Assembly Code:

;

; DA1A.asm

;

; Created: 2/7/2019 4:39:46 PM

; Author : Chris

;

.EQU multiplicand **=** 0xFFFF ; multiplicand variable declaration

.EQU multiplier **=** 0xFF ; multiplier variabledeclaration

initialize**:** ; INITIALIZING THE MULTIPLICAND AND MULTIPLIER

ldi r24**,** low**(**multiplicand**)** ; loads in low value of multiplicand

ldi r25**,** high**(**multiplicand**)** ; loads in high value of multiplicand

ldi r23**,** 0 ; 0 for comparison when decrementing multiplier

ldi r22**,** multiplier ; loads in multiplier value

**loop:** ; LOOP TO MULTIPLY

**add** r18**,** r24 ; adds low value of the multiplicand into register R18

**adc** r19**,** r25 ; adds high value of the multiplicand and the carry into register R19

**adc** r20**,** r23 ; adds the carry into register R20

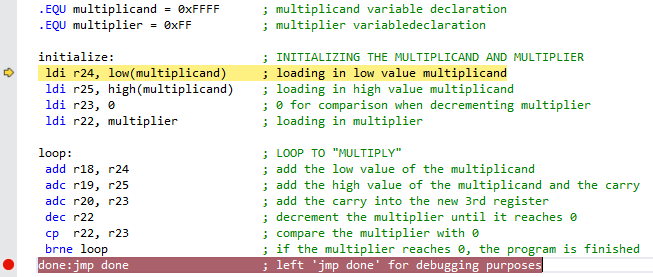
**dec** r22 ; decrements the multiplier until it reaches 0

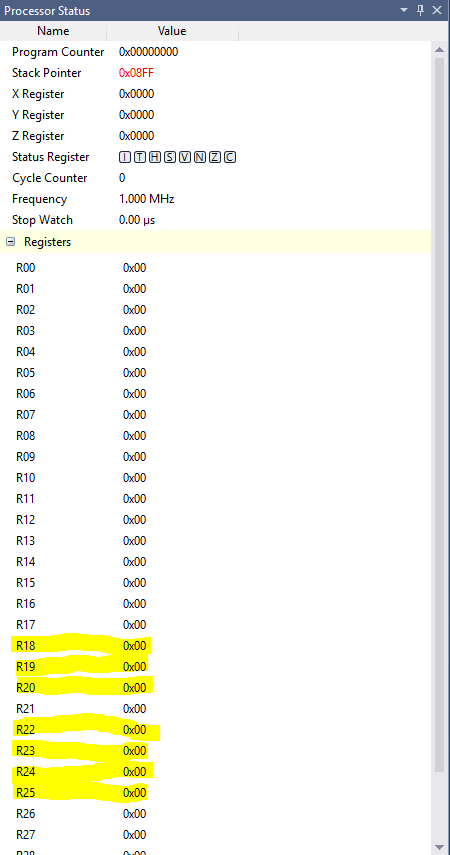
cp r22**,** r23 ; compares the multiplier with 0

brne **loop** ; if the multiplier equals 0 the program’s done

done**:jmp** done ; 'jmp done' for debugging purposes

Screenshots:

  
Screenshot of the **beginning** process of debugging

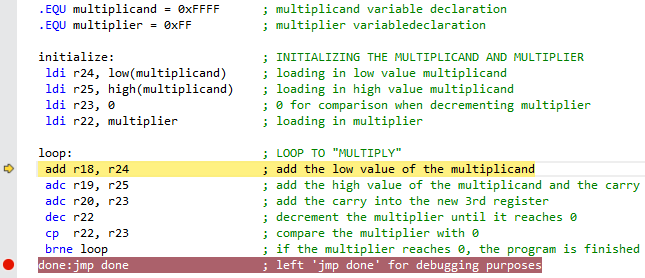
  
Screenshot of the Registers at the **beginning** of debugging

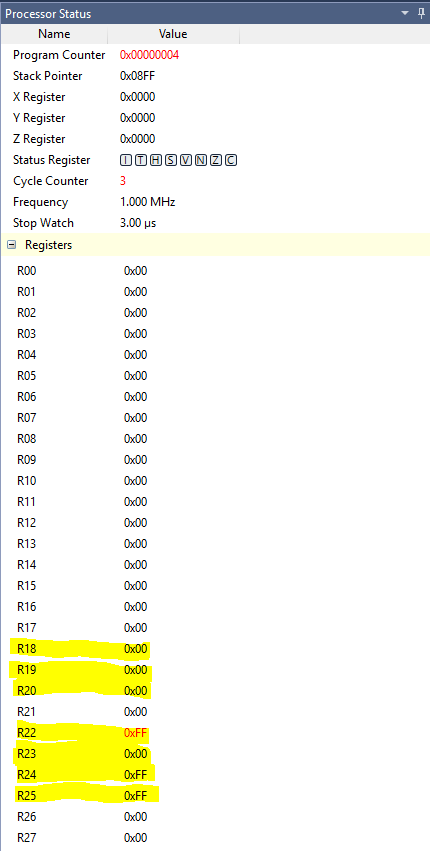
Multiplicand value

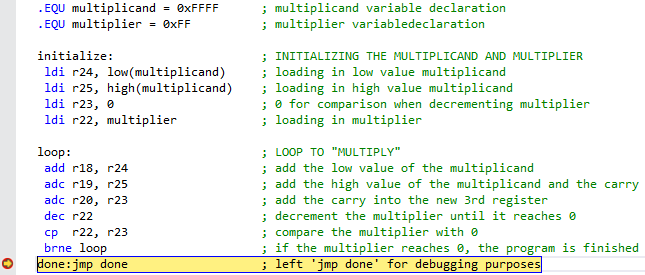
Multiplier value

Result value

Comparator of value ‘0’

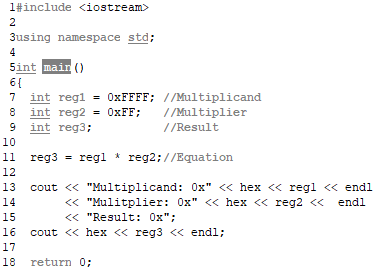
  
Screenshot **after initialization** of registers

  
Screenshot of the Registers **after initialization**

  
Screenshot of the **end** process of debugging

  
Screenshot of the Registers at the **end** of debugging

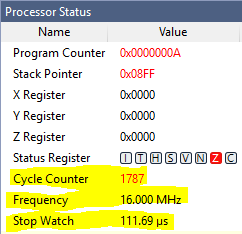
Verification (Using C++ @ Bobby compiler):

  
Screenshot of C++ verification code

  
Screenshot of the Output

Both the result in AVR and C++ show the multiplication of hex values 0xFFFF and 0xFF equals 0xFEFF01

Execution Time @ 16MHz/# of Cycles:



The highlighted screenshot shows frequency @ 16MHz results:  
**Number of Cycles: 1787  
Execution Time: 111.69 us**