

Sommario

Manuale Easy	3
<i>Introduzione alla programmazione</i>	3
Unità di configurazione	3
Tipi di variabili	4
Gli Identificatori	5
Le costanti	5
Le variabili SYSTEM	5
Le variabili GLOBAL	5
Utilizzo variabili SYSTEM e GLOBAL	6
Le variabili ARRAY SYSTEM	6
Le variabili ARRAY GLOBAL	6
Utilizzo variabili ARRAY SYSTEM ed ARRAY GLOBAL	7
Le variabili TIMER	7
Utilizzo variabili TIMER	7
Le variabili INPUT ed OUTPUT	8
Le variabili DATAGROUP	8
Utilizzo variabili DATAGROUP	10
Sezione BUS	10
<i>Simboli Speciali</i>	10
<i>Istruzioni QCL</i>	11
Gli operatori del QCL	11
Esempio variabili GLOBAL	12
Utilizzo variabili SYSTEM e GLOBAL	12
Esempio variabili ARRAY GLOBAL	12
Utilizzo variabili ARRAY SYSTEM ed ARRAY GLOBAL	12
Esempio variabili TIMER	13
Utilizzo variabili TIMER	13
Esempio variabili INPUT ed OUTPUT	13
Esempio variabili DATAGROUP	14

Manuale Easy

• TIPI DI VARIABILI [Dettagli](#)



FLAG	Variabili booleane. Range di valori: 0 a 1. Spazio memoria: dipendente dal loro numero. Sintassi: <nome variabile> F
BYTE	Range di valori: -128 a +127. Spazio memoria: 1 byte cadauna. Sintassi: <nome variabile> B
WORD	Range di valori: -32768 a +32767. Spazio memoria: 2 byte cadauna. Sintassi <nome variabile> W
LONG	Range di valori: -2147483648 a +2147483648. Spazio memoria: 4 byte cadauna. Sintassi <nome variabile> L
SINGLE	Range di valori: -3.4 x 1038 a +3.4 x 1038. Spazio memoria: 4 byte cadauna. Sintassi <nome variabile> S

- **IDENTIFICATORI** “;” seguito da un commento [Dettagli](#)
- **COSTANTI** Keyword: CONST Sintassi: <nome costante> <valore> [Dettagli](#)
- **LE VARIABILI** [Dettagli](#)

SYSTEM	Variabili ritentivi. Sintassi: Titolo SYSTEM <nome variabile> <tipo> Dettagli
GLOBAL	Variabili non ritentivi. Sintassi: Titolo GLOBAL <nome variabile> <tipo> Dettagli
ARRAY SYSTEM	Variabili ritentivi uguali. Sintassi: Titolo ARRSYS <nome variabile> <tipo> <numero elementi> Dettagli
ARRAY GLOBAL	Variabili non ritentivi uguali. Sintassi: Titolo ARRGBL <nome variabile> <tipo> <numero elementi> Dettagli
TIMER	Variabili temporizzatori. Sintassi: Titolo TIMER <nome variabile> Dettagli
INPUT	Variabili con riferimenti a input. Sintassi: Titolo INPUT <nome variabile> <tipo> <io address> Dettagli
OUTPUT	Variabili con riferimenti a output. Sintassi: Titolo OUTPUT <nome variabile> <tipo> <io address> Dettagli
DATAGROUP	Variabili XXXXXXXX XXXXXX. Sintassi: Titolo DATAGROUP <xxxxxxx> <xxxx> <xxxxxxx> Dettagli

- **Sezione BUS XXXX XXX XXXXXX** [Dettagli](#)
- **SIMBOLI SPECIALI** [Dettagli](#)
- **ISTRUZIONI QCL** [Dettagli](#)
- **OPERATORI** [Dettagli](#)

| Operatore di assegnamento | ...|

ARITMETICI	somma (+), sottrazione (-), moltiplicazione (*), divisione (/) ed il resto della divisione (%)XXX
LOGICI	(AND, OR e NOT) xxx
BINARI	(ANDB, ORB, NOTB, XORB)XXX
RELAZIONALI	EQ( LT(<) LE (≡) GT (>) GE (>  XXX
LIVELLO DI PRECEDENZA	() XXX

Introduzione alla programmazione

Unità di configurazione

L'unità di configurazione è un componente fondamentale di un progetto per QMOVE; è unica nel progetto e si accompagna al resto delle unità componenti il progetto (che possono essere più di una).

L'unità di configurazione contiene le dichiarazioni di tutte le variabili e di tutte le costanti utilizzate nell'applicativo; inoltre viene anche definita la composizione hardware del sistema QMOVE adottato, specificando il tipo di scheda CPU, le schede intelligenti e le schede non intelligenti presenti sul rack. L'unità di configurazione deve essere editata tramite l'editor dell'ambiente QVIEW.

L'unità di configurazione è composta da una serie di sezioni a seconda del gruppo di variabili da dichiarare. Per la definizione specifica delle varie sezioni si rimanda ai paragrafi seguenti; ogni sezione è identificata da una parola chiave che indica l'inizio della sezione stessa all'interno dell'unità di configurazione.

Le parole chiavi sono:

- CONST
- SYSTEM
- GLOBAL
- ARRSYS
- ARRGBL
- TIMER
- INPUT
- OUTPUT
- DATAGROUP
- BUS
- INTDEVICE (sezione inserita nel capitolo dedicato ai device)
- EXTDEVICE (sezione inserita nel capitolo dedicato ai device)

Gli esempi di composizione dell'unità di configurazione vengono presentati nel resto del manuale al momento delle dichiarazioni delle variabili.

Tipi di variabili

QCL prevede cinque tipi principali di dato:

Flag

Il dato di tipo FLAG è utilizzato per la definizione di variabili booleane che hanno un range di valori compreso tra 0 e 1. L'occupazione della memoria da parte delle variabili di questo tipo dipende dal loro numero. La sintassi per la definizione di una variabile di tipo FLAG nell'unità di configurazione è la seguente:

```
<nome variabile> F
```

Byte

Il dato di tipo BYTE è utilizzato per la definizione di variabili che hanno un range di valori compreso tra -128 e +127; ogni variabile occupa un byte di memoria. La sintassi per la definizione di una variabile BYTE nell'unità di configurazione è la seguente:

```
<nome variabile> B
```

Word

Il dato di tipo WORD è utilizzato per la definizione di variabili che hanno un range di valori compreso tra -32768 e +32767; ogni variabile occupa due byte di memoria. La sintassi per la definizione di una variabile WORD nell'unità di configurazione è la seguente:

```
<nome variabile> W
```

Long

Il dato di tipo LONG è utilizzato per la definizione di variabili che hanno un range di valori compreso tra -2147483648 e +2147483647; ogni variabile occupa quattro byte di memoria. La sintassi per la definizione di una variabile LONG nell'unità di configurazione è la seguente:

```
<nome variabile> L
```

Single

Il dato di tipo SINGLE è utilizzato per la definizione di variabili reali che hanno un range di valori compreso tra -3.4×10^{38} e $+3.4 \times 10^{38}$; ogni variabile occupa quattro byte di memoria. La sintassi per la definizione di una variabile SINGLE nell'unità di configurazione è la seguente:

```
<nome variabile> S
```

La precisione massima delle variabili SINGLE è di sette cifre (contando le cifre prima e dopo la virgola). Esempio:
 1,234567 x 10³ Precisione del millesimo; incremento minimo = 1 millesimo.
 1,234567 x 10⁴ Precisione del centesimo; incremento minimo = 1 centesimo.
 1,234567 x 10⁹ Precisione delle centinaia; incremento minimo = 100.

Tabella riassuntiva dei tipi di variabili utilizzabili

Assegnando ad una variabile un valore esterno al range consentito si verifica la condizione di overflow.

Tipo dato	Codice	Spazio occupato memoria (Bit)	Intervallo
FLAG	F	1	0 ÷ 1
BYTE	B	8	-128 ÷ 127
WORD	W	16	-32768 ÷ 32767
LONG	L	32	-2147483648 ÷ 2147483647
SINGLE	S	32	-3.4×10^{38} ÷ $+3.4 \times 10^{38}$

Gli Identificatori

Il carattere punto e virgola (“;”) ha lo scopo di identificare una riga di commento. Tutto il testo inserito dopo questo carattere non verrà considerato codice QCL. Il carattere punto e virgola vale solo per la linea in cui è inserito.

Gli identificatori sono dei nomi tramite i quali è possibile fare dei riferimenti ad oggetti o ad etichette. Ogni identificatore è formato da uno o più caratteri alfanumerici tenendo presente che il primo deve essere una lettera. I caratteri alfanumerici possono essere riassunti in:

- dalla “A” alla “Z”
- dalla “a” alla “z”
- da “0” a “9”
- “ ”
- “_”

I nomi

I nomi vengono usati per agevolare l'identificazione di un oggetto all'interno del sistema QMOVE. Per oggetto si intende qualsiasi entità gestibile a livello di linguaggio e provvista di caratteristiche fisiche proprie come ad esempio: variabili, ingressi, uscite e devices interni od esterni. I nomi sono limitati ad una lunghezza massima di 12 caratteri ed essendo immagazzinati nella memoria del QMOVE incidono per un byte a carattere nell'occupazione di tale risorsa. In tutto il progetto non vi possono essere due oggetti con lo stesso nome.

Le costanti

Una costante è un carattere o una stringa di caratteri utilizzabile come valore in un applicativo. Le costanti vengono dichiarate nell'unità di configurazione e devono essere poste in seguito alla parola chiave “CONST”. Viene riportata la sintassi per la definizione di costanti:

```

;-----
; Definizione costanti
;-----
CONST
<nome costante> <valore>

```

dove:

CONST	Parola chiave che indica la definizione delle costanti.
<nome costante>	Nome della costante.
<valore>	valore associato alla costante.

Esempio

Le variabili SYSTEM

Le variabili SYSTEM mantengono il loro valore anche allo spegnimento del sistema.

Con il nome SYSTEM vengono raggruppate tutte le variabili ritentive di uso generico; possono essere di tipo FLAG, BYTE, WORD, LONG o SINGLE e sono accessibili sia in scrittura che in lettura. Vengono dichiarate nell'unità di configurazione e devono essere poste in seguito alla parola chiave “SYSTEM”. Viene riportata la sintassi per la definizione di variabili SYSTEM.

```

;-----
; Definizione variabili SYSTEM
;-----
SYSTEM
<nome variabile> <tipo>

```

dove:

SYSTEM	Parola chiave che indica la definizione di variabili SYSTEM.
<nome variabile>	Nome della variabile SYSTEM.
<tipo>	Tipo della variabile e può essere: F (FLAG) B (BYTE) W (WORD) L (LONG) S (SINGLE)

Esempio

Le variabili GLOBAL

Con il nome GLOBAL vengono raggruppate tutte le variabili non ritentive ad uso generico; hanno la caratteristica di venire

azzerate ad ogni accensione del sistema; possono essere di tipo FLAG, BYTE, WORD, LONG e SINGLE e sono accessibili sia in scrittura che in lettura. Vengono dichiarate nell'unità di configurazione e devono essere poste in seguito alla parola chiave "GLOBAL". Viene riportata la sintassi per la definizione di variabili GLOBAL.

```

;-----
; Definizione variabili GLOBAL
;-----
GLOBAL
<nome variabile> <tipo>

```

dove:

GLOBAL	Parola chiave che indica per la definizione di variabili GLOBAL.
<nome variabile>	Nome della variabile GLOBAL.
<tipo>	Tipo della variabile e può essere: F (FLAG) B (BYTE) W (WORD) L (LONG) S (SINGLE)

Per esempi relativi alla sintassi per la definizione, fare riferimento a quelli indicati per le variabili SYSTEM.

Quanto descritto in questo esempio deve essere inserito nel progetto Firstapp.qm4 in modo da poter mettere immediatamente in pratica quanto appreso.

Esempio

Utilizzo variabili SYSTEM e GLOBAL

Le variabili SYSTEM e GLOBAL possono essere utilizzate sia a destra che a sinistra di un'assegnazione o dentro un'espressione (indicandone soltanto il nome). Non è possibile indicizzare una variabile di questo tipo.

Esempi di scrittura ed utilizzo delle variabili system e global sono inseriti nello sviluppo del progetto **Firstapp.qm4**.

Le variabili ARRAY SYSTEM

Una variabile Array System è un insieme di variabili ritentive dello stesso tipo, aventi la stessa dimensione ed alle quali è possibile accedere tramite un nome comune, riferendosi ad uno specifico elemento tramite un indice. Non sono previsti array di variabili tipo FLAG.

Come le SYSTEM sono variabili ritentive di uso generale e devono essere dichiarate di seguito alla parola chiave "ARRSYS". Viene riportata la sintassi per la definizione di variabili ARRAY SYSTEM.

```

;-----
; Definizione variabili ARRAY SYSTEM
;-----
ARRSYS
<nome variabile> <tipo> <numero_elementi>

```

dove:

GLOBAL	Parola chiave che per la definizione di variabili ARRAY SYSTEM.
<nome variabile>	Nome della variabile ARRAY SYSTEM.
<tipo>	Tipo della variabile e può essere: B (BYTE) W (WORD) L (LONG) S (SINGLE)
<numero_elementi>	Numero di elementi che compongono la variabile.

Il numero massimo di elementi consentiti in un array è di 65535. Non si possono indicare dimensioni negative o dimensioni con parte decimale. Come dimensione dell'array è possibile usare delle costanti già definite nella sezione CONST; se questa avesse valore decimale verrà troncato: ad esempio il valore 200.34 viene forzato a 200.

Esempio

Le variabili ARRAY GLOBAL

Una variabile Array Global è un insieme di variabili non ritentive dello stesso tipo, aventi la stessa dimensione ed alle quali è possibile accedere tramite un nome comune, riferendosi ad uno specifico elemento tramite un indice. Non sono previsti array di variabili tipo FLAG.

Come le GLOBAL sono variabili non ritentive di uso generale e devono essere dichiarate di seguito alla parola chiave "ARRGBL".

Viene riportata la sintassi per la definizione di variabili ARRAY GLOBAL:

```

; Definizione variabili ARRAY GLOBAL
;-----
ARRGBL
<nome variabile> <tipo> <numero_elementi>

```

dove:

ARRGBL	Parola chiave per la definizione di variabili ARRAY GLOBAL.
<nome variabile>	Nome della variabile ARRAY GLOBAL.
<tipo>	Tipo della variabile e può essere: B (BYTE) W (WORD) L (LONG) S (SINGLE)
<numero_elementi>	Numero di elementi che compongono la variabile.

Per esempi relativi alla sintassi per la definizione, fare riferimento a quelli indicati per le variabili ARRAY SYSTEM.

[Esempio](#)

Utilizzo variabili ARRAY SYSTEM ed ARRAY GLOBAL

Le variabili Array possono essere utilizzate sia a destra che a sinistra di un'assegnazione o dentro un'espressione con la seguente sintassi:

```
< Nome Array >[ i ]
```

dove "i" può essere un numero, una costante, una variabile (non di tipo Single) o un'espressione complessa. Gli indici di un array partono sempre da 1 e non sono ammessi valori superiori al numero massimo di elementi.

Le variabili TIMER

Sono variabili utilizzate per realizzare temporizzazioni alle quali può essere assegnato un valore intero (espresso in ms) che rappresenta il tempo che deve trascorrere (dal momento dell'assegnazione); in lettura è disponibile lo stato di "temporizzazione terminata" (1) o "temporizzazione attiva" (0).

Mediante l'istruzione *<nome variabile>:remain* è possibile leggere il valore del tempo rimanente prima della fine del timer. Le variabili timer vengono dichiarate nell'unità di configurazione e devono essere poste in seguito alla parola chiave "TIMER". Viene riportata la sintassi per la definizione di variabili TIMER.

```

; Definizione variabili TIMER
;-----
TIMER
<nome variabile>

```

dove:

TIMER	Parola chiave per la definizione di variabili TIMER.
<nome variabile>	Nome della variabile TIMER.

[Esempio](#)

Utilizzo variabili TIMER

Le variabili timer possono essere utilizzate sia a destra che a sinistra di un'assegnazione o dentro un'espressione indicando semplicemente il nome della variabile:

```
< Nome Timer >
```

Quando la variabile timer si trova a sinistra dell'assegnazione si definisce il caricamento di un valore sul timer (valore espresso in millisecondi):

```
tMioTimer = 1000 ;Imposta il timer tMioTimer a 1 secondo.
```

Quando la variabile timer si trova a destra dell'assegnazione o all'interno di una espressione ne viene letto lo stato (0 = Temporizzazione attiva, 1 = Temporizzazione terminata):

```
gfIsTimerEnd = tMioTimer ;Assegna alla variabile gfIsTimerEnd lo stato del timer.
```

oppure

```
IF(tMioTimer) ;Se timer tMioTimer terminato esegue il blocco codice all'interno dell'IF.
.....
ENDIF
```

È, inoltre, possibile leggere il valore del tempo rimanente prima dello scadere del timer (il valore restituito è espresso in millisecondi):

```
< Nome Timer >:remain
```

Esempio:

```
glTempoRiman = tMioTimer:remain
```

Le variabili INPUT ed OUTPUT

Sono tutte le variabili che fanno riferimento ad ingressi od uscite digitali. Vengono dichiarate nell'unità di configurazione; devono essere poste in seguito alle parole chiave "INPUT" per gli ingressi o "OUTPUT" per le uscite. Viene riportata la sintassi per la definizione di variabili INPUT ed OUTPUT.

```
-----
; Definizione variabili INPUT ed OUTPUT
-----
INPUT
<nome variabile> <tipo> <io address>
OUTPUT
<nome variabile> <tipo> <io address>
```

Gli I/O address sono disponibili sulle schede tecniche hardware delle schede utilizzate.

dove:

INPUT	Parola chiave per la definizione di variabili INPUT.
OUTPUT	Parola chiave per la definizione di variabili OUTPUT.
<nome variabile>	Nome della variabile.
<tipo>	Tipo della variabile e può essere: F per (FLAG) B per (BYTE)
<io address>	Indirizzo dell'INPUT o OUTPUT così composto: <i>Slot number.name</i> : <i>Slot number</i> è il numero dello slot nel quale è posta la scheda con la risorsa hardware. <i>name</i> : è il nome che fa riferimento all'indirizzo fisico dell'I/O (definito nei riferimenti hardware).

Un'applicazione interessante degli ingressi e delle uscite digitali è quello del raggruppamento degli stessi in un unico identificatore. Questo identificatore è analogo ad una variabile di otto bit dove ogni ingresso o uscita digitale rappresenta un bit.

Se per esempio abbiamo una terza scheda MIX montata sullo slot 3 sono possibile le ulteriori dichiarazioni:

Nella sezione INPUT

```
ibIngresso B 3.INPB ;8 ingressi digitali formano un unico ingresso di dimensione 1 byte.
```

Nella sezione OUTPUT

```
obUscita B 3.OUTB ;8 uscite digitali formano un'unica uscita di dimensione 1 byte.
```

Se si dispone di schede con più di otto ingressi o uscite digitali (per esempio D24, I24 oppure O24), è possibile raggrupparli in gruppi di otto modificando la dichiarazione:

Nella sezione INPUT

```
ibIngresso1 B 3.INPB1 ;Primo gruppo di otto (1÷8) ingressi
;digitali raggruppati in un byte.
ibIngresso2 B 3.INPB2 ;Secondo gruppo di otto (9÷16)
;ingressi digitali raggruppati in
;un byte.
```

Nella sezione OUTPUT

```
obUscita1 B 3.OUTB1 ;Primo gruppo di otto (1÷8) uscite
;digitali raggruppate in un byte.
obUscita2 B 3.OUTB2 ;Secondo gruppo di otto (9÷16) uscite
;digitali raggruppate in un byte.
```

Esempio

Le variabili DATAGROUP

Le variabili finora presentate possono essere pensate come una zona di memoria, di dimensioni diverse a seconda del tipo della

variabile, contenente un singolo valore. Nel progetto si fa riferimento a tale zona di memoria tramite un identificatore che è il nome assegnato alla variabile nell'unità di configurazione.

Le variabili Datagroup sono una particolare struttura di dati. Quando si dichiara un datagroup, si organizza una parte della memoria come una tabella formata da righe e colonne. Le colonne sono chiamate "programmi", mentre le righe "passi" (in inglese "step").

Ogni programma (colonna) contiene due tipologie di variabili:

- Statiche.
- Indicizzate.

Le statiche sono delle variabili che possono assumere un valore diverso a seconda del programma (colonna) a cui si fa riferimento. Nella dichiarazione, ognuna di queste variabili è identificata con un unico nome quindi, per poter far riferimento ai diversi valori che può assumere, si deve utilizzare un metodo di indicizzazione. Per esempio, per far riferimento alla variabile "dsIveMa" del programma (colonna) 5, viene adottato questo metodo:

dsIveMa[5]

Le indicizzate sono variabili che possono assumere un valore diverso a seconda del programma e del passo (riga) a cui si fa riferimento. Nella dichiarazione, ognuna di queste variabili è identificata con un unico nome quindi, per potersi riferire ai diversi valori che può assumere, si deve utilizzare un metodo di indicizzazione. Per esempio, per far riferimento alla variabile "ddwLuPe" del programma (colonna) 5 e del passo (riga) 3, viene adottato questo metodo:

ddwLuPe[5,3]

Graficamente il datagroup si può rappresentare in questo modo:

	Prog. 1	Prog. 2	Prog. 3	Prog. 4	Prog. 5	
	dsIveMa[1]	dsIveMa[2]	dsIveMa[3]	dsIveMa[4]	dsIveMa[5]	Statiche
Step 1	ddwLuPe[1,1]	—	—	—	—	Indicizzate
Step 2	ddwLuPe[1,2]	—	—	—	—	
Step 3	ddwLuPe[1,3]	—	—	—	ddwLuPe[5,3]	
Step 4	—	—	—	—	ddwLuPe[5,4]	
Step 5	—	—	—	—	ddwLuPe[5,5]	

Nelle due sezioni, statiche e indicizzate, del datagroup è possibile dichiarare più di una variabile.

Viene riportata la sintassi per la definizione di variabili DATAGROUP:

```

;-----
; Definizione del DataGroup
;-----
DATAGROUP
<nome DataGroup>
;Definizione del numero programmi
  DATAPROGRAM
    <numero programmi>
;Definizione variabili statiche
  <nome variabile> <tipo>
  <nome variabile> <tipo>
  <nome variabile> <tipo>
;Definizione del numero di passi
  STEP
    <numero passi>
;Definisce le variabili indicizzate di ciascun passo
  <nome variabile> <tipo>
  <nome variabile> <tipo>
  <nome variabile> <tipo>

```

dove:

DATAGROUP	Parola chiave per la definizione di un DataGroup.
<nome DataGroup>	Nome associato al DataGroup.
DATAPROGRAM	Parola chiave per la definizione delle variabili statiche nel DataGroup.
<num. programmi>	Numero di programmi (DataProgram) di cui è composto il DataGroup.
<nome variabile>	Nome della variabile statica del DataGroup.
<tipo>	Tipo della variabile statica e può essere: F (FLAG) B (BYTE) W (WORD) L (LONG) S (SINGLE)
STEP	Parola chiave per la definizione delle variabili indicizzate nel DataGroup.
<numero passi>	Numero di passi (Step) di cui è composto il DataGroup.
<nome variabile>	Nome della variabile indicizzata del DataGroup.
<tipo>	Tipo della variabile indicizzata e può essere: F (FLAG) B (BYTE) W (WORD) L (LONG) S (SINGLE)

La definizione del DATAGROUP è composta di 3 parti:

- una relativa alla definizione del nome del datagroup.
- una relativa all'impostazione del numero programmi e delle variabili statiche (inizia con la parola chiave DATAPROGRAM).
- una relativa all'impostazione del numero di passi di programma e delle variabili indicizzate (inizia con la parola chiave STEP).

Il nome datagroup segue tutte le regole generali finora incontrate per la sintassi dei nomi variabili.

Il numero programmi va scritto in forma numerica, oppure tramite l'ausilio di costanti e deve essere diverso da zero; il numero massimo di programmi è 65534.

Il numero passi va scritto in forma numerica, oppure tramite l'ausilio di costanti e deve essere diverso da zero. Il numero massimo di passi è 65534.

La sottosezione DATAPROGRAM è obbligatoria, mentre quella STEP è opzionale.

Non è possibile dichiarare una sezione STEP senza dichiarare almeno una variabile indicizzata. È possibile dichiarare una sezione STEP senza dichiarare una sezione DATAPROGRAM. Non è possibile dichiarare una sezione DATAPROGRAM senza dichiarare almeno una variabile statica. È possibile dichiarare una sezione DATAPROGRAM senza dichiarare una sezione STEP. In un DATAGROUP tutte le variabili, sia statiche che indicizzate, sono ritentive.

Per calcolare l'occupazione in memoria totale del DATAGROUP si deve tener presente che ogni variabile inserita nel datagroup occupata 4 byte (qualsiasi sia il tipo scelto per la variabile stessa). Quindi l'occupazione in byte è pari a:

$(N.\text{Programmi} \times N.\text{Variabili statiche} \times 4) + (N.\text{Programmi} \times N.\text{Passi} \times N.\text{Variabili indicizzate} \times 4)$.

Esempio

Utilizzo variabili DATAGROUP

Una variabile DataGroup (static o index), può essere utilizzata sia a destra che a sinistra di un'assegnazione o dentro un'espressione.

Sintassi nel caso di variabile statiche:

```
< Nome variabile Static > < [ num_prog ] >
```

Sintassi nel caso di variabile indicizzate:

```
< Nome variabile Index > < [ num_prog, num_step ] >
```

num_prog e *num_step* possono essere un numero (non SINGLE), una costante, una variabile o un'espressione complessa. Nel caso di numero o di costante, viene eseguito il controllo durante la compilazione che l'indice non superi la dimensione massima dichiarata in configurazione (rispettivamente per il numero programmi e il numero step); gli altri indici - *num_prog* e *num_step* - partono (in valore) da uno.

Un esempio di utilizzo dei datagroups viene presentato nell'esempio dell'istruzione FOR / NEXT.

Sezione BUS

La sezione BUS nell'unità di configurazione è indispensabile per dichiarare quali siano le dotazioni hardware che il programmatore ha a disposizione.

In questa sezione si dovrà indicare quale è la CPU utilizzata e inserita nello slot 1 e quali siano le altre schede negli altri slots. Ogni scheda è individuata da una parola chiave che ne identifica tipologia di hardware e, nel caso di schede intelligenti, versione del firmware; le parole chiavi sono reperibili nei manuali installazione e manutenzione dell'hardware utilizzato.

Esempio

Simboli Speciali

Vi sono dei simboli che sono dichiarabili esclusivamente nello slot 1 (CPU) e che servono per leggere gli ingressi in interruzione e per i device simulati. Di seguito è riportata una tabella riepilogativa per questi simboli:

Simbolo	Tipo segnale	Dimensione	Accesso
INT01	Input	Flag	Read
INT02	Input	Flag	Read
INT03	Input	Flag	Read
INT04	Input	Flag	Read
INT05	Input	Flag	Read

INT06	Input	Flag	Read
INT07	Input	Flag	Read
INT08	Input	Flag	Read
CNT01	Conteggio	Word	Read
CNT02	Conteggio	Word	Read
CNT03	Conteggio	Word	Read
CNT04	Conteggio	Word	Read
CNT05	Conteggio	Word	Read
AN01	DAC	Word	Write
AN02	DAC	Word	Write
AN03	DAC	Word	Write
AN04	DAC	Word	Write
AN05	DAC	Word	Write

Con il simbolo speciale INTnn è possibile dichiarare un ingresso fittizio che legge la linea di interrupt in ingresso alla CPU. Un esempio di utilizzo di questo simbolo speciale è:

```

;-----
; Definizione variabili INPUT ed OUTPUT
;-----
INPUT
  ifInter01      F      1.INT01
  ifInter02      F      1.INT02
  ifInter05      F      1.INT05
  ifInter06      F      1.INT06

```

Gli altri simboli della tabella sono utilizzati per creare dei device simulati. Utilizzando i simboli CNTnn in una dichiarazione di un normale device il conteggio che il device acquisisce non è reale, ma viene simulato dalla CPU stessa. Se per esempio vengono dichiarati i seguenti device (si veda dichiarazione dei device nei relativi manuali):

```

;-----
; INTDEVICE Declaration
;-----
INTDEVICE
eaSim EANPOS      2      1.CNT01  X  X.X  X.X
cnSim COUNTER3    4      1.CNT01  X  X.X  X.X

```

Il device EANPOS, che è un posizionatore analogico, genererà un profilo ideale della posizione nel tempo e il device COUNTER3 leggerà semplicemente la posizione di un asse dall'ingresso di conteggio virtuale 1.CNT01.

Se inoltre il device EANPOS viene dichiarato nel seguente modo:

```

;-----
; INTDEVICE Declaration
;-----
INTDEVICE
eaSim EANPOS      2      1.CNT01  X  X.X  1.AN01

```

la CPU genererà, oltre al profilo di posizione nel tempo, anche l'andamento di una tensione virtuale nel tempo per realizzare tale posizionamento. L'andamento della tensione nel tempo si può leggere da un apposito parametro del device EANPOS.

Istruzioni QCL

Il QCL (QEM Control Language) è un linguaggio nato appositamente per la programmazione del sistema QMOVE. Le caratteristiche principali del QCL sono la semplicità (poche ma potenti istruzioni) la facilità d'uso (somiglianza con il linguaggio BASIC) e l'orientamento verso l'automazione industriale grazie alle istruzioni appositamente studiate per il controllo assi.

Gli operatori del QCL

Nel linguaggio QCL vengono messi a disposizione tutti gli operatori elementari per la manipolazione dei dati.

Esempio Costanti

Quanto descritto in questo esempio deve essere inserito nel progetto Firstapp.qm4 in modo da poter mettere immediatamente in pratica quanto appreso.

Nell'unità di configurazione (firstapp) aggiungiamo la sezione per la dichiarazione delle costanti:

```

;-----
; Definizione COSTANTI
;-----
CONST
  TM SECONDO 1000 ;Costante con valore 1000 per il tempo di un secondo.
  DIM_ARRAY 7     ;Costante con valore 7 per la dimensione dell'array.
  DIM_PROG 10     ;Costante con valore 10 per il numero dei programmi nel datagroup.
  DIM_STEP 5      ;Costante con valore 5 per il numero degli step nel datagroup.

```

Esempio Variabili SYSTEM

Quanto descritto in questo esempio deve essere inserito nel progetto Firstapp.qm4 in modo da poter mettere immediatamente in pratica quanto appreso.

Attualmente nell'unità firstapp di esempio è già presente la sezione SYSTEM (nella quale è dichiarata la variabile slProva); aggiungiamo le seguenti variabili:

```

;-----
; Definizione Variabili SYSTEM
;-----
SYSTEM
slProva L      ;Variabile di prova dimensione LONG tipo SYSTEM.
sbSecondi B    ;Variabile per i secondi.
sbMinuti B     ;Variabile per i minuti.
swOre W        ;Variabile per le ore.

```

Esempio variabili GLOBAL

Nell'unità di configurazione (firstapp), in seguito alla sezione SYSTEM aggiungiamo la sezione per la dichiarazione del gruppo GLOBAL:

```

;-----
; Definizione Variabili GLOBAL
;-----
GLOBAL
gfMioFlag      F      ;Variabile global di tipo Flag.
gbTuoByte      B      ;Variabile global di tipo Byte.
gwSuoWord      W      ;Variabile global di tipo Word.
glNostroLong   L      ;Variabile global di tipo Long.
gsVostroSing   S      ;Variabile global di tipo Single.

```

Utilizzo variabili SYSTEM e GLOBAL

Le variabili SYSTEM e GLOBAL possono essere utilizzate sia a destra che a sinistra di un'assegnazione o dentro un'espressione (indicandone soltanto il nome). Non è possibile indicizzare una variabile di questo tipo.

Esempi di scrittura ed utilizzo delle variabili system e global sono inseriti nello sviluppo del progetto **Firstapp.qm4**.

Esempio variabili ARRAY SYSTEM

Quanto descritto in questo esempio deve essere inserito nel progetto Firstapp.qm4 in modo da poter mettere immediatamente in pratica quanto appreso.

Nell'unità di configurazione (firstapp), in seguito alla sezione GLOBAL aggiungiamo la sezione per la dichiarazione del gruppo ARRAY SYSTEM.

```

;-----
; Definizione variabili ARRAY SYSTEM
;-----
ARRSYS
asbMioArray B 10      ;Dichiarazione di un array system di byte di dimensione 10.
aslTuoArray L DIM_ARRAY ;Dichiarazione di un array system di long di dimensione DIM_ARRAY.

```

Si osservi che nel secondo array si è usata una costante per la dichiarazione della dimensione.

Esempio variabili ARRAY GLOBAL

Quanto descritto in questo esempio deve essere inserito nel progetto Firstapp.qm4 in modo da poter mettere immediatamente in pratica quanto appreso.

Nell'unità di configurazione (firstapp), in seguito alla sezione ARRAY SYSTEM aggiungiamo la sezione per la dichiarazione del gruppo ARRAY GLOBAL.

```

;-----
; Definizione variabili ARRAY GLOBAL
;-----
ARRGBL
arwMioArray W 15 ;Dichiarazione di un array global di byte di dimensione 15.

```

Un esempio della modalità di scrittura in un'array viene fatto in seguito nell'esempio relativo all'istruzione FOR / NEXT.

Utilizzo variabili ARRAY SYSTEM ed ARRAY GLOBAL

Le variabili Array possono essere utilizzate sia a destra che a sinistra di un'assegnazione o dentro un'espressione con la seguente sintassi:

```
< Nome Array >[ i ]
```

dove "i" può essere un numero, una costante, una variabile (non di tipo Single) o un'espressione complessa. Gli indici di un

array partono sempre da 1 e non sono ammessi valori superiori al numero massimo di elementi.

Esempio variabili TIMER

Quanto descritto in questo esempio deve essere inserito nel progetto Firstapp.qm4 in modo da poter mettere immediatamente in pratica quanto appreso.

Nell'unità di configurazione (firstapp), in seguito alla sezione ARRAY GLOBAL aggiungiamo la sezione per la dichiarazione del gruppo TIMER.

```

;-----
; Definizione TIMER
;-----
TIMER
  tTempo1 ;Identificativo del primo timer.
  tTempo2 ;Identificativo del secondo timer.
  tSecondi ;Identificativo del timer per i secondi.
  tDelay ;Identificativo del timer per un ritardo.

```

Un'esempio di uso della variabile timer viene presentato in seguito nell'esempio per l'istruzione IF / ELSE / ENDIF per la realizzazione di un semplice orologio.

Utilizzo variabili TIMER

Le variabili timer possono essere utilizzate sia a destra che a sinistra di un'assegnazione o dentro un'espressione indicando semplicemente il nome della variabile:

< Nome Timer >

Quando la variabile timer si trova a sinistra dell'assegnazione si definisce il caricamento di un valore sul timer (valore espresso in millisecondi):

```
tMioTimer = 1000 ;Imposta il timer tMioTimer a 1 secondo.
```

Quando la variabile timer si trova a destra dell'assegnazione o all'interno di una espressione ne viene letto lo stato (0 = Temporizzazione attiva, 1 = Temporizzazione terminata):

```
gfIsTimerEnd = tMioTimer ;Assegna alla variabile gfIsTimerEnd lo stato del timer.
```

oppure

```

IF(tMioTimer) ;Se timer tMioTimer terminato esegue il blocco codice all'interno dell'IF.
  ....
ENDIF

```

È, inoltre, possibile leggere il valore del tempo rimanente prima dello scadere del timer (il valore restituito è espresso in millisecondi):

< Nome Timer >:remain

Esempio:

```
glTempoRiman = tMioTimer:remain
```

Esempio variabili INPUT ed OUTPUT

Quanto descritto in questo esempio deve essere inserito nel progetto Firstapp.qm4 in modo da poter mettere immediatamente in pratica quanto appreso.

Per inserire la sezione relativa alla dichiarazione del gruppo di INPUT e di OUTPUT è necessario disporre di una scheda che metta a disposizione un insieme di ingressi e di uscite digitali. In questo esempio faremo riferimento ad una scheda MIX, la quale dispone di otto ingressi e otto uscite digitali identificabili nei manuali delle schede tecniche con X.INP01 ... X.INP08 e X.OUT01 ... X.OUT08, rispettivamente, dove X è il numero dello slot in cui è installata la scheda. Nell'unità di configurazione (firstapp), in seguito alla sezione TIMER aggiungiamo la sezione per la dichiarazione delle variabili INPUT ed OUTPUT.

```

;-----
; Definizione degli INPUT
;-----
INPUT
  ifSetUp0re F 2.INP01 ;Primo ingresso digitale della scheda
                       ;nello slot 2.
  ifSetUpMin F 2.INP02 ;Secondo ingresso digitale della scheda
                       ;nello slot 2.
  ifAbilUI F 2.INP03 ;Terzo ingresso digitale della scheda
                    ;nello slot 2.
;-----
; Definizione degli OUTPUT
;-----
OUTPUT
  ofUscita1 F 2.OUT01 ;Prima uscita digitale della scheda
                    ;nello slot 2.

```

```
ofUscita2 F 2.OUT02 ;Seconda uscita digitale della scheda
;nello slot 2.
```

Esempio variabili DATAGROUP

Quanto descritto in questo esempio deve essere inserito nel progetto Firstapp.qm4 in modo da poter mettere immediatamente in pratica quanto appreso.

Nell'unità di configurazione (firstapp), in seguito alla sezione di dichiarazione degli INPUT ed OUTPUT inseriamo le dichiarazioni del gruppo DATAGROUP:

```
-----
; Definizione del DATAGROUP
-----
DATAGROUP
dmioDataGrp          ;Nome identificativo del
                    ;Datagroup.
  DATAPROGRAM
  DIM_PROG           ;Numero dei programmi.
  -----Definizione variabili statiche-----
  dsfStat1 F         ;variabile statica di tipo Flag.
  dswStat2 W         ;Variabile statica di tipo Word.
  STEP
  DIM_STEP           ; numero di passi per programma
  -----Definisce le variabili di ciascun passo-----
  ddbDin1 B          ;Variabile dinamica di tipo Byte.
  ddlDin2 L          ;Variabile dinamica di tipo long.
```

Esempio configurazione BUS

Nell'unità di configurazione (firstapp), in seguito alla sezione di dichiarazione dei DATAGROUP inseriamo le dichiarazioni relative alla configurazione del BUS:

```
-----
; Configurazione BUS
-----
BUS
1 CPUB 02 ;Scheda CPUB-02 installata (slot 1).
2 IMIXA 00 ;Scheda MIX installata (slot 2).
3 . . . ;Lo slot 3 è il primo libero.
```

Dopo aver configurato in questo modo il BUS è possibile compilare il progetto. Se ci sono errori nella compilazione verrà visualizzato un messaggio di errore che darà una descrizione sommaria del tipo di errore, indicando il file, riga e colonna in cui è localizzato.

Una volta compilato ed eseguito correttamente il download del progetto è possibile visualizzare, con i comandi del menu monitor, tutte le variabili introdotte, compresi gli elementi degli arrays e le variabili dei datagroup; con un doppio clic sul nome della variabile appare una piccola finestra per l'inserzione del valore.

Immettendo un valore diverso da zero in una variabile global, quando si mette in stato di stop la CPU questa variabile perde il valore impostato e si azzerà. Ora abbiamo a disposizione una serie di variabili su cui sperimentare le istruzioni QCL descritte in seguito.

Se non si ha a disposizione l'hardware indicato nella configurazione del bus, dovranno essere inserite le parole chiave relative alle schede in possesso. È chiaro che per il corretto funzionamento dell'esempio Firstapp.qm4 è necessario che nello slot 2 venga inserito un hardware che metta a disposizione ingressi ed uscite digitali.

- [Istruzioni QCL](#)
- [Funzioni di libreria QCL](#)
- [Editor Ladder](#)
- [Multitasking](#)
- [Device](#)
- [Interfaccia Qview](#)
- [Debug](#)
- [Le librerie QCL](#)
- [Particolarità del QCL](#)
- [Appendice A: Limitazioni QCL](#)
- [Appendice B: Conversione e promozione di tipo](#)
- [Appendice C: Convenzioni di scrittura](#)
- [Appendice D: Parole chiave](#)
- [Appendice E: Tasti caldi](#)
- [Appendice F: Files generati](#)
- [Appendice G: Compatibilità con le versioni precedenti](#)
- [Appendice H: Segnalazioni di errore](#)
- [Appendice I: Errori durante il download](#)
- [Appendice L: Errori durante la compilazione Ladder](#)
- [Appendice M: Qpaint synchronization](#)

- [Appendice N: Creare le funzioni utente](#)

Documento generato automaticamente da **Qem Wiki** - <https://wiki.qem.it/>

Il contenuto wiki è costantemente aggiornato dal team di sviluppo, è quindi possibile che la versione online contenga informazioni più recenti di questo documento.