OPERAZIONI ARITMETICHE

Merker speciali:

- **SM1.0** risultato uguale a zero
- **SM1.1** overflow
- **SM1.2** risultato negativo
- **SM1.3** divisione per zero

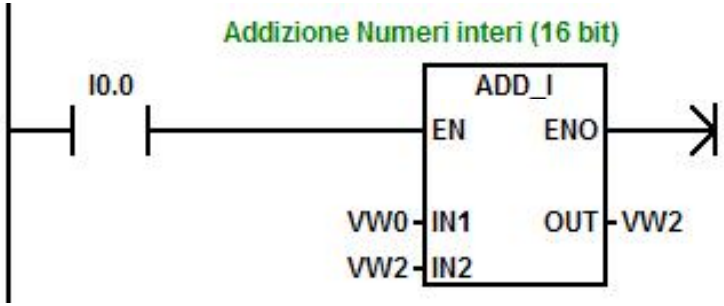
Tabella riassuntiva

(codici operativi in IL)

	INTERI (16 Bit)	DOUBLE W. (32 Bit)	REALI (32 Bit)
ADDIZIONE	+I 46 «s	+D 55 «s	+R 110-163 «s
SOTTRAZIONE	-I 47 «s	-D 55 «s	-R 113-166 «s
MOLTIPLICAZIONE	*I 71 «s	*D 92 «s	*R 100-130 «s
	MUL 70 «s		
DIVISIONE	/I 115 «s	/D 376 «s	/R 300-360 «s
	DIV 119 «s		

ADDIZIONE

Numeri interi (16 bit)



Il blocco ADD_I esegue la seguente somma

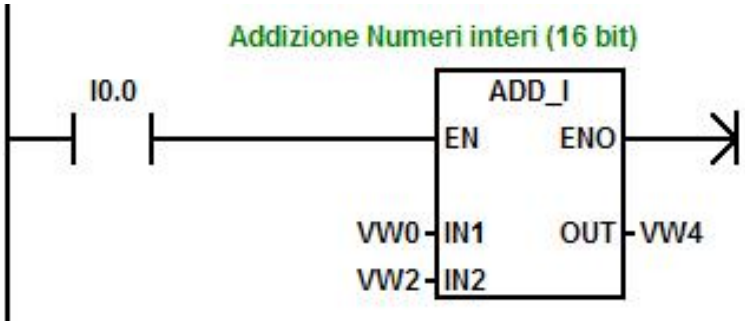
IN1 + IN2 ì OUT

```
LD      I0.0
              1°    2°
+I      VW0 , VW2      Non modifica lo stack
```

L'istruzione +I esegue la seguente somma:

2° + 1° → 2°

NB: Quando l'uscita non è costituita da uno dei due operandi:



Il blocco ADD_I esegue la seguente somma

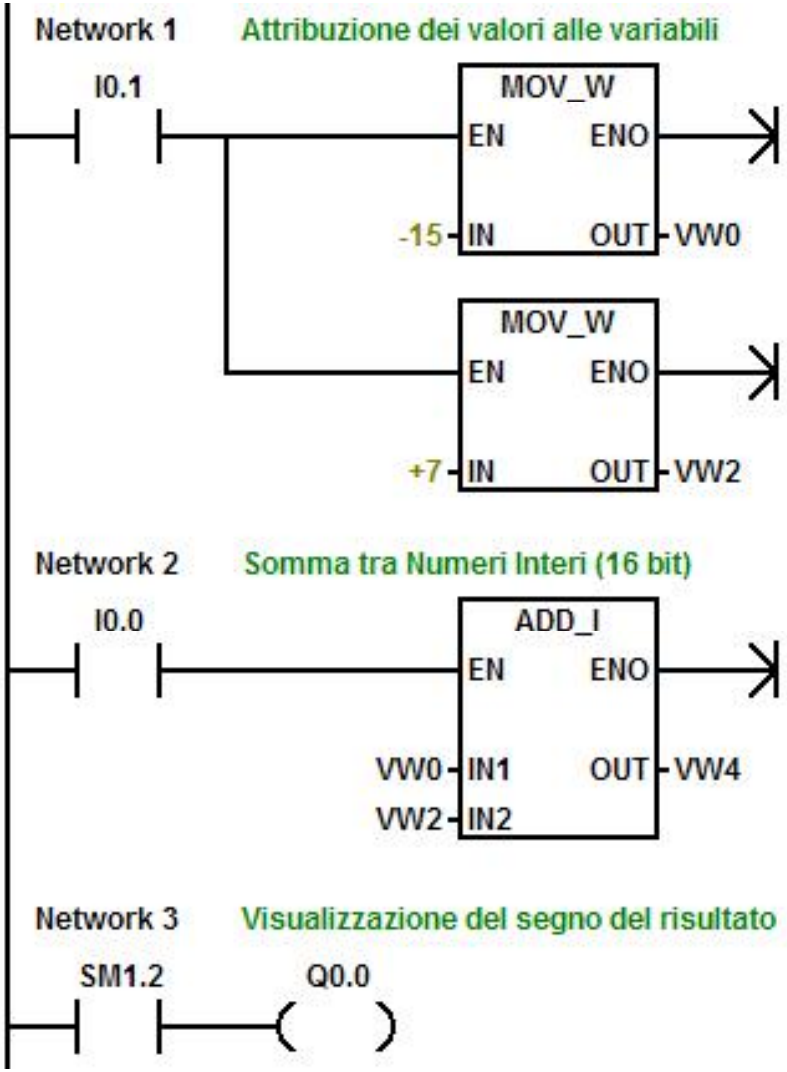
IN1 + IN2 ì OUT

```
LD      I0.0
MOVW    VW0 , VW4
+I      VW2 , VW4
```

Carica il contenuto di VW0 in VW4

In questo caso occorre, con MOVW, *caricare l'ingresso IN1 nell'uscita e poi eseguire la somma tra il IN2 e l'uscita.*

ESEMPIO: **Svolgere la somma tra i contenuti delle variabili VW0 (-15) e VW2 (+7) e impostare una segnalazione luminosa in caso di risultato negativo.**



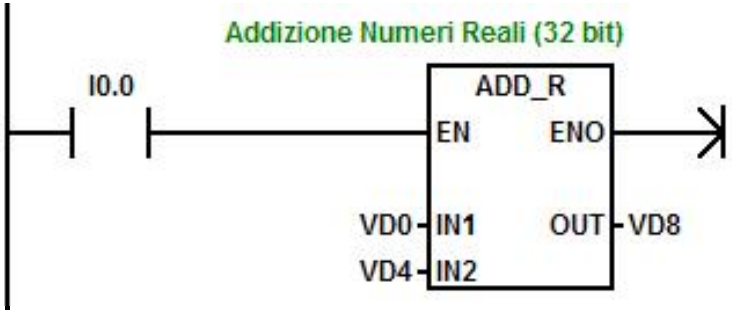

```
Network 1
LD      I0.1
MOVW    -15,  VW0
MOVW    7, VW2

Network 2
LD      I0.0
MOVW    VW0,  VW4
+I      VW2,  VW4

Network 3
LD      SM1.2
=       Q0.0
```

L'uscita Q0.0 si attiva se il risultato dell'operazione è negativo.

Numeri reali (32 bit)

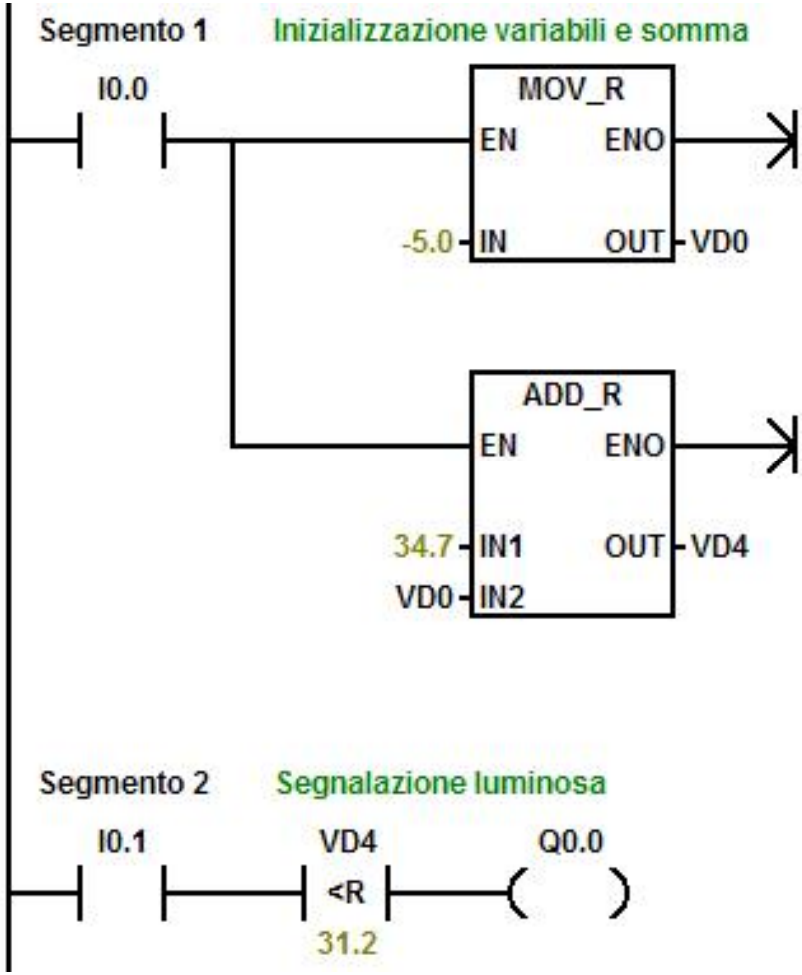



```
LD      I0.0
MOVR    VD0 , VD8
+R      VD4 , VD8
```

ESEMPIO: Svolgere la somma tra i contenuti delle variabili VD0 (-5) e VD4 (+34.7) e impostare una segnalazione luminosa in caso di risultato < 31.2

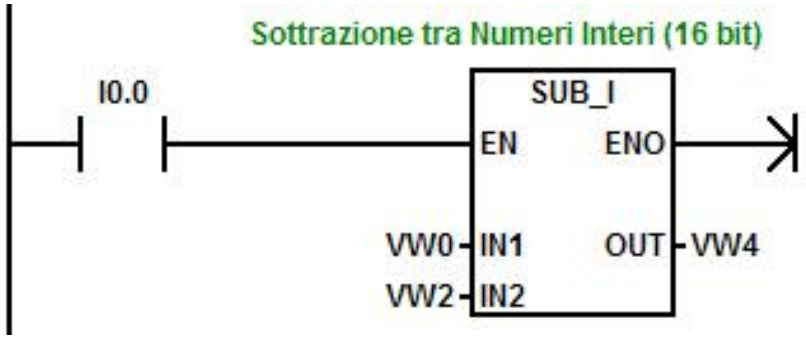
```
Segmento 1      Inizializzazione variabili e somma
LD      I0.0
MOVR    -5.0, VD0
MOVR    34.7, VD4
+R      VD0, VD4

Segmento 2      Segnalazione luminosa
LD      I0.1
AR<     VD4, 31.2
=       Q0.0
```



SOTTRAZIONE

Numeri interi (16 bit)



Il blocco SUB_I esegue la seguente sottrazione

IN1 - IN2 ì OUT

```
LD      I0.0
MOVW    VW0 , VW4
-I      VW2 , VW4
```

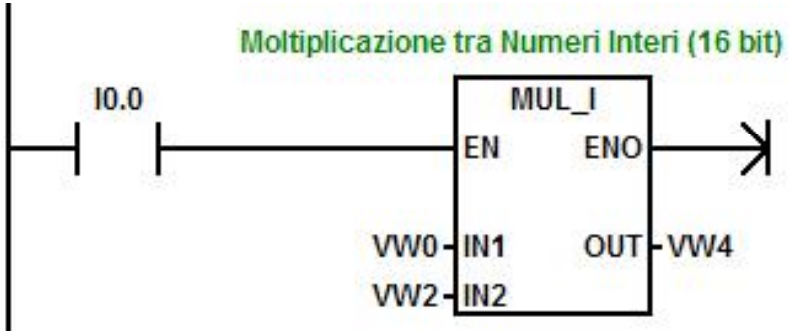
L'istruzione -I esegue la seguente sottrazione:

2° - 1° ì 2°

In questo caso occorre, con MOVW, caricare IN1 nell'uscita e poi eseguire la differenza tra l'uscita e il IN2.

MOLTIPLICAZIONE

Moltiplicazione tra *numeri* interi (16 bit) con *risultato* in word (16 bit)



Il blocco MUL_I esegue la seguente moltiplicazione IN1 * IN2 ì OUT

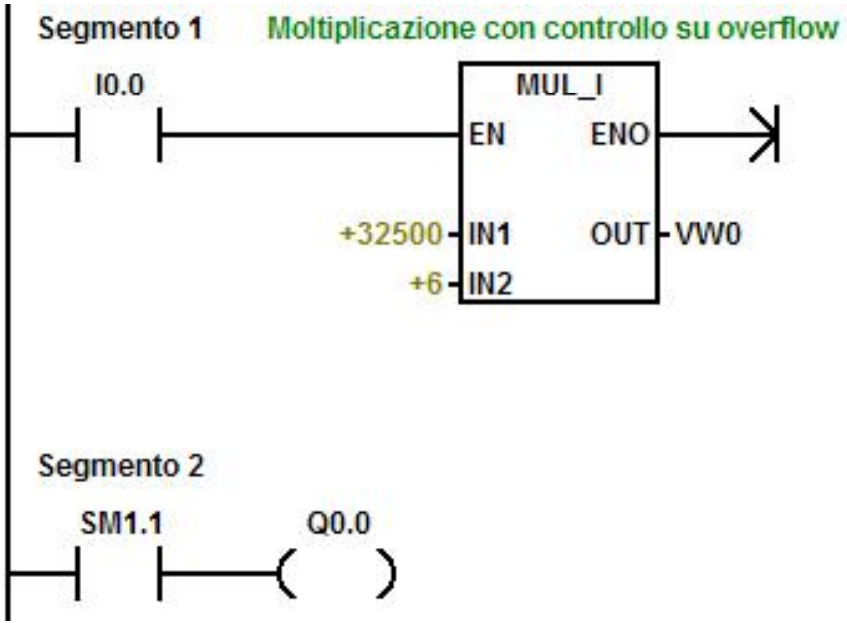
```
LD      I0.0
MOVW    VW0 , VW4
*I      VW2 , VW4
```

L'istruzione *I esegue la seguente moltiplicazione:

2° * 1° → 2°

Occorre quindi, con MOVW, caricare IN1 nell'uscita e poi eseguire il prodotto tra l'uscita e IN2.

ESEMPIO: *Controllo su overflow*



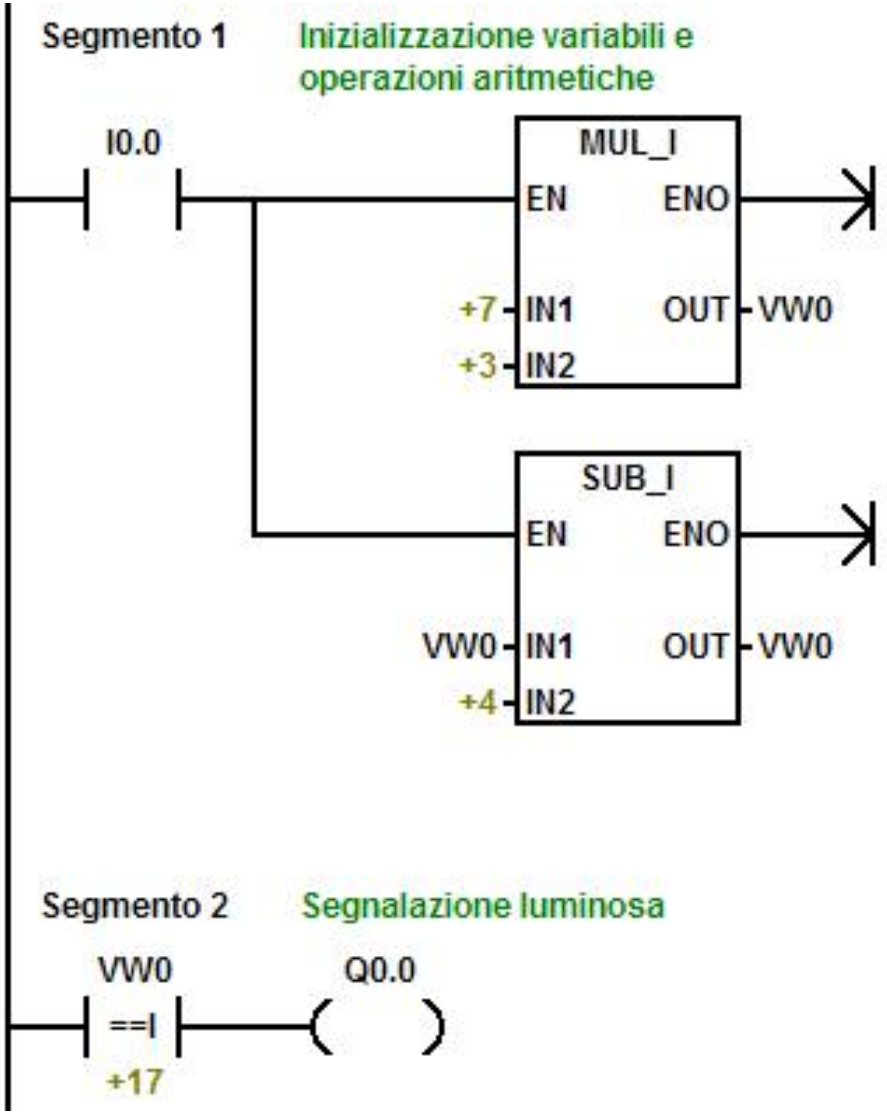

```
Segmento 1    Moltiplicazione con controllo su overflow
LD      I0.0
MOVW    +32500, VW0
*I      +6, VW0

Segmento 2
LD      SM1.1
=       Q0.0
```

ESEMPIO: Svolgere la seguente operazione: 7*3 – 4 , se il risultato è uguale a 17 attivare una segnalazione luminosa.

```
Segmento 1      Inizializzazione variabili e
                operazioni aritmetiche
LD      I0.0
MOVW    +7, VW0
*I      +3, VW0
-I      +4, VW0

Segmento 2      Segnalazione luminosa
LDW=    VW0, +17
=       Q0.0
```

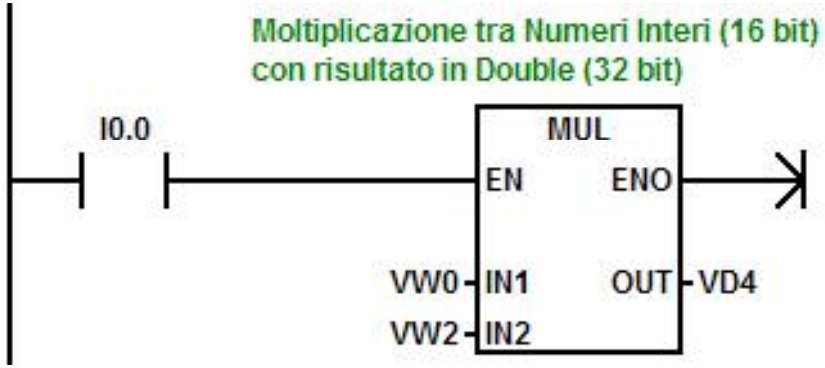


ESERCIZIO: Impostare la seguente operazione: 2.5*3 – 1 , se il risultato è uguale a 6.5 attivare una segnalazione luminosa.

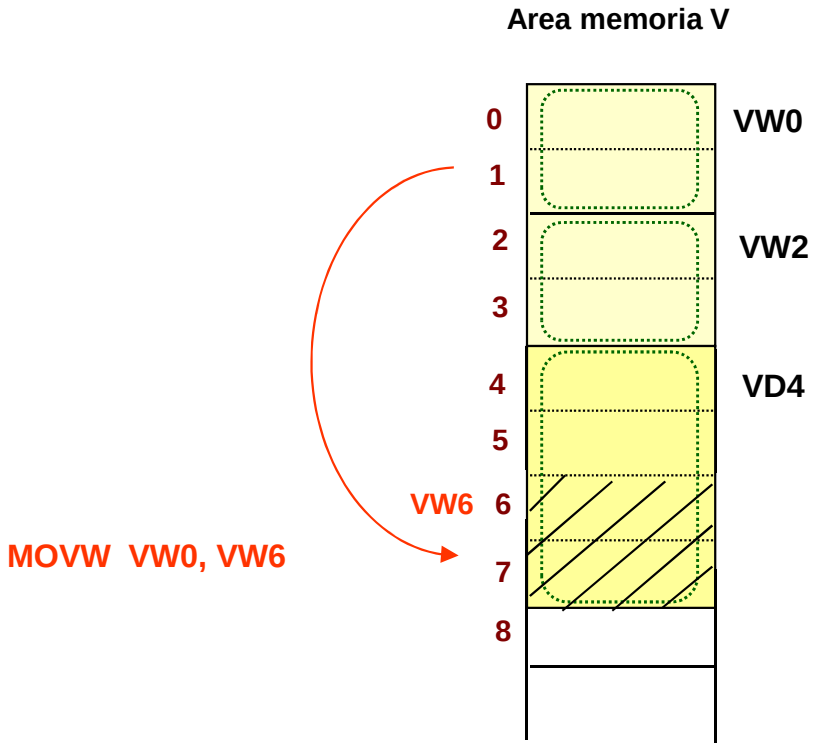
ESERCIZIO: Impostare la seguente operazione: 1.5*(4 – 1) , se il risultato è uguale a 4.5 attivare una segnalazione luminosa.

NB: prevedere un comando di reset del calcolo.

Moltiplicazione tra numeri interi (16 bit) con risultato in double word (32 bit)




```
LD      I0.0
MOVW    VW0 , VW6
*I      VW2 , VD4
```

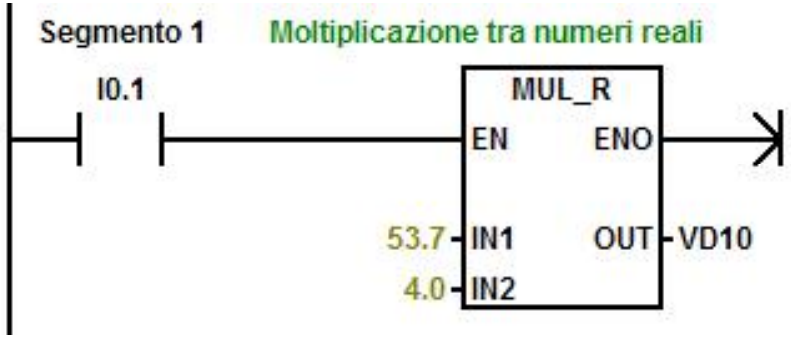


NB: in IL la word meno significativa dell'uscita double VD4 è utilizzata come uno dei fattori.

L'ingresso VW0 viene caricato nella word meno significativa VW6 di VD4.

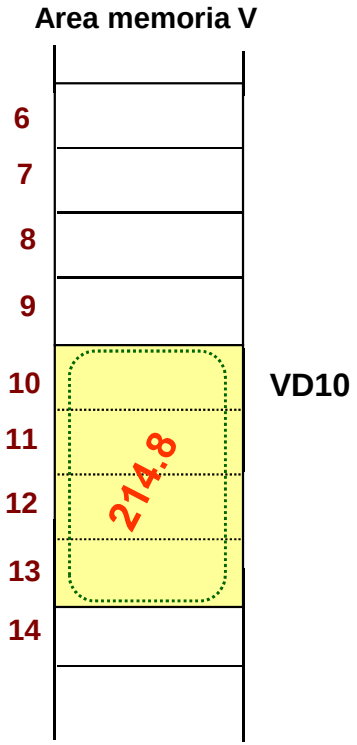
NB: l'indirizzo è sempre quello del byte il più significativo.

Moltiplicazione tra numeri reali (32 bit)



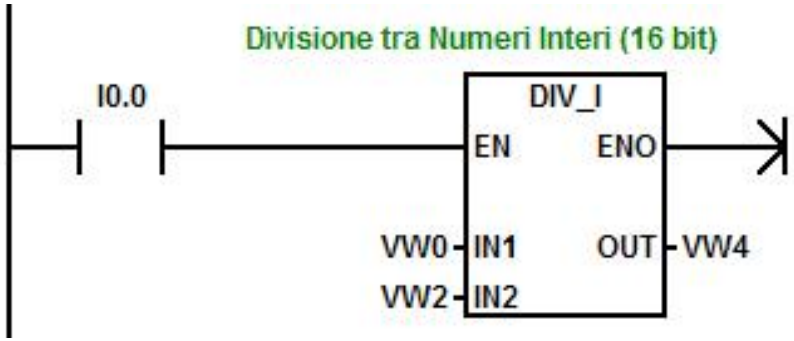

```
Segmento 1      Moltiplicazione tra numeri reali

LD      I0.1
MOVR    53.7, VD10
*R      4.0, VD10
```



DIVISIONE

Divisione tra numeri interi (16 bit) con risultato in word (16 bit)




```
LD      I0.0
MOVW    VW0 , VW4
/I      VW2 , VW4
```

(1° = VW2, 2° = VW4)

Il blocco DIV_I esegue la divisione IN1 / IN2 ì OUT

NB: in caso di risultato con cifre decimali, queste vengono troncate.

Attivazione merker speciali:

- **SM1.0 risultato uguale a zero**
- **SM1.1 overflow**
- **SM1.2 risultato negativo**
- **SM1.3 divisione per zero**

L'istruzione **/I** esegue la divisione

2° / 1° → 2°

Occorre, con MOVW, caricare IN1 nell'uscita (2° operando) e poi eseguire la divisione tra l'uscita e IN2 (1° operando).

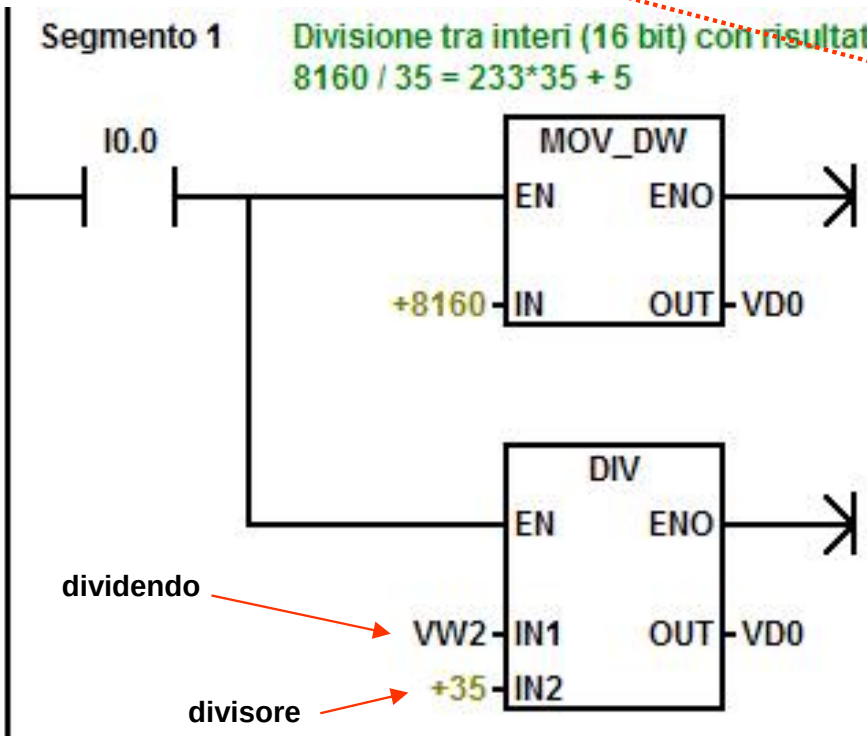
Divisione tra numeri interi (16 bit) con risultato in double word (32 bit)

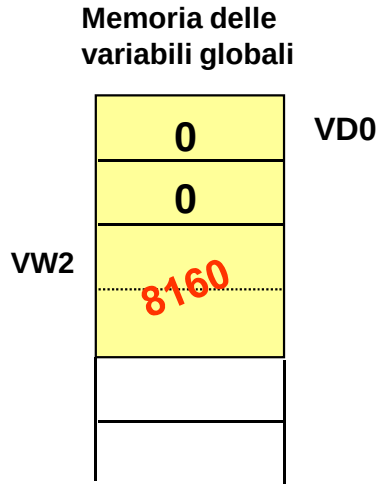
DIV divisore, dividendo

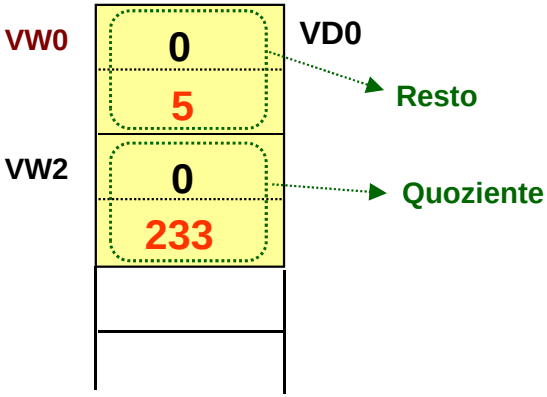
divisore: ***word*** **(16 bit)**
dividendo: ***double word*** **(32 bit)**

Segmento 1 Divisione tra interi (16 bit) con risultato in double word (32 bit)
8160 / 35 = 233*35 + 5

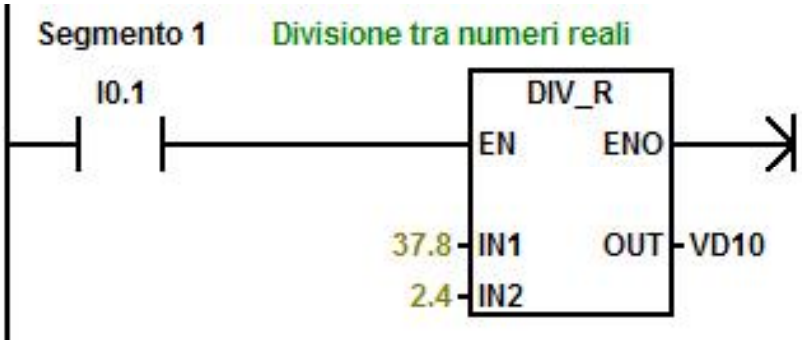
```
LD      I0.0
MOVD    +8160, VD0
DIV     +35, VD0
```



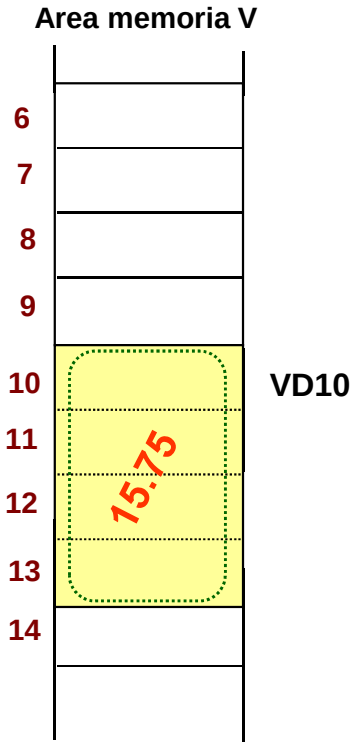




Divisione tra numeri reali (32 bit)



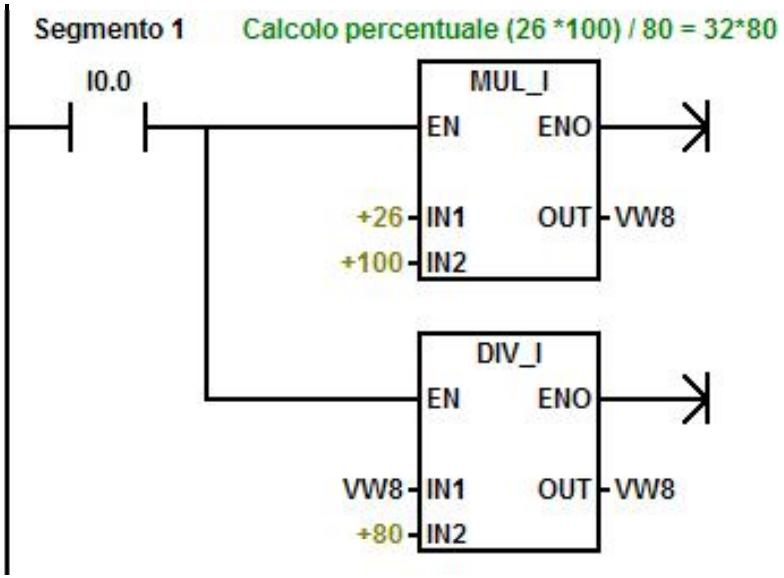

```
Segmento 1      Divisione tra numeri reali
LD      I0.1
MOVR    37.8, VD10
/R      2.4, VD10
```



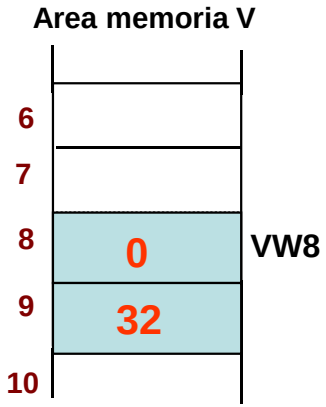
Esempio: calcolo di una percentuale

$$\frac{26}{80}\cdot 100 = 32.5\%$$

1° versione




```
Segmento 1      Calcolo percentuale (26 *100) / 80 = 32*80
LD      I0.0
MOVW    +26, VW8
*I      +100, VW8
/I      +80, VW8
```

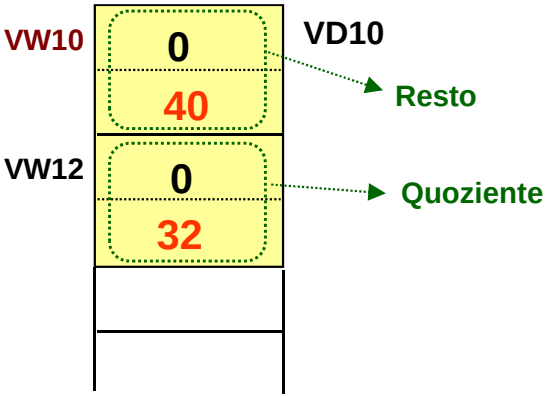


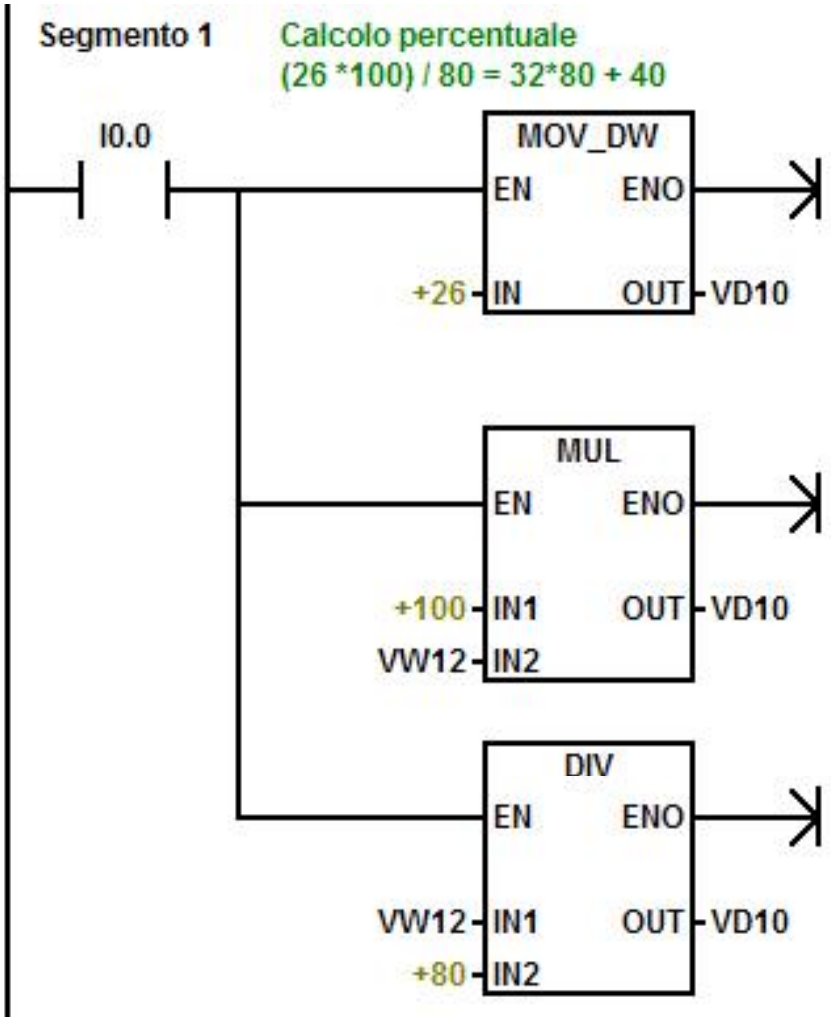
NB: nella word VW8 viene caricato solo il quoziente, il resto invece, se presente, è perso.

2° versione

Segmento 1 Calcolo percentuale
(26 *100) / 80 = 32*80 + 40

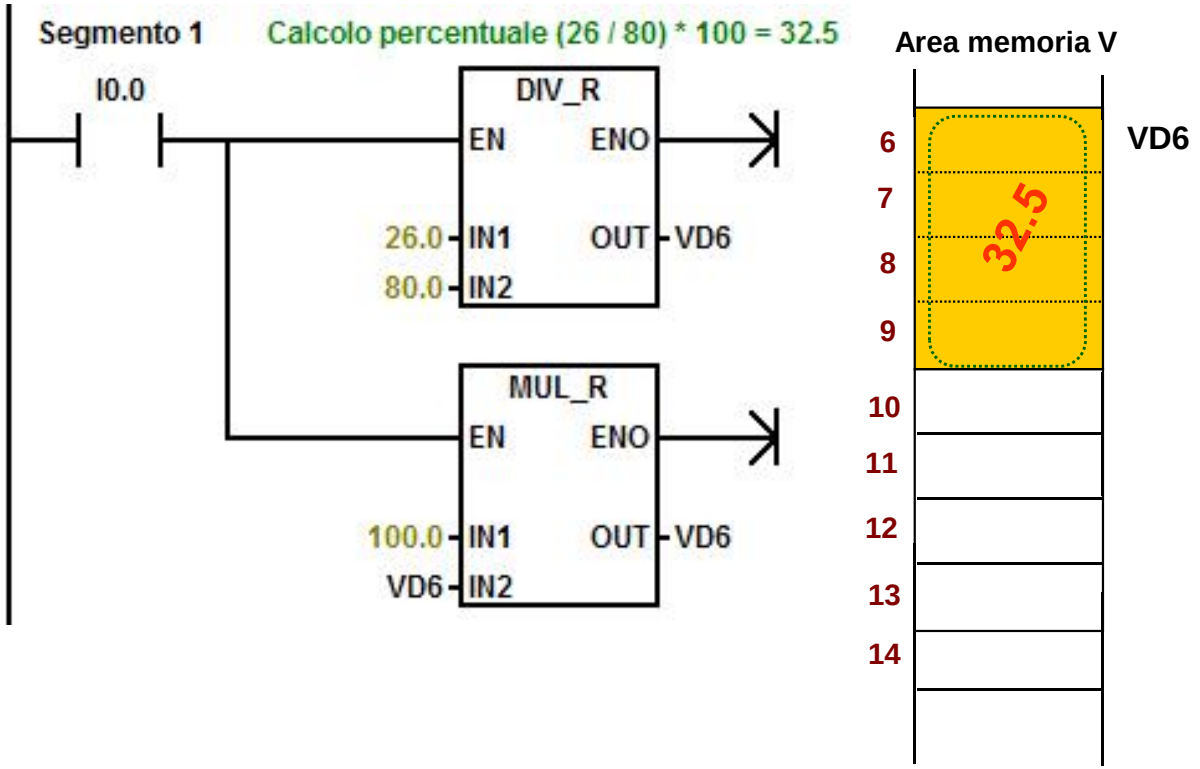
```
LD      I0.0
MOVD    +26, VD10
MUL     +100, VD10
DIV     +80, VD10
```





3° versione

```
Segmento 1      Calcolo percentuale (26 / 80) * 100 = 32.5
LD      I0.0
MOVR    26.0, VD6
/R      80.0, VD6
*R      100.0, VD6
```



<u>Tempi di esecuzione</u>:

***I 71 «s**

MUL 70 «s

***R 100-130 «s**

DIV 119 «s

/R 300-360 «s

<u>NB</u>: quando non strettamente necessario conviene evitare le operazioni con numeri reali.