Tutorial 05: ARM Data Definition Directives and ARM Pseudo Instructions

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

Data Definition Directives

- Assembly language directives include:

AREA		To name a region of **code** or **data**
ENTRY		The execution starting point
END		The physical end of the program
name EQU	*value*	Equate a *name* to a *value* ***Will not make any memory allocation, i.e., Similar to `#define` in C***
{label} DCD	v. expr {, v. expr} …	Set up one or more 32-bit constant in memory *Must start at a multiple of 4 address location*
{label} DCW	v. expr {, v. expr} …	Set up one or more 16-bit constant in memory *Must start at an even address location*
{label} DCB	v. expr {, v. expr} …	Set up one or more 8-bit constant in memory *Can start anywhere*
{label} SPACE	size expr	Reserves a zeroed block of memory *Can start anywhere*
ALIGN		Ensures that next instruction is correctly aligned on 32-bit boundaries, i.e., to start at a multiple of 4 address location

Data Definition Directives

- Some symbols in Keil assembler have different meanings, based on their location within the instruction:
  - **Equal sign "="**
    - at the opcode column *means* **DCB**
    - as a prefix of the 2nd operand of an LDR instruction *means* pseudo instruction

    Example 1:

    ```
    XYZ =   0x41  ;the = sign in this context means DCB, i.e.,
    XYZ DCB 0x41
    ```

    What will happen if the "=" sign is omitted?

    Example 2:

    ```
        LDR r0,=0x12345678  ; to LDR the 32-bit value 0x12345678 into r0
        LDR r0,=PPP         ; to LDR the 32-bit address of PPP into r0
    ```

    the **=** sign in this context means the **LDR** here is a pseudo instruction

  - **Ampersand sign "&"**
    - at the opcode column *means* **DCD**
    - as a prefix of an operand *means* a HEX value of a ***single byte*** (i.e., similar to **0x**)

    Example 3:

    ```
    AAA &   0x123456  ;the & sign in this context means DCD, i.e.,
    AAA DCD 0x123456
    ```

    Example 4:

    ```
        MOV r0,#&8F   ;the & sign in this context means a HEX value of a single byte
    ```

  - **Percent sign "%"**
    - at the opcode column *means* **SPACE**

    Example 5:

    ```
    BBB %     0x40   ;the % sign in this context means SPACE, i.e.,
    BBB SPACE 0x40
    ```

Writing Numbers with Various Radix

- The Keil assembler uses
  - a prefix **0x** to indicate hexadecimal, e.g.,
    ```
    MOV r1, #0x9C
    ```
    or
    ```
    DCD 0x9C
    ```
  - a prefix **2_** to indicate binary, e.g.,
    ```
    MOV r1, #2_10011100
    ```
    or
    ```
    DCD 2_10011100
    ```
  - a prefix **8_** to indicate octal, e.g.,
    ```
    MOV r1, #8_234
    ```
    or
    ```
    DCD 8_234
    ```
  - **no** prefix to indicate decimal, e.g.,
    ```
    MOV r1, #156
    ```
    or
    ```
    DCD 156
    ```

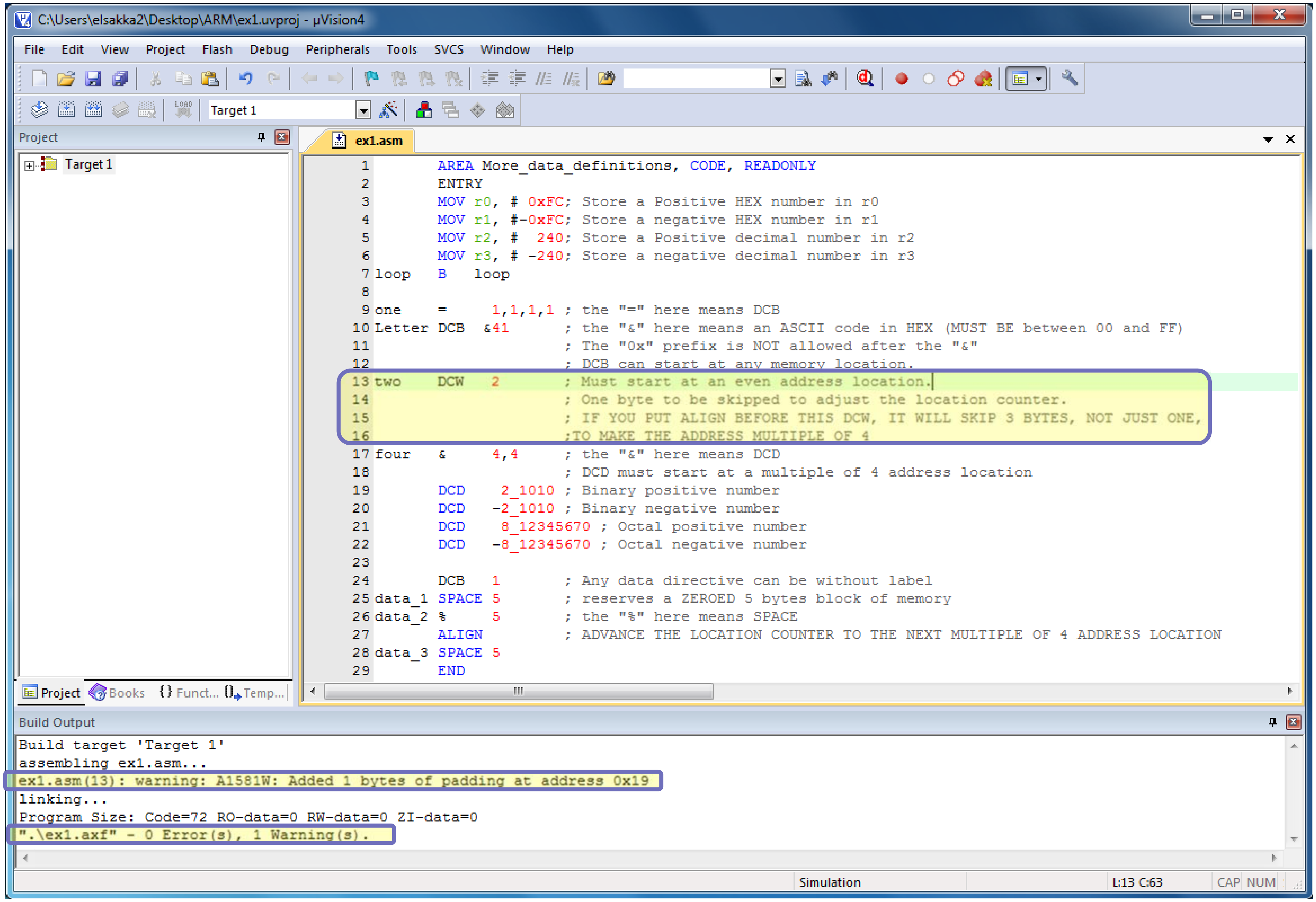
In ARM assembly,
the `"#"` means
Literal or immediate
addressing mode

In ARM assembly,
It is ***illegal*** to use `"#"` with
`DCD`, **`DCW`**, or **`DCB`**

Data Definition Directives

```
       AREA More_data_definitions, CODE, READONLY
       ENTRY
       MOV r0, # 0xFC; Store a Positive HEX number in r0
       MOV r1, #-0xFC; Store a negative HEX number in r1
       MOV r2, #  240; Store a Positive decimal number in r2
       MOV r3, # -240; Store a negative decimal number in r3
loop   B   loop

one    =   1,1,1,1 ; the "=" here means DCB
Letter DCB &41     ; the "&" here means an ASCII code in HEX (MUST BE between 00 and FF)
                   ; The "0x" prefix is NOT allowed after the "&"
                   ; DCB can start at any memory location.
two    DCW  2      ; Must start at an even address location.
                   ; One byte to be skipped to adjust the location counter.
                   ; IF YOU PUT ALIGN BEFORE THIS DCW, IT WILL SKIP 3 BYTES, NOT JUST ONE,
                   ; TO MAKE THE ADDRESS MULTIPLE OF 4
four   &     4,4   ; the "&" here means DCD
                   ; DCD must start at a multiple of 4 address location
       DCD   2_1010      ; Binary positive number
       DCD   -2_1010     ; Binary negative number
       DCD    8_12345670 ; Octal positive number
       DCD   -8_12345670 ; Octal negative number

       DCB   1           ; Any data directive can be without label
data_1 SPACE 5           ; reserves a ZEROED 5 bytes block of memory
data_2 %     5           ; the "%" here means SPACE
       ALIGN     ; ADVANCE THE LOCATION COUNTER TO THE NEXT MULTIPLE OF 4 ADDRESS LOCATION
data_3 SPACE 5
       END
```

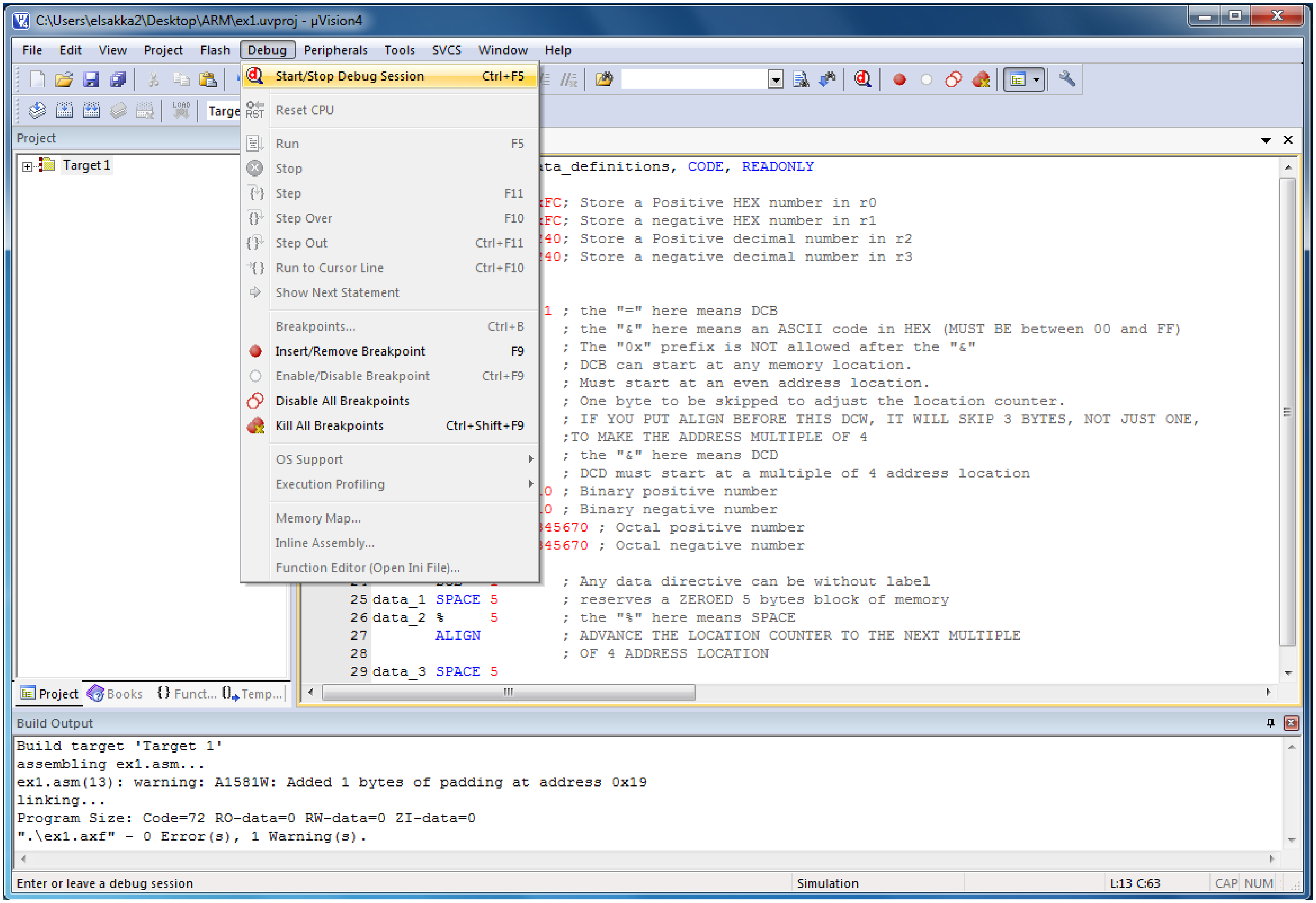
Data Definition Directives


C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Target 1
Project
Target 1
Project Books {} Funct... Temp...
ex1.asm
Build Output
Simulation
L:13 C:63
CAP NUM

```
 1        AREA More_data_definitions, CODE, READONLY
 2        ENTRY
 3        MOV r0, # 0xFC; Store a Positive HEX number in r0
 4        MOV r1, #-0xFC; Store a negative HEX number in r1
 5        MOV r2, #  240; Store a Positive decimal number in r2
 6        MOV r3, # -240; Store a negative decimal number in r3
 7 loop   B   loop
 8
 9 one    =     1,1,1,1 ; the "=" here means DCB
10 Letter DCB  &41      ; the "&" here means an ASCII code in HEX (MUST BE between 00 and FF)
11                      ; The "0x" prefix is NOT allowed after the "&"
12                      ; DCB can start at any memory location.
13 two    DCW   2       ; Must start at an even address location.
14                      ; One byte to be skipped to adjust the location counter.
15                      ; IF YOU PUT ALIGN BEFORE THIS DCW, IT WILL SKIP 3 BYTES, NOT JUST ONE,
16                      ;TO MAKE THE ADDRESS MULTIPLE OF 4
17 four   &     4,4     ; the "&" here means DCD
18                      ; DCD must start at a multiple of 4 address location
19        DCD    2_1010 ; Binary positive number
20        DCD   -2_1010 ; Binary negative number
21        DCD    8_12345670 ; Octal positive number
22        DCD   -8_12345670 ; Octal negative number
23
24        DCB   1       ; Any data directive can be without label
25 data_1 SPACE 5       ; reserves a ZEROED 5 bytes block of memory
26 data_2 %     5       ; the "%" here means SPACE
27        ALIGN         ; ADVANCE THE LOCATION COUNTER TO THE NEXT MULTIPLE OF 4 ADDRESS LOCATION
28 data_3 SPACE 5
29        END
```

```
Build target 'Target 1'
assembling ex1.asm...
ex1.asm(13): warning: A1581W: Added 1 bytes of padding at address 0x19
linking...
Program Size: Code=72 RO-data=0 RW-data=0 ZI-data=0
".\ex1.axf" - 0 Error(s), 1 Warning(s).
```

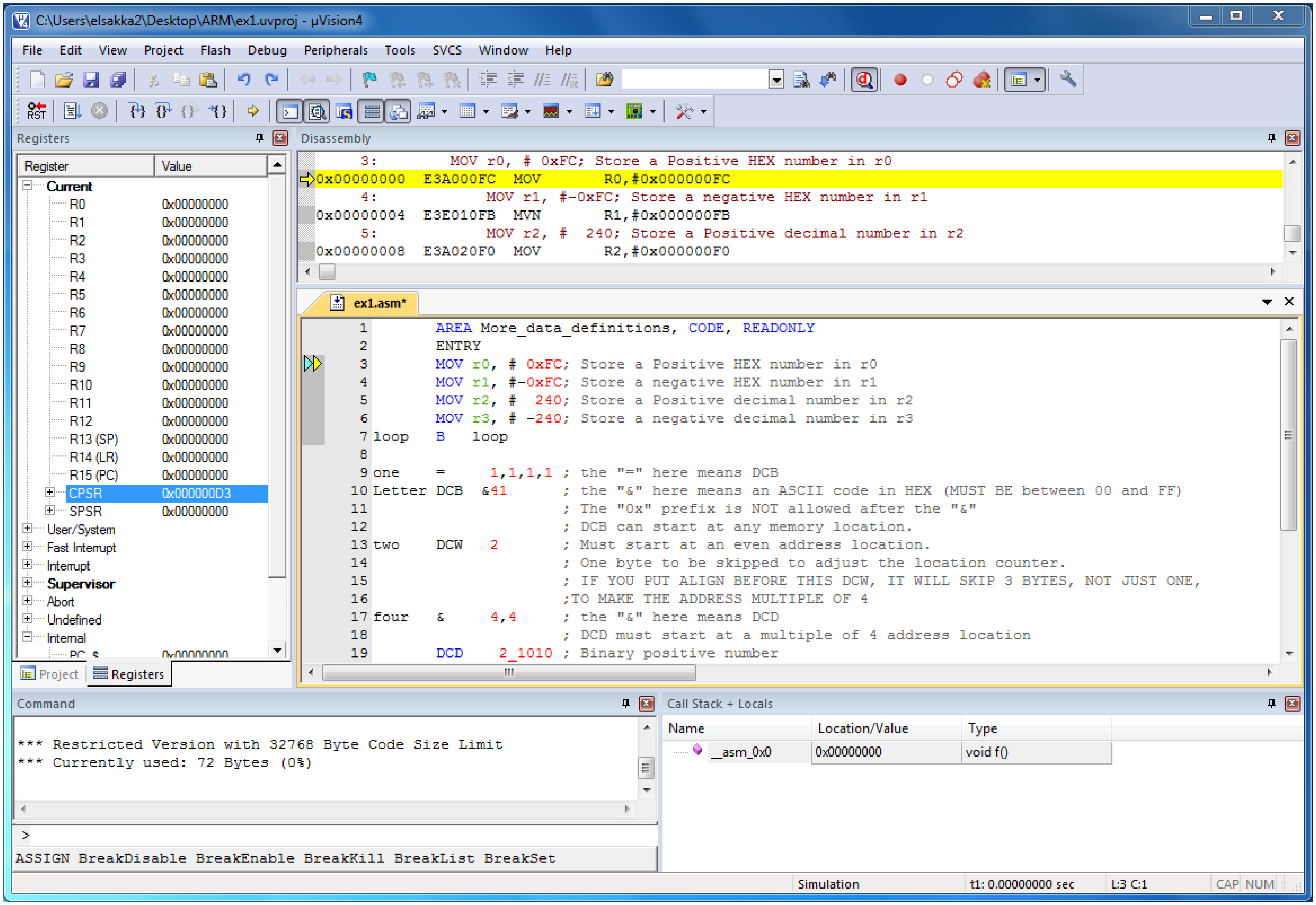
Data Definition Directives


C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Start/Stop Debug Session Ctrl+F5
Reset CPU
Run F5
Stop
Step F11
Step Over F10
Step Out Ctrl+F11
Run to Cursor Line Ctrl+F10
Show Next Statement
Breakpoints... Ctrl+B
Insert/Remove Breakpoint F9
Enable/Disable Breakpoint Ctrl+F9
Disable All Breakpoints
Kill All Breakpoints Ctrl+Shift+F9
OS Support
Execution Profiling
Memory Map...
Inline Assembly...
Function Editor (Open Ini File)...
Project
Target 1
Project Books Funct... Temp...
Build Output
Enter or leave a debug session
Simulation
L:13 C:63
CAP NUM

```
ta_definitions, CODE, READONLY

FC; Store a Positive HEX number in r0
FC; Store a negative HEX number in r1
40; Store a Positive decimal number in r2
40; Store a negative decimal number in r3


1  ; the "=" here means DCB
   ; the "&" here means an ASCII code in HEX (MUST BE between 00 and FF)
   ; The "0x" prefix is NOT allowed after the "&"
   ; DCB can start at any memory location.
   ; Must start at an even address location.
   ; One byte to be skipped to adjust the location counter.
   ; IF YOU PUT ALIGN BEFORE THIS DCW, IT WILL SKIP 3 BYTES, NOT JUST ONE,
   ;TO MAKE THE ADDRESS MULTIPLE OF 4
   ; the "&" here means DCD
   ; DCD must start at a multiple of 4 address location
0  ; Binary positive number
0  ; Binary negative number
45670 ; Octal positive number
45670 ; Octal negative number

         ; Any data directive can be without label
data_1 SPACE 5      ; reserves a ZEROED 5 bytes block of memory
data_2 %     5      ; the "%" here means SPACE
       ALIGN        ; ADVANCE THE LOCATION COUNTER TO THE NEXT MULTIPLE
                    ; OF 4 ADDRESS LOCATION
data_3 SPACE 5
```

```
Build target 'Target 1'
assembling ex1.asm...
ex1.asm(13): warning: A1581W: Added 1 bytes of padding at address 0x19
linking...
Program Size: Code=72 RO-data=0 RW-data=0 ZI-data=0
".\ex1.axf" - 0 Error(s), 1 Warning(s).
```

Data Definition Directives


C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4

Registers

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)	0x00000000
CPSR	0x000000D3
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x00000000

Disassembly

```
     3:         MOV r0, # 0xFC; Store a Positive HEX number in r0
0x00000000  E3A000FC  MOV      R0,#0x000000FC
     4:             MOV r1, #-0xFC; Store a negative HEX number in r1
0x00000004  E3E010FB  MVN      R1,#0x000000FB
     5:             MOV r2, #  240; Store a Positive decimal number in r2
0x00000008  E3A020F0  MOV      R2,#0x000000F0
```

ex1.asm*

```
        AREA More_data_definitions, CODE, READONLY
        ENTRY
        MOV r0, # 0xFC; Store a Positive HEX number in r0
        MOV r1, #-0xFC; Store a negative HEX number in r1
        MOV r2, #  240; Store a Positive decimal number in r2
        MOV r3, # -240; Store a negative decimal number in r3
loop    B   loop

one     =      1,1,1,1 ; the "=" here means DCB
Letter DCB   &41       ; the "&" here means an ASCII code in HEX (MUST BE between 00 and FF)
                       ; The "0x" prefix is NOT allowed after the "&"
                       ; DCB can start at any memory location.
two     DCW    2       ; Must start at an even address location.
                       ; One byte to be skipped to adjust the location counter.
                       ; IF YOU PUT ALIGN BEFORE THIS DCW, IT WILL SKIP 3 BYTES, NOT JUST ONE,
                       ;TO MAKE THE ADDRESS MULTIPLE OF 4
four    &      4,4     ; the "&" here means DCD
                       ; DCD must start at a multiple of 4 address location
        DCD     2_1010 ; Binary positive number
```

Command

```
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 72 Bytes (0%)

>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet
```

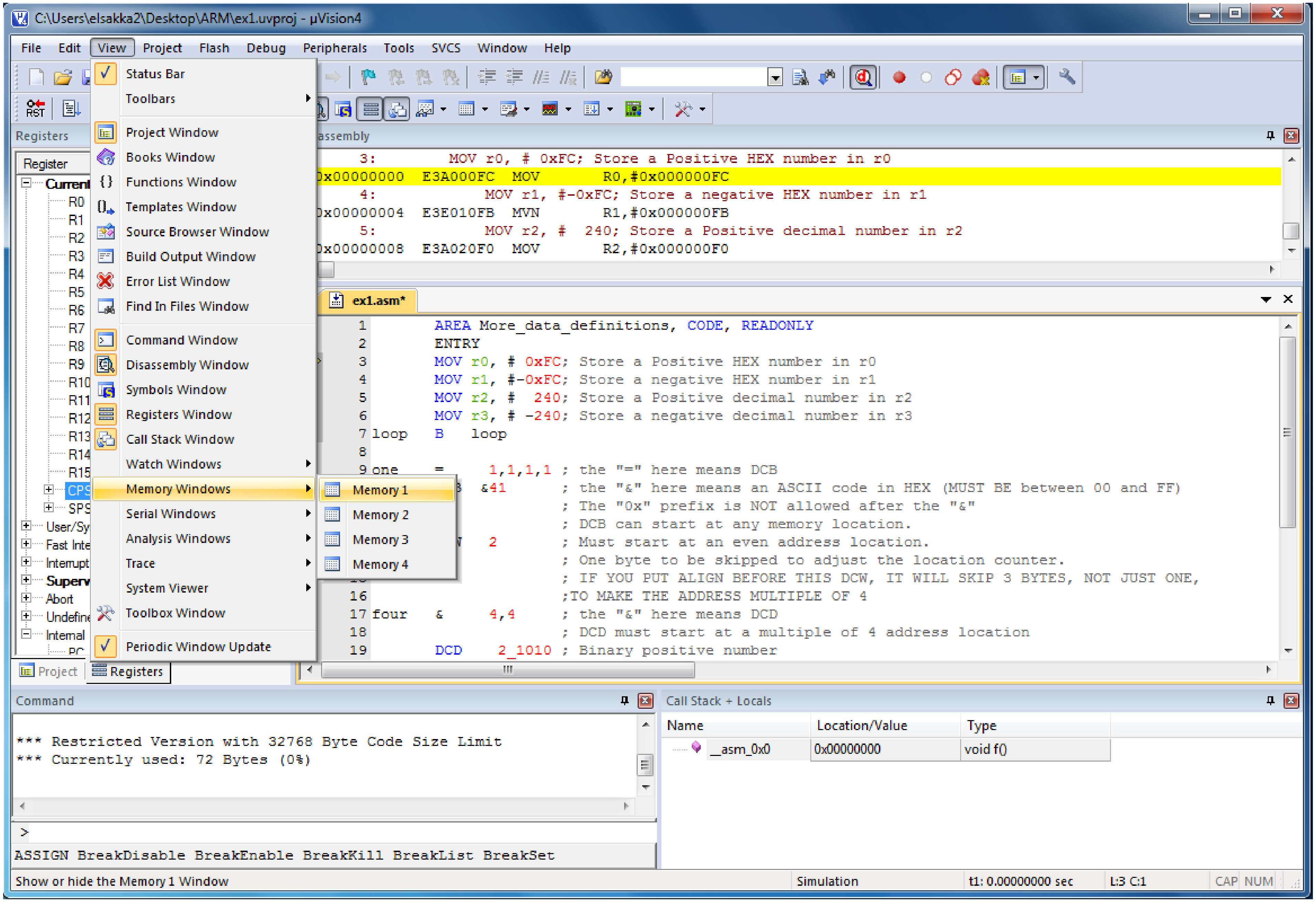
Call Stack + Locals

Name	Location/Value	Type
_asm_0x0	0x00000000	void f()

Simulation t1: 0.00000000 sec L:3 C:1 CAP NUM

File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Project
Registers

Data Definition Directives

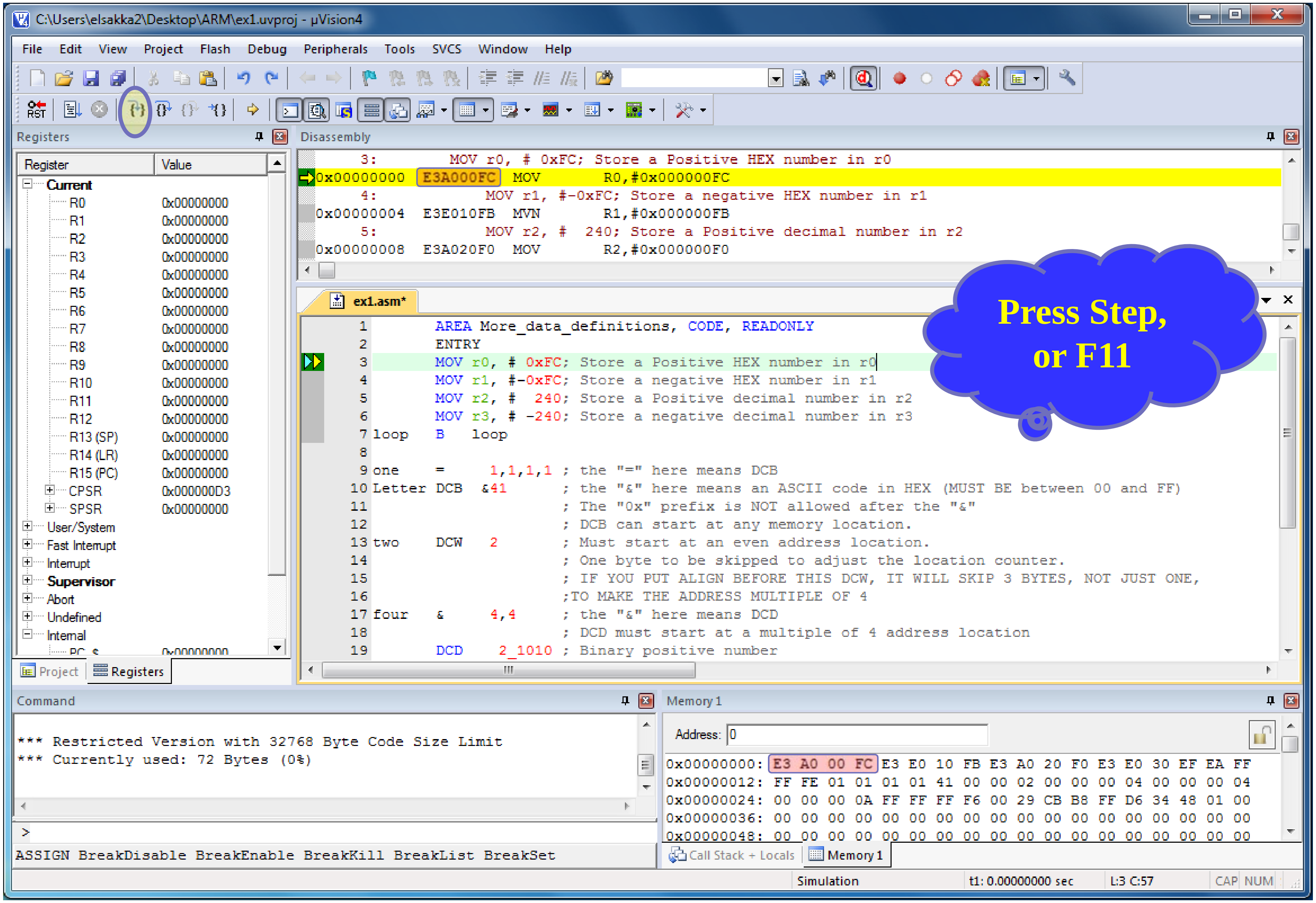


```
     3:          MOV r0, # 0xFC; Store a Positive HEX number in r0
0x00000000  E3A000FC  MOV      R0,#0x000000FC
     4:              MOV r1, #-0xFC; Store a negative HEX number in r1
0x00000004  E3E010FB  MVN      R1,#0x000000FB
     5:              MOV r2, #  240; Store a Positive decimal number in r2
0x00000008  E3A020F0  MOV      R2,#0x000000F0
```

```
        AREA More_data_definitions, CODE, READONLY
        ENTRY
        MOV r0, # 0xFC; Store a Positive HEX number in r0
        MOV r1, #-0xFC; Store a negative HEX number in r1
        MOV r2, #  240; Store a Positive decimal number in r2
        MOV r3, # -240; Store a negative decimal number in r3
loop    B   loop

one     =      1,1,1,1 ; the "=" here means DCB
      B    &41         ; the "&" here means an ASCII code in HEX (MUST BE between 00 and FF)
                       ; The "0x" prefix is NOT allowed after the "&"
                       ; DCB can start at any memory location.
      W    2           ; Must start at an even address location.
                       ; One byte to be skipped to adjust the location counter.
                       ; IF YOU PUT ALIGN BEFORE THIS DCW, IT WILL SKIP 3 BYTES, NOT JUST ONE,
                       ;TO MAKE THE ADDRESS MULTIPLE OF 4
four    &      4,4     ; the "&" here means DCD
                       ; DCD must start at a multiple of 4 address location
        DCD    2_1010  ; Binary positive number
```

```
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 72 Bytes (0%)

>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet
```

Data Definition Directives



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)	0x00000000
CPSR	0x000000D3
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	

```
     3:         MOV r0, # 0xFC; Store a Positive HEX number in r0
0x00000000  E3A000FC  MOV       R0,#0x000000FC
     4:         MOV r1, #-0xFC; Store a negative HEX number in r1
0x00000004  E3E010FB  MVN       R1,#0x000000FB
     5:         MOV r2, #  240; Store a Positive decimal number in r2
0x00000008  E3A020F0  MOV       R2,#0x000000F0
```

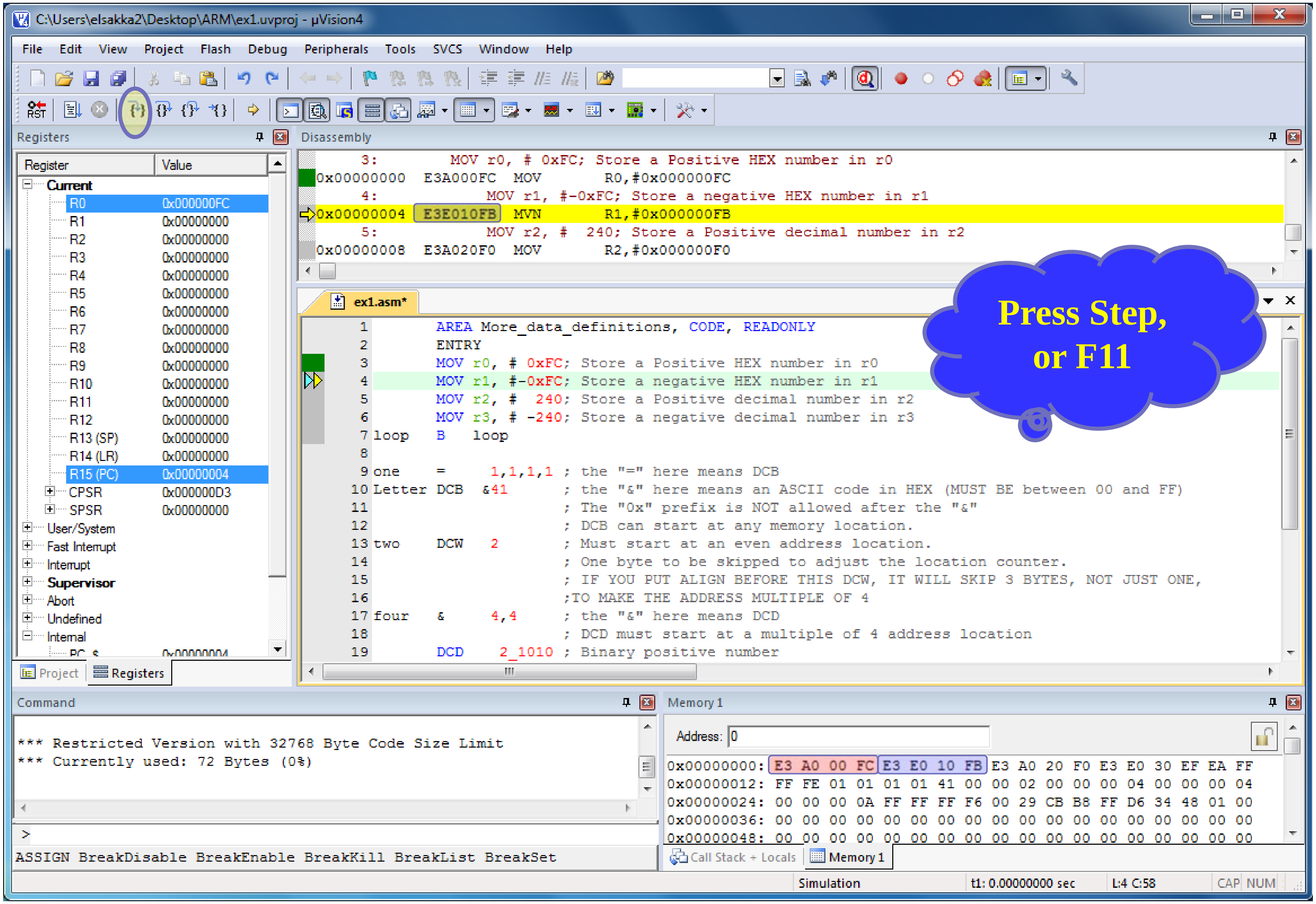
```
        AREA More_data_definitions, CODE, READONLY
        ENTRY
        MOV r0, # 0xFC; Store a Positive HEX number in r0
        MOV r1, #-0xFC; Store a negative HEX number in r1
        MOV r2, #  240; Store a Positive decimal number in r2
        MOV r3, # -240; Store a negative decimal number in r3
loop    B    loop

one     =       1,1,1,1  ; the "=" here means DCB
Letter  DCB    &41       ; the "&" here means an ASCII code in HEX (MUST BE between 00 and FF)
                         ; The "0x" prefix is NOT allowed after the "&"
                         ; DCB can start at any memory location.
two     DCW     2        ; Must start at an even address location.
                         ; One byte to be skipped to adjust the location counter.
                         ; IF YOU PUT ALIGN BEFORE THIS DCW, IT WILL SKIP 3 BYTES, NOT JUST ONE,
                         ;TO MAKE THE ADDRESS MULTIPLE OF 4
four    &       4,4      ; the "&" here means DCD
                         ; DCD must start at a multiple of 4 address location
        DCD     2_1010   ; Binary positive number
```

Press Step, or F11

```
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 72 Bytes (0%)

>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet
```

```
Address: 0
0x00000000: E3 A0 00 FC E3 E0 10 FB E3 A0 20 F0 E3 E0 30 EF EA FF
0x00000012: FF FE 01 01 01 01 41 00 00 02 00 00 00 04 00 00 00 04
0x00000024: 00 00 00 0A FF FF FF F6 00 29 CB B8 FF D6 34 48 01 00
0x00000036: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00000048: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Data Definition Directives



Register	Value
Current	
R0	0x000000FC
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)	0x00000004
CPSR	0x000000D3
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	

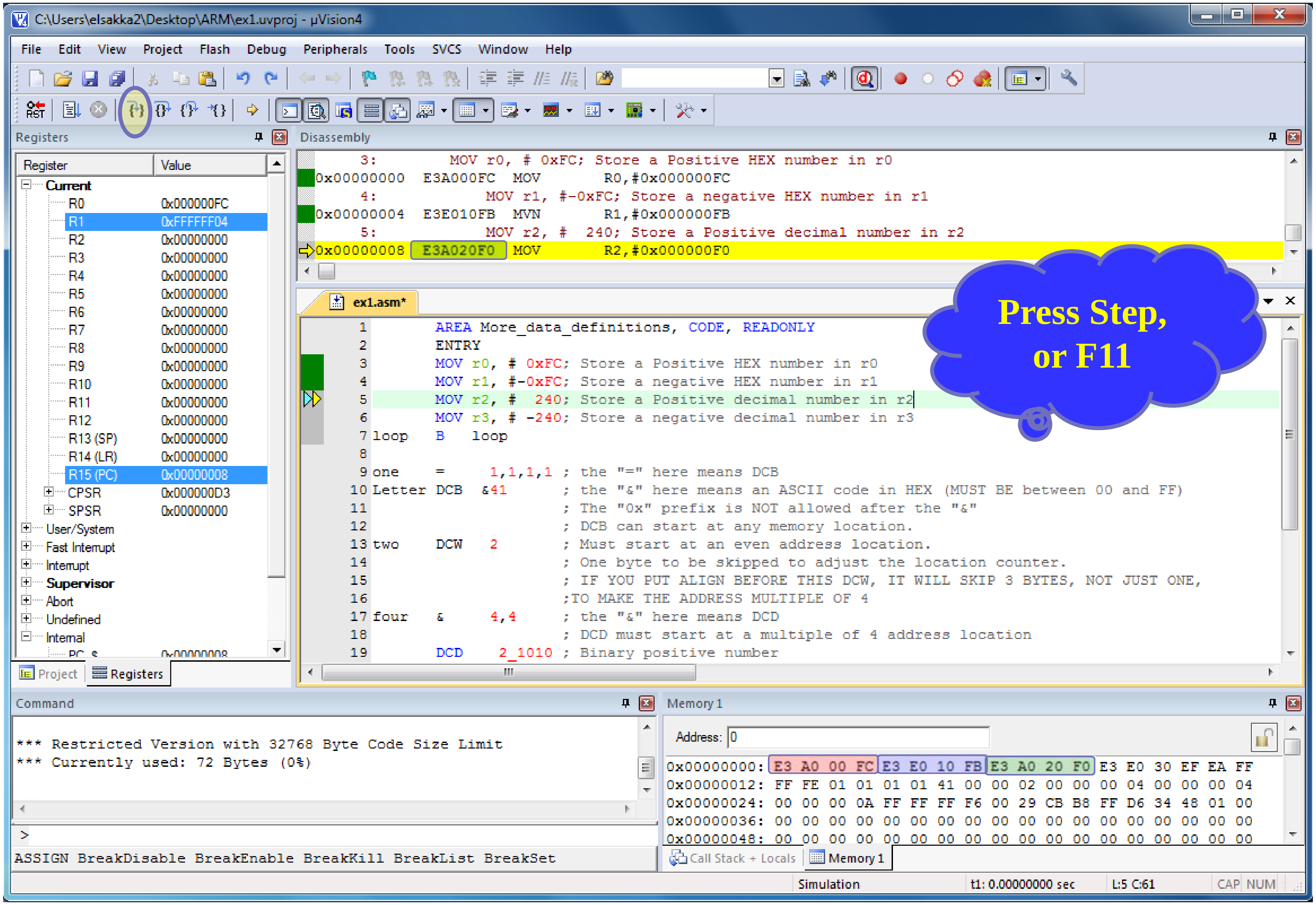
```
      3:          MOV r0, # 0xFC; Store a Positive HEX number in r0
0x00000000  E3A000FC  MOV       R0,#0x000000FC
      4:              MOV r1, #-0xFC; Store a negative HEX number in r1
0x00000004  E3E010FB  MVN       R1,#0x000000FB
      5:              MOV r2, #  240; Store a Positive decimal number in r2
0x00000008  E3A020F0  MOV       R2,#0x000000F0
```

```
        AREA More_data_definitions, CODE, READONLY
        ENTRY
        MOV r0, # 0xFC; Store a Positive HEX number in r0
        MOV r1, #-0xFC; Store a negative HEX number in r1
        MOV r2, #  240; Store a Positive decimal number in r2
        MOV r3, # -240; Store a negative decimal number in r3
loop    B    loop

one     =       1,1,1,1 ; the "=" here means DCB
Letter DCB    &41       ; the "&" here means an ASCII code in HEX (MUST BE between 00 and FF)
                        ; The "0x" prefix is NOT allowed after the "&"
                        ; DCB can start at any memory location.
two     DCW     2       ; Must start at an even address location.
                        ; One byte to be skipped to adjust the location counter.
                        ; IF YOU PUT ALIGN BEFORE THIS DCW, IT WILL SKIP 3 BYTES, NOT JUST ONE,
                        ;TO MAKE THE ADDRESS MULTIPLE OF 4
four    &       4,4     ; the "&" here means DCD
                        ; DCD must start at a multiple of 4 address location
        DCD     2_1010  ; Binary positive number
```

```
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 72 Bytes (0%)
>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet
```

```
Address: 0
0x00000000: E3 A0 00 FC E3 E0 10 FB E3 A0 20 F0 E3 E0 30 EF EA FF
0x00000012: FF FE 01 01 01 01 41 00 00 02 00 00 00 04 00 00 00 04
0x00000024: 00 00 00 0A FF FF FF F6 00 29 CB B8 FF D6 34 48 01 00
0x00000036: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00000048: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Data Definition Directives


C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Disassembly
ex1.asm*
Press Step, or F11
Project
Registers
Command
Memory 1
Call Stack + Locals
Memory 1

Register	Value
Current	
R0	0x000000FC
R1	0xFFFFFF04
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)	0x00000008
CPSR	0x000000D3
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x00000008

```
     3:           MOV r0, # 0xFC; Store a Positive HEX number in r0
0x00000000  E3A000FC  MOV       R0,#0x000000FC
     4:               MOV r1, #-0xFC; Store a negative HEX number in r1
0x00000004  E3E010FB  MVN       R1,#0x000000FB
     5:               MOV r2, #  240; Store a Positive decimal number in r2
0x00000008  E3A020F0  MOV       R2,#0x000000F0
```

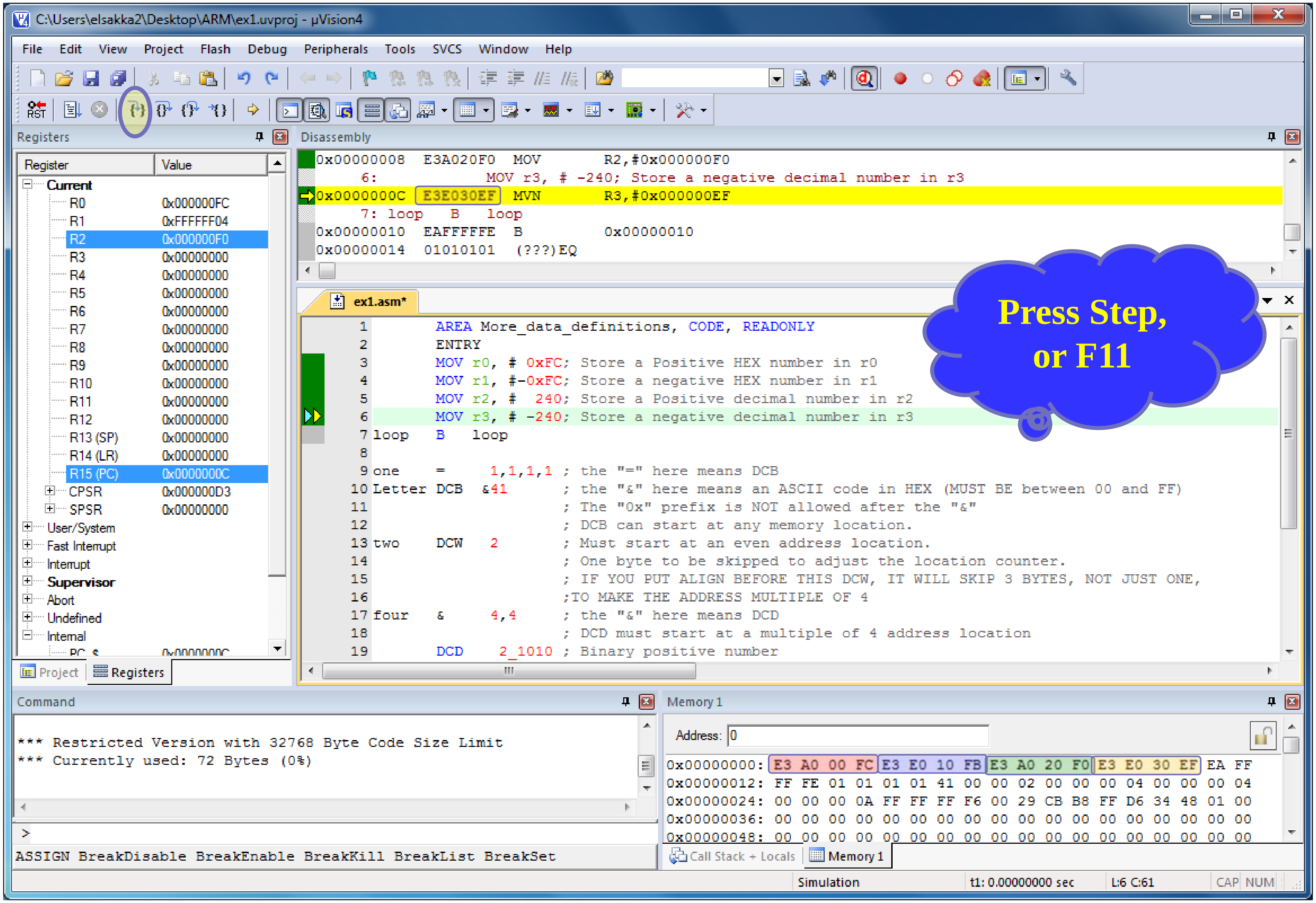
```
        AREA More_data_definitions, CODE, READONLY
        ENTRY
        MOV r0, # 0xFC; Store a Positive HEX number in r0
        MOV r1, #-0xFC; Store a negative HEX number in r1
        MOV r2, #  240; Store a Positive decimal number in r2
        MOV r3, # -240; Store a negative decimal number in r3
loop    B    loop

one     =       1,1,1,1  ; the "=" here means DCB
Letter  DCB    &41       ; the "&" here means an ASCII code in HEX (MUST BE between 00 and FF)
                         ; The "0x" prefix is NOT allowed after the "&"
                         ; DCB can start at any memory location.
two     DCW     2        ; Must start at an even address location.
                         ; One byte to be skipped to adjust the location counter.
                         ; IF YOU PUT ALIGN BEFORE THIS DCW, IT WILL SKIP 3 BYTES, NOT JUST ONE,
                         ;TO MAKE THE ADDRESS MULTIPLE OF 4
four    &       4,4      ; the "&" here means DCD
                         ; DCD must start at a multiple of 4 address location
        DCD     2_1010 ; Binary positive number
```

```
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 72 Bytes (0%)

>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet
```

```
Address: 0
0x00000000: E3 A0 00 FC E3 E0 10 FB E3 A0 20 F0 E3 E0 30 EF EA FF
0x00000012: FF FE 01 01 01 01 41 00 00 02 00 00 00 04 00 00 00 04
0x00000024: 00 00 00 0A FF FF FF F6 00 29 CB B8 FF D6 34 48 01 00
0x00000036: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00000048: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Simulation   t1: 0.00000000 sec   L:5 C:61   CAP NUM

Data Definition Directives


C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Disassembly
ex1.asm*
Press Step, or F11
Project
Registers
Command
Memory 1
Call Stack + Locals

Register	Value
Current	
R0	0x000000FC
R1	0xFFFFFF04
R2	0x000000F0
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)	0x0000000C
CPSR	0x000000D3
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	

```
0x00000008  E3A020F0  MOV       R2,#0x000000F0
     6:            MOV r3, # -240; Store a negative decimal number in r3
0x0000000C  E3E030EF  MVN       R3,#0x000000EF
     7: loop    B    loop
0x00000010  EAFFFFFE  B         0x00000010
0x00000014  01010101  (???)EQ
```

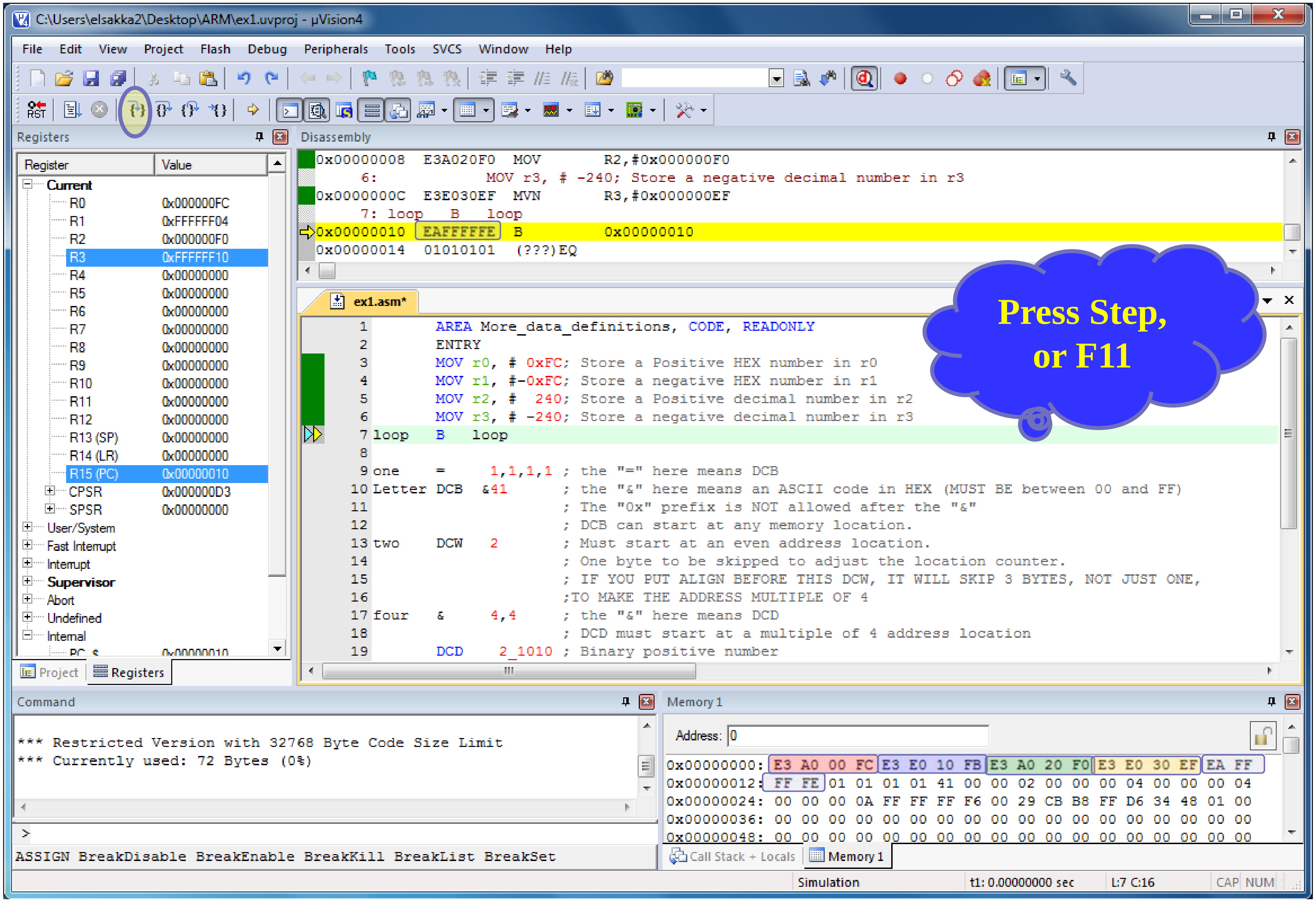
```
        AREA More_data_definitions, CODE, READONLY
        ENTRY
        MOV r0, # 0xFC; Store a Positive HEX number in r0
        MOV r1, #-0xFC; Store a negative HEX number in r1
        MOV r2, #  240; Store a Positive decimal number in r2
        MOV r3, # -240; Store a negative decimal number in r3
loop    B    loop

one     =      1,1,1,1  ; the "=" here means DCB
Letter  DCB   &41       ; the "&" here means an ASCII code in HEX (MUST BE between 00 and FF)
                        ; The "0x" prefix is NOT allowed after the "&"
                        ; DCB can start at any memory location.
two     DCW     2       ; Must start at an even address location.
                        ; One byte to be skipped to adjust the location counter.
                        ; IF YOU PUT ALIGN BEFORE THIS DCW, IT WILL SKIP 3 BYTES, NOT JUST ONE,
                        ;TO MAKE THE ADDRESS MULTIPLE OF 4
four    &      4,4      ; the "&" here means DCD
                        ; DCD must start at a multiple of 4 address location
        DCD     2_1010  ; Binary positive number
```

```
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 72 Bytes (0%)

>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet
```

```
Address: 0
0x00000000: E3 A0 00 FC E3 E0 10 FB E3 A0 20 F0 E3 E0 30 EF EA FF
0x00000012: FF FE 01 01 01 01 41 00 00 02 00 00 00 04 00 00 00 04
0x00000024: 00 00 00 0A FF FF FF F6 00 29 CB B8 FF D6 34 48 01 00
0x00000036: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00000048: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

```
Simulation    t1: 0.00000000 sec    L:6 C:61    CAP NUM
```

Data Definition Directives


C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Disassembly
ex1.asm*
Press Step, or F11
Project Registers
Command
Memory 1
Call Stack + Locals Memory 1
Simulation t1: 0.00000000 sec L:7 C:16 CAP NUM

Register	Value
Current	
R0	0x000000FC
R1	0xFFFFFF04
R2	0x000000F0
R3	0xFFFFFF10
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
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x00000010

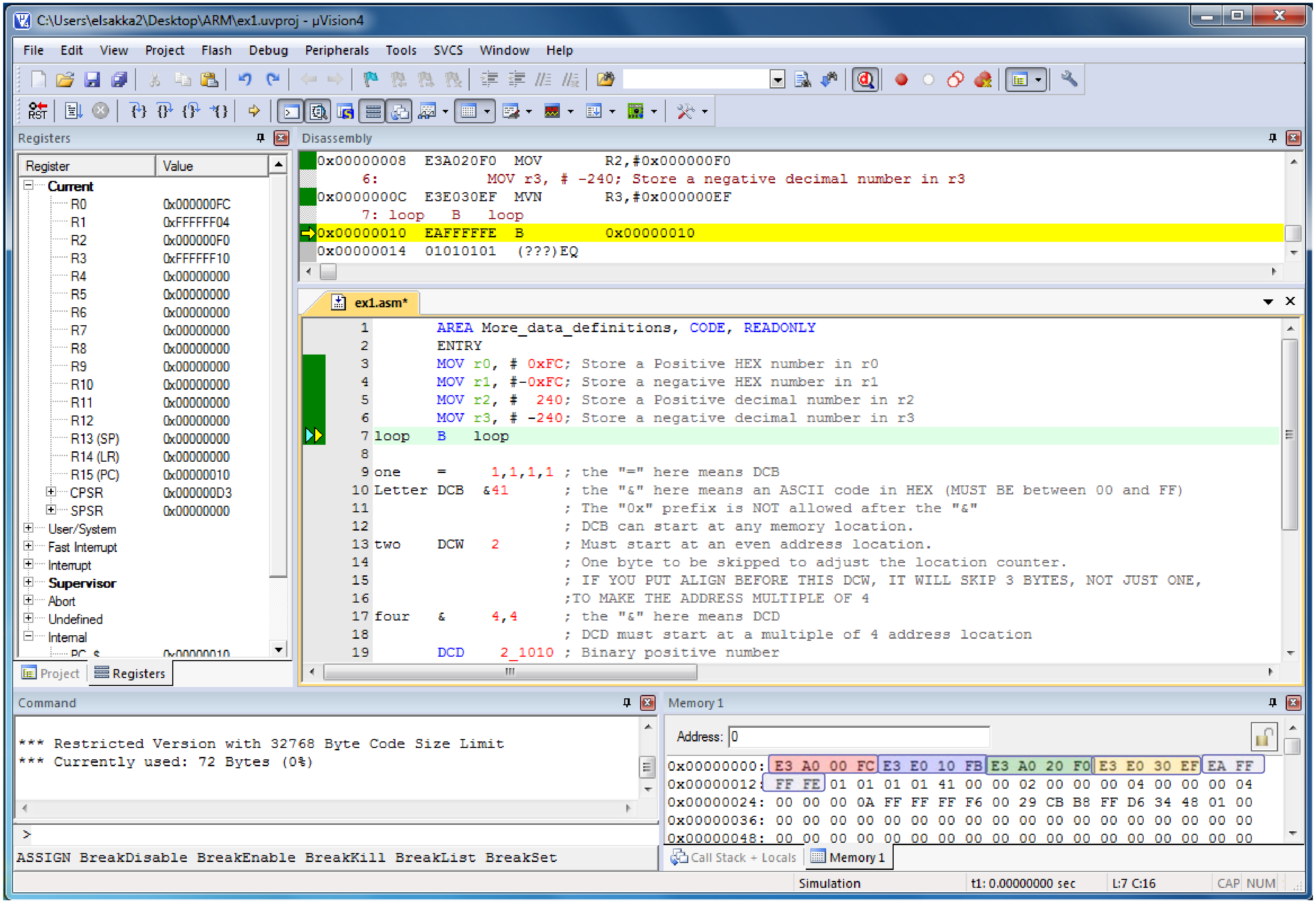
```
0x00000008  E3A020F0  MOV      R2,#0x000000F0
     6:          MOV r3, # -240; Store a negative decimal number in r3
0x0000000C  E3E030EF  MVN      R3,#0x000000EF
     7: loop    B    loop
0x00000010  EAFFFFFE  B        0x00000010
0x00000014  01010101  (???)EQ
```

```
        AREA More_data_definitions, CODE, READONLY
        ENTRY
        MOV r0, # 0xFC; Store a Positive HEX number in r0
        MOV r1, #-0xFC; Store a negative HEX number in r1
        MOV r2, #  240; Store a Positive decimal number in r2
        MOV r3, # -240; Store a negative decimal number in r3
loop    B    loop

one     =      1,1,1,1  ; the "=" here means DCB
Letter  DCB   &41       ; the "&" here means an ASCII code in HEX (MUST BE between 00 and FF)
                        ; The "0x" prefix is NOT allowed after the "&"
                        ; DCB can start at any memory location.
two     DCW    2        ; Must start at an even address location.
                        ; One byte to be skipped to adjust the location counter.
                        ; IF YOU PUT ALIGN BEFORE THIS DCW, IT WILL SKIP 3 BYTES, NOT JUST ONE,
                        ;TO MAKE THE ADDRESS MULTIPLE OF 4
four    &      4,4      ; the "&" here means DCD
                        ; DCD must start at a multiple of 4 address location
        DCD     2_1010  ; Binary positive number
```

```
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 72 Bytes (0%)

>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet
```

```
Address: 0
0x00000000: E3 A0 00 FC E3 E0 10 FB E3 A0 20 F0 E3 E0 30 EF EA FF
0x00000012: FF FE 01 01 01 01 41 00 00 02 00 00 00 04 00 00 00 04
0x00000024: 00 00 00 0A FF FF FF F6 00 29 CB B8 FF D6 34 48 01 00
0x00000036: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00000048: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Data Definition Directives



C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4

Registers

Register	Value
Current	
R0	0x000000FC
R1	0xFFFFFF04
R2	0x000000F0
R3	0xFFFFFF10
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
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x00000010

Disassembly

```
0x00000008  E3A020F0  MOV       R2,#0x000000F0
     6:          MOV r3, # -240; Store a negative decimal number in r3
0x0000000C  E3E030EF  MVN       R3,#0x000000EF
     7: loop    B    loop
0x00000010  EAFFFFFE  B         0x00000010
0x00000014  01010101  (???)EQ
```

ex1.asm*

```
        AREA More_data_definitions, CODE, READONLY
        ENTRY
        MOV r0, # 0xFC; Store a Positive HEX number in r0
        MOV r1, #-0xFC; Store a negative HEX number in r1
        MOV r2, #  240; Store a Positive decimal number in r2
        MOV r3, # -240; Store a negative decimal number in r3
loop    B    loop

one     =      1,1,1,1 ; the "=" here means DCB
Letter  DCB   &41      ; the "&" here means an ASCII code in HEX (MUST BE between 00 and FF)
                       ; The "0x" prefix is NOT allowed after the "&"
                       ; DCB can start at any memory location.
two     DCW    2       ; Must start at an even address location.
                       ; One byte to be skipped to adjust the location counter.
                       ; IF YOU PUT ALIGN BEFORE THIS DCW, IT WILL SKIP 3 BYTES, NOT JUST ONE,
                       ;TO MAKE THE ADDRESS MULTIPLE OF 4
four    &      4,4     ; the "&" here means DCD
                       ; DCD must start at a multiple of 4 address location
        DCD     2_1010 ; Binary positive number
```

Command

```
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 72 Bytes (0%)

>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet
```

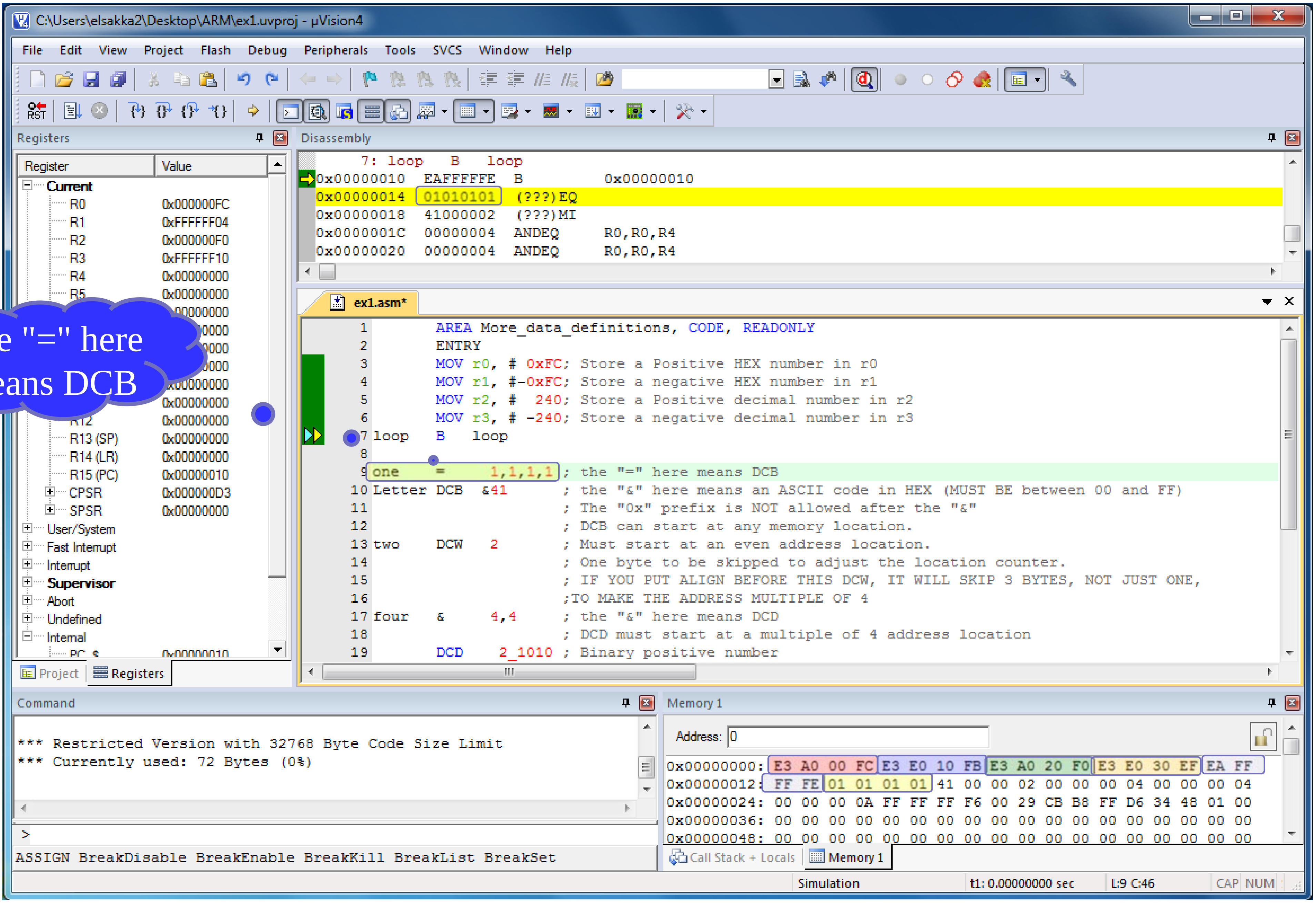
Memory 1

Address: 0

```
0x00000000: E3 A0 00 FC E3 E0 10 FB E3 A0 20 F0 E3 E0 30 EF EA FF
0x00000012: FF FE 01 01 01 01 41 00 00 02 00 00 00 04 00 00 00 04
0x00000024: 00 00 00 0A FF FF FF F6 00 29 CB B8 FF D6 34 48 01 00
0x00000036: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00000048: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Simulation   t1: 0.00000000 sec   L:7 C:16   CAP NUM

Data Definition Directives


C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Disassembly
ex1.asm*
Command
Memory 1
Project Registers
Call Stack + Locals Memory 1
The "=" here means DCB

Register	Value
Current	
R0	0x000000FC
R1	0xFFFFFF04
R2	0x000000F0
R3	0xFFFFFF10
R4	0x00000000
R5	0x00000000
	00000000
	0000
	0000
	0000
	00000000
	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x00000010
CPSR	0x000000D3
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	

```
     7: loop    B    loop
0x00000010  EAFFFFFE  B         0x00000010
0x00000014  01010101  (???)EQ
0x00000018  41000002  (???)MI
0x0000001C  00000004  ANDEQ     R0,R0,R4
0x00000020  00000004  ANDEQ     R0,R0,R4
```

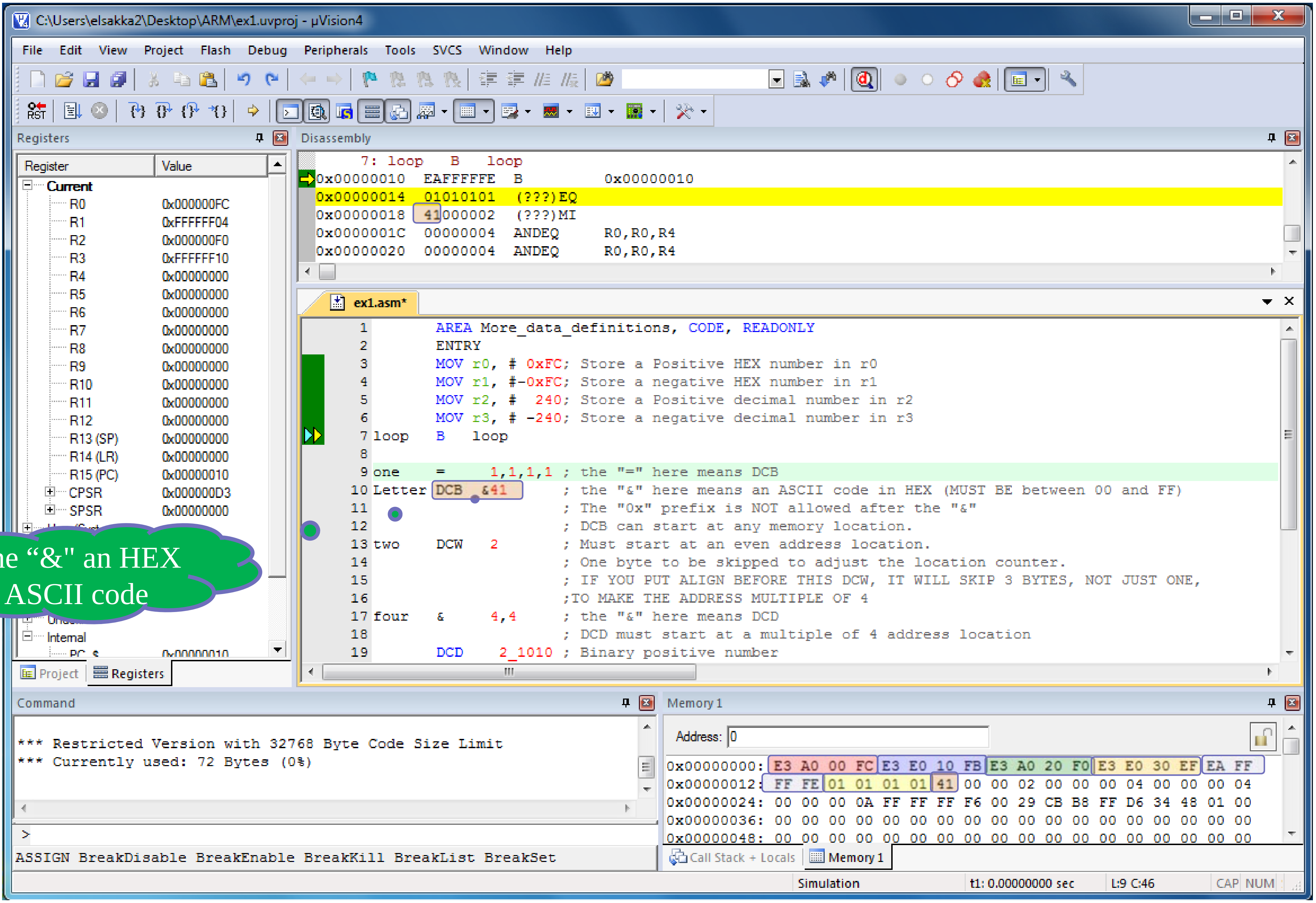
```
        AREA More_data_definitions, CODE, READONLY
        ENTRY
        MOV r0, # 0xFC; Store a Positive HEX number in r0
        MOV r1, #-0xFC; Store a negative HEX number in r1
        MOV r2, #  240; Store a Positive decimal number in r2
        MOV r3, # -240; Store a negative decimal number in r3
loop    B   loop

one     =       1,1,1,1 ; the "=" here means DCB
Letter  DCB   &41       ; the "&" here means an ASCII code in HEX (MUST BE between 00 and FF)
                        ; The "0x" prefix is NOT allowed after the "&"
                        ; DCB can start at any memory location.
two     DCW     2       ; Must start at an even address location.
                        ; One byte to be skipped to adjust the location counter.
                        ; IF YOU PUT ALIGN BEFORE THIS DCW, IT WILL SKIP 3 BYTES, NOT JUST ONE,
                        ;TO MAKE THE ADDRESS MULTIPLE OF 4
four    &       4,4     ; the "&" here means DCD
                        ; DCD must start at a multiple of 4 address location
        DCD     2_1010  ; Binary positive number
```

```
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 72 Bytes (0%)

>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet
```

```
Address: 0
0x00000000: E3 A0 00 FC E3 E0 10 FB E3 A0 20 F0 E3 E0 30 EF EA FF
0x00000012: FF FE 01 01 01 01 41 00 00 02 00 00 00 04 00 00 00 04
0x00000024: 00 00 00 0A FF FF FF F6 00 29 CB B8 FF D6 34 48 01 00
0x00000036: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00000048: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Simulation   t1: 0.00000000 sec   L:9 C:46   CAP NUM

Data Definition Directives



C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4

Registers

Register	Value
Current	
R0	0x000000FC
R1	0xFFFFFF04
R2	0x000000F0
R3	0xFFFFFF10
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
SPSR	0x00000000
Internal	
PC $	0x00000010

Disassembly

```
     7: loop    B    loop
0x00000010  EAFFFFFE  B         0x00000010
0x00000014  01010101  (???)EQ
0x00000018  41000002  (???)MI
0x0000001C  00000004  ANDEQ     R0,R0,R4
0x00000020  00000004  ANDEQ     R0,R0,R4
```

ex1.asm*

```
        AREA More_data_definitions, CODE, READONLY
        ENTRY
        MOV r0, # 0xFC; Store a Positive HEX number in r0
        MOV r1, #-0xFC; Store a negative HEX number in r1
        MOV r2, #  240; Store a Positive decimal number in r2
        MOV r3, # -240; Store a negative decimal number in r3
loop    B    loop

one     =       1,1,1,1  ; the "=" here means DCB
Letter  DCB   &41        ; the "&" here means an ASCII code in HEX (MUST BE between 00 and FF)
                         ; The "0x" prefix is NOT allowed after the "&"
                         ; DCB can start at any memory location.
two     DCW     2        ; Must start at an even address location.
                         ; One byte to be skipped to adjust the location counter.
                         ; IF YOU PUT ALIGN BEFORE THIS DCW, IT WILL SKIP 3 BYTES, NOT JUST ONE,
                         ;TO MAKE THE ADDRESS MULTIPLE OF 4
four    &       4,4      ; the "&" here means DCD
                         ; DCD must start at a multiple of 4 address location
        DCD     2_1010   ; Binary positive number
```

Command

```
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 72 Bytes (0%)

>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet
```

Memory 1

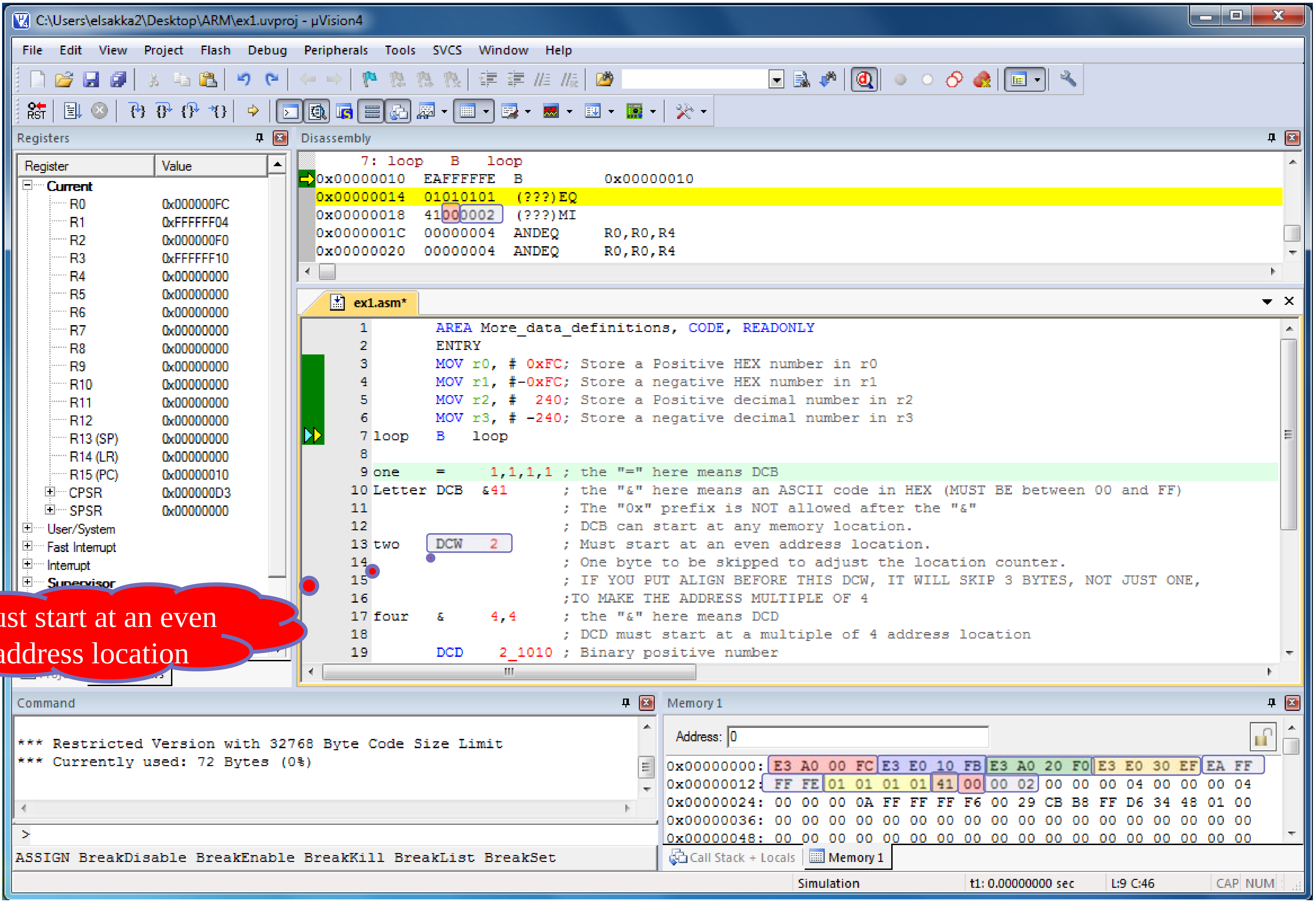
Address: 0

```
0x00000000: E3 A0 00 FC E3 E0 10 FB E3 A0 20 F0 E3 E0 30 EF EA FF
0x00000012: FF FE 01 01 01 01 41 00 00 02 00 00 00 04 00 00 00 04
0x00000024: 00 00 00 0A FF FF FF F6 00 29 CB B8 FF D6 34 48 01 00
0x00000036: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00000048: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Simulation   t1: 0.00000000 sec   L:9 C:46   CAP NUM

The “&" an HEX ASCII code

Data Definition Directives


C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Disassembly
ex1.asm*
Command
Memory 1
Call Stack + Locals
Simulation
t1: 0.00000000 sec
L:9 C:46
CAP NUM

Register	Value
Current	
R0	0x000000FC
R1	0xFFFFFF04
R2	0x000000F0
R3	0xFFFFFF10
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
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	

```
     7: loop    B    loop
0x00000010  EAFFFFFE  B         0x00000010
0x00000014  01010101  (???)EQ
0x00000018  41000002  (???)MI
0x0000001C  00000004  ANDEQ     R0,R0,R4
0x00000020  00000004  ANDEQ     R0,R0,R4
```

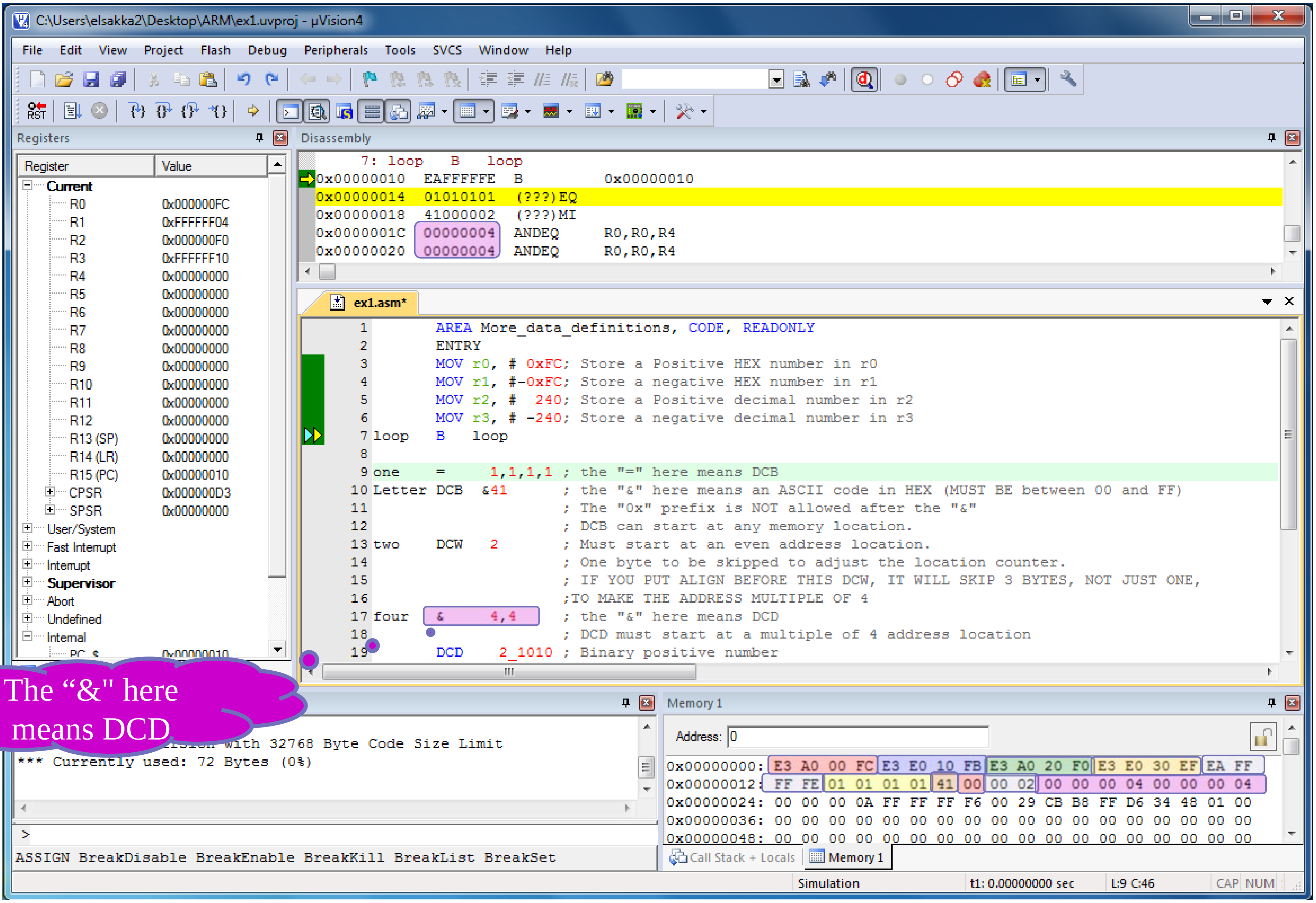
```
        AREA More_data_definitions, CODE, READONLY
        ENTRY
        MOV r0, # 0xFC; Store a Positive HEX number in r0
        MOV r1, #-0xFC; Store a negative HEX number in r1
        MOV r2, #  240; Store a Positive decimal number in r2
        MOV r3, # -240; Store a negative decimal number in r3
loop    B    loop

one     =       1,1,1,1  ; the "=" here means DCB
Letter  DCB    &41       ; the "&" here means an ASCII code in HEX (MUST BE between 00 and FF)
                         ; The "0x" prefix is NOT allowed after the "&"
                         ; DCB can start at any memory location.
two      DCW     2       ; Must start at an even address location.
                         ; One byte to be skipped to adjust the location counter.
                         ; IF YOU PUT ALIGN BEFORE THIS DCW, IT WILL SKIP 3 BYTES, NOT JUST ONE,
                         ;TO MAKE THE ADDRESS MULTIPLE OF 4
four    &       4,4      ; the "&" here means DCD
                         ; DCD must start at a multiple of 4 address location
        DCD     2_1010   ; Binary positive number
```

```
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 72 Bytes (0%)

>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet
```

```
Address: 0
0x00000000: E3 A0 00 FC E3 E0 10 FB E3 A0 20 F0 E3 E0 30 EF EA FF
0x00000012: FF FE 01 01 01 01 41 00 00 02 00 00 00 04 00 00 00 04
0x00000024: 00 00 00 0A FF FF FF F6 00 29 CB B8 FF D6 34 48 01 00
0x00000036: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00000048: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Must start at an even address location

Data Definition Directives


C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Disassembly
ex1.asm*
Memory 1
Call Stack + Locals
The "&" here means DCD

Register	Value
Current	
R0	0x000000FC
R1	0xFFFFFF04
R2	0x000000F0
R3	0xFFFFFF10
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
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x00000010

```
      7: loop    B    loop
0x00000010  EAFFFFFE  B         0x00000010
0x00000014  01010101  (???)EQ
0x00000018  41000002  (???)MI
0x0000001C  00000004  ANDEQ     R0,R0,R4
0x00000020  00000004  ANDEQ     R0,R0,R4
```

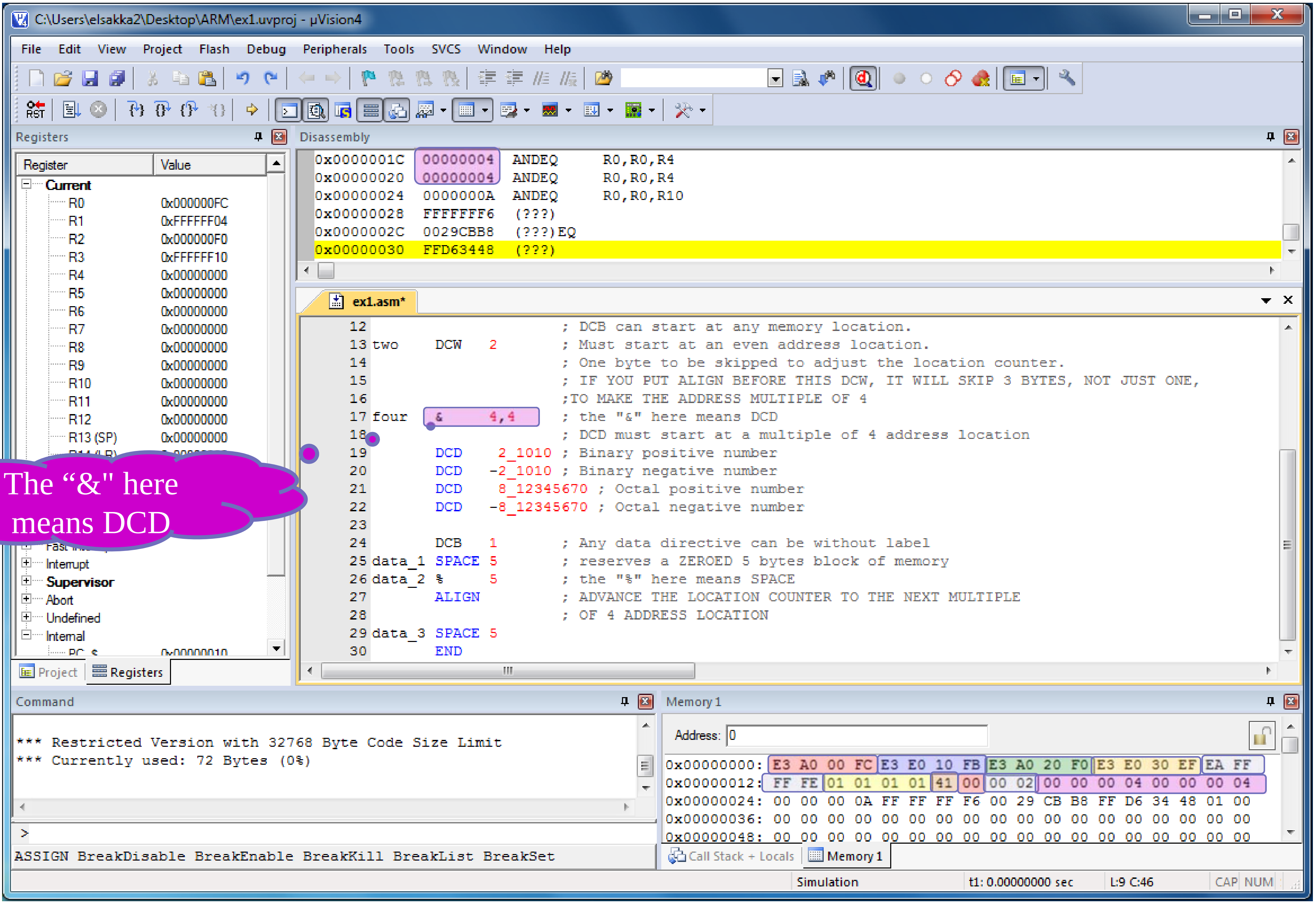
```
        AREA More_data_definitions, CODE, READONLY
        ENTRY
        MOV r0, # 0xFC; Store a Positive HEX number in r0
        MOV r1, #-0xFC; Store a negative HEX number in r1
        MOV r2, #  240; Store a Positive decimal number in r2
        MOV r3, # -240; Store a negative decimal number in r3
loop    B    loop

one     =       1,1,1,1 ; the "=" here means DCB
Letter DCB   &41        ; the "&" here means an ASCII code in HEX (MUST BE between 00 and FF)
                        ; The "0x" prefix is NOT allowed after the "&"
                        ; DCB can start at any memory location.
two     DCW    2        ; Must start at an even address location.
                        ; One byte to be skipped to adjust the location counter.
                        ; IF YOU PUT ALIGN BEFORE THIS DCW, IT WILL SKIP 3 BYTES, NOT JUST ONE,
                        ;TO MAKE THE ADDRESS MULTIPLE OF 4
four    &       4,4     ; the "&" here means DCD
                        ; DCD must start at a multiple of 4 address location
        DCD     2_1010 ; Binary positive number
```

```
...ision with 32768 Byte Code Size Limit
*** Currently used: 72 Bytes (0%)
>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet
```

```
Address: 0
0x00000000: E3 A0 00 FC E3 E0 10 FB E3 A0 20 F0 E3 E0 30 EF EA FF
0x00000012: FF FE 01 01 01 01 41 00 00 02 00 00 00 04 00 00 00 04
0x00000024: 00 00 00 0A FF FF FF F6 00 29 CB B8 FF D6 34 48 01 00
0x00000036: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00000048: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Simulation   t1: 0.00000000 sec   L:9 C:46   CAP NUM

Data Definition Directives


C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Disassembly
ex1.asm*
Project
Registers
Command
Memory 1
Call Stack + Locals
The “&" here means DCD

Register	Value
Current	
R0	0x000000FC
R1	0xFFFFFF04
R2	0x000000F0
R3	0xFFFFFF10
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	

```
0x0000001C  00000004  ANDEQ     R0,R0,R4
0x00000020  00000004  ANDEQ     R0,R0,R4
0x00000024  0000000A  ANDEQ     R0,R0,R10
0x00000028  FFFFFFF6  (???)
0x0000002C  0029CBB8  (???)EQ
0x00000030  FFD63448  (???)
```

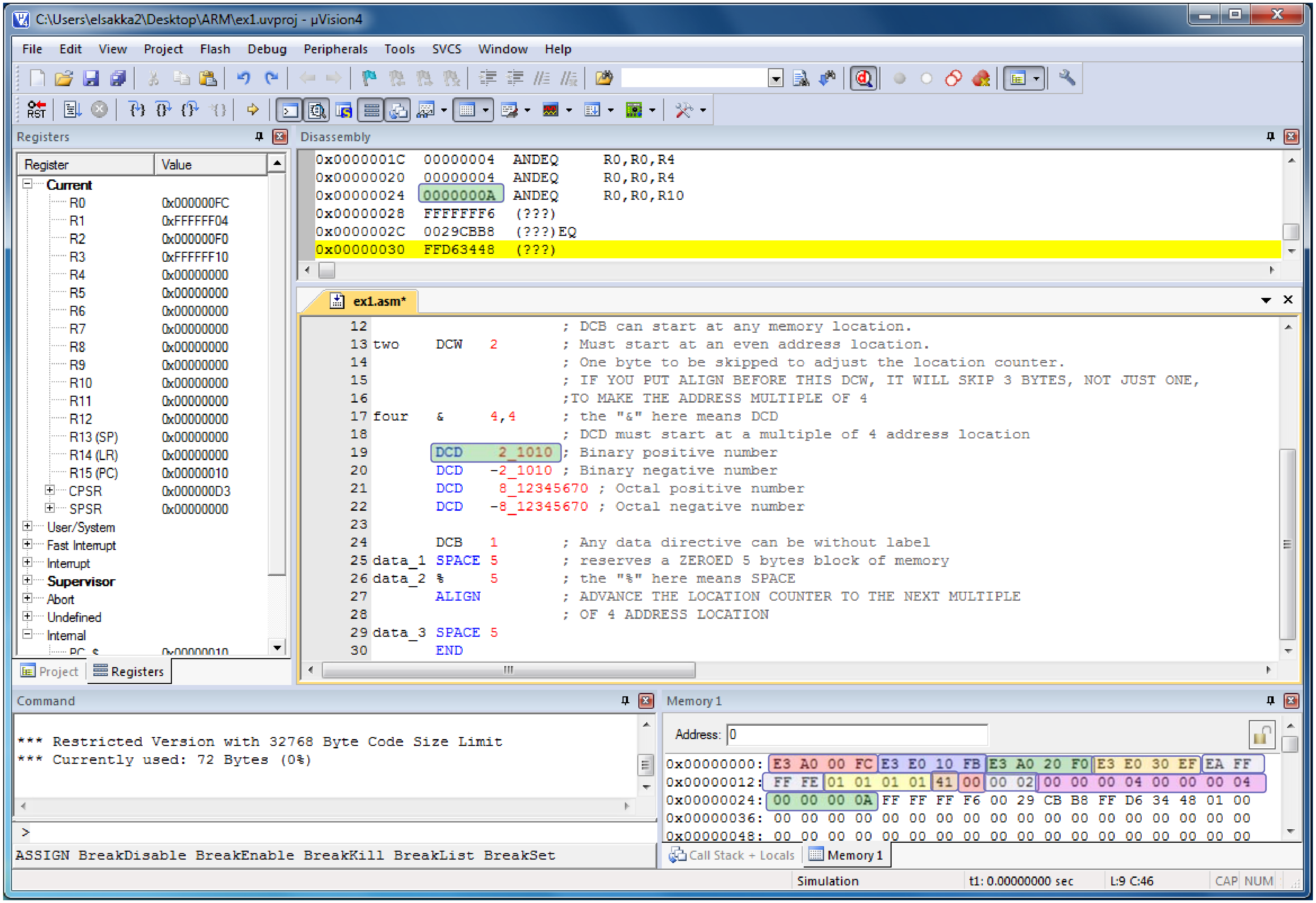
```
                                 ; DCB can start at any memory location.
two     DCW    2                 ; Must start at an even address location.
                                 ; One byte to be skipped to adjust the location counter.
                                 ; IF YOU PUT ALIGN BEFORE THIS DCW, IT WILL SKIP 3 BYTES, NOT JUST ONE,
                                 ;TO MAKE THE ADDRESS MULTIPLE OF 4
four    &      4,4               ; the "&" here means DCD
                                 ; DCD must start at a multiple of 4 address location
        DCD     2_1010 ; Binary positive number
        DCD    -2_1010 ; Binary negative number
        DCD     8_12345670 ; Octal positive number
        DCD    -8_12345670 ; Octal negative number

        DCB    1                 ; Any data directive can be without label
data_1 SPACE 5                   ; reserves a ZEROED 5 bytes block of memory
data_2 %      5                  ; the "%" here means SPACE
        ALIGN                    ; ADVANCE THE LOCATION COUNTER TO THE NEXT MULTIPLE
                                 ; OF 4 ADDRESS LOCATION
data_3 SPACE 5
        END
```

```
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 72 Bytes (0%)

>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet
```

```
Address: 0
0x00000000: E3 A0 00 FC E3 E0 10 FB E3 A0 20 F0 E3 E0 30 EF EA FF
0x00000012: FF FE 01 01 01 01 41 00 00 02 00 00 00 04 00 00 00 04
0x00000024: 00 00 00 0A FF FF FF F6 00 29 CB B8 FF D6 34 48 01 00
0x00000036: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00000048: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Simulation   t1: 0.00000000 sec   L:9 C:46   CAP NUM

Data Definition Directives



C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4

Registers

Register	Value
Current	
R0	0x000000FC
R1	0xFFFFFF04
R2	0x000000F0
R3	0xFFFFFF10
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
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x00000010

Disassembly

```
0x0000001C  00000004  ANDEQ     R0,R0,R4
0x00000020  00000004  ANDEQ     R0,R0,R4
0x00000024  0000000A  ANDEQ     R0,R0,R10
0x00000028  FFFFFFF6  (???)
0x0000002C  0029CBB8  (???)EQ
0x00000030  FFD63448  (???)
```

ex1.asm*

```
                                  ; DCB can start at any memory location.
two     DCW    2                  ; Must start at an even address location.
                                  ; One byte to be skipped to adjust the location counter.
                                  ; IF YOU PUT ALIGN BEFORE THIS DCW, IT WILL SKIP 3 BYTES, NOT JUST ONE,
                                  ;TO MAKE THE ADDRESS MULTIPLE OF 4
four    &      4,4                ; the "&" here means DCD
                                  ; DCD must start at a multiple of 4 address location
        DCD     2_1010            ; Binary positive number
        DCD    -2_1010            ; Binary negative number
        DCD     8_12345670        ; Octal positive number
        DCD    -8_12345670        ; Octal negative number

        DCB    1                  ; Any data directive can be without label
data_1  SPACE  5                  ; reserves a ZEROED 5 bytes block of memory
data_2  %      5                  ; the "%" here means SPACE
        ALIGN                     ; ADVANCE THE LOCATION COUNTER TO THE NEXT MULTIPLE
                                  ; OF 4 ADDRESS LOCATION
data_3  SPACE  5
        END
```

Command

```
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 72 Bytes (0%)

>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet
```

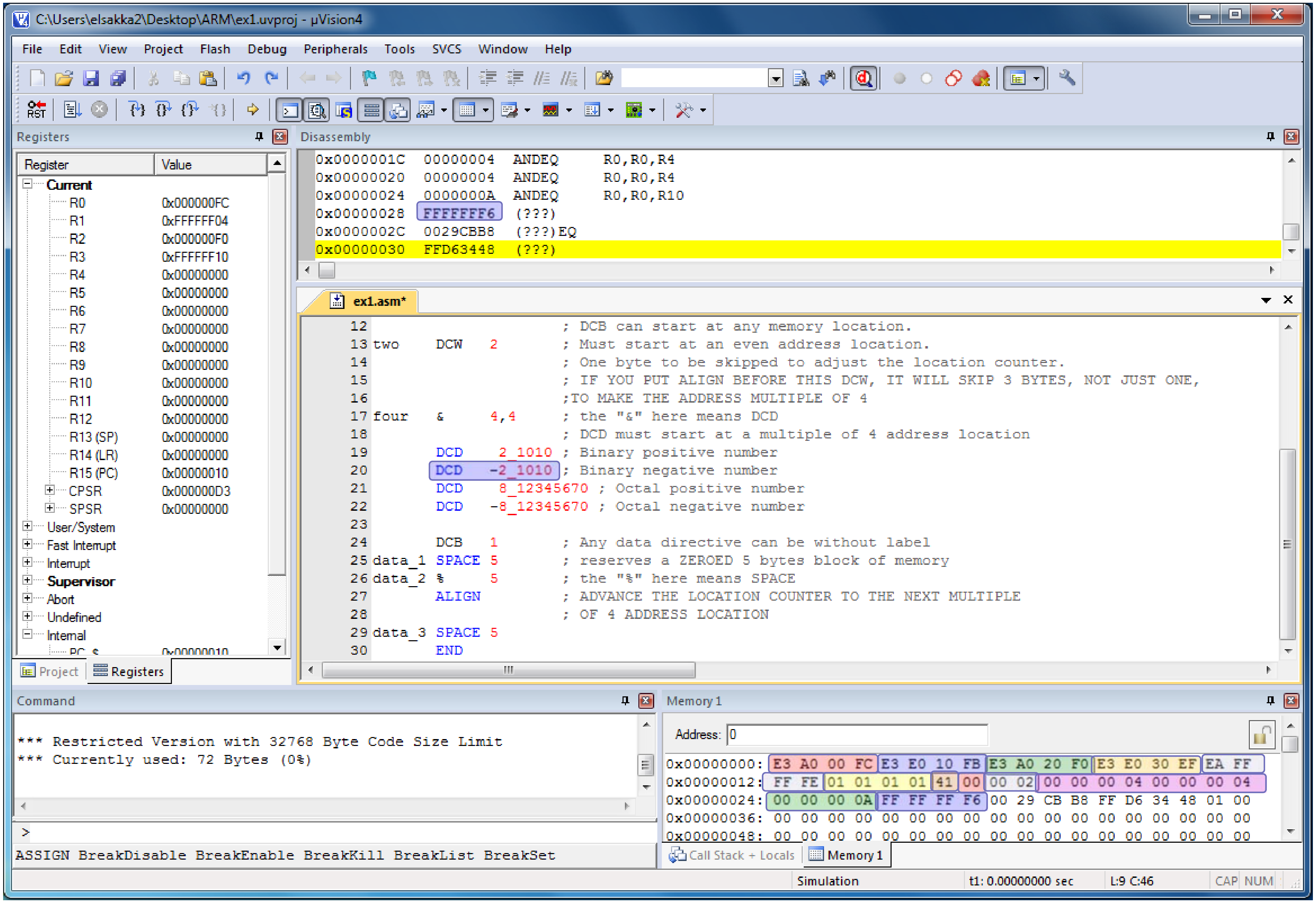
Memory 1

Address: 0

```
0x00000000: E3 A0 00 FC E3 E0 10 FB E3 A0 20 F0 E3 E0 30 EF EA FF
0x00000012: FF FE 01 01 01 01 41 00 00 02 00 00 00 04 00 00 00 04
0x00000024: 00 00 00 0A FF FF FF F6 00 29 CB B8 FF D6 34 48 01 00
0x00000036: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00000048: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Simulation   t1: 0.00000000 sec   L:9 C:46   CAP NUM

Data Definition Directives


C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Disassembly
ex1.asm*
Command
Memory 1
Project
Registers
Call Stack + Locals
Memory 1

Register	Value
Current	
R0	0x000000FC
R1	0xFFFFFF04
R2	0x000000F0
R3	0xFFFFFF10
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
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x00000010

```
0x0000001C  00000004  ANDEQ     R0,R0,R4
0x00000020  00000004  ANDEQ     R0,R0,R4
0x00000024  0000000A  ANDEQ     R0,R0,R10
0x00000028  FFFFFFF6  (???)
0x0000002C  0029CBB8  (???)EQ
0x00000030  FFD63448  (???)
```

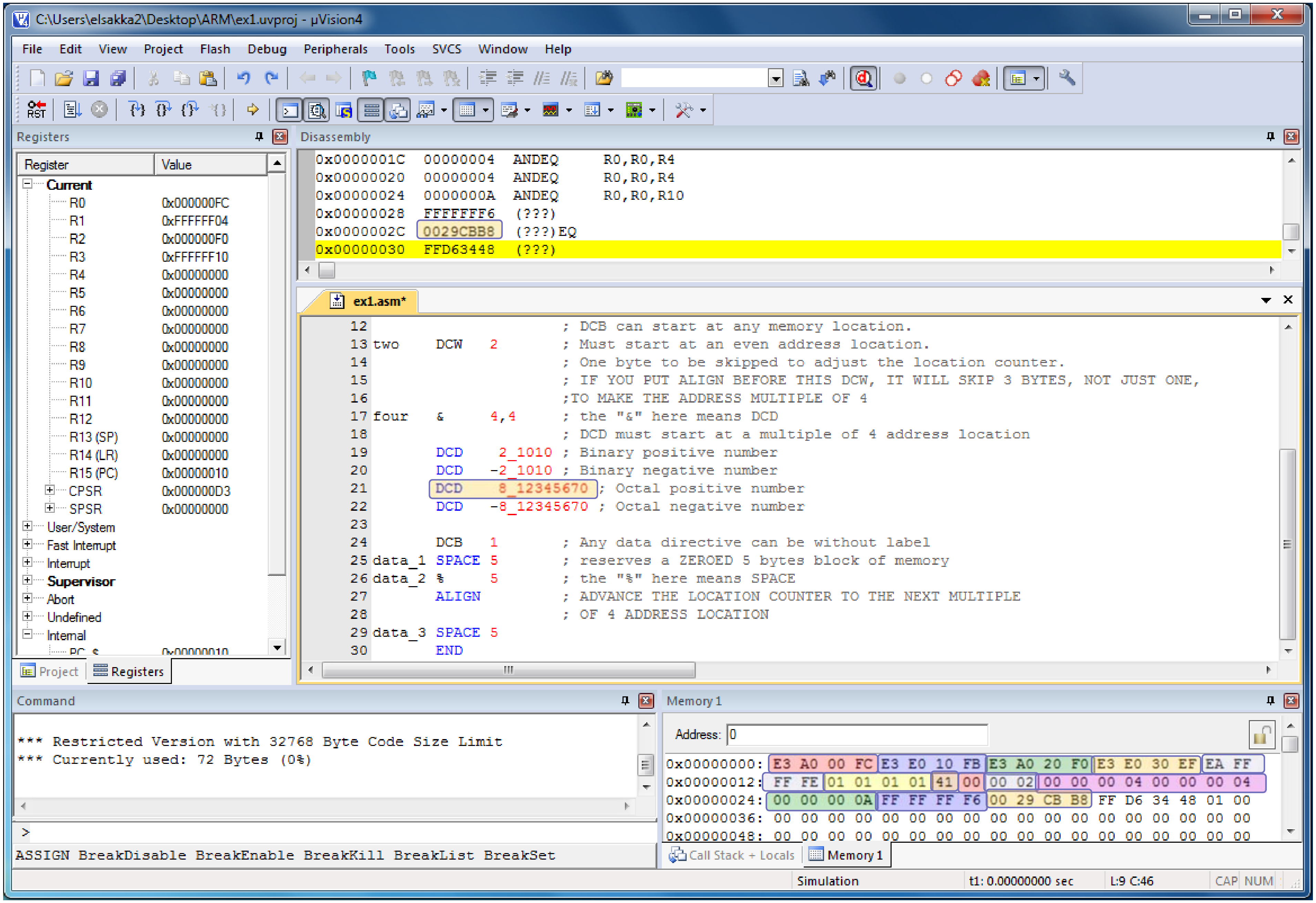
```
                             ; DCB can start at any memory location.
two     DCW     2            ; Must start at an even address location.
                             ; One byte to be skipped to adjust the location counter.
                             ; IF YOU PUT ALIGN BEFORE THIS DCW, IT WILL SKIP 3 BYTES, NOT JUST ONE,
                             ;TO MAKE THE ADDRESS MULTIPLE OF 4
four    &       4,4          ; the "&" here means DCD
                             ; DCD must start at a multiple of 4 address location
        DCD      2_1010 ; Binary positive number
        DCD     -2_1010 ; Binary negative number
        DCD      8_12345670 ; Octal positive number
        DCD     -8_12345670 ; Octal negative number

        DCB     1            ; Any data directive can be without label
data_1  SPACE 5              ; reserves a ZEROED 5 bytes block of memory
data_2  %       5            ; the "%" here means SPACE
        ALIGN                ; ADVANCE THE LOCATION COUNTER TO THE NEXT MULTIPLE
                             ; OF 4 ADDRESS LOCATION
data_3  SPACE 5
        END
```

```
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 72 Bytes (0%)

>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet
```

```
Address: 0

0x00000000: E3 A0 00 FC E3 E0 10 FB E3 A0 20 F0 E3 E0 30 EF EA FF
0x00000012: FF FE 01 01 01 01 41 00 00 02 00 00 00 04 00 00 00 04
0x00000024: 00 00 00 0A FF FF FF F6 00 29 CB B8 FF D6 34 48 01 00
0x00000036: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00000048: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

```
Simulation    t1: 0.00000000 sec    L:9 C:46    CAP NUM
```

Data Definition Directives


C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Disassembly
ex1.asm*
Command
Memory 1
Project
Registers
Call Stack + Locals
Memory 1

Register	Value
Current	
R0	0x000000FC
R1	0xFFFFFF04
R2	0x000000F0
R3	0xFFFFFF10
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
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x00000010

```
0x0000001C  00000004  ANDEQ     R0,R0,R4
0x00000020  00000004  ANDEQ     R0,R0,R4
0x00000024  0000000A  ANDEQ     R0,R0,R10
0x00000028  FFFFFFF6  (???)
0x0000002C  0029CBB8  (???)EQ
0x00000030  FFD63448  (???)
```

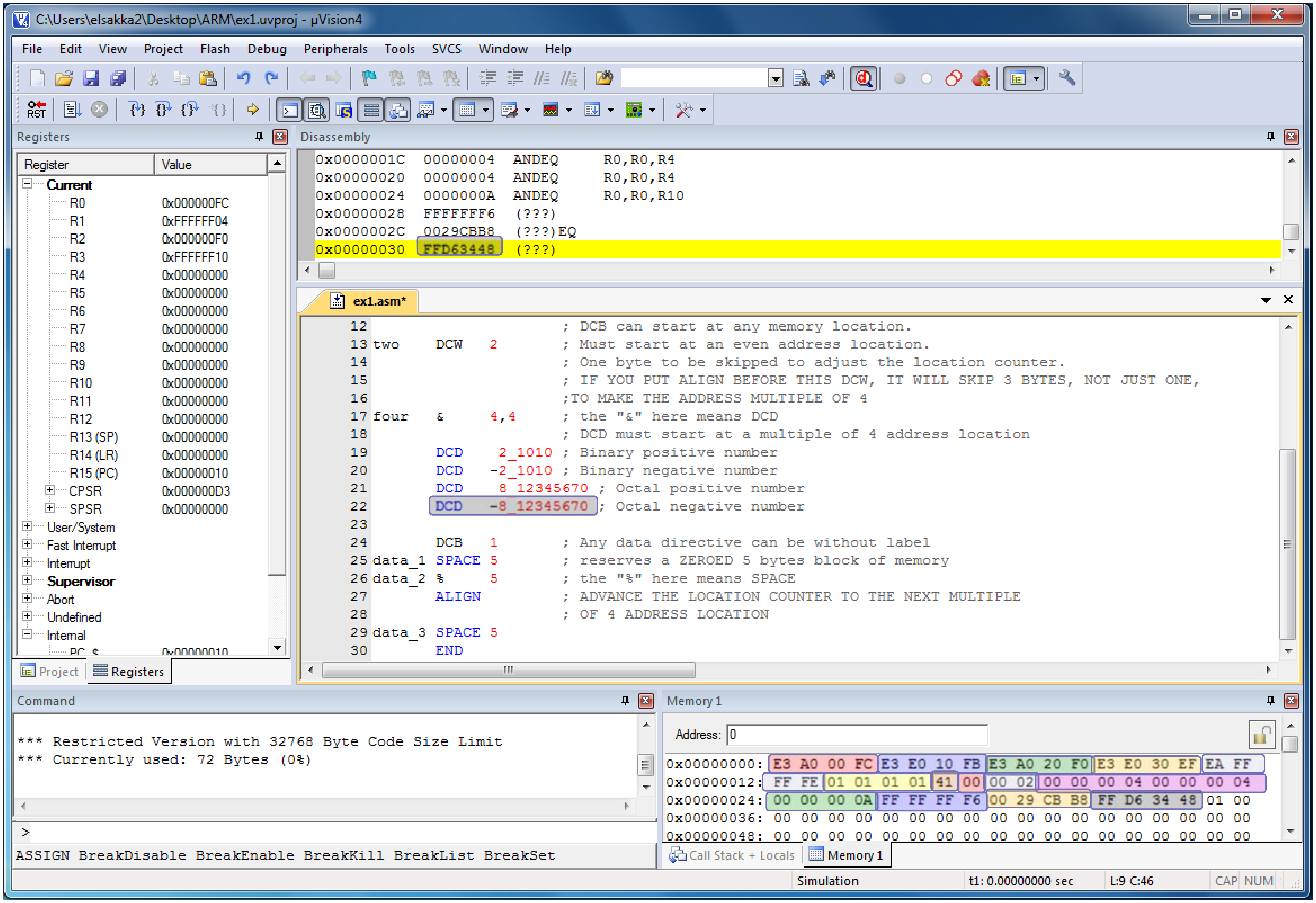
```
                                    ; DCB can start at any memory location.
two     DCW    2                    ; Must start at an even address location.
                                    ; One byte to be skipped to adjust the location counter.
                                    ; IF YOU PUT ALIGN BEFORE THIS DCW, IT WILL SKIP 3 BYTES, NOT JUST ONE,
                                    ;TO MAKE THE ADDRESS MULTIPLE OF 4
four    &      4,4                  ; the "&" here means DCD
                                    ; DCD must start at a multiple of 4 address location
        DCD     2_1010              ; Binary positive number
        DCD    -2_1010              ; Binary negative number
        DCD     8_12345670          ; Octal positive number
        DCD    -8_12345670          ; Octal negative number

        DCB    1                    ; Any data directive can be without label
data_1  SPACE  5                    ; reserves a ZEROED 5 bytes block of memory
data_2  %      5                    ; the "%" here means SPACE
        ALIGN                       ; ADVANCE THE LOCATION COUNTER TO THE NEXT MULTIPLE
                                    ; OF 4 ADDRESS LOCATION
data_3  SPACE  5
        END
```

```
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 72 Bytes (0%)

>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet
```

```
Address: 0
0x00000000: E3 A0 00 FC E3 E0 10 FB E3 A0 20 F0 E3 E0 30 EF EA FF
0x00000012: FF FE 01 01 01 01 41 00 00 02 00 00 00 04 00 00 00 04
0x00000024: 00 00 00 0A FF FF FF F6 00 29 CB B8 FF D6 34 48 01 00
0x00000036: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00000048: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

```
Simulation    t1: 0.00000000 sec    L:9 C:46    CAP NUM
```

Data Definition Directives



Register	Value
Current	
R0	0x000000FC
R1	0xFFFFFF04
R2	0x000000F0
R3	0xFFFFFF10
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
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x00000010

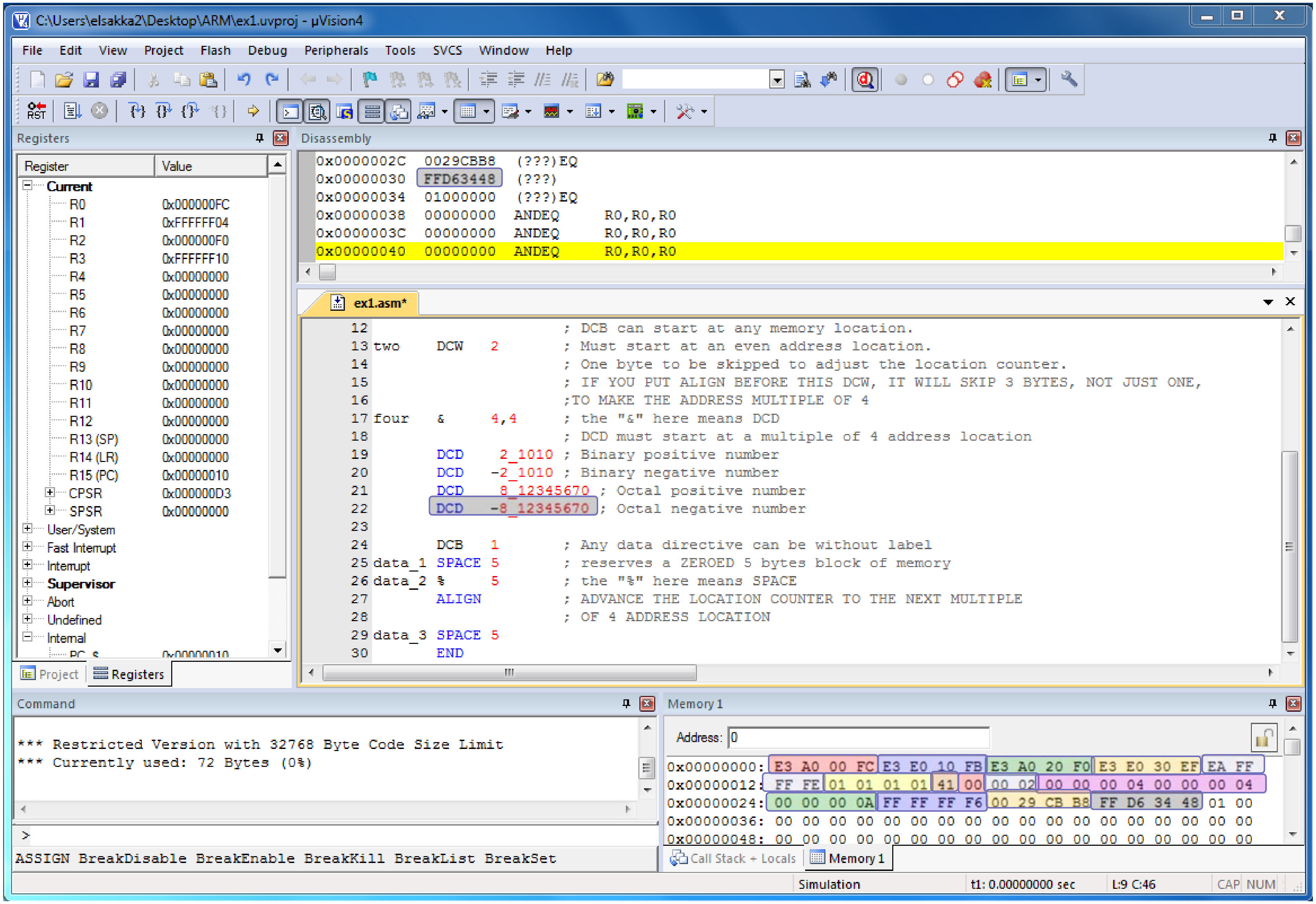
```
0x0000001C  00000004  ANDEQ     R0,R0,R4
0x00000020  00000004  ANDEQ     R0,R0,R4
0x00000024  0000000A  ANDEQ     R0,R0,R10
0x00000028  FFFFFFF6  (???)
0x0000002C  0029CBB8  (???)EQ
0x00000030  FFD63448  (???)
```

```
                                ; DCB can start at any memory location.
two     DCW     2               ; Must start at an even address location.
                                ; One byte to be skipped to adjust the location counter.
                                ; IF YOU PUT ALIGN BEFORE THIS DCW, IT WILL SKIP 3 BYTES, NOT JUST ONE,
                                ;TO MAKE THE ADDRESS MULTIPLE OF 4
four    &       4,4             ; the "&" here means DCD
                                ; DCD must start at a multiple of 4 address location
        DCD      2_1010 ; Binary positive number
        DCD     -2_1010 ; Binary negative number
        DCD      8_12345670 ; Octal positive number
        DCD     -8_12345670 ; Octal negative number

        DCB     1               ; Any data directive can be without label
data_1  SPACE   5               ; reserves a ZEROED 5 bytes block of memory
data_2  %       5               ; the "%" here means SPACE
        ALIGN                   ; ADVANCE THE LOCATION COUNTER TO THE NEXT MULTIPLE
                                ; OF 4 ADDRESS LOCATION
data_3  SPACE   5
        END
```

```
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 72 Bytes (0%)

>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet
```

```
Address: 0
0x00000000: E3 A0 00 FC E3 E0 10 FB E3 A0 20 F0 E3 E0 30 EF EA FF
0x00000012: FF FE 01 01 01 01 41 00 00 02 00 00 00 04 00 00 00 04
0x00000024: 00 00 00 0A FF FF FF F6 00 29 CB B8 FF D6 34 48 01 00
0x00000036: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00000048: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Data Definition Directives



C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4

Registers

```
Register        Value
Current
    R0          0x000000FC
    R1          0xFFFFFF04
    R2          0x000000F0
    R3          0xFFFFFF10
    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
    SPSR        0x00000000
User/System
Fast Interrupt
Interrupt
Supervisor
Abort
Undefined
Internal
```

Disassembly

```
0x0000002C  0029CBB8  (???)EQ
0x00000030  FFD63448  (???)
0x00000034  01000000  (???)EQ
0x00000038  00000000  ANDEQ     R0,R0,R0
0x0000003C  00000000  ANDEQ     R0,R0,R0
0x00000040  00000000  ANDEQ     R0,R0,R0
```

ex1.asm*

```
                             ; DCB can start at any memory location.
two     DCW    2             ; Must start at an even address location.
                             ; One byte to be skipped to adjust the location counter.
                             ; IF YOU PUT ALIGN BEFORE THIS DCW, IT WILL SKIP 3 BYTES, NOT JUST ONE,
                             ;TO MAKE THE ADDRESS MULTIPLE OF 4
four    &      4,4           ; the "&" here means DCD
                             ; DCD must start at a multiple of 4 address location
        DCD     2_1010       ; Binary positive number
        DCD    -2_1010       ; Binary negative number
        DCD     8_12345670  ; Octal positive number
        DCD    -8_12345670  ; Octal negative number

        DCB    1             ; Any data directive can be without label
data_1  SPACE  5             ; reserves a ZEROED 5 bytes block of memory
data_2  %      5             ; the "%" here means SPACE
        ALIGN                ; ADVANCE THE LOCATION COUNTER TO THE NEXT MULTIPLE
                             ; OF 4 ADDRESS LOCATION
data_3  SPACE  5
        END
```

Command

```
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 72 Bytes (0%)

>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet
```

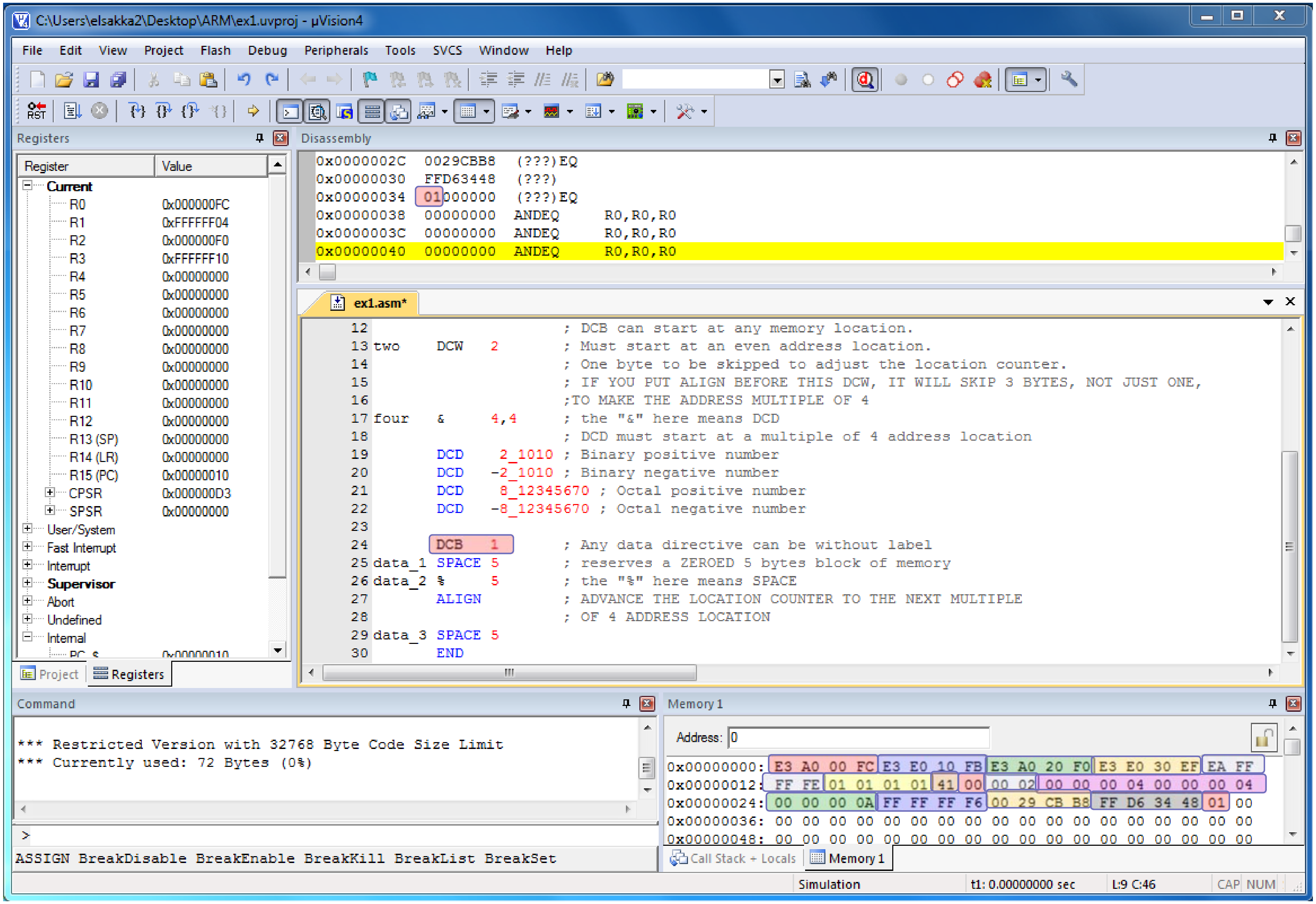
Memory 1

Address: 0

```
0x00000000: E3 A0 00 FC E3 E0 10 FB E3 A0 20 F0 E3 E0 30 EF EA FF
0x00000012: FF FE 01 01 01 01 41 00 00 02 00 00 00 04 00 00 00 04
0x00000024: 00 00 00 0A FF FF FF F6 00 29 CB B8 FF D6 34 48 01 00
0x00000036: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00000048: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Simulation   t1: 0.00000000 sec   L:9 C:46   CAP NUM

Data Definition Directives


C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Disassembly
ex1.asm*
Project
Registers
Command
Memory 1
Call Stack + Locals
Memory 1

Register	Value
Current	
R0	0x000000FC
R1	0xFFFFFF04
R2	0x000000F0
R3	0xFFFFFF10
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
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x00000010

```
0x0000002C  0029CBB8  (???)EQ
0x00000030  FFD63448  (???)
0x00000034  01000000  (???)EQ
0x00000038  00000000  ANDEQ     R0,R0,R0
0x0000003C  00000000  ANDEQ     R0,R0,R0
0x00000040  00000000  ANDEQ     R0,R0,R0
```

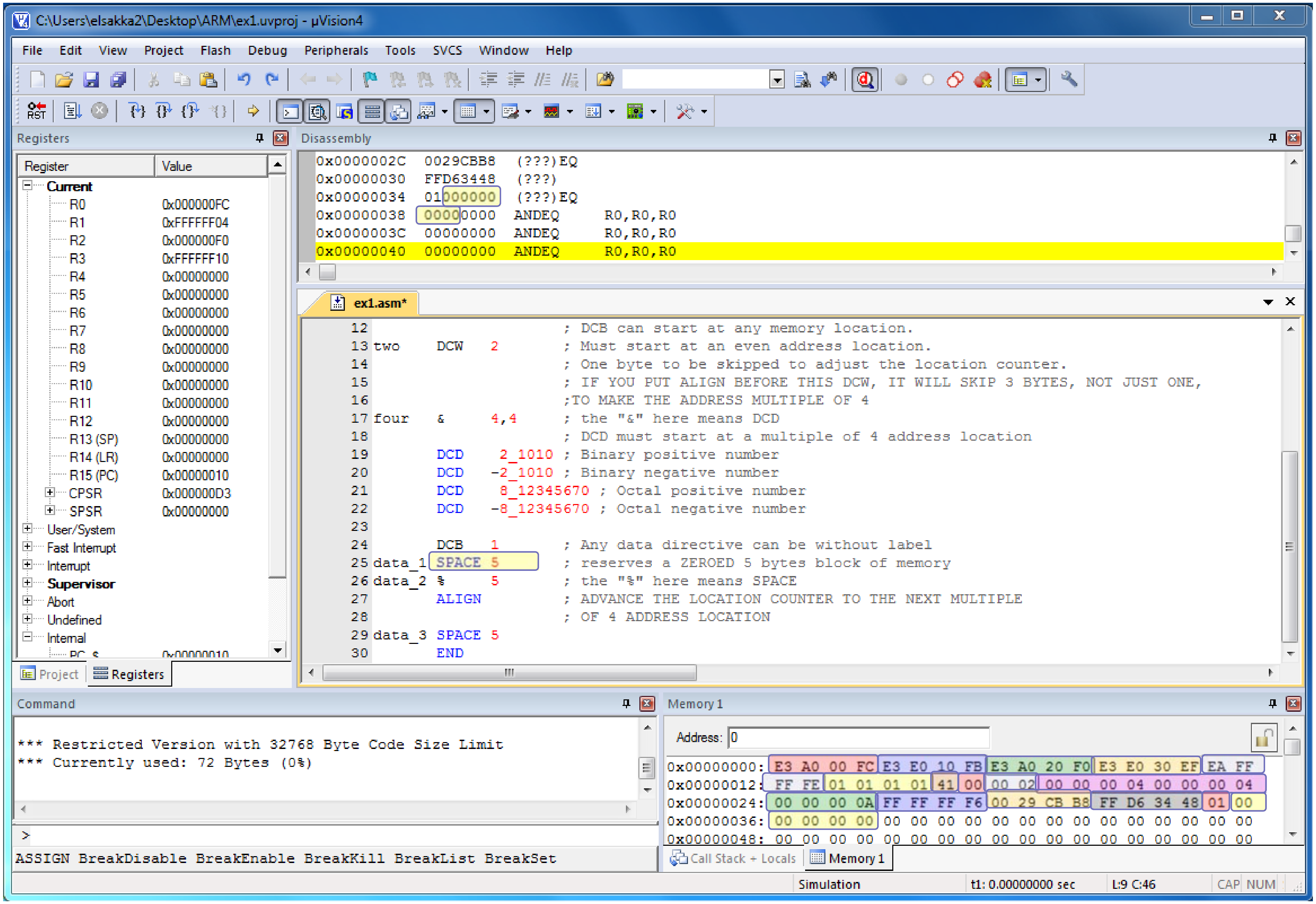
```
                                ; DCB can start at any memory location.
two     DCW    2                ; Must start at an even address location.
                                ; One byte to be skipped to adjust the location counter.
                                ; IF YOU PUT ALIGN BEFORE THIS DCW, IT WILL SKIP 3 BYTES, NOT JUST ONE,
                                ;TO MAKE THE ADDRESS MULTIPLE OF 4
four    &      4,4              ; the "&" here means DCD
                                ; DCD must start at a multiple of 4 address location
        DCD    2_1010           ; Binary positive number
        DCD    -2_1010          ; Binary negative number
        DCD    8_12345670 ; Octal positive number
        DCD    -8_12345670 ; Octal negative number

        DCB    1                ; Any data directive can be without label
data_1  SPACE  5                ; reserves a ZEROED 5 bytes block of memory
data_2  %      5                ; the "%" here means SPACE
        ALIGN                   ; ADVANCE THE LOCATION COUNTER TO THE NEXT MULTIPLE
                                ; OF 4 ADDRESS LOCATION
data_3  SPACE  5
        END
```

```
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 72 Bytes (0%)

>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet
```

```
Address: 0
0x00000000: E3 A0 00 FC E3 E0 10 FB E3 A0 20 F0 E3 E0 30 EF EA FF
0x00000012: FF FE 01 01 01 01 41 00 00 02 00 00 00 04 00 00 00 04
0x00000024: 00 00 00 0A FF FF FF F6 00 29 CB B8 FF D6 34 48 01 00
0x00000036: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00000048: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Simulation t1: 0.00000000 sec L:9 C:46 CAP NUM

Data Definition Directives


C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4

Registers

Register	Value
Current	
R0	0x000000FC
R1	0xFFFFFF04
R2	0x000000F0
R3	0xFFFFFF10
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
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x00000010

Disassembly

```
0x0000002C  0029CBB8  (???)EQ
0x00000030  FFD63448  (???)
0x00000034  01000000  (???)EQ
0x00000038  00000000  ANDEQ     R0,R0,R0
0x0000003C  00000000  ANDEQ     R0,R0,R0
0x00000040  00000000  ANDEQ     R0,R0,R0
```

ex1.asm*

```
                                ; DCB can start at any memory location.
two     DCW    2                ; Must start at an even address location.
                                ; One byte to be skipped to adjust the location counter.
                                ; IF YOU PUT ALIGN BEFORE THIS DCW, IT WILL SKIP 3 BYTES, NOT JUST ONE,
                                ;TO MAKE THE ADDRESS MULTIPLE OF 4
four    &      4,4              ; the "&" here means DCD
                                ; DCD must start at a multiple of 4 address location
        DCD     2_1010 ; Binary positive number
        DCD    -2_1010 ; Binary negative number
        DCD     8_12345670 ; Octal positive number
        DCD    -8_12345670 ; Octal negative number

        DCB    1                ; Any data directive can be without label
data_1  SPACE  5                ; reserves a ZEROED 5 bytes block of memory
data_2  %      5                ; the "%" here means SPACE
        ALIGN                   ; ADVANCE THE LOCATION COUNTER TO THE NEXT MULTIPLE
                                ; OF 4 ADDRESS LOCATION
data_3  SPACE  5
        END
```

Command

```
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 72 Bytes (0%)
>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet
```

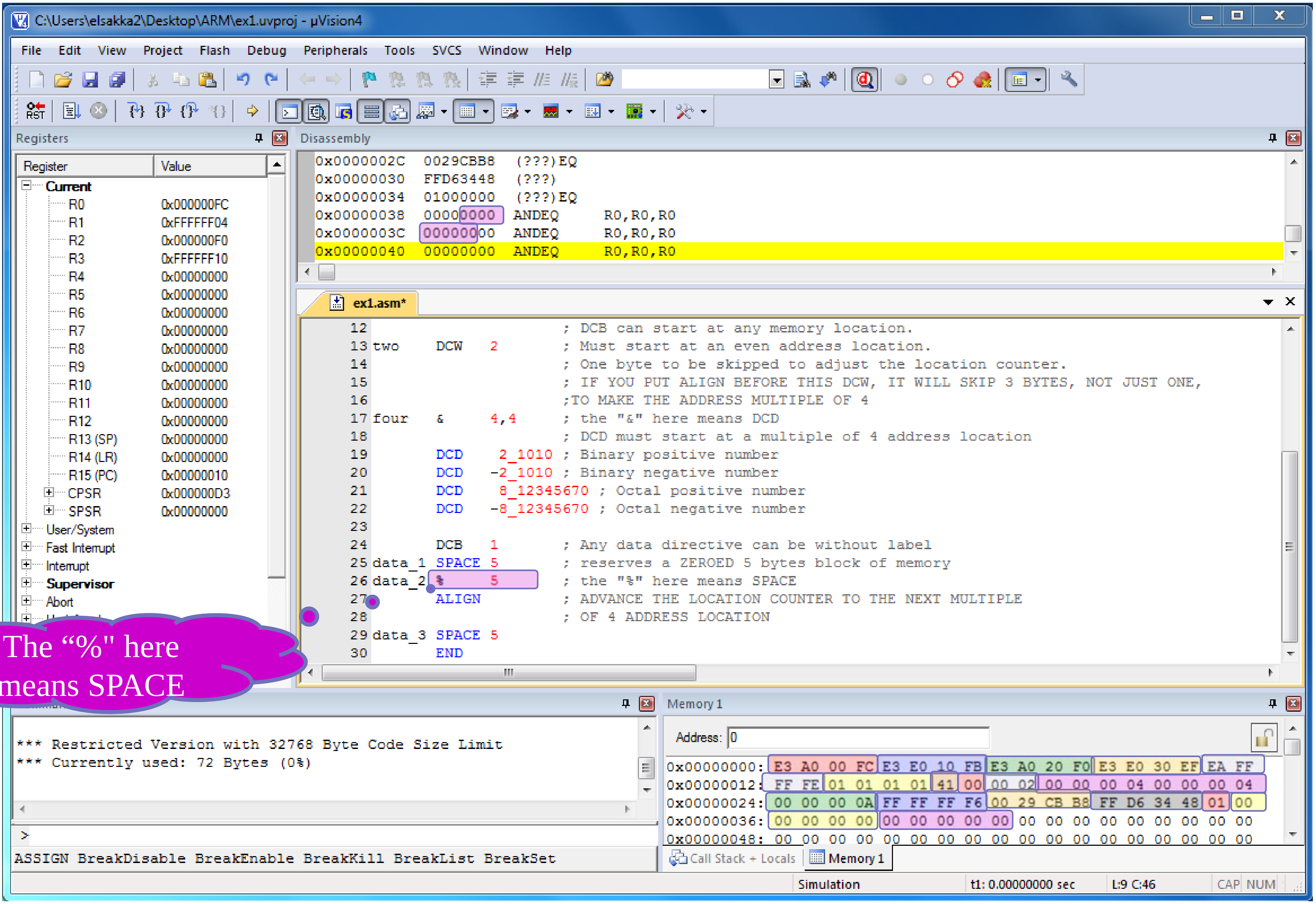
Memory 1

Address: 0

```
0x00000000: E3 A0 00 FC E3 E0 10 FB E3 A0 20 F0 E3 E0 30 EF EA FF
0x00000012: FF FE 01 01 01 01 41 00 00 02 00 00 00 04 00 00 00 04
0x00000024: 00 00 00 0A FF FF FF F6 00 29 CB B8 FF D6 34 48 01 00
0x00000036: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00000048: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Simulation t1: 0.00000000 sec L:9 C:46 CAP NUM

Data Definition Directives


C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Disassembly
ex1.asm*
Memory 1
Call Stack + Locals
Simulation
t1: 0.00000000 sec
L:9 C:46
CAP NUM

Register	Value
Current	
R0	0x000000FC
R1	0xFFFFFF04
R2	0x000000F0
R3	0xFFFFFF10
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
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	

```
0x0000002C  0029CBB8  (???)EQ
0x00000030  FFD63448  (???)
0x00000034  01000000  (???)EQ
0x00000038  00000000  ANDEQ     R0,R0,R0
0x0000003C  00000000  ANDEQ     R0,R0,R0
0x00000040  00000000  ANDEQ     R0,R0,R0
```

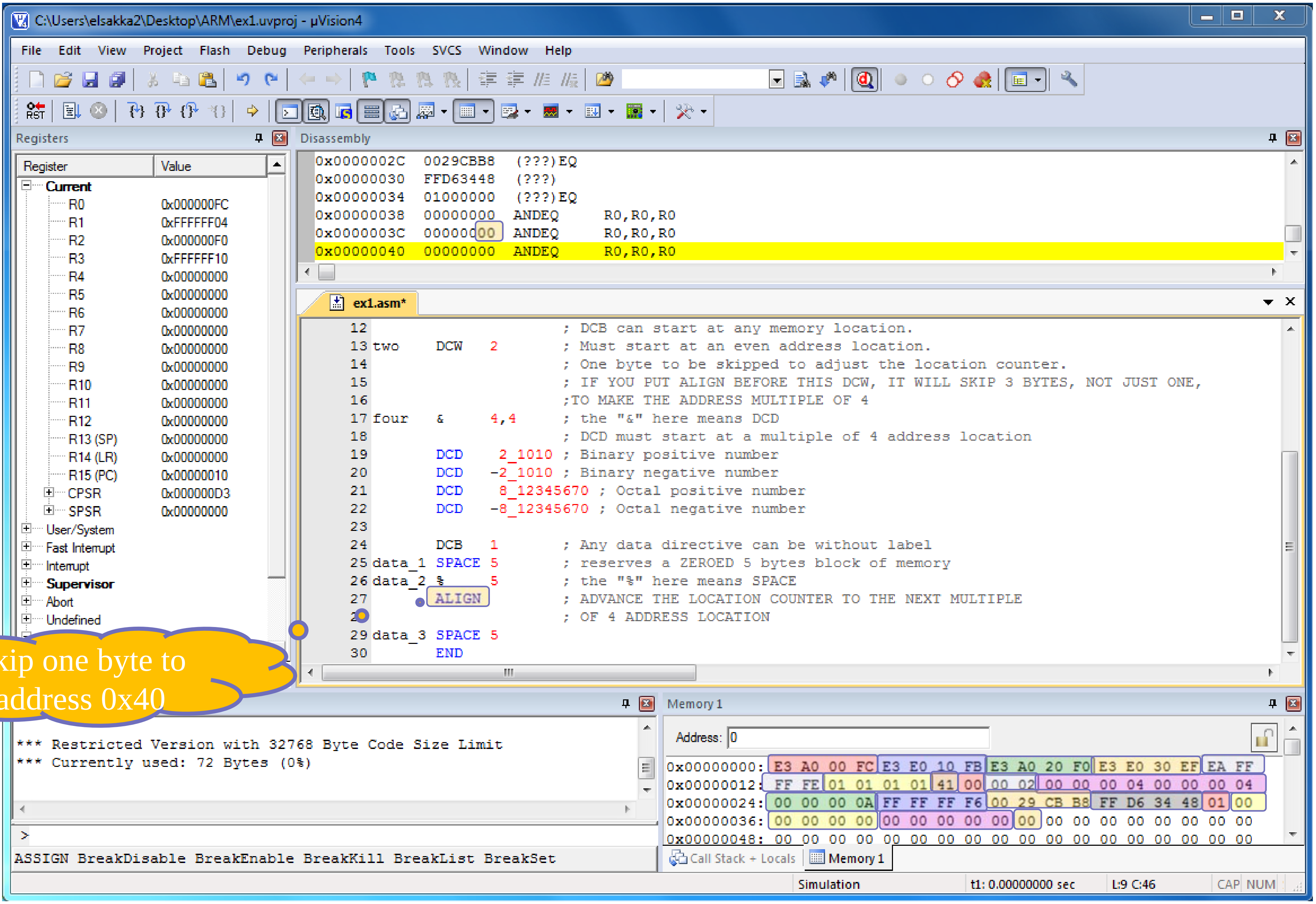
```
                                ; DCB can start at any memory location.
two     DCW     2               ; Must start at an even address location.
                                ; One byte to be skipped to adjust the location counter.
                                ; IF YOU PUT ALIGN BEFORE THIS DCW, IT WILL SKIP 3 BYTES, NOT JUST ONE,
                                ;TO MAKE THE ADDRESS MULTIPLE OF 4
four    &       4,4             ; the "&" here means DCD
                                ; DCD must start at a multiple of 4 address location
        DCD     2_1010          ; Binary positive number
        DCD     -2_1010         ; Binary negative number
        DCD     8_12345670      ; Octal positive number
        DCD     -8_12345670     ; Octal negative number

        DCB     1               ; Any data directive can be without label
data_1  SPACE   5               ; reserves a ZEROED 5 bytes block of memory
data_2  %       5               ; the "%" here means SPACE
        ALIGN                   ; ADVANCE THE LOCATION COUNTER TO THE NEXT MULTIPLE
                                ; OF 4 ADDRESS LOCATION
data_3  SPACE   5
        END
```

```
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 72 Bytes (0%)

>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet
```

```
Address: 0
0x00000000: E3 A0 00 FC E3 E0 10 FB E3 A0 20 F0 E3 E0 30 EF EA FF
0x00000012: FF FE 01 01 01 01 41 00 00 02 00 00 00 04 00 00 00 04
0x00000024: 00 00 00 0A FF FF FF F6 00 29 CB B8 FF D6 34 48 01 00
0x00000036: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00000048: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

The “%" here means SPACE

Data Definition Directives


C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Disassembly
ex1.asm*
Memory 1
Call Stack + Locals
Skip one byte to address 0x40

Register	Value
Current	
R0	0x000000FC
R1	0xFFFFFF04
R2	0x000000F0
R3	0xFFFFFF10
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
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	

```
0x0000002C  0029CBB8  (???)EQ
0x00000030  FFD63448  (???)
0x00000034  01000000  (???)EQ
0x00000038  00000000  ANDEQ     R0,R0,R0
0x0000003C  00000000  ANDEQ     R0,R0,R0
0x00000040  00000000  ANDEQ     R0,R0,R0
```

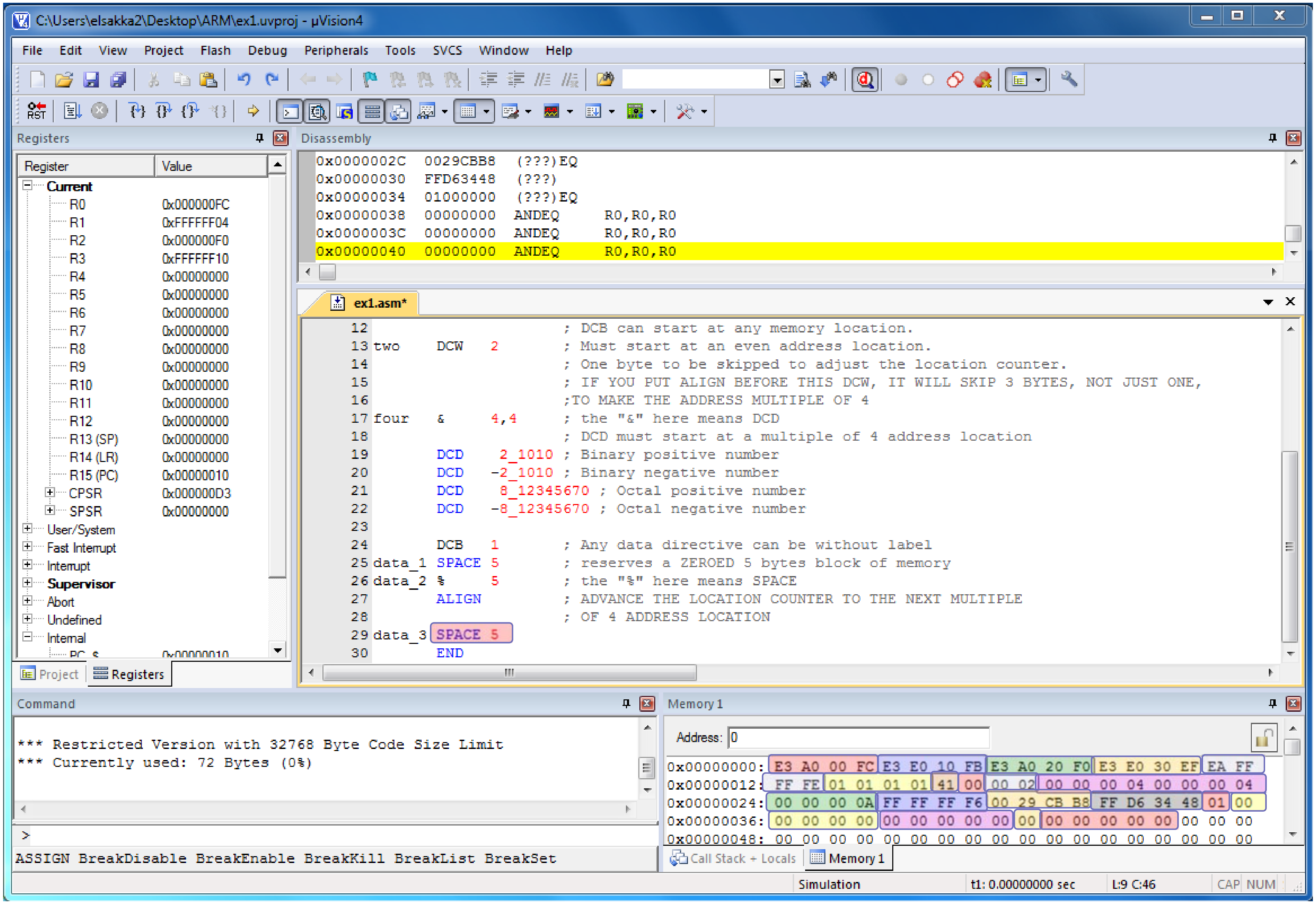
```
                                ; DCB can start at any memory location.
two     DCW    2                ; Must start at an even address location.
                                ; One byte to be skipped to adjust the location counter.
                                ; IF YOU PUT ALIGN BEFORE THIS DCW, IT WILL SKIP 3 BYTES, NOT JUST ONE,
                                ;TO MAKE THE ADDRESS MULTIPLE OF 4
four    &      4,4              ; the "&" here means DCD
                                ; DCD must start at a multiple of 4 address location
        DCD     2_1010 ; Binary positive number
        DCD    -2_1010 ; Binary negative number
        DCD     8_12345670 ; Octal positive number
        DCD    -8_12345670 ; Octal negative number

        DCB    1                ; Any data directive can be without label
data_1  SPACE  5                ; reserves a ZEROED 5 bytes block of memory
data_2  %      5                ; the "%" here means SPACE
        ALIGN                   ; ADVANCE THE LOCATION COUNTER TO THE NEXT MULTIPLE
                                ; OF 4 ADDRESS LOCATION
data_3  SPACE  5
        END
```

```
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 72 Bytes (0%)

>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet
```

```
Address: 0
0x00000000: E3 A0 00 FC E3 E0 10 FB E3 A0 20 F0 E3 E0 30 EF EA FF
0x00000012: FF FE 01 01 01 01 41 00 00 02 00 00 00 04 00 00 00 04
0x00000024: 00 00 00 0A FF FF FF F6 00 29 CB B8 FF D6 34 48 01 00
0x00000036: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00000048: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

```
Simulation    t1: 0.00000000 sec    L:9 C:46    CAP NUM
```

Data Definition Directives



C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4

Register	Value
Current	
R0	0x000000FC
R1	0xFFFFFF04
R2	0x000000F0
R3	0xFFFFFF10
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
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x00000010

Disassembly

```
0x0000002C  0029CBB8  (???)EQ
0x00000030  FFD63448  (???)
0x00000034  01000000  (???)EQ
0x00000038  00000000  ANDEQ     R0,R0,R0
0x0000003C  00000000  ANDEQ     R0,R0,R0
0x00000040  00000000  ANDEQ     R0,R0,R0
```

ex1.asm*

```
                          ; DCB can start at any memory location.
two     DCW    2          ; Must start at an even address location.
                          ; One byte to be skipped to adjust the location counter.
                          ; IF YOU PUT ALIGN BEFORE THIS DCW, IT WILL SKIP 3 BYTES, NOT JUST ONE,
                          ;TO MAKE THE ADDRESS MULTIPLE OF 4
four    &      4,4        ; the "&" here means DCD
                          ; DCD must start at a multiple of 4 address location
        DCD     2_1010 ; Binary positive number
        DCD    -2_1010 ; Binary negative number
        DCD     8_12345670 ; Octal positive number
        DCD    -8_12345670 ; Octal negative number

        DCB    1          ; Any data directive can be without label
data_1  SPACE  5          ; reserves a ZEROED 5 bytes block of memory
data_2  %      5          ; the "%" here means SPACE
        ALIGN             ; ADVANCE THE LOCATION COUNTER TO THE NEXT MULTIPLE
                          ; OF 4 ADDRESS LOCATION
data_3  SPACE  5
        END
```

Command

```
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 72 Bytes (0%)

>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet
```

Memory 1

Address: 0

```
0x00000000: E3 A0 00 FC E3 E0 10 FB E3 A0 20 F0 E3 E0 30 EF EA FF
0x00000012: FF FE 01 01 01 01 41 00 00 02 00 00 00 04 00 00 00 04
0x00000024: 00 00 00 0A FF FF FF F6 00 29 CB B8 FF D6 34 48 01 00
0x00000036: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00000048: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

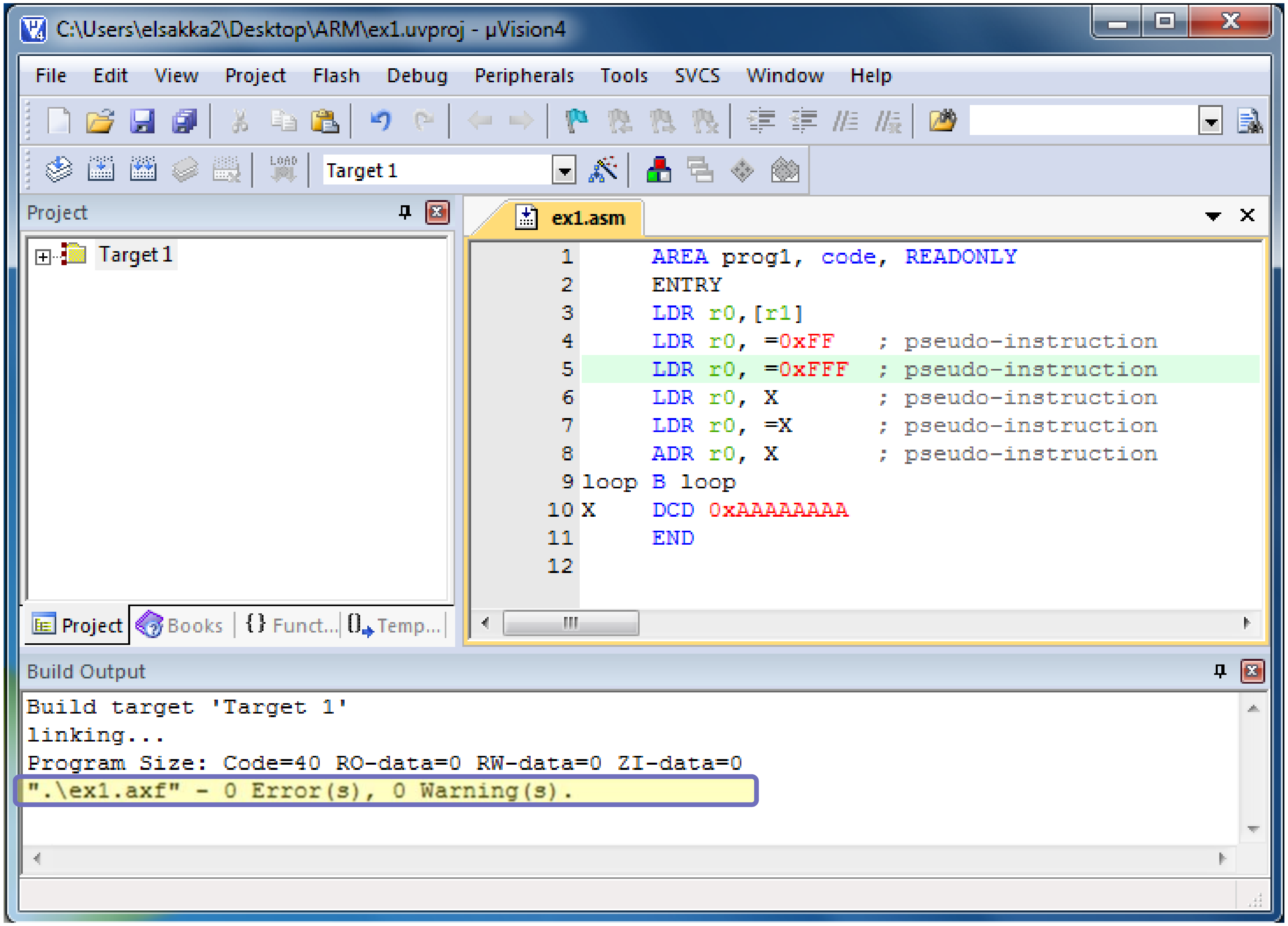
Simulation   t1: 0.00000000 sec   L:9 C:46   CAP NUM

ARM pseudo-instructions

- The ARM assembler supports a number of pseudo-instructions that are translated into the appropriate combination of ARM instructions at assembly time.
- Consider the following assembly program:

```
     AREA prog1, code, READONLY
     ENTRY
     LDR r0,[r1]
     LDR r0, =0xFF   ; pseudo-instruction
     LDR r0, =0xFFF  ; pseudo-instruction
     LDR r0, X       ; pseudo-instruction
     LDR r0, =X      ; pseudo-instruction
     ADR r0, X       ; pseudo-instruction
loop B loop
X    DCD 0xAAAAAAAA
     END
```

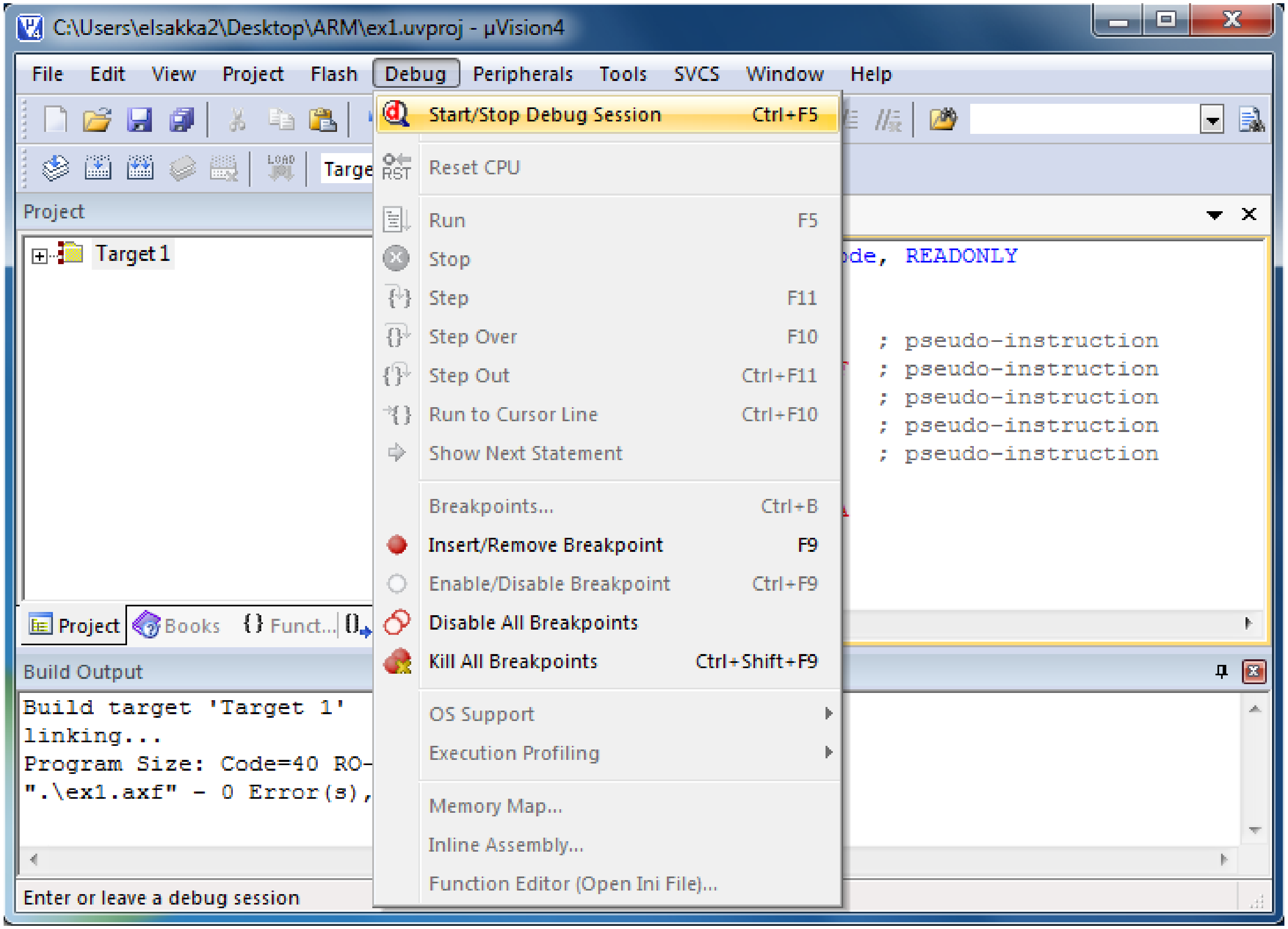
ARM pseudo-instructions


C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Target 1
Project
Target 1
Project Books {} Funct... Temp...
ex1.asm
Build Output

```
        AREA prog1, code, READONLY
        ENTRY
        LDR r0,[r1]
        LDR r0, =0xFF    ; pseudo-instruction
        LDR r0, =0xFFF   ; pseudo-instruction
        LDR r0, X        ; pseudo-instruction
        LDR r0, =X       ; pseudo-instruction
        ADR r0, X        ; pseudo-instruction
loop B loop
X       DCD 0xAAAAAAAA
        END
```

```
Build target 'Target 1'
linking...
Program Size: Code=40 RO-data=0 RW-data=0 ZI-data=0
".\ex1.axf" - 0 Error(s), 0 Warning(s).
```

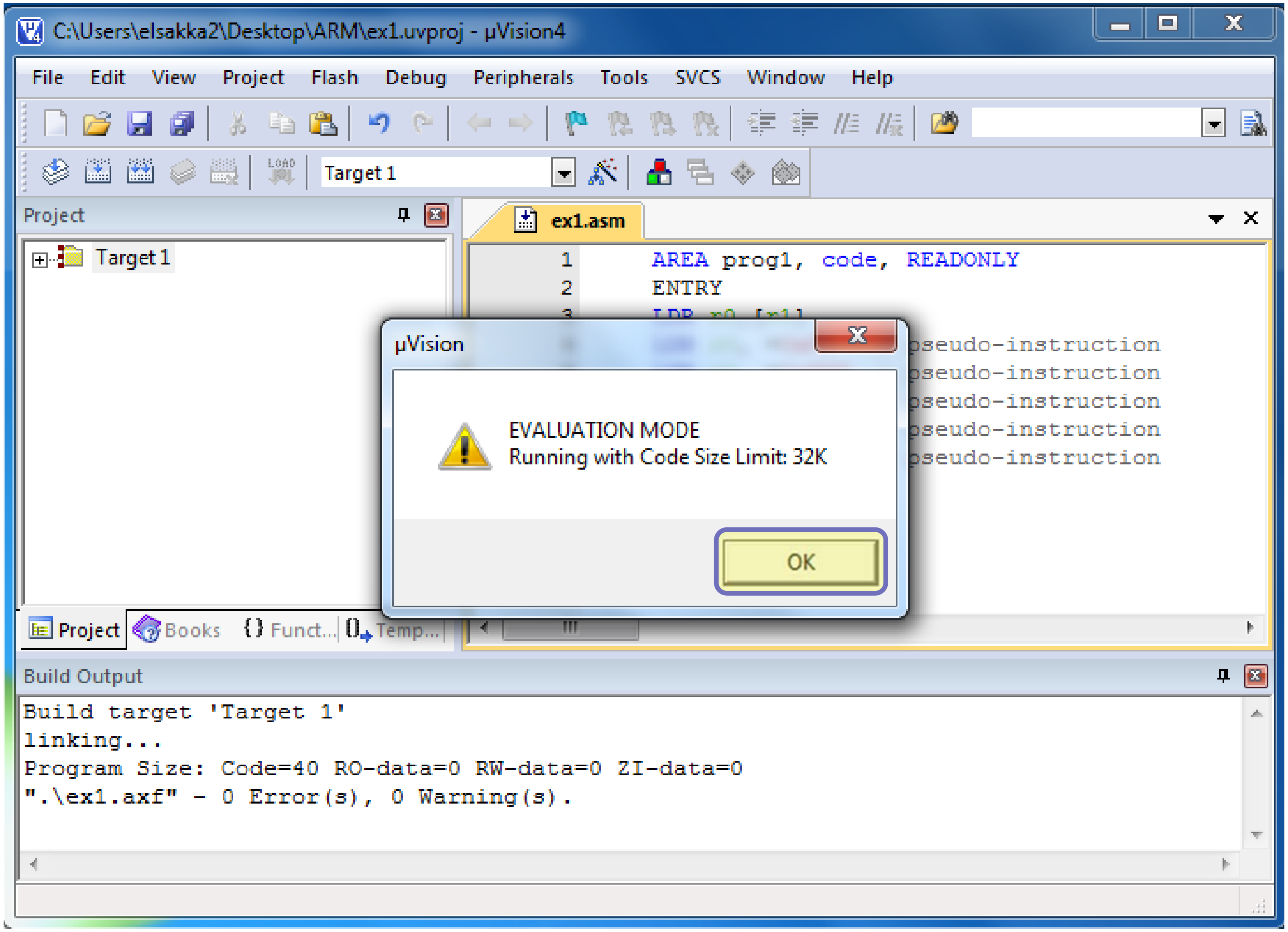
ARM pseudo-instructions


C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Start/Stop Debug Session Ctrl+F5
Reset CPU
Run F5
Stop
Step F11
Step Over F10
Step Out Ctrl+F11
Run to Cursor Line Ctrl+F10
Show Next Statement
Breakpoints... Ctrl+B
Insert/Remove Breakpoint F9
Enable/Disable Breakpoint Ctrl+F9
Disable All Breakpoints
Kill All Breakpoints Ctrl+Shift+F9
OS Support
Execution Profiling
Memory Map...
Inline Assembly...
Function Editor (Open Ini File)...
Project
Target 1
Project Books Funct...
Build Output
Enter or leave a debug session

```
ode, READONLY

 ; pseudo-instruction
 ; pseudo-instruction
 ; pseudo-instruction
 ; pseudo-instruction
 ; pseudo-instruction
```

```
Build target 'Target 1'
linking...
Program Size: Code=40 RO-
".\ex1.axf" - 0 Error(s),
```

ARM pseudo-instructions


C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Target 1
Project
ex1.asm
Project Books Funct... Temp...
Build Output
OK

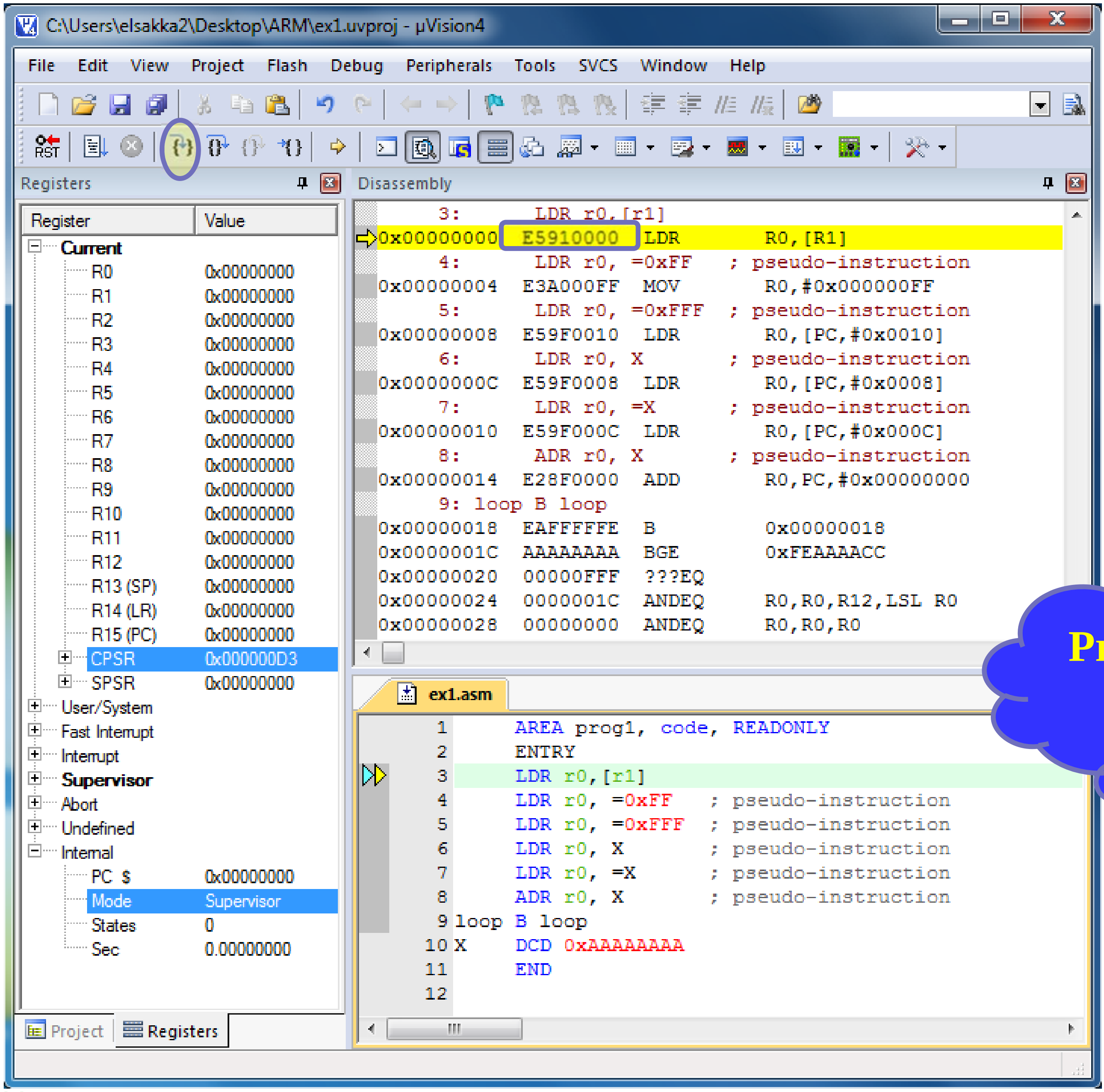
```
        AREA prog1, code, READONLY
        ENTRY
        LDR r0,[r1]
                            pseudo-instruction
                            pseudo-instruction
                            pseudo-instruction
                            pseudo-instruction
                            pseudo-instruction
```

µVision

EVALUATION MODE
Running with Code Size Limit: 32K

```
Build target 'Target 1'
linking...
Program Size: Code=40 RO-data=0 RW-data=0 ZI-data=0
".\ex1.axf" - 0 Error(s), 0 Warning(s).
```

ARM pseudo-instructions


C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Disassembly
ex1.asm
Project
Registers

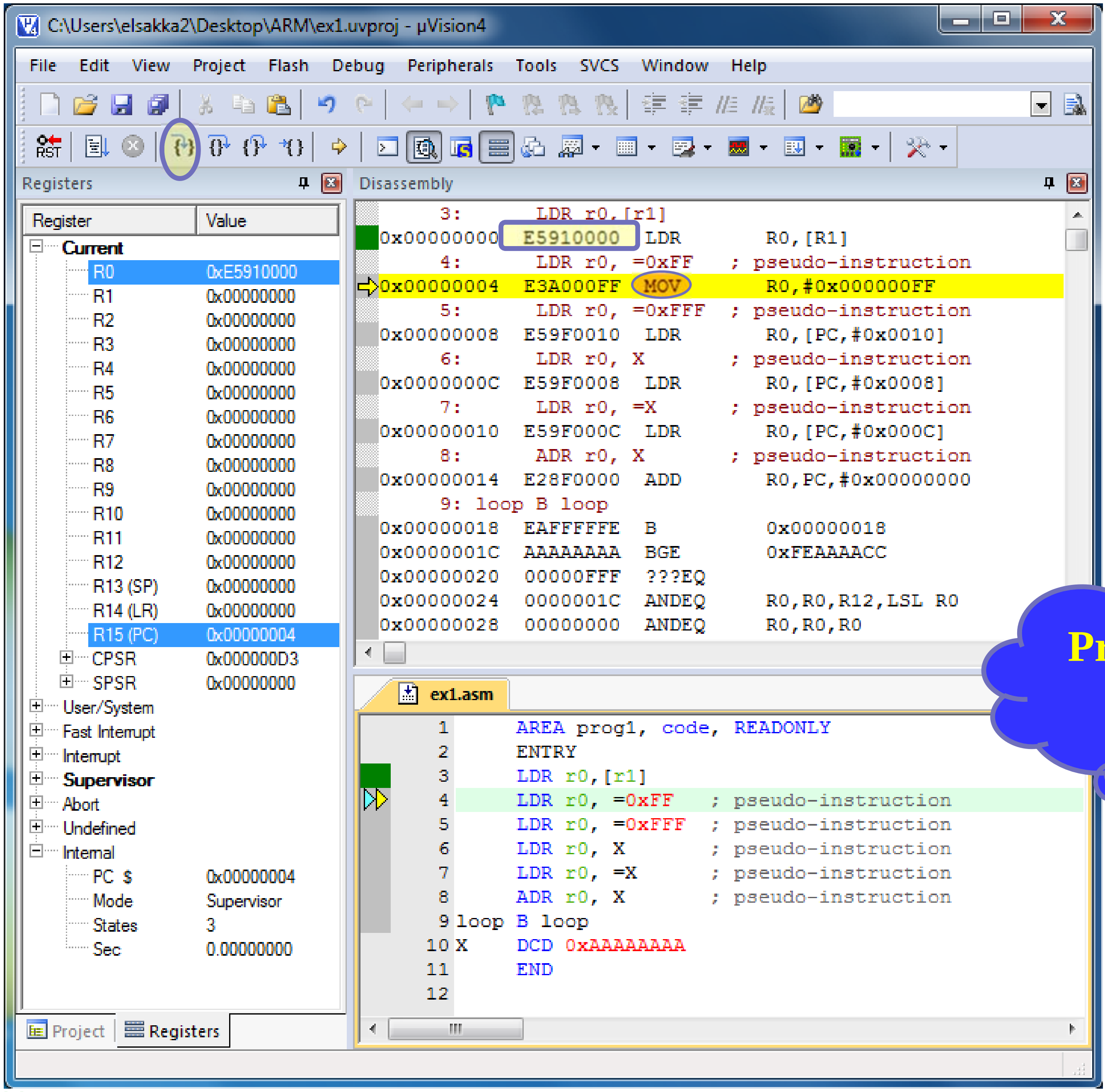
Register	Value
Current	
R0	0x00000000
R1	0x00000000
R2	0x00000000
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x00000000
CPSR	0x000000D3
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x00000000
Mode	Supervisor
States	0
Sec	0.00000000

```
     3:        LDR r0,[r1]
0x00000000  E5910000  LDR       R0,[R1]
     4:        LDR r0, =0xFF    ; pseudo-instruction
0x00000004  E3A000FF  MOV       R0,#0x000000FF
     5:        LDR r0, =0xFFF   ; pseudo-instruction
0x00000008  E59F0010  LDR       R0,[PC,#0x0010]
     6:        LDR r0, X        ; pseudo-instruction
0x0000000C  E59F0008  LDR       R0,[PC,#0x0008]
     7:        LDR r0, =X       ; pseudo-instruction
0x00000010  E59F000C  LDR       R0,[PC,#0x000C]
     8:        ADR r0, X        ; pseudo-instruction
0x00000014  E28F0000  ADD       R0,PC,#0x00000000
     9: loop B loop
0x00000018  EAFFFFFE  B         0x00000018
0x0000001C  AAAAAAAA  BGE       0xFEAAAACC
0x00000020  00000FFF  ???EQ
0x00000024  0000001C  ANDEQ     R0,R0,R12,LSL R0
0x00000028  00000000  ANDEQ     R0,R0,R0
```

```
        AREA prog1, code, READONLY
        ENTRY
        LDR r0,[r1]
        LDR r0, =0xFF    ; pseudo-instruction
        LDR r0, =0xFFF   ; pseudo-instruction
        LDR r0, X        ; pseudo-instruction
        LDR r0, =X       ; pseudo-instruction
        ADR r0, X        ; pseudo-instruction
loop    B loop
X       DCD 0xAAAAAAAA
        END
```

Press Step, or F11

ARM pseudo-instructions



C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4

File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
RST
Registers
Disassembly
Project
Registers
ex1.asm

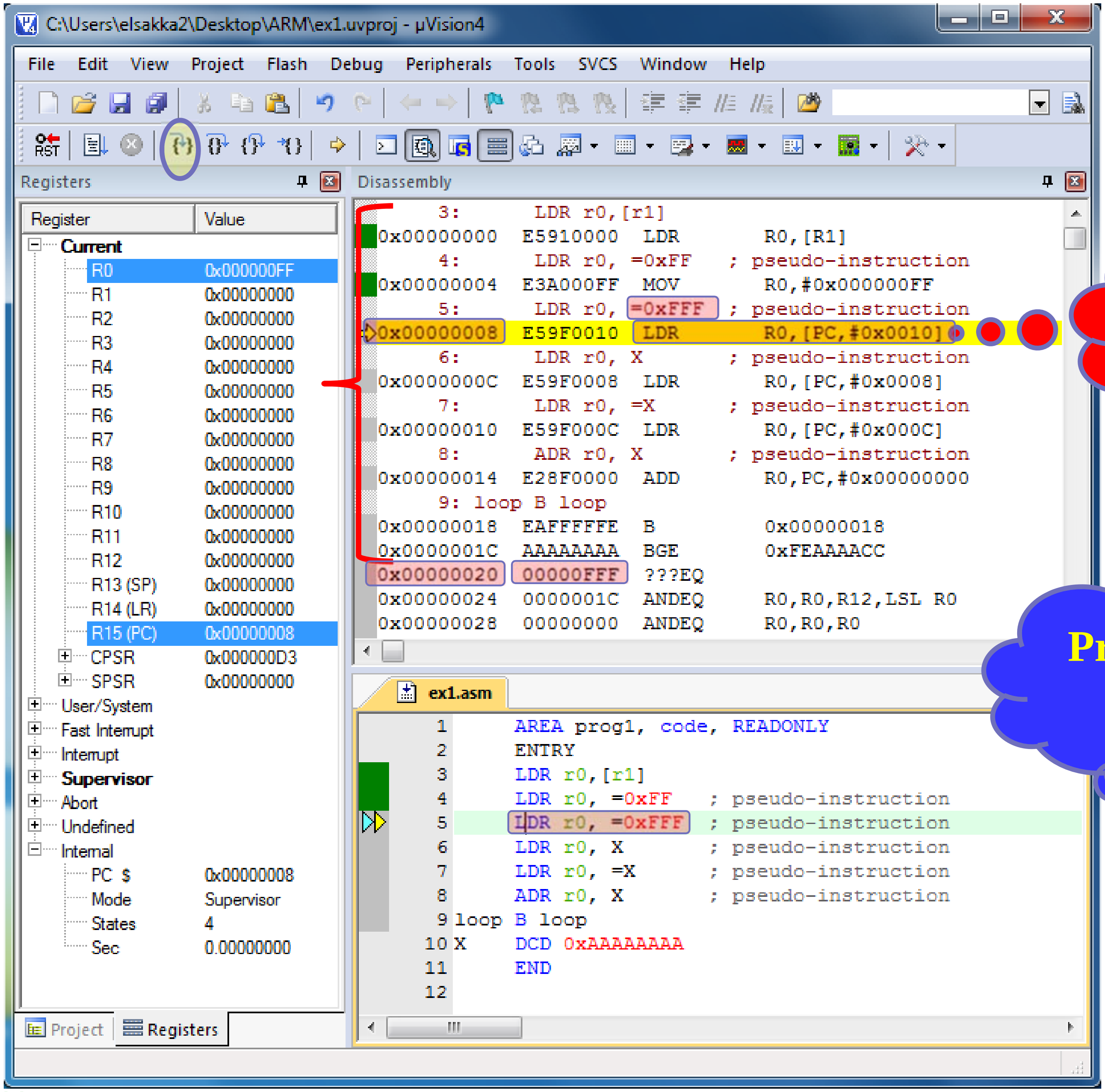
Register	Value
Current	
R0	0xE5910000
R1	0x00000000
R2	0x00000000
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x00000004
CPSR	0x000000D3
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x00000004
Mode	Supervisor
States	3
Sec	0.00000000

```
      3:       LDR r0,[r1]
0x00000000  E5910000  LDR      R0,[R1]
      4:       LDR r0, =0xFF    ; pseudo-instruction
0x00000004  E3A000FF  MOV      R0,#0x000000FF
      5:       LDR r0, =0xFFF   ; pseudo-instruction
0x00000008  E59F0010  LDR      R0,[PC,#0x0010]
      6:       LDR r0, X        ; pseudo-instruction
0x0000000C  E59F0008  LDR      R0,[PC,#0x0008]
      7:       LDR r0, =X       ; pseudo-instruction
0x00000010  E59F000C  LDR      R0,[PC,#0x000C]
      8:       ADR r0, X        ; pseudo-instruction
0x00000014  E28F0000  ADD      R0,PC,#0x00000000
      9: loop B loop
0x00000018  EAFFFFFE  B        0x00000018
0x0000001C  AAAAAAAA  BGE      0xFEAAAACC
0x00000020  00000FFF  ???EQ
0x00000024  0000001C  ANDEQ    R0,R0,R12,LSL R0
0x00000028  00000000  ANDEQ    R0,R0,R0
```

```
 1      AREA prog1, code, READONLY
 2      ENTRY
 3      LDR r0,[r1]
 4      LDR r0, =0xFF    ; pseudo-instruction
 5      LDR r0, =0xFFF   ; pseudo-instruction
 6      LDR r0, X        ; pseudo-instruction
 7      LDR r0, =X       ; pseudo-instruction
 8      ADR r0, X        ; pseudo-instruction
 9 loop B loop
10 X    DCD 0xAAAAAAAA
11      END
12
```

Press Step,
or F11

ARM pseudo-instructions



C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Disassembly
ex1.asm
Project
Registers

Register	Value
Current	
R0	0x000000FF
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)	0x00000008
CPSR	0x000000D3
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x00000008
Mode	Supervisor
States	4
Sec	0.00000000

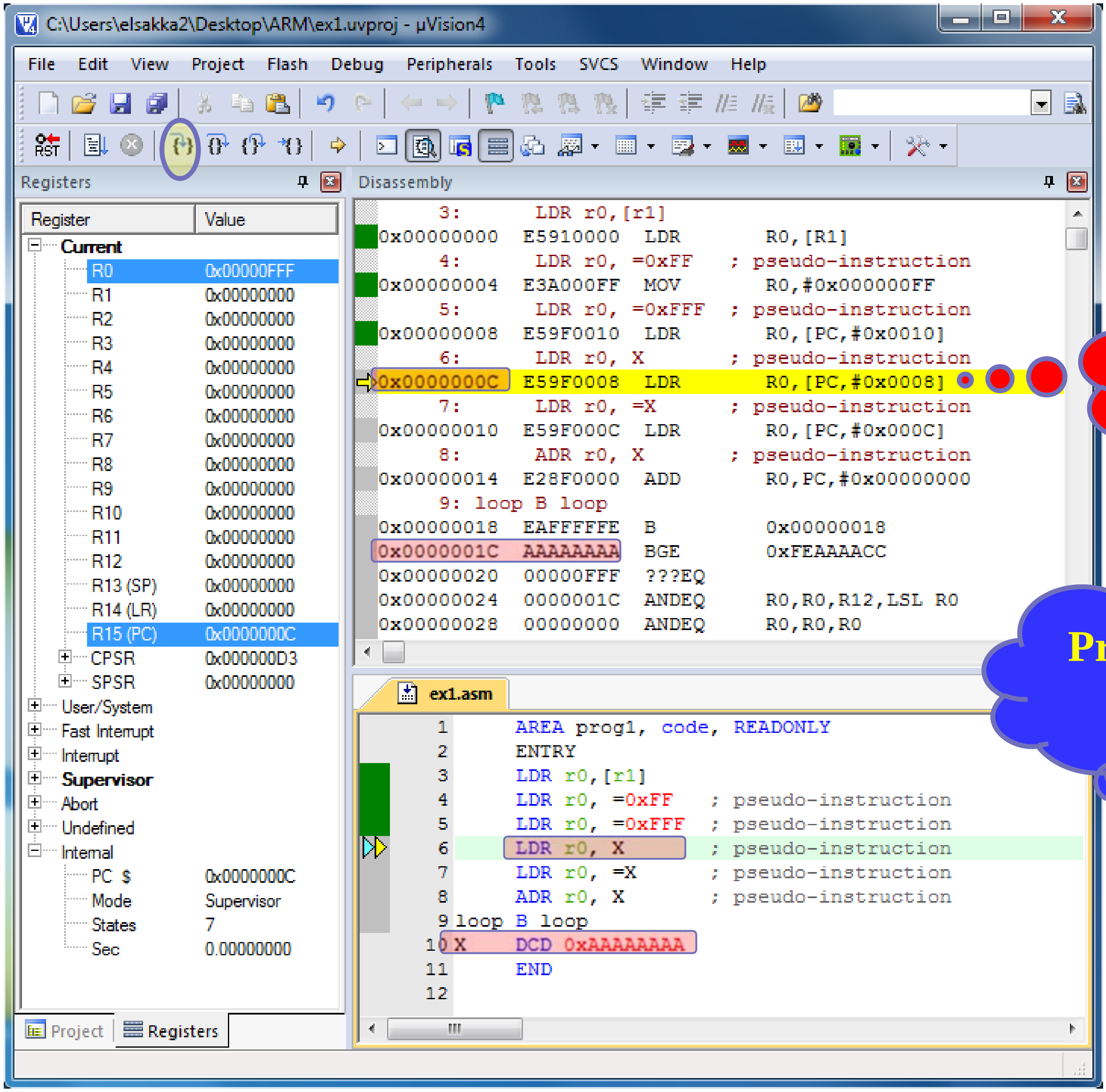
```
     3:         LDR r0,[r1]
0x00000000  E5910000  LDR      R0,[R1]
     4:         LDR r0, =0xFF   ; pseudo-instruction
0x00000004  E3A000FF  MOV      R0,#0x000000FF
     5:         LDR r0, =0xFFF  ; pseudo-instruction
0x00000008  E59F0010  LDR      R0,[PC,#0x0010]
     6:         LDR r0, X       ; pseudo-instruction
0x0000000C  E59F0008  LDR      R0,[PC,#0x0008]
     7:         LDR r0, =X      ; pseudo-instruction
0x00000010  E59F000C  LDR      R0,[PC,#0x000C]
     8:         ADR r0, X       ; pseudo-instruction
0x00000014  E28F0000  ADD      R0,PC,#0x00000000
     9: loop B loop
0x00000018  EAFFFFFE  B        0x00000018
0x0000001C  AAAAAAAA  BGE      0xFEAAAACC
0x00000020  00000FFF  ???EQ
0x00000024  0000001C  ANDEQ    R0,R0,R12,LSL R0
0x00000028  00000000  ANDEQ    R0,R0,R0
```

```
 1        AREA prog1, code, READONLY
 2        ENTRY
 3        LDR r0,[r1]
 4        LDR r0, =0xFF    ; pseudo-instruction
 5        LDR r0, =0xFFF   ; pseudo-instruction
 6        LDR r0, X        ; pseudo-instruction
 7        LDR r0, =X       ; pseudo-instruction
 8        ADR r0, X        ; pseudo-instruction
 9  loop  B loop
10  X     DCD 0xAAAAAAAA
11        END
12
```

How is this offset calculated?

Press Step, or F11

ARM pseudo-instructions



C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4

Registers

Register	Value
Current	
R0	0x00000FFF
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)	0x0000000C
CPSR	0x000000D3
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x0000000C
Mode	Supervisor
States	7
Sec	0.00000000

Disassembly

```
     3:        LDR r0,[r1]
0x00000000  E5910000  LDR       R0,[R1]
     4:        LDR r0, =0xFF   ; pseudo-instruction
0x00000004  E3A000FF  MOV       R0,#0x000000FF
     5:        LDR r0, =0xFFF  ; pseudo-instruction
0x00000008  E59F0010  LDR       R0,[PC,#0x0010]
     6:        LDR r0, X       ; pseudo-instruction
0x0000000C  E59F0008  LDR       R0,[PC,#0x0008]
     7:        LDR r0, =X      ; pseudo-instruction
0x00000010  E59F000C  LDR       R0,[PC,#0x000C]
     8:        ADR r0, X       ; pseudo-instruction
0x00000014  E28F0000  ADD       R0,PC,#0x00000000
     9: loop B loop
0x00000018  EAFFFFFE  B         0x00000018
0x0000001C  AAAAAAAA  BGE       0xFEAAAACC
0x00000020  00000FFF  ???EQ
0x00000024  0000001C  ANDEQ     R0,R0,R12,LSL R0
0x00000028  00000000  ANDEQ     R0,R0,R0
```

ex1.asm

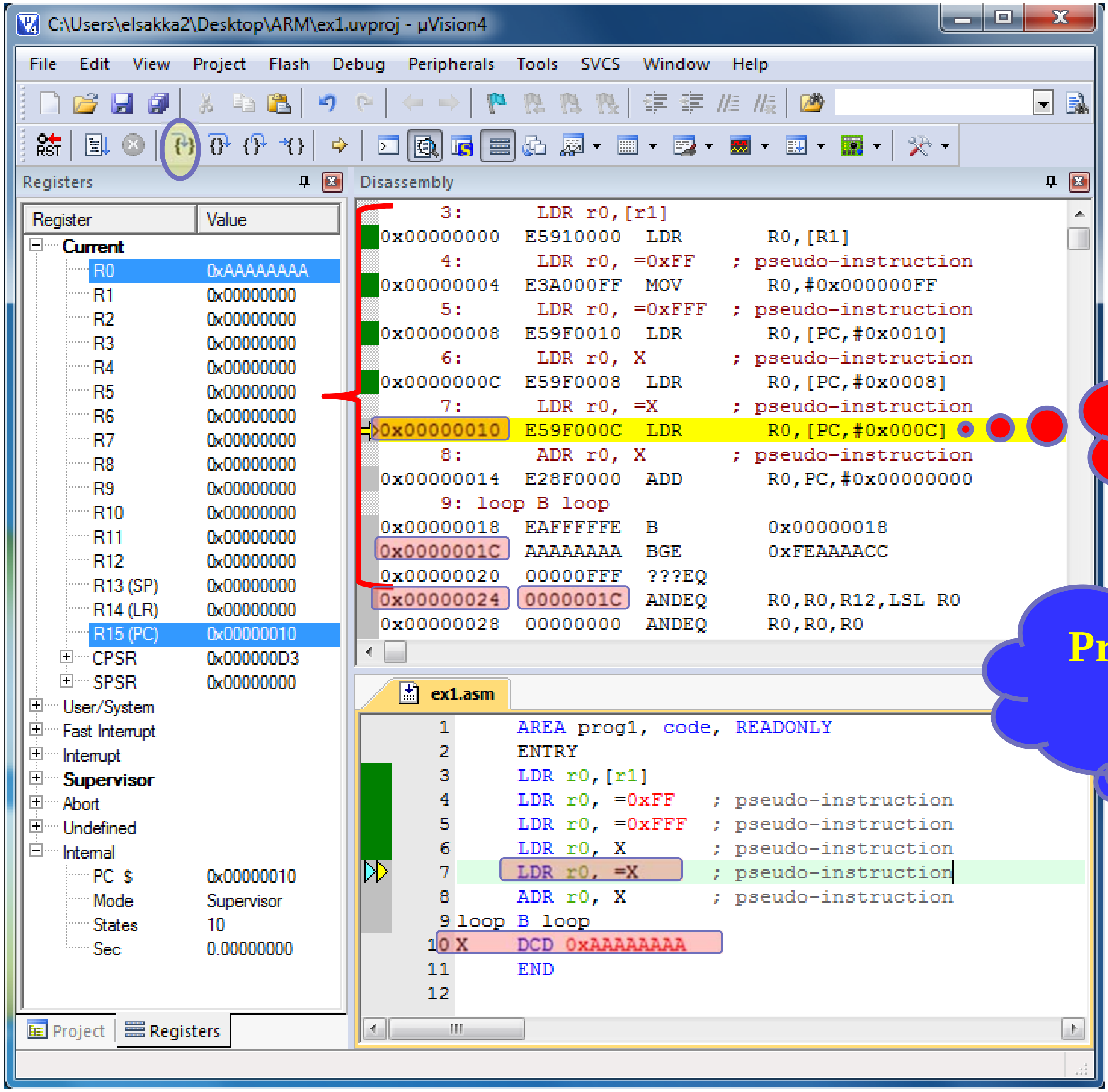
```
        AREA prog1, code, READONLY
        ENTRY
        LDR r0,[r1]
        LDR r0, =0xFF   ; pseudo-instruction
        LDR r0, =0xFFF  ; pseudo-instruction
        LDR r0, X       ; pseudo-instruction
        LDR r0, =X      ; pseudo-instruction
        ADR r0, X       ; pseudo-instruction
loop    B loop
X       DCD 0xAAAAAAAA
        END
```

File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Project
Registers

How is this offset calculated?

Press Step, or F11

ARM pseudo-instructions

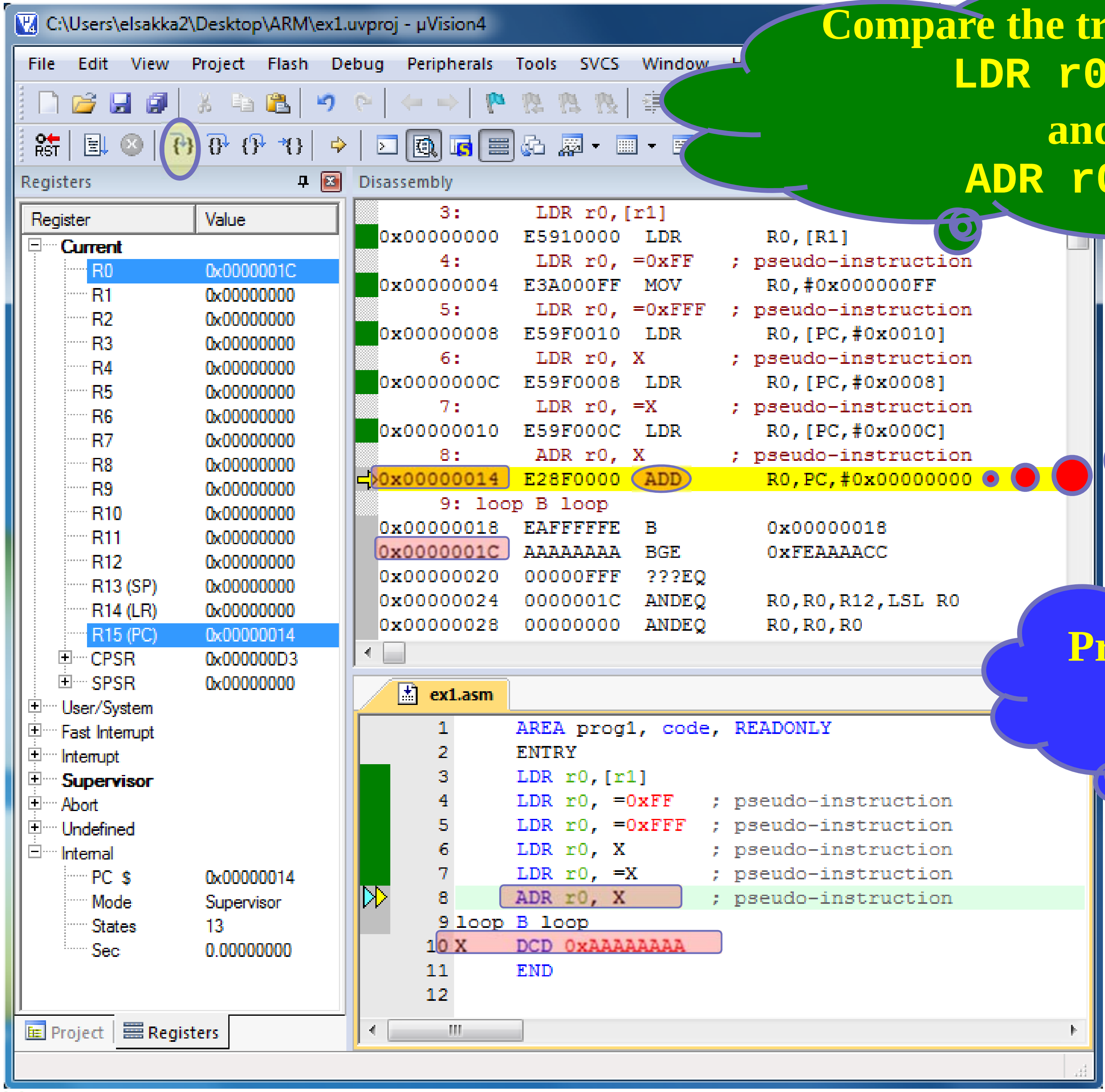

C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Disassembly
ex1.asm
Project
Registers
How is this offset calculated?
Press Step, or F11

Register	Value
Current	
R0	0xAAAAAAAA
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)	0x00000010
CPSR	0x000000D3
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x00000010
Mode	Supervisor
States	10
Sec	0.00000000

```
     3:      LDR r0,[r1]
0x00000000  E5910000  LDR      R0,[R1]
     4:      LDR r0, =0xFF   ; pseudo-instruction
0x00000004  E3A000FF  MOV      R0,#0x000000FF
     5:      LDR r0, =0xFFF  ; pseudo-instruction
0x00000008  E59F0010  LDR      R0,[PC,#0x0010]
     6:      LDR r0, X       ; pseudo-instruction
0x0000000C  E59F0008  LDR      R0,[PC,#0x0008]
     7:      LDR r0, =X      ; pseudo-instruction
0x00000010  E59F000C  LDR      R0,[PC,#0x000C]
     8:      ADR r0, X       ; pseudo-instruction
0x00000014  E28F0000  ADD      R0,PC,#0x00000000
     9: loop B loop
0x00000018  EAFFFFFE  B        0x00000018
0x0000001C  AAAAAAAA  BGE      0xFEAAAACC
0x00000020  00000FFF  ???EQ
0x00000024  0000001C  ANDEQ    R0,R0,R12,LSL R0
0x00000028  00000000  ANDEQ    R0,R0,R0
```

```
        AREA prog1, code, READONLY
        ENTRY
        LDR r0,[r1]
        LDR r0, =0xFF   ; pseudo-instruction
        LDR r0, =0xFFF  ; pseudo-instruction
        LDR r0, X       ; pseudo-instruction
        LDR r0, =X      ; pseudo-instruction
        ADR r0, X       ; pseudo-instruction
loop    B loop
X       DCD 0xAAAAAAAA
        END
```

ARM pseudo-instructions


C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window
Registers
Disassembly
ex1.asm
Project
Registers

Register	Value
Current	
R0	0x0000001C
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	0x000000D3
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x00000014
Mode	Supervisor
States	13
Sec	0.00000000

```
     3:        LDR r0,[r1]
0x00000000  E5910000  LDR      R0,[R1]
     4:        LDR r0, =0xFF   ; pseudo-instruction
0x00000004  E3A000FF  MOV      R0,#0x000000FF
     5:        LDR r0, =0xFFF  ; pseudo-instruction
0x00000008  E59F0010  LDR      R0,[PC,#0x0010]
     6:        LDR r0, X       ; pseudo-instruction
0x0000000C  E59F0008  LDR      R0,[PC,#0x0008]
     7:        LDR r0, =X      ; pseudo-instruction
0x00000010  E59F000C  LDR      R0,[PC,#0x000C]
     8:        ADR r0, X       ; pseudo-instruction
0x00000014  E28F0000  ADD      R0,PC,#0x00000000
     9: loop B loop
0x00000018  EAFFFFFE  B        0x00000018
0x0000001C  AAAAAAAA  BGE      0xFEAAAACC
0x00000020  00000FFF  ???EQ
0x00000024  0000001C  ANDEQ    R0,R0,R12,LSL R0
0x00000028  00000000  ANDEQ    R0,R0,R0
```

```
        AREA prog1, code, READONLY
        ENTRY
        LDR r0,[r1]
        LDR r0, =0xFF   ; pseudo-instruction
        LDR r0, =0xFFF  ; pseudo-instruction
        LDR r0, X       ; pseudo-instruction
        LDR r0, =X      ; pseudo-instruction
        ADR r0, X       ; pseudo-instruction
loop    B loop
X       DCD 0xAAAAAAAA
        END
```

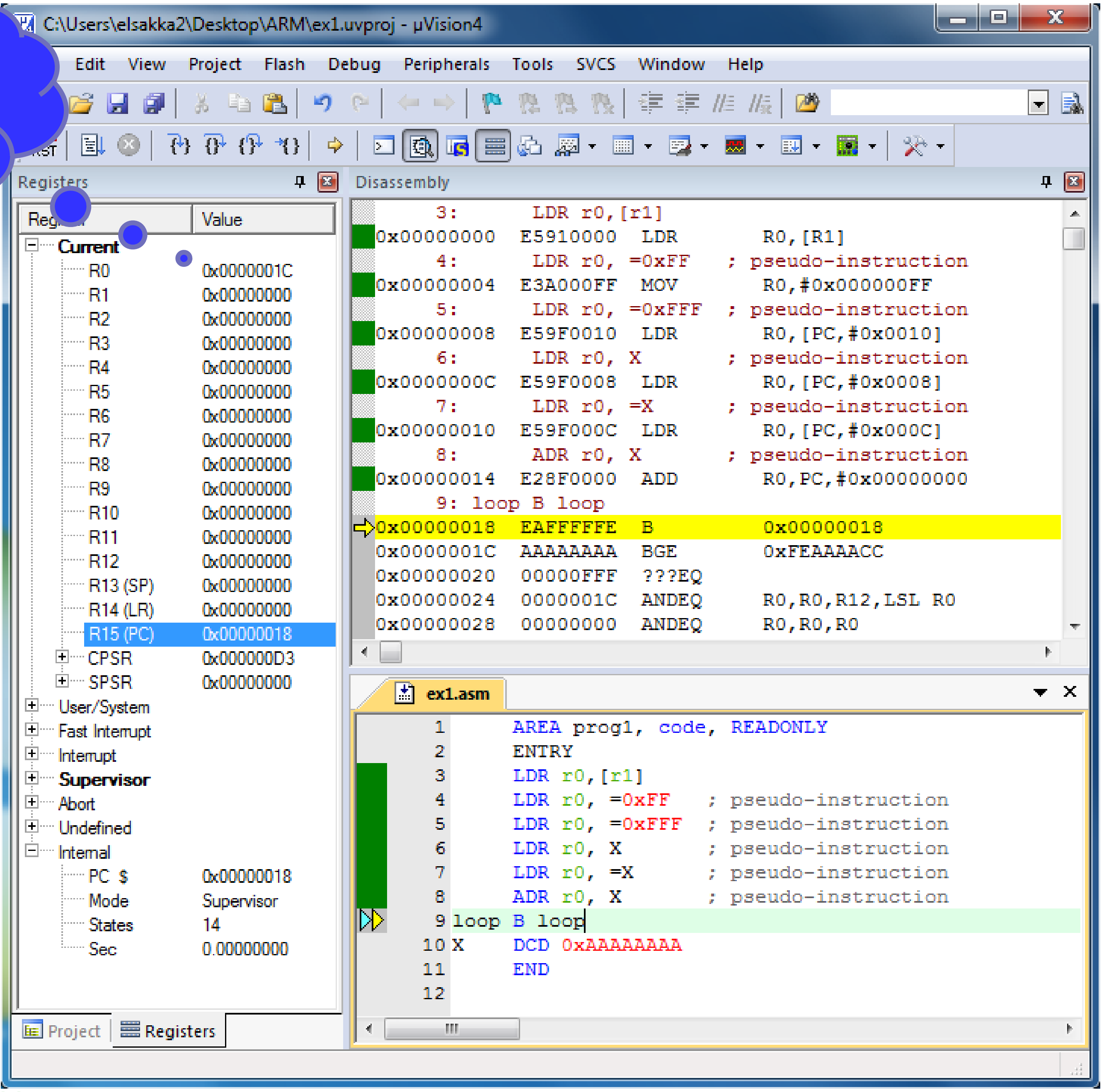
Compare the translations of
`LDR r0, =X`
and
`ADR r0, X`

How is this offset calculated?

Press Step, or F11

ARM pseudo-instructions

Same address (no change)


C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4
Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Disassembly
ex1.asm
Project
Registers

Register	Value
Current	
R0	0x0000001C
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)	0x00000018
CPSR	0x000000D3
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x00000018
Mode	Supervisor
States	14
Sec	0.00000000

```
     3:        LDR r0,[r1]
0x00000000  E5910000  LDR       R0,[R1]
     4:        LDR r0, =0xFF    ; pseudo-instruction
0x00000004  E3A000FF  MOV       R0,#0x000000FF
     5:        LDR r0, =0xFFF   ; pseudo-instruction
0x00000008  E59F0010  LDR       R0,[PC,#0x0010]
     6:        LDR r0, X        ; pseudo-instruction
0x0000000C  E59F0008  LDR       R0,[PC,#0x0008]
     7:        LDR r0, =X       ; pseudo-instruction
0x00000010  E59F000C  LDR       R0,[PC,#0x000C]
     8:        ADR r0, X        ; pseudo-instruction
0x00000014  E28F0000  ADD       R0,PC,#0x00000000
     9: loop B loop
0x00000018  EAFFFFFE  B         0x00000018
0x0000001C  AAAAAAAA  BGE       0xFEAAAACC
0x00000020  00000FFF  ???EQ
0x00000024  0000001C  ANDEQ     R0,R0,R12,LSL R0
0x00000028  00000000  ANDEQ     R0,R0,R0
```

```
        AREA prog1, code, READONLY
        ENTRY
        LDR r0,[r1]
        LDR r0, =0xFF    ; pseudo-instruction
        LDR r0, =0xFFF   ; pseudo-instruction
        LDR r0, X        ; pseudo-instruction
        LDR r0, =X       ; pseudo-instruction
        ADR r0, X        ; pseudo-instruction
loop    B loop
X       DCD 0xAAAAAAAA
        END
```

ARM pseudo-instructions

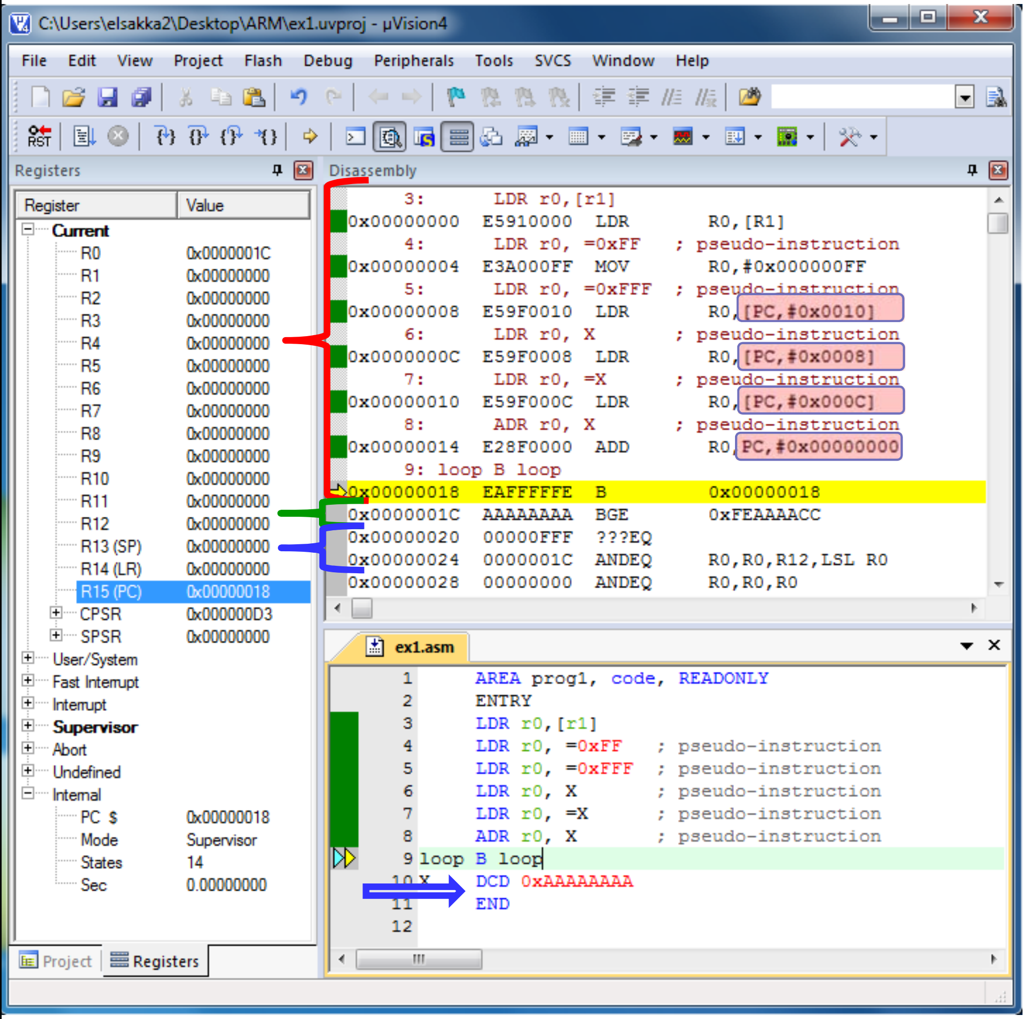
- Consider we changed the previous program as follow:

```
AREA prog1, code, READONLY
     ENTRY
     LDR r0,[r1]
     LDR r0, =0xFF    ; pseudo-instruction
     LDR r0, =0xFFF   ; pseudo-instruction
     LDR r0, X        ; pseudo-instruction
     LDR r0, =X       ; pseudo-instruction
     ADR r0, X        ; pseudo-instruction
loop B loop
X    DCD 0xAAAAAAAA
     END
```

```
     AREA prog1, code, READONLY
     ENTRY
     LDR r0,[r1]
     LDR r0, =0xFF    ; pseudo-instruction
     LDR r0, =0xFFF   ; pseudo-instruction
     LDR r0, X        ; pseudo-instruction
     LDR r0, =X       ; pseudo-instruction
     ADR r0, X        ; pseudo-instruction
loop B loop

     AREA prog1, data, READONLY
X    DCD 0xAAAAAAAA
     END
```

- What is the effect of this change on the generated code?

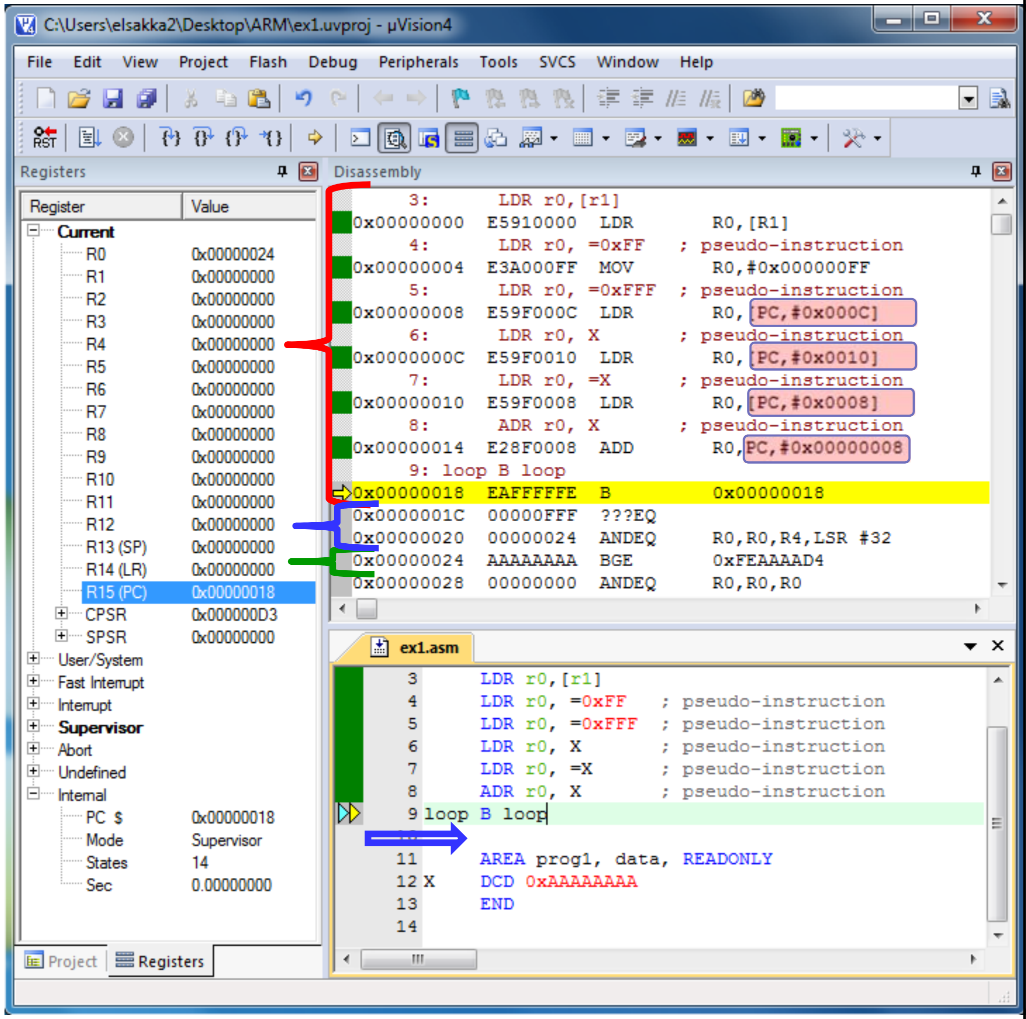
ARM pseudo-instructions


C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Disassembly
ex1.asm
Project
Registers

Register	Value
Current	
R0	0x0000001C
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)	0x00000018
CPSR	0x000000D3
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x00000018
Mode	Supervisor
States	14
Sec	0.00000000

```
     3:         LDR r0,[r1]
0x00000000  E5910000  LDR       R0,[R1]
     4:         LDR r0, =0xFF   ; pseudo-instruction
0x00000004  E3A000FF  MOV       R0,#0x000000FF
     5:         LDR r0, =0xFFF  ; pseudo-instruction
0x00000008  E59F0010  LDR       R0,[PC,#0x0010]
     6:         LDR r0, X       ; pseudo-instruction
0x0000000C  E59F0008  LDR       R0,[PC,#0x0008]
     7:         LDR r0, =X      ; pseudo-instruction
0x00000010  E59F000C  LDR       R0,[PC,#0x000C]
     8:         ADR r0, X       ; pseudo-instruction
0x00000014  E28F0000  ADD       R0,PC,#0x00000000
     9: loop B loop
0x00000018  EAFFFFFE  B         0x00000018
0x0000001C  AAAAAAAA  BGE       0xFEAAAACC
0x00000020  00000FFF  ???EQ
0x00000024  0000001C  ANDEQ     R0,R0,R12,LSL R0
0x00000028  00000000  ANDEQ     R0,R0,R0
```

```
        AREA prog1, code, READONLY
        ENTRY
        LDR r0,[r1]
        LDR r0, =0xFF    ; pseudo-instruction
        LDR r0, =0xFFF   ; pseudo-instruction
        LDR r0, X        ; pseudo-instruction
        LDR r0, =X       ; pseudo-instruction
        ADR r0, X        ; pseudo-instruction
loop B loop
X       DCD 0xAAAAAAAA
        END
```


C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Disassembly
ex1.asm
Project
Registers

Register	Value
Current	
R0	0x00000024
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)	0x00000018
CPSR	0x000000D3
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x00000018
Mode	Supervisor
States	14
Sec	0.00000000

```
     3:         LDR r0,[r1]
0x00000000  E5910000  LDR       R0,[R1]
     4:         LDR r0, =0xFF   ; pseudo-instruction
0x00000004  E3A000FF  MOV       R0,#0x000000FF
     5:         LDR r0, =0xFFF  ; pseudo-instruction
0x00000008  E59F000C  LDR       R0,[PC,#0x000C]
     6:         LDR r0, X       ; pseudo-instruction
0x0000000C  E59F0010  LDR       R0,[PC,#0x0010]
     7:         LDR r0, =X      ; pseudo-instruction
0x00000010  E59F0008  LDR       R0,[PC,#0x0008]
     8:         ADR r0, X       ; pseudo-instruction
0x00000014  E28F0008  ADD       R0,PC,#0x00000008
     9: loop B loop
0x00000018  EAFFFFFE  B         0x00000018
0x0000001C  00000FFF  ???EQ
0x00000020  00000024  ANDEQ     R0,R0,R4,LSR #32
0x00000024  AAAAAAAA  BGE       0xFEAAAAD4
0x00000028  00000000  ANDEQ     R0,R0,R0
```

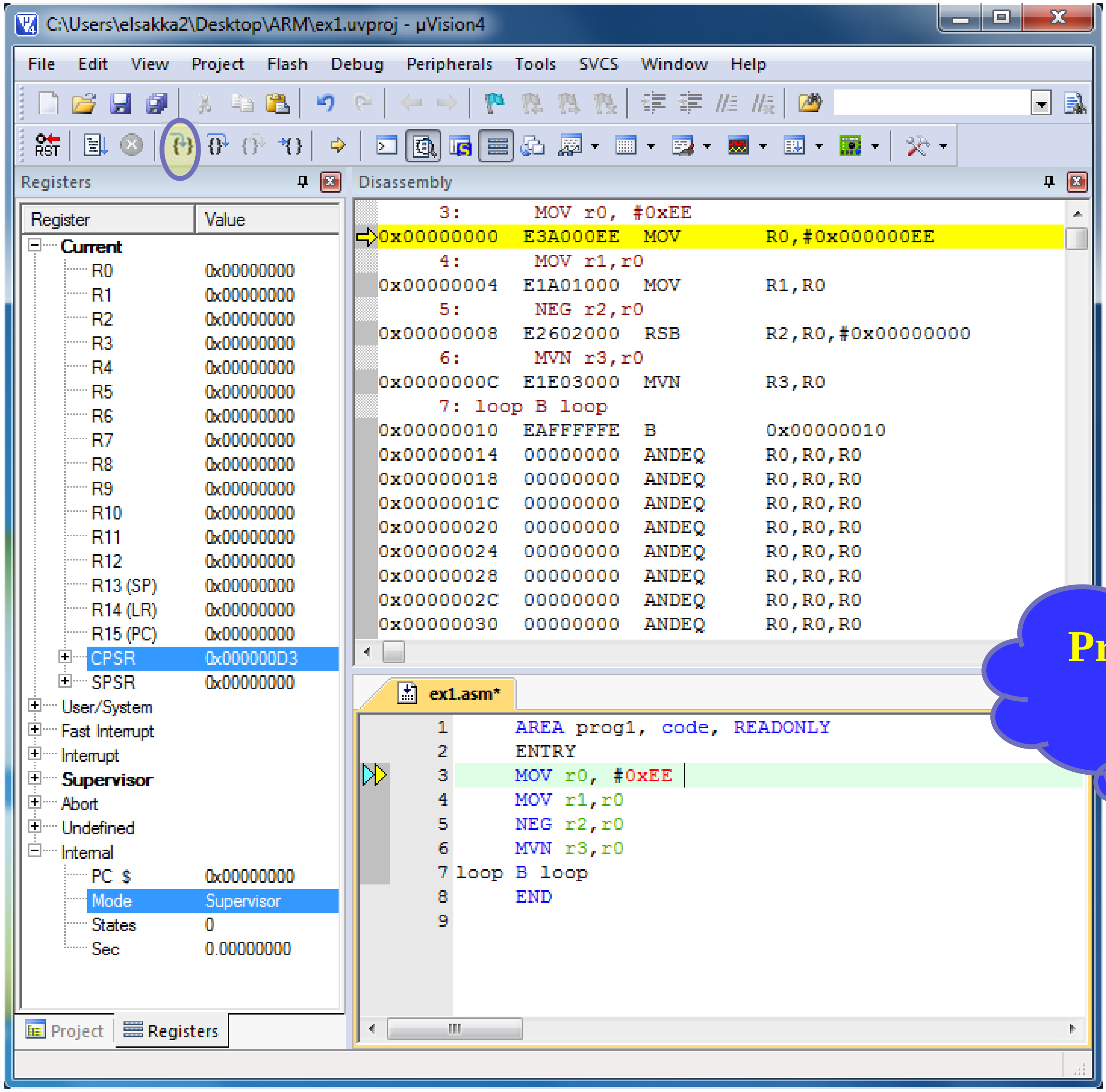
```
        LDR r0,[r1]
        LDR r0, =0xFF    ; pseudo-instruction
        LDR r0, =0xFFF   ; pseudo-instruction
        LDR r0, X        ; pseudo-instruction
        LDR r0, =X       ; pseudo-instruction
        ADR r0, X        ; pseudo-instruction
loop B loop

        AREA prog1, data, READONLY
X       DCD 0xAAAAAAAA
        END
```

ARM pseudo-instructions

- Consider the following assembly program:

```
     AREA prog1, code, READONLY
     ENTRY
     MOV r0, #0xEE
     MOV r1,r0
     NEG r2,r0
     MVN r3,r0
loop B loop
     END
```

ARM pseudo-instructions


C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Disassembly
ex1.asm*
Project
Registers

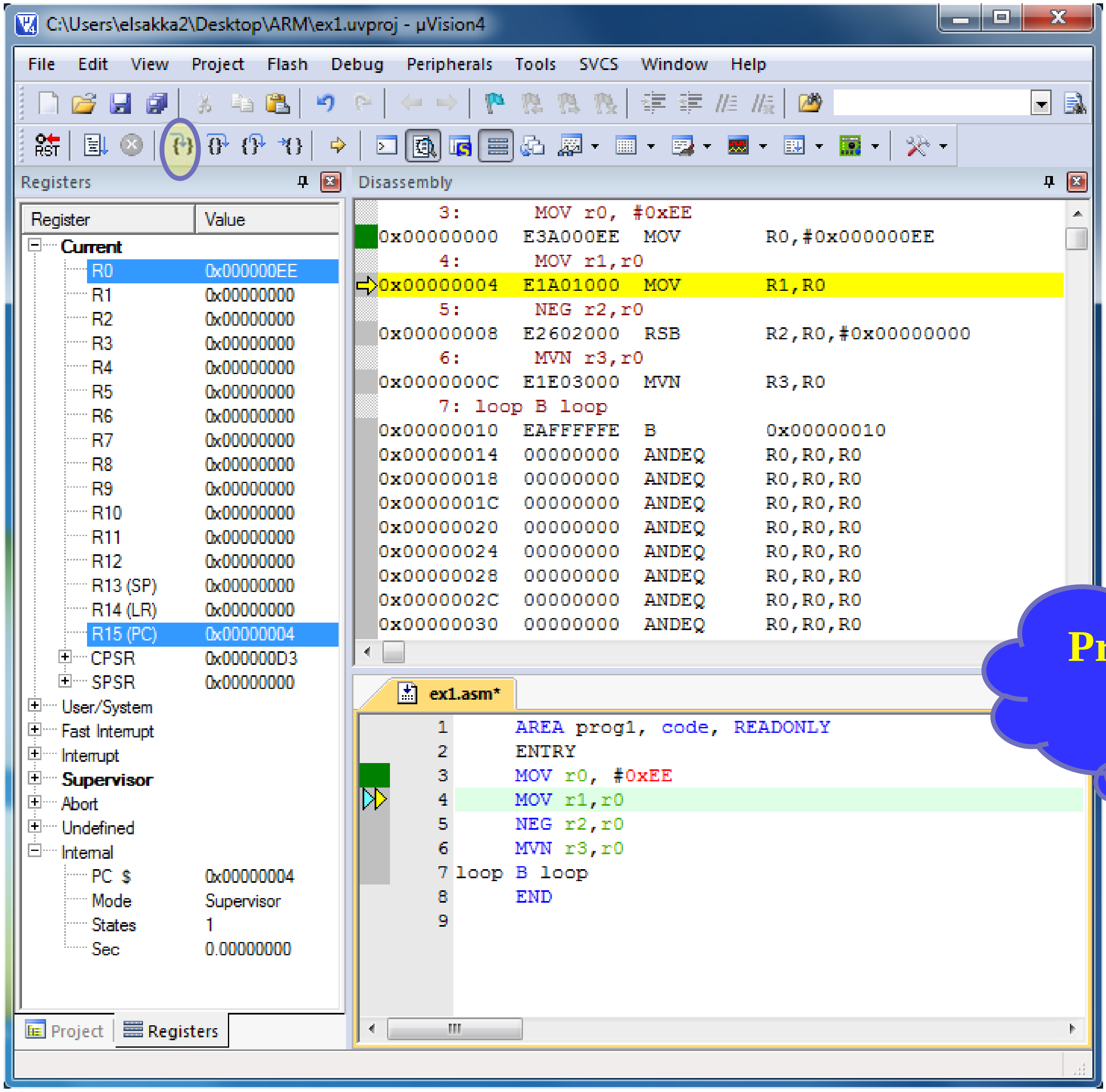
Register	Value
Current	
R0	0x00000000
R1	0x00000000
R2	0x00000000
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x00000000
CPSR	0x000000D3
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x00000000
Mode	Supervisor
States	0
Sec	0.00000000

```
      3:      MOV r0, #0xEE
0x00000000  E3A000EE  MOV       R0,#0x000000EE
      4:      MOV r1,r0
0x00000004  E1A01000  MOV       R1,R0
      5:      NEG r2,r0
0x00000008  E2602000  RSB       R2,R0,#0x00000000
      6:      MVN r3,r0
0x0000000C  E1E03000  MVN       R3,R0
      7: loop B loop
0x00000010  EAFFFFFE  B         0x00000010
0x00000014  00000000  ANDEQ     R0,R0,R0
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
0x00000030  00000000  ANDEQ     R0,R0,R0
```

```
     AREA prog1, code, READONLY
     ENTRY
     MOV r0, #0xEE
     MOV r1,r0
     NEG r2,r0
     MVN r3,r0
loop B loop
     END
```

Press Step, or F11

ARM pseudo-instructions


C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Disassembly
ex1.asm*
Project
Registers

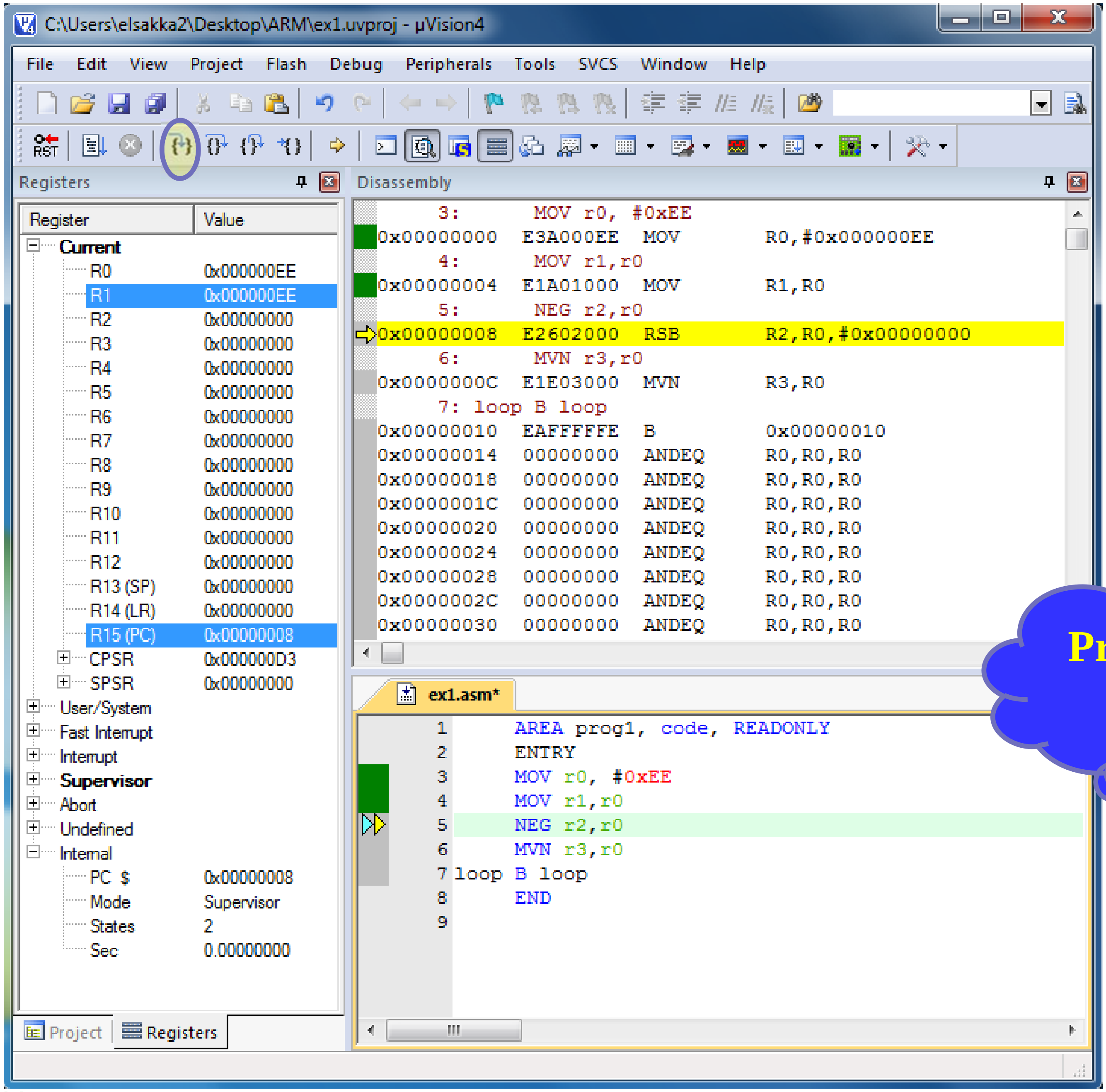
Register	Value
Current	
R0	0x000000EE
R1	0x00000000
R2	0x00000000
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x00000004
CPSR	0x000000D3
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x00000004
Mode	Supervisor
States	1
Sec	0.00000000

```
     3:        MOV r0, #0xEE
0x00000000  E3A000EE  MOV       R0,#0x000000EE
     4:        MOV r1,r0
0x00000004  E1A01000  MOV       R1,R0
     5:        NEG r2,r0
0x00000008  E2602000  RSB       R2,R0,#0x00000000
     6:        MVN r3,r0
0x0000000C  E1E03000  MVN       R3,R0
     7: loop B loop
0x00000010  EAFFFFFE  B         0x00000010
0x00000014  00000000  ANDEQ     R0,R0,R0
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
0x00000030  00000000  ANDEQ     R0,R0,R0
```

```
      AREA prog1, code, READONLY
      ENTRY
      MOV r0, #0xEE
      MOV r1,r0
      NEG r2,r0
      MVN r3,r0
loop  B loop
      END
```

Press Step, or F11

ARM pseudo-instructions


C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Disassembly
ex1.asm*
Project
Registers

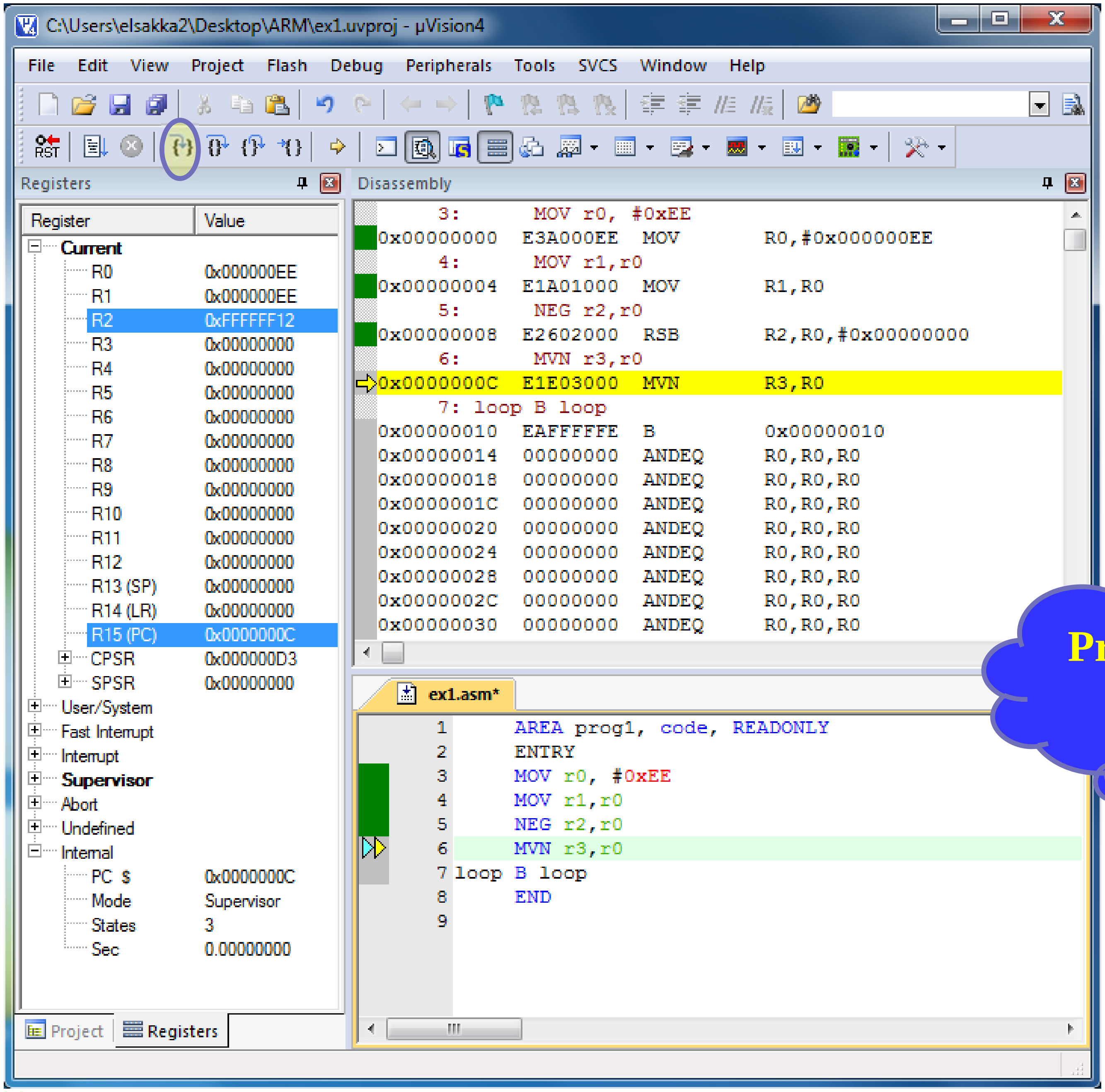
Register	Value
Current	
R0	0x000000EE
R1	0x000000EE
R2	0x00000000
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x00000008
CPSR	0x000000D3
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x00000008
Mode	Supervisor
States	2
Sec	0.00000000

```
      3:        MOV r0, #0xEE
0x00000000  E3A000EE  MOV        R0,#0x000000EE
      4:        MOV r1,r0
0x00000004  E1A01000  MOV        R1,R0
      5:        NEG r2,r0
0x00000008  E2602000  RSB        R2,R0,#0x00000000
      6:        MVN r3,r0
0x0000000C  E1E03000  MVN        R3,R0
      7: loop B loop
0x00000010  EAFFFFFE  B          0x00000010
0x00000014  00000000  ANDEQ      R0,R0,R0
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
0x00000030  00000000  ANDEQ      R0,R0,R0
```

```
     AREA prog1, code, READONLY
     ENTRY
     MOV r0, #0xEE
     MOV r1,r0
     NEG r2,r0
     MVN r3,r0
loop B loop
     END
```

Press Step, or F11

ARM pseudo-instructions


C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Disassembly
ex1.asm*
Project
Registers

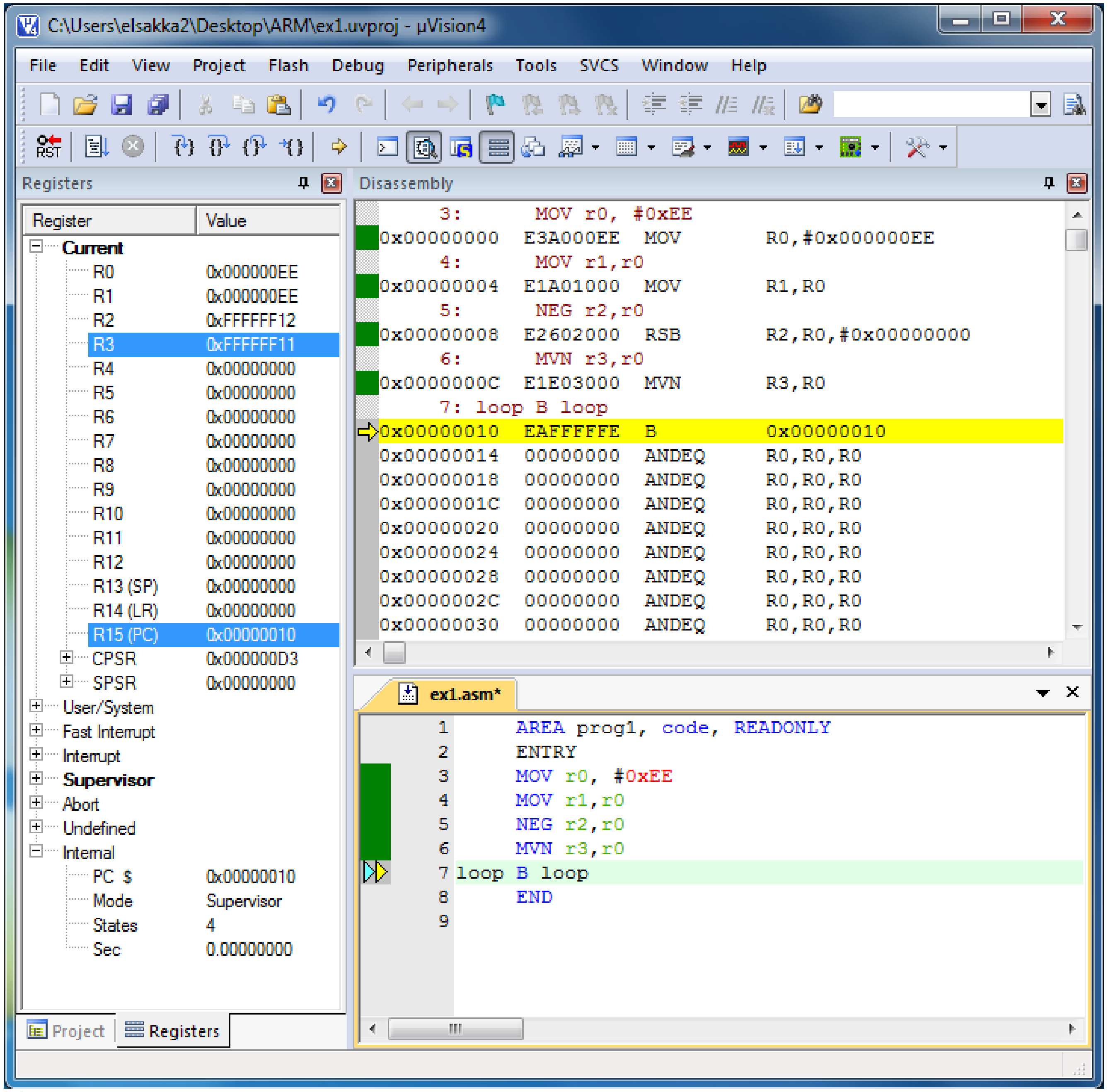
Register	Value
Current	
R0	0x000000EE
R1	0x000000EE
R2	0xFFFFFF12
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x0000000C
CPSR	0x000000D3
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x0000000C
Mode	Supervisor
States	3
Sec	0.00000000

```
     3:       MOV r0, #0xEE
0x00000000  E3A000EE  MOV       R0,#0x000000EE
     4:       MOV r1,r0
0x00000004  E1A01000  MOV       R1,R0
     5:       NEG r2,r0
0x00000008  E2602000  RSB       R2,R0,#0x00000000
     6:       MVN r3,r0
0x0000000C  E1E03000  MVN       R3,R0
     7: loop B loop
0x00000010  EAFFFFFE  B         0x00000010
0x00000014  00000000  ANDEQ     R0,R0,R0
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
0x00000030  00000000  ANDEQ     R0,R0,R0
```

```
      AREA prog1, code, READONLY
      ENTRY
      MOV r0, #0xEE
      MOV r1,r0
      NEG r2,r0
      MVN r3,r0
loop B loop
      END
```

Press Step, or F11

ARM pseudo-instructions


C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Disassembly
ex1.asm*
Project
Registers

Register	Value
Current	
R0	0x000000EE
R1	0x000000EE
R2	0xFFFFFF12
R3	0xFFFFFF11
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
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x00000010
Mode	Supervisor
States	4
Sec	0.00000000

```
     3:      MOV r0, #0xEE
0x00000000  E3A000EE  MOV      R0,#0x000000EE
     4:      MOV r1,r0
0x00000004  E1A01000  MOV      R1,R0
     5:      NEG r2,r0
0x00000008  E2602000  RSB      R2,R0,#0x00000000
     6:      MVN r3,r0
0x0000000C  E1E03000  MVN      R3,R0
     7: loop B loop
0x00000010  EAFFFFFE  B        0x00000010
0x00000014  00000000  ANDEQ    R0,R0,R0
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
0x00000030  00000000  ANDEQ    R0,R0,R0
```

```
        AREA prog1, code, READONLY
        ENTRY
        MOV r0, #0xEE
        MOV r1,r0
        NEG r2,r0
        MVN r3,r0
loop    B loop
        END
```

ARM pseudo-instructions

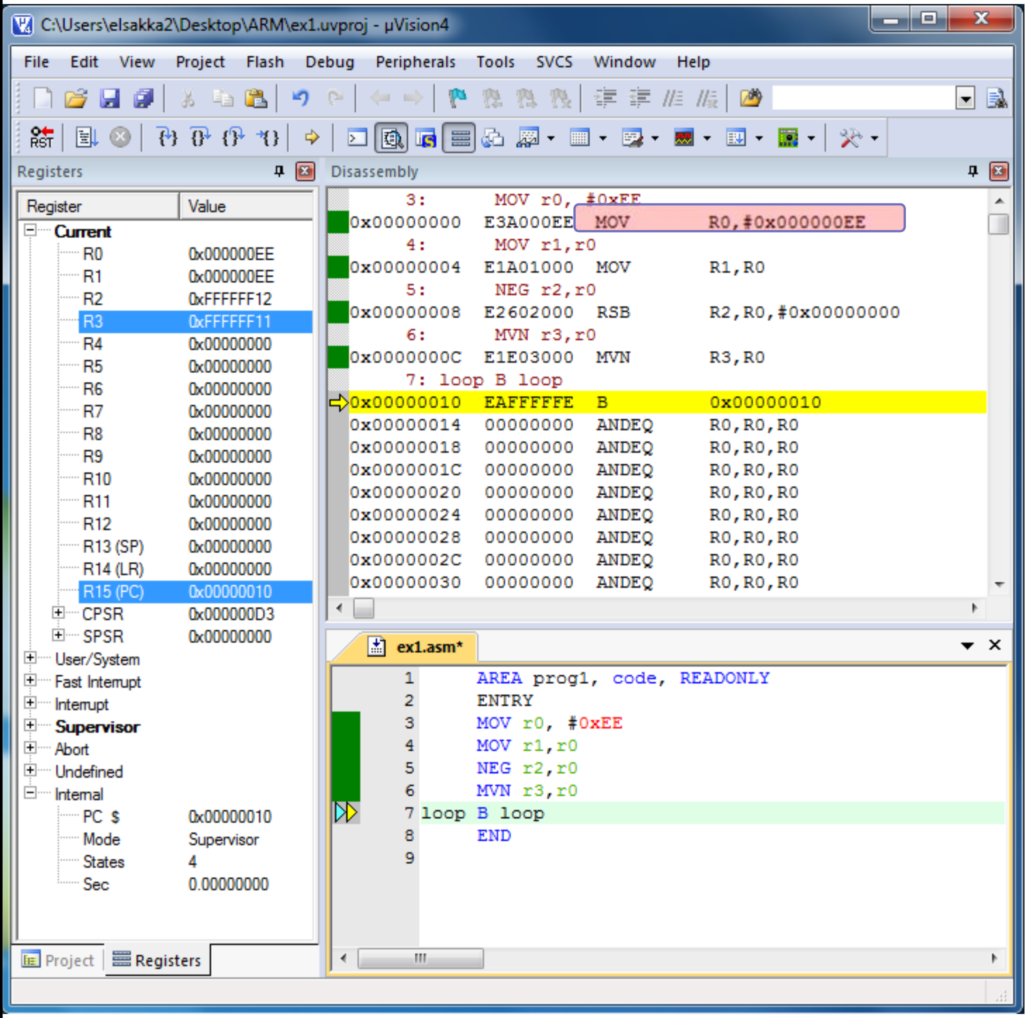
- Consider we changed the previous program as follow:

```
 AREA prog1, code, READONLY
     ENTRY
     MOV r0, #0xEE
     MOV r1,r0
     NEG r2,r0
     MVN r3,r0
loop B loop
     END
```

```
 AREA prog1, code, READONLY
     ENTRY
     MOV r0, #-0xEE
     MOV r1,r0
     NEG r2,r0
     MVN r3,r0
loop B loop
     END     END
```

- What is the effect of this change on the generated code?

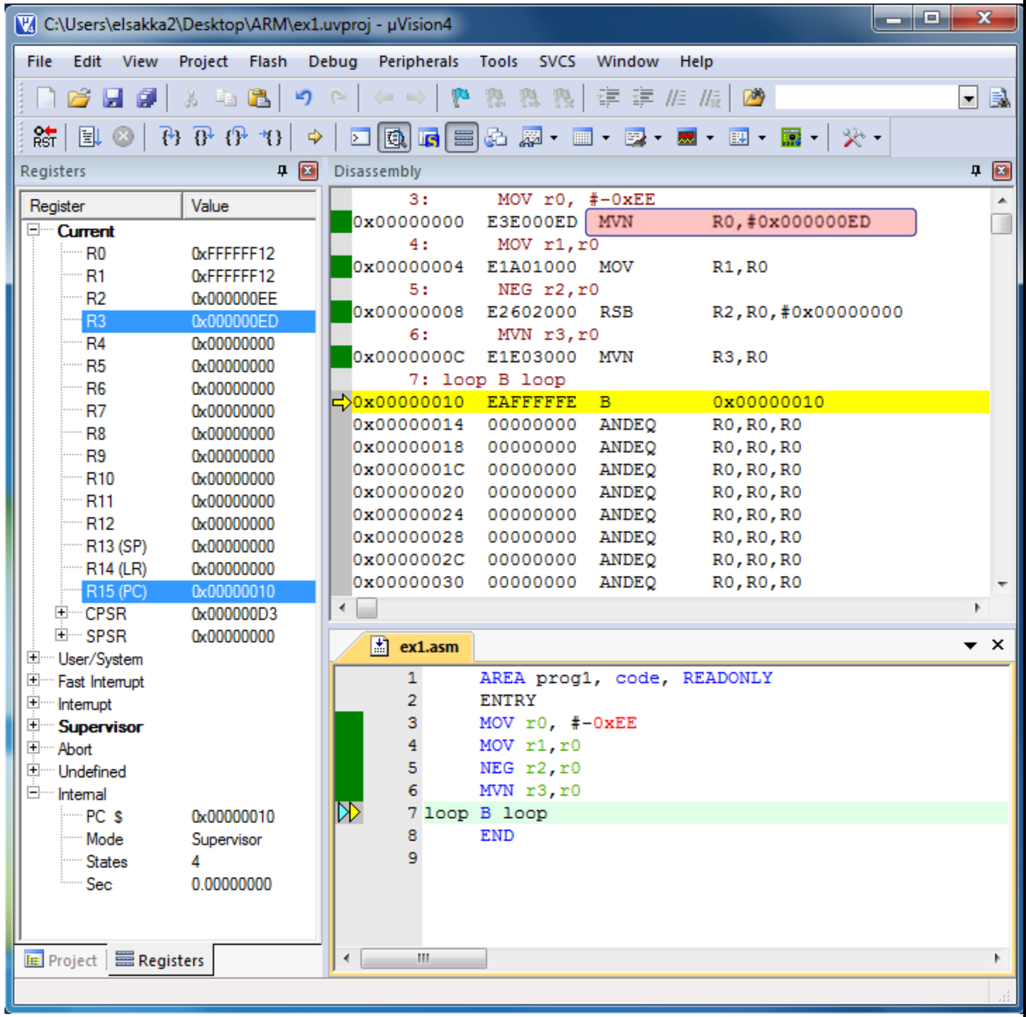
ARM pseudo-instructions


C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Disassembly
ex1.asm*
Project
Registers

Register	Value
Current	
R0	0x000000EE
R1	0x000000EE
R2	0xFFFFFF12
R3	0xFFFFFF11
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
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x00000010
Mode	Supervisor
States	4
Sec	0.00000000

```
     3:      MOV r0, #0xEE
0x00000000  E3A000EE  MOV      R0,#0x000000EE
     4:      MOV r1,r0
0x00000004  E1A01000  MOV      R1,R0
     5:      NEG r2,r0
0x00000008  E2602000  RSB      R2,R0,#0x00000000
     6:      MVN r3,r0
0x0000000C  E1E03000  MVN      R3,R0
     7: loop B loop
0x00000010  EAFFFFFE  B        0x00000010
0x00000014  00000000  ANDEQ    R0,R0,R0
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
0x00000030  00000000  ANDEQ    R0,R0,R0
```

```
      AREA prog1, code, READONLY
      ENTRY
      MOV r0, #0xEE
      MOV r1,r0
      NEG r2,r0
      MVN r3,r0
loop  B loop
      END
```


C:\Users\elsakka2\Desktop\ARM\ex1.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Disassembly
ex1.asm
Project
Registers

Register	Value
Current	
R0	0xFFFFFF12
R1	0xFFFFFF12
R2	0x000000EE
R3	0x000000ED
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
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x00000010
Mode	Supervisor
States	4
Sec	0.00000000

```
     3:      MOV r0, #-0xEE
0x00000000  E3E000ED  MVN      R0,#0x000000ED
     4:      MOV r1,r0
0x00000004  E1A01000  MOV      R1,R0
     5:      NEG r2,r0
0x00000008  E2602000  RSB      R2,R0,#0x00000000
     6:      MVN r3,r0
0x0000000C  E1E03000  MVN      R3,R0
     7: loop B loop
0x00000010  EAFFFFFE  B        0x00000010
0x00000014  00000000  ANDEQ    R0,R0,R0
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
0x00000030  00000000  ANDEQ    R0,R0,R0
```

```
      AREA prog1, code, READONLY
      ENTRY
      MOV r0, #-0xEE
      MOV r1,r0
      NEG r2,r0
      MVN r3,r0
loop  B loop
      END
```