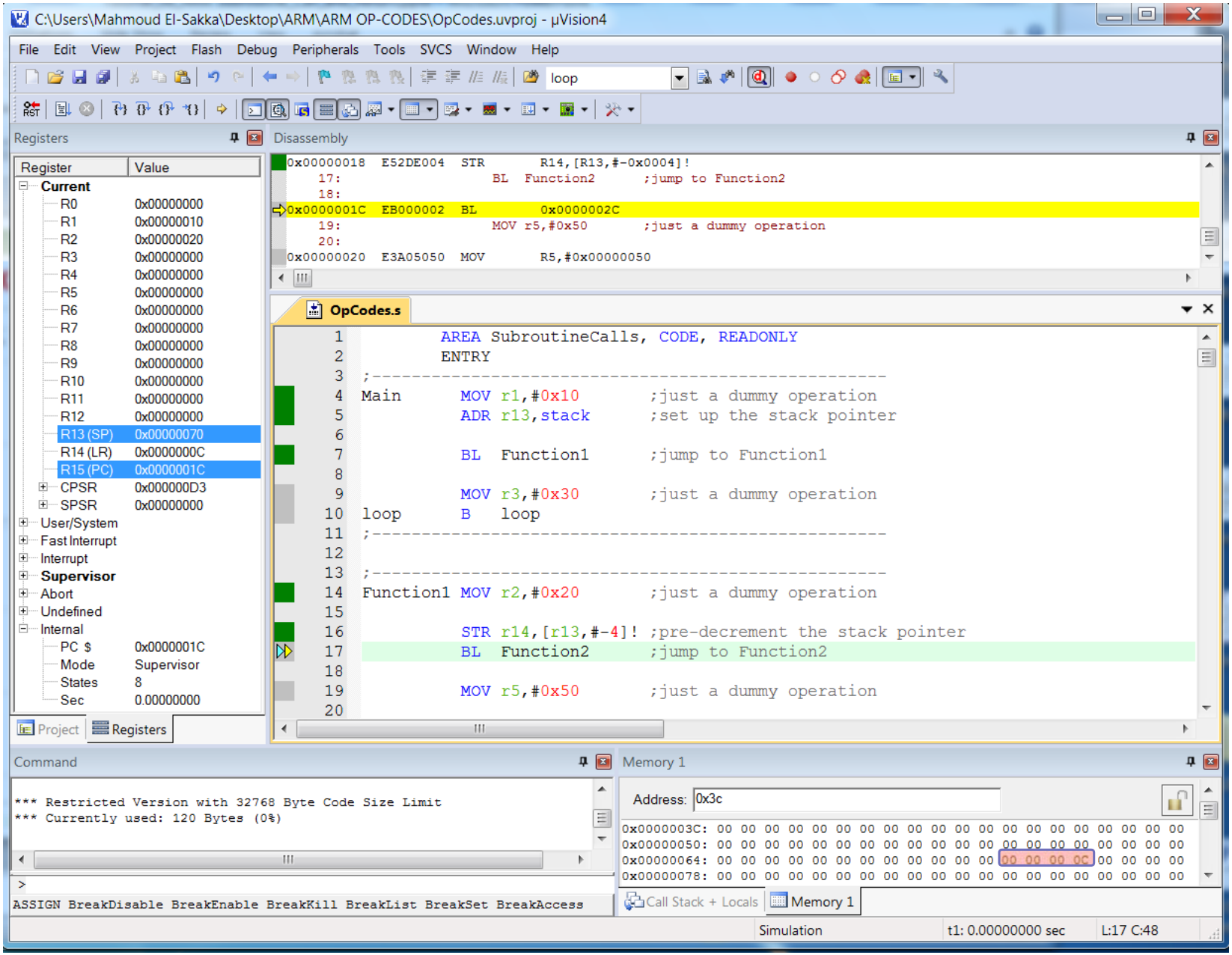
Example 5: Subroutine call and Return



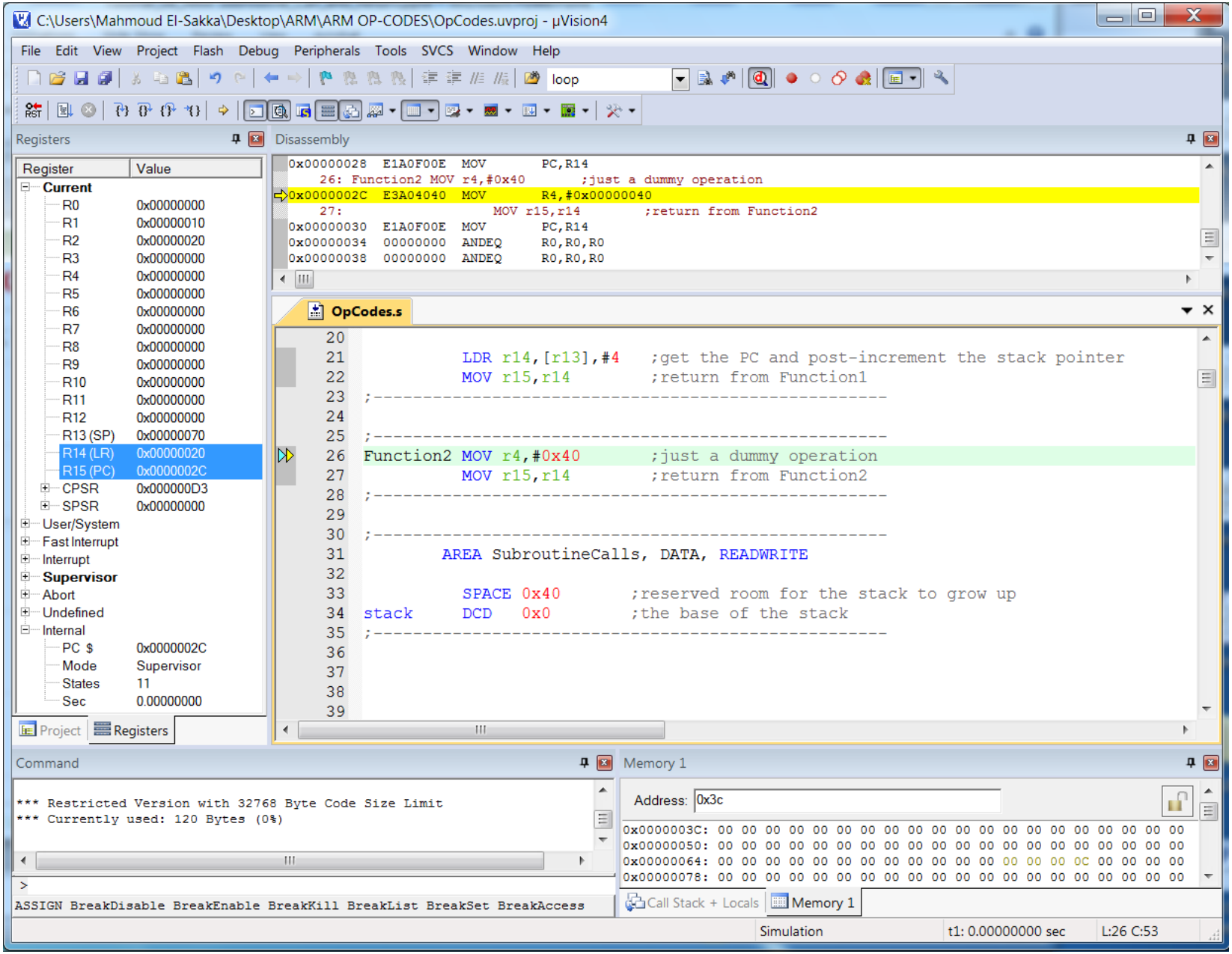
Example 5: Subroutine call and Return



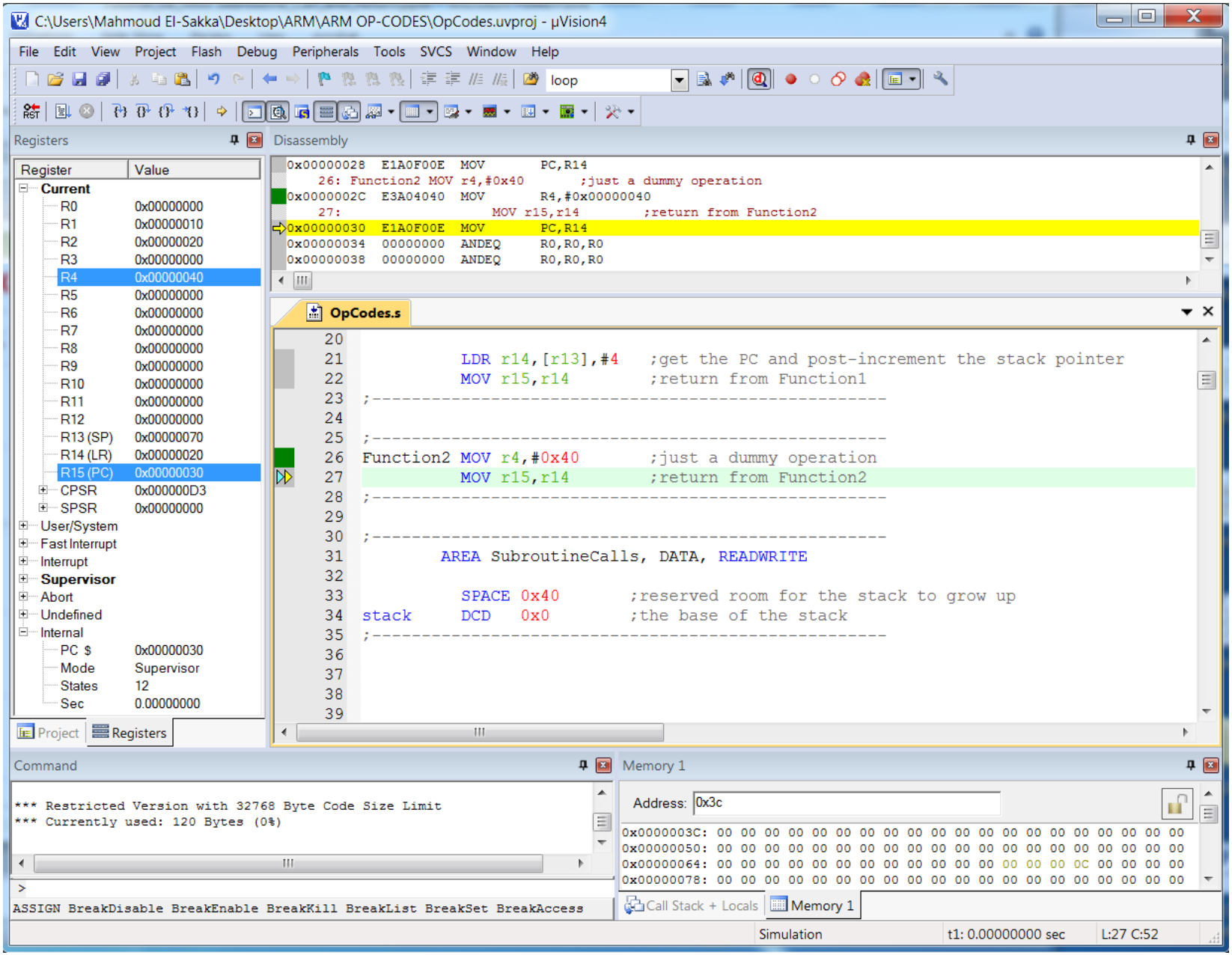
Example 5: Subroutine call and Return



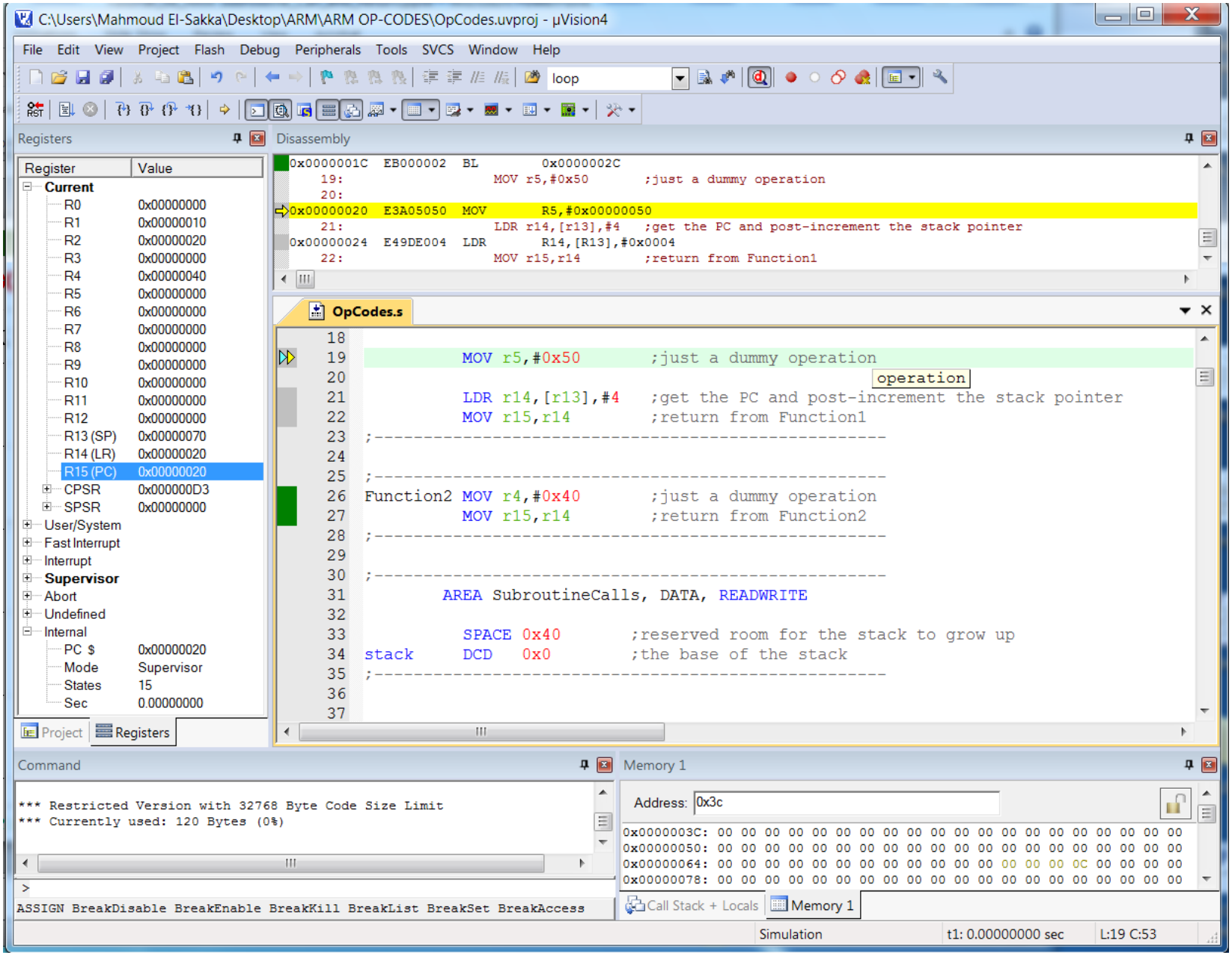
Example 5: Subroutine call and Return



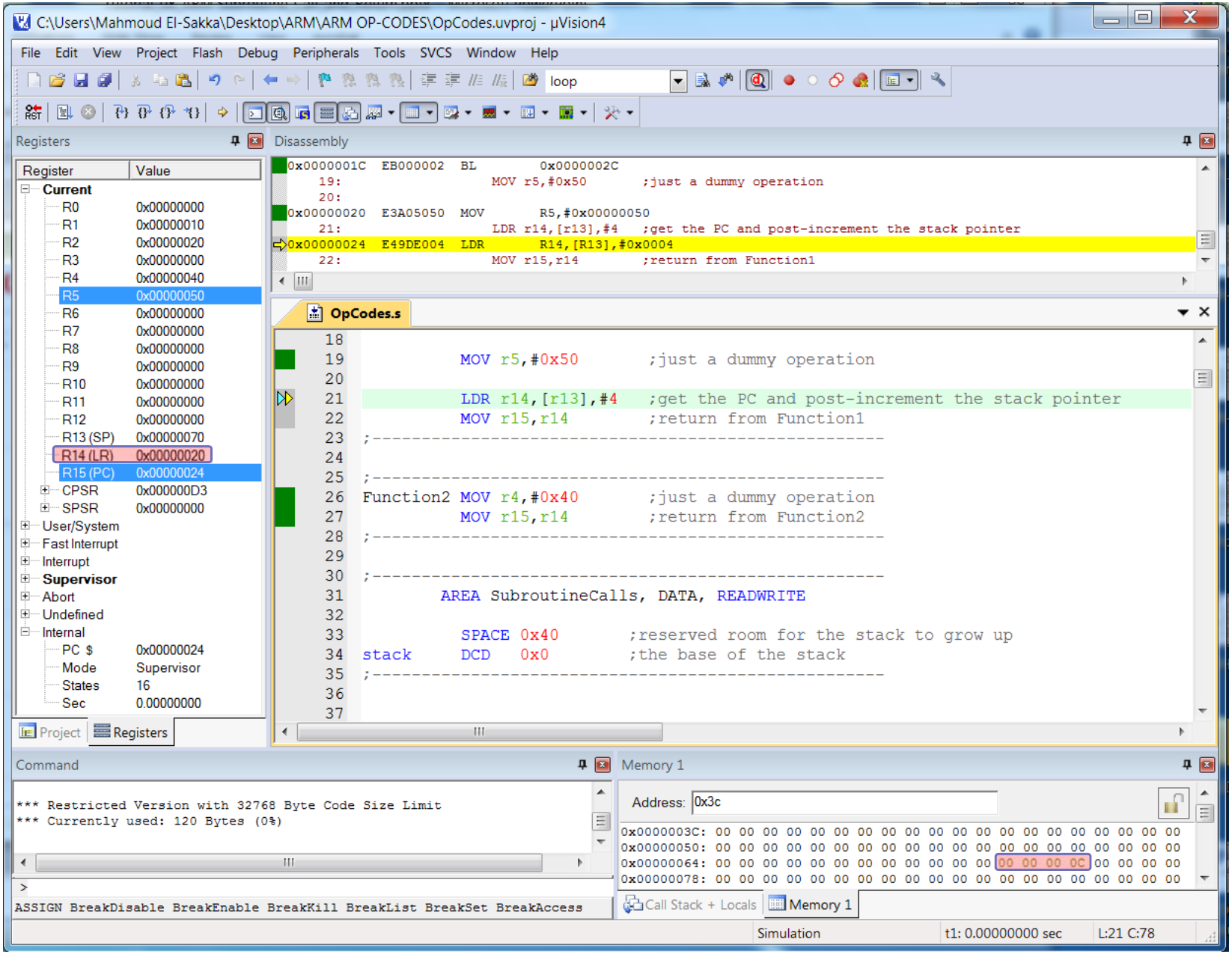
Example 5: Subroutine call and Return



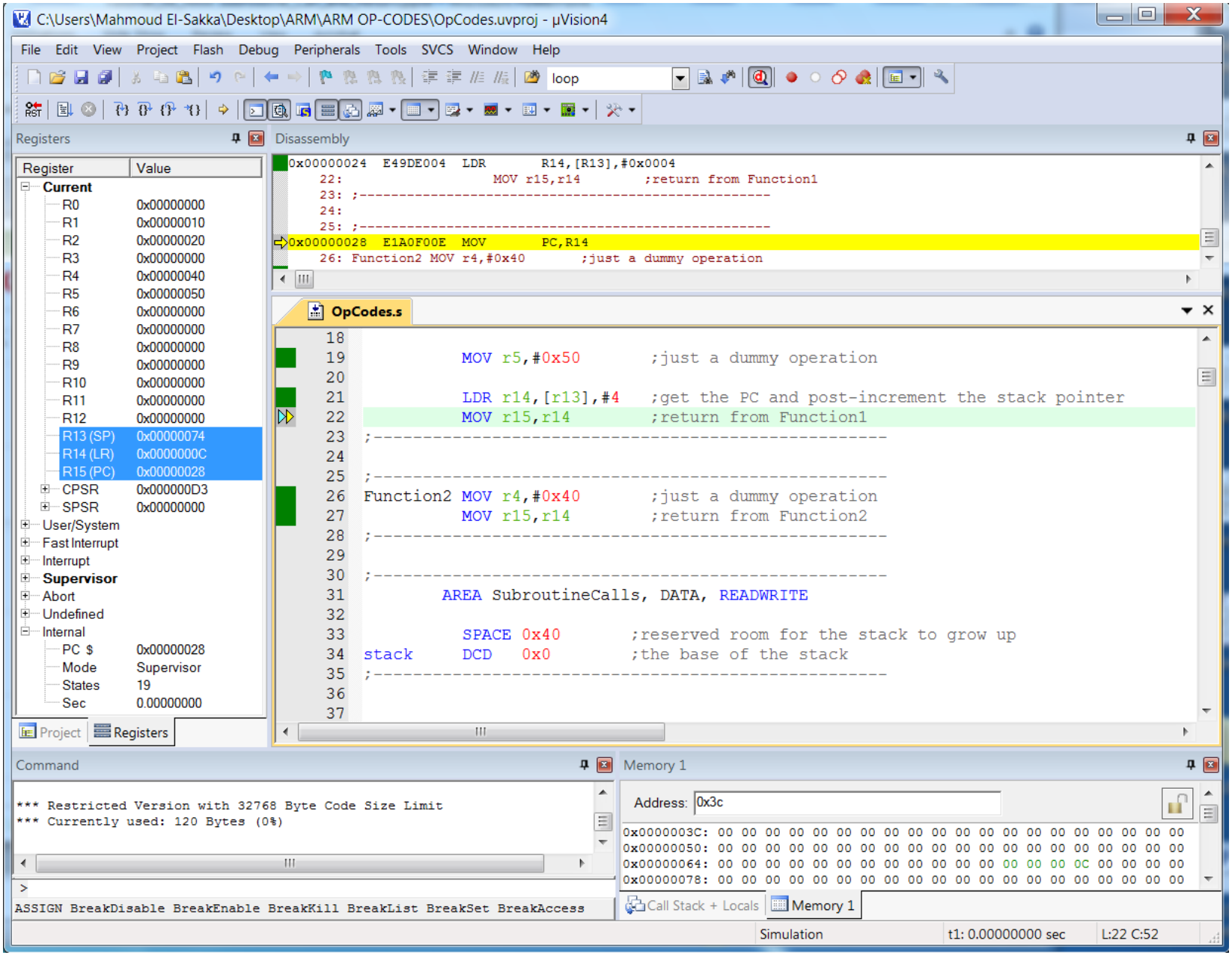
Example 5: Subroutine call and Return



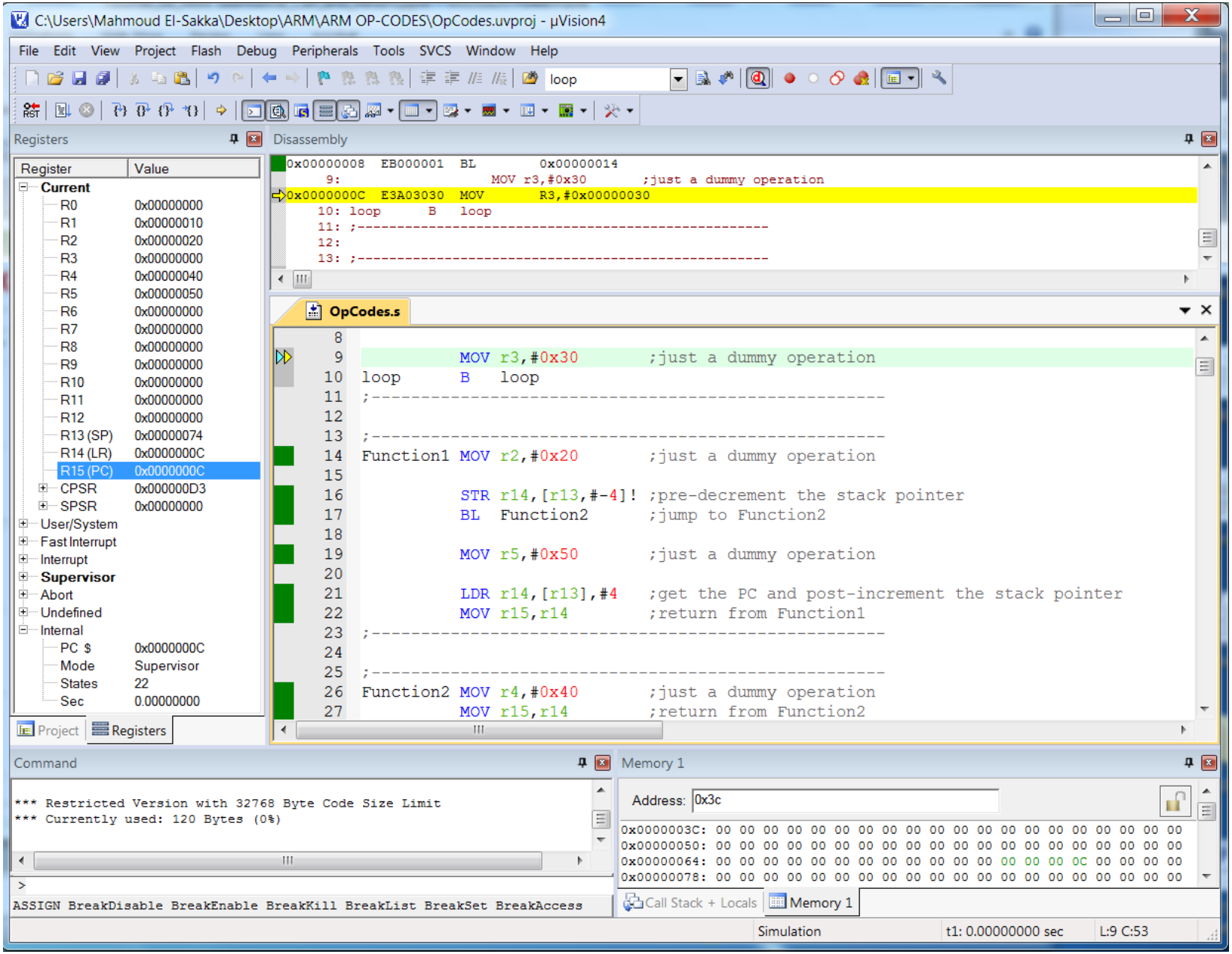
Example 5: Subroutine call and Return



Example 5: Subroutine call and Return



Example 5: Subroutine call and Return



Example 5: Subroutine call and Return