A_Cube: Perform A^3

Calculate A^3 where A is an 8-bit unsigned variable. The result is placed into 24-bit variable C. The 24-bit result is saved using little endian byte ordering.
C2:C1:C0 = A^3
Simulation of the multiplication problem 50^3. The answer should equal 125,000 (0x2625A0).

```
.CSEG

A_Cubed:
    lds  r26,A     ; load
    mul  r26,r26
    movw r25:r24, r1:r0
    rcall muls16x8_24
    sts  C,r2      ; least significant byte (little end)
    sts  C+1,r3    ; most  significant byte (big end)
    sts  C+2, r4

    rjmp A_Cubed
```

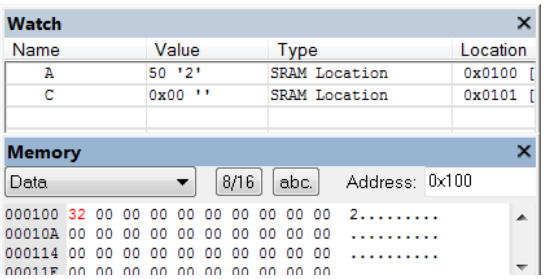

Figure 1: Start of program with A initialized to 0x32 (50_{10}) by double click on variable A on the Watch and enter “50”

```
.CSEG

A_Cubed:
    lds  r26,A     ; load
    mul  r26,r26
    movw r25:r24, r1:r0
    rcall muls16x8_24
    sts  C,r2      ; least significant byte (little end)
    sts  C+1,r3    ; most  significant byte (big end)
    sts  C+2, r4

    rjmp A_Cubed
```

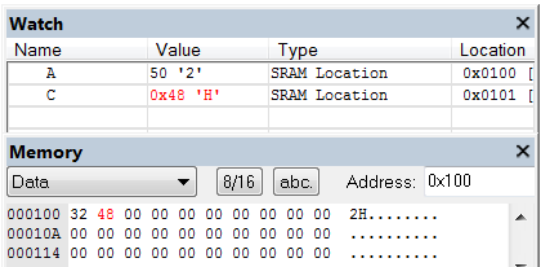

Figure 2: After performing calculation of A^3 (by performing A^2 with command “mul r26, r26”and A^2*A with subroutine muls16x8_24), result of C0 is 0x48

```
.CSEG

A_Cubed:
    lds  r26,A     ; load
    mul  r26,r26
    movw r25:r24, r1:r0
    rcall muls16x8_24
    sts  C,r2      ; least significant byte (little end)
    sts  C+1,r3    ; most  significant byte (big end)
    sts  C+2, r4

    rjmp A_Cubed
```

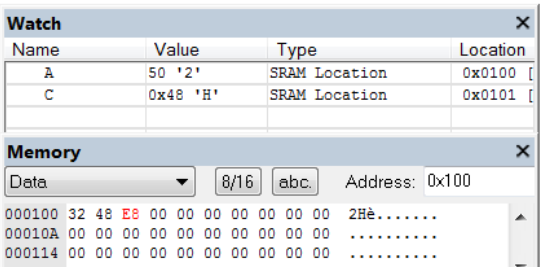

Figure 3: Result of C1 is 0xE8

```
.CSEG

A_Cubed:
    lds  r26,A     ; load
    mul  r26,r26
    movw r25:r24, r1:r0
    rcall muls16x8_24
    sts  C,r2      ; least significant byte (little end)
    sts  C+1,r3    ; most  significant byte (big end)
    sts  C+2, r4

    rjmp A_Cubed
```

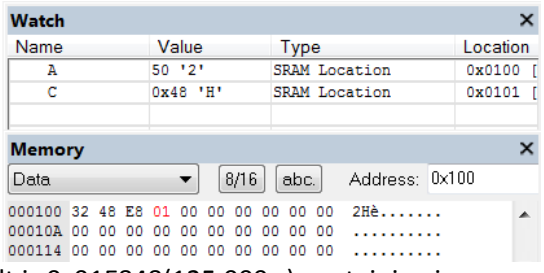

Figure 4: Result of C2 is 0x01. End of program with the result is 0x01E848($125{,}000_{10}$) containing in C2:C1:C0.