Unsigned 16-bit Multiply 8-bit result in 24-bit number

Write a program to multiply a 16-bit unsigned number in the r25:r24 register pair by an 8-bit number in r26.return the answer in r4:r3:r2
r4:r3:32 = r25:r24 x r26
Simulation of the multiplication problem 10,000x250. The answer should equal 2,500,000 (0x2625A0).

```
.INCLUDE <m328pdef.inc>

muls16x8_24_test:
    ldi r25,HIGH(10000) ;
    ldi r24,LOW(10000)  ;
    ldi r26,250         ;
    rcall muls16x8_24   ;
    rjmp muls16x8_24_test
```

Watch

Name	Value	Type	Location
R25	0x27 '''	Register	R25
R24	0x00 ''	Register	R24
R26	0x00 ''	Register	R26
R4	0x00 ''	Register	R4
R3	0x00 ''	Register	R3
R2	0x00 ''	Register	R2

Figure 1: Start of program with r25 initialized to 0x27

```
.INCLUDE <m328pdef.inc>

muls16x8_24_test:
    ldi r25,HIGH(10000) ;
    ldi r24,LOW(10000)  ;
    ldi r26,250         ;
    rcall muls16x8_24   ;
    rjmp muls16x8_24_test
```

Watch

Name	Value	Type	Location
R25	0x27 '''	Register	R25
R24	0x10 '┼'	Register	R24
R26	0x00 ''	Register	R26
R4	0x00 ''	Register	R4
R3	0x00 ''	Register	R3
R2	0x00 ''	Register	R2

Figure 2: variable r24 is initialized to 0x10. Now r25:r24 is 0x2710 ($10{,}000_{10}$)

```
.INCLUDE <m328pdef.inc>

muls16x8_24_test:
    ldi r25,HIGH(10000) ;
    ldi r24,LOW(10000)  ;
    ldi r26,250         ;
    rcall muls16x8_24   ;
    rjmp muls16x8_24_test
```

Watch

Name	Value	Type	Location
R25	0x27 '''	Register	R25
R24	0x10 '┼'	Register	R24
R26	0xFA 'ú'	Register	R26
R4	0x00 ''	Register	R4
R3	0x00 ''	Register	R3
R2	0x00 ''	Register	R2

Figure 3: variable r26 is initialized to 0xFA (250_{10})

```
.INCLUDE <m328pdef.inc>

muls16x8_24_test:
    ldi r25,HIGH(10000) ;
    ldi r24,LOW(10000)  ;
    ldi r26,250         ;
    rcall muls16x8_24   ;
    rjmp muls16x8_24_test
```

Watch

Name	Value	Type	Location
R25	0x27 '''	Register	R25
R24	0x10 '┼'	Register	R24
R26	0xFA 'ú'	Register	R26
R4	0x26 '&'	Register	R4
R3	0x25 '%'	Register	R3
R2	0xA0 ' '	Register	R2

Figure 4: End of program with the result is 0x2625A0($2{,}500{,}000_{10}$) containing in r4:r3:r2 registers