

# WAIT

## WAIT action

### Function

Temporary suspension of the execution of actions.

### Declaration

```
WAIT expression
```

or

```
WAIT
```

### Parameters

|            |    |                           |
|------------|----|---------------------------|
| expression | in | Expressions of BOOL type. |
|------------|----|---------------------------|

### Description

#### **WAIT action with an expression:**

Holding the execution of an action till the condition is not met. The condition is met, when a value of the expression is TRUE.

#### **WAIT action without expression:**

Holding the execution of actions till the last assignment is executed. This variant of the action WAIT allows to remove time hazards and also check the assignment success. Assignment to a local variable is immediate (synchronously with the execution of actions). On the other hand, assignment to an object (setting an object value) represents only a request and its execution may take a while. There is the possible risk of time hazard.

### Example

It assumes the existence of the object `U.Int` of `User variable` type, integer value type.

```
; assignment with delay
U.Int := 1 ; value assignment
WAIT ; waiting for the assignment
```

```
; if the action WAIT is successful (no error),
```

```
; the value of the object U.Int is certainly 1
IF U.Int = 1 THEN ; value test
    ; the branch THEN-ELSE is executed whenever the value of the object U.
    Int is 1
ELSE
    ; the branch ELSE-ENDIF won't be never executed
```

```
ENDIF
```

```
U.Int := 2
; In this case there is not guaranteed that the value of the object U.Int
will be 2
; it depends on the current system load ...
IF U.Int = 1 THEN
ELSE
ENDIF
```

The action WAIT allows to test the success of the assignment of a value to an object. For example, writing into an output I/O tag, that is performed via process [D2000 KOM](#).

## Example

It assumes the existence of the object `M.Cmd` of I/O tag type, output integer type.

```
INT _maxWriteCount = 10 ; maximal count of writings
INT _writeCount ; writing counter

ON ERROR WriteFailed ; error handle

_writeCount := 0
WriteRetry:
  IF _writeCount >= _maxWriteCount GOTO WriteLoopFailed
  _writeCount := _writeCount + 1
  M.Cmd := 1
  WAIT
  ; Successful writing
  END

; Writing failed _maxWriteCount-times
WriteLoopFailed:
  END
; unsuccessful writing handle
WriteFailed:
  GOTO WriteRetry
```

The listed example handles all the error states, that may occur during a writing:

- process [D2000 KOM](#) is not running,
- station communication failure,
- writing into a device failed,
- object is not in the manual mode.

In case of assigning a selection, then this is controlled as the whole. Every fractional writing must be well executed. Such a situation occurs, for the assignment of a value of *structure* type, if references to objects are in individual items.

## Example

[Structure definition](#) `SD.Rec` contains one item of *Integer* type with the name `Int` and two items of *Object* type with the names `s Obj1` and `Obj2`.

The structure type of the object `SV.Rec` of [Structured variable](#) type is `SD.Rec` and its row number is 2. There are the objects `M.1`, `M.2`, `M.3`, `M.4` - output I/O tags of *Integer* type.

```
RECORD NOALIAS (SD.Rec) _locArr ; local variable of Structure type
                                ; disables references to objects
REDIM _locArr[2]                ; resizing the array length
```

```
; assignment of values to the local variable of Structure type
```

```
_locArr[1]^Int := 1
_locArr[1]^Obj1 := 1
_locArr[1]^Obj2 := 2
_locArr[2]^Int := 2
_locArr[2]^Obj1 := 3
_locArr[2]^Obj2 := 4
```

```
; assign references to the objects
SET SV.Rec[1] ^Obj1 AS M.1
SET SV.Rec[1] ^Obj2 AS M.2
SET SV.Rec[2] ^Obj1 AS M.3
SET SV.Rec[2] ^Obj2 AS M.4
WAIT ; execution of the reference change in the system
```

```
P1:  
SET SV.Rec AS _locArr ; assignment of the whole value  
WAIT
```

In the example, in the line marked by the label P1, there is performed the assignment of quite 6 values, 4 of which execute assignments to the I/O tags (using the reference to object). The operation success may be verified the following action WAIT. Possible failure of arbitrary assignment will cause the error.



**Related pages:**

[Script actions](#)