

Sommario

RULES OF USING A FUNCTION	3
<i>Codificazione delle funzioni</i>	4
<i>Note per l'utente</i>	4

RULES OF USING A FUNCTION

A QCL function is a piece of code that allows you to solve particular problems, to execute data processing, to provide specific functionality with a QMove application. In order to use a QCL function in a task code you just call it passing the required arguments, as is described below in this simple manual. This approach has some advantages: are found the tested and prewritten pieces of code, that solve some of the most frequent problems in an optimized faced in drafting an application for QMove; provide the ability to reuse as many times as you want the same function, without having to worry about rewriting it every time; there is a possibility to simply update the library as it is equipped with new features, maintaining an absolute compatibility with the applications already developed.

For as defined, QCL functions do not return a value, they can only do so through variables that are their past. If we denote with FuncQCL01(...) any QCL library function these situations cannot verify and give an error in compiling

- cannot be used on the right of an assignment

```
sInt01 = FuncQCL01(...) ERROR
```

- cannot be used as parameters of IF, FOR, WHILE instructions

```
IF (FuncQCL01(...) AND ...) ERROR FOR (sInt01 = 1, sInt01 LT FuncQCL01(...), ...) ERROR WHILE (FuncQCL01(...) LT 24) ERROR
```

The only correct way to call a function QCL is this:

```
FuncQCL01(...)
```

that is, as if it were a new QCL instruction. Although a QCL function does not return a value, it's however possible for a QCL function set of values to parameters passed as argument and so basically it's like if you had more return values. This is one more feature than the normal functions or procedures. See the description of the parameters of a function like below explained:



[IN] indicates that the parameter is passed by QCL task is used by the function to its proper execution.

[OUT] indicates that the parameter is passed by QCL task will be changed by the function QCL to indicate possible errors in its performance or to indicate return values. In this way it is possible to get even more return values.

The *variable type* and the size (*DIM*) can have multiple elements separated by a bar (/) showing that the same variable can be configured in different ways according to the needs of the user of the function. The *variable names* passed to the function are left to the imagination of the user they must respect only the limit of 12 characters, in fact, names listed on table only serve as an example.

See some simple examples:

AC10AvergArr (Myarray, val_medio, ok_calc)

The function calculates the average value of an array passed as an argument. In this case (as you can see from the description) MyArray is an input parameter: is the array to which you want to calculate the average value. val_medio and ok_calc are output parameters: denote the mean value calculated by the function and a flag that indicates when the calculation is completed. When ok_calc takes value 1 means that the function has finished his calculations, and then you can retrieve this value by reading the val_medio variable.

In the *Note*, for each function, whether you need to call at any logical application per round. There are functions that for operations that they do, should be called in a cyclical part of the code, While a few may also be called non-cyclical part of the code, as they quickly solve the problem. Remember what are these two parts in which you divide a task QCL

```
; start task
; non-cyclical part of the task code
MAIN:
; cyclical part of the task code
JUMP MAIN
END
```

The designation on the "where" call a function is given in the description of each function. There are functions that can be called either a cyclical area than in a non-cyclical area; in the description then it will not be made any reference to this way of working. Instead for those functions that require expressly the use in a cyclical area (for example the Blink function) this will be written clearly in the description.

Another feature to consider for certain functions is: when you need lengthy operations, the function itself is concerned to execute a WAIT statement inside, every 180 milliseconds. This is to prevent the "hang" QCL task execution due to special and onerous tasks. Usually such functions there is an argument of flag type that can be monitored to verify when the function has finished its task.

Codificazione delle funzioni

Tutte le funzioni della libreria prosseguono un prefisso a due lettere che varia di significato in base alla loro funzionalità.

Le due lettere possono identificare un specifico gruppo di appartenenza oppure descrivere singolarmente le caratteristiche di tale funzione.

In questo caso le prime due lettere della funzione servono rispettivamente per dare un'indicazione sui dati di utilizzo della funzione e per indicare il tipo di azione che essa compie.

Il loro significato è spiegato di seguito.

Codice unico a due lettere

SY = Funzioni di sistema

Codice a due lettere: prima lettera

A = Array

B = Buffer

C = Counter

D = Device

G = Datagroup

I = Input

O = Output

T = Timer

V = Variable

Codice a due lettere: seconda lettera

C = *Funzioni di Calcolo*

W = *Funzioni di Scrittura*

R = *Funzioni di Lettura*

A = *Funzioni che eseguono un'azione*

T = *Funzioni di Controllo*

N = *Funzioni contemplate dalle norme IEC1131*

Note per l'utente

1) L'utilizzo delle funzioni può portare ad una modifica del file simboli, quindi, nel caso di utilizzo di una interfaccia che colloquia con Qmove (pc o terminali vari), c'è la necessità di reimportare i simboli nel progetto.

2) Le variabili interne utilizzate dalle funzioni QCL assumono nomi che iniziano con ZZ_ZZqem_nnn dove con nnn si indica un numero interno di riferimento. L'uso di variabili con tale nome da parte dell'applicativo è sconsigliato, in quanto il comportamento del sistema stesso, risulterebbe imprevedibile.

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.